5.0 Deployment and User Guide

Last Updated: 4/10/2023

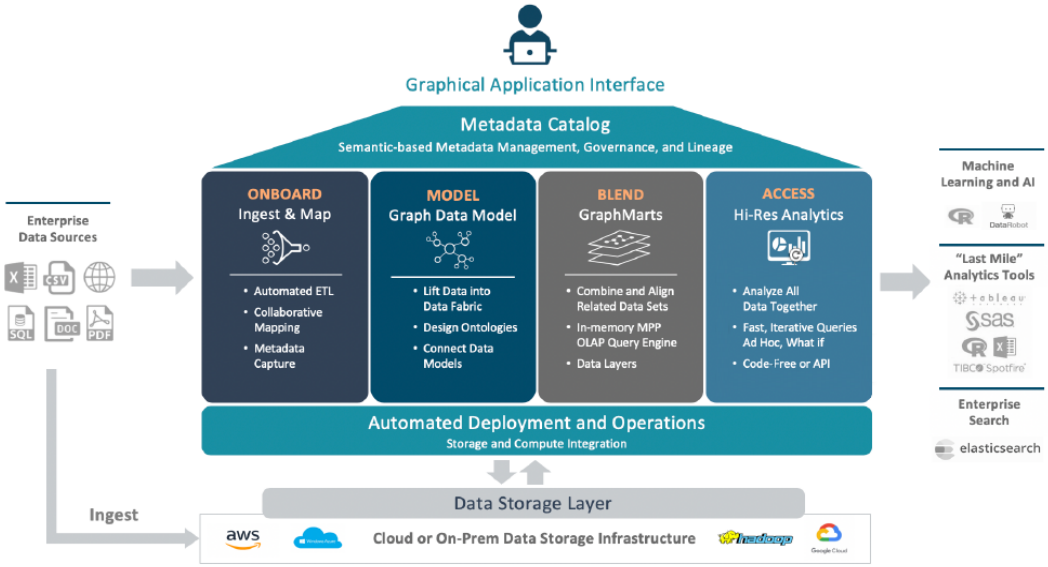
Table of Contents

Anzo 5.0 Deployment and User Guide

Welcome to the Anzo 5.0 Deployment and User Guide! This guide provides deployment instructions, guidance on getting started, administration and configuration information, and instructions for using Anzo Version 5.0 components.

Anzo Concepts and Vocabulary

This topic introduces you to key features, concepts, and vocabulary to know when working with Anzo. The diagram below shows a high-level overview of Anzo components and concepts. Details about the components in the image are described below, followed by a glossary that defines common Anzo terms and phrases.



Component	Description
Enterprise Data Sources	Anzo onboards data from many structured and unstructured data sources. Structured data sources such as relational databases or flat files are onboarded using Anzo's built-in pipelines. These pipelines natively support CSV, JSON, XML, and SAS files, along with all common database connections, including SQL, Oracle, MySQL, HIVE and others. Unstructured data sources, such as documents, PDFs, text snippets, web pages, and content from knowledgebases, are onboarded using configurable unstructured data pipelines. These pipelines onboard unstructured source files, integrate metadata about those files into the Anzo catalog, and make the full text in those files and key facts available as part of graph data models.
Data Storage Layer	The Anzo platform components, AnzoGraph, Spark, Elasticsearch, etc., share a file system for maintaining onboarded graph data and supporting files. Anzo supports storage systems such as HDFS, AWS S3, and NFS.

Component	Description
Metadata Catalog	Anzo's catalog is a special-purpose graph data model within Anzo. It combines traditional technical, operational, and business metadata with a semantic layer to describe all aspects of enterprise data elements. The catalog enables Anzo's unique use of semantics and graph models and is the system of record for data in Anzo. Anzo collects and generates metadata at every stage in the data discovery and integration process. Metadata in the catalog documents how data is converted during the onboarding process from its original format into a graph model. Subsequent data blending, transformation, and preparation steps are captured as additional metadata. Anzo also captures new metadata to describe all actions taken against data within Anzo. Anzo uses the metadata to enable users to visualize their data, understand business contexts, identify connections, and blend and prepare data.
Onboard	When data is onboarded from its source platform to Anzo, it is converted from its original format to a new format that describes the data as a graph data model. This format, Resource Description Framework (RDF), captures each data value and relationship. Anzo stores the converted RDF data in files that Anzo's catalog manages. RDF data, in Turtle (TTL) format, is efficiently laid out on disk for optimal loading into Anzo's in-memory graph engine, AnzoGraph. Mappings describe how data from source systems is transformed into Anzo's RDF format. These mappings can be automatically generated from the source system's schema or custom-defined to perform additional transformation steps as part of the onboarding process. At run time, Anzo converts these mappings into the code that is executed on the ETL engine.
Data Dictionary	Data Dictionaries enable users to automate, accelerate, and simplify the process of mapping data from enterprise data sources into semantic graph business models. A data dictionary is a centralized repository of the concepts that link the logical business models to the physical schemas of the data sources that feed it. The data dictionary structure becomes the basis for creating and reusing models and mappings across data sources.

Component	Description
Model	Anzo establishes the semantic layer by enabling users to convert diverse enterprise data models into graph data models and then enhance the data by adding new business definitions, names, and tags. Further insight is added when data from separate graph data maps are linked, connecting shared business definitions across previously siloed sources. Anzo employs open World Wide Web Consortium (W3C) standards, including Web Ontology Language (OWL), RDF, and SPARQL to model, connect, and query interconnected graphs.
Blend	When users select one or more data sets from the Anzo catalog to blend and access, Anzo loads the corresponding files from the file store into memory for rapid analysis and manipulation. Anzo delivers blending and access through “graphmarts” that give users the flexibility to combine and analyze any subset of data in Anzo.
Access	Once data has been onboarded, modeled, and blended into the dataset catalog and graphmarts, users have several options for accessing and analyzing the data. Anzo’s Hi-Res Analytics application enables users to create dashboards for exploring and visualizing the data without needing to have specialized query knowledge. The Query Builder provides access for finding specific statements or writing and running SPARQL queries. And the Anzo Data on Demand service provides access to data from business intelligence tools (see Last Mile Analytics Tools).
Graphical Application Interface	The graphical application interface is organized and compartmentalized by the concepts and processes described above. The design accommodates use cases where users with different permissions and responsibilities build various parts of a solution. For a tour of the graphical user interface, see Introduction to the User Interface.
Last Mile Analytics Tools	In line with its open standard architecture, Anzo graphmarts can be accessed using modern application program interfaces (APIs). In addition to using SPARQL-compliant query endpoints, Anzo offers standards-compliant Open Data Protocol (OData)-based REST data feed endpoints as part of its data on demand service.

Anzo Glossary

The table below defines commonly used Anzo terms and phrases.

Phrase	Description
Anzo for Office	Anzo for Office (AFO or A4O) is the Microsoft Excel plugin that enables you to create and edit source to target ETL mappings.
Anzo Data Store	An Anzo data store (previously known as a graph data source) defines an endpoint for writing data. It specifies the file store and directory on the file store where Anzo can generate file-based linked data sets (see File-Based Linked Data Set). It also defines write properties such as the maximum file size and whether files should be compressed.
AnzoGraph	AnzoGraph (AZG) is Anzo's in-memory massively parallel processing (MPP) graph OLAP engine.
Data Layers	Data layers enable users to enhance graphmarts dynamically. Users can create layers to load additional data sets, clean, conform, or transform data, infer new information, or export data to a file-based linked data set (FLDS).
Data Toolkit	Anzo's Data Toolkit is a SPARQL service that enables users to query data from external endpoints that are accessible over JDBC or HTTP. Information from the external sources can augment data in the Anzo catalog without having to onboard the data to Anzo.
ELT	In addition to traditional ETL, Anzo's data layers capability enables users to transform, blend, and prepare any data that has been added to the catalog into analytics-ready data sets using an extract, load, transform (ELT) flow. Data layers are Anzo's mechanism for flexibly transforming data in memory.
ETL	The extract, transform, and load (ETL) process takes source data and converts it to the graph data model using a source to target mapping. Anzo's mapping tool enables users to define field-level transformations, including type casting, date conversions, unit conversions, etc., as data is onboarded to Anzo.

Phrase	Description
File-Based Linked Data Set	When the onboarding process is complete, Anzo creates a data set in the metadata catalog. The data set in the catalog is registered in the Anzo system data source (see Journal or Volume) and includes metadata about the data, including a pointer to the data store location for the RDF files generated by the ETL pipeline. The catalog data set and the files on disk are known as a file-based linked data set (FLDS).
File Store	A file store is the file storage system, such as NFS, HDFS, or cloud storage, that is shared between servers in an Anzo solution. Anzo, AnzoGraph, Elasticsearch, and other systems share data in a file store.
Graphmart	Graphmarts are metadata descriptions of collections of datasets that users can share, discover, and enhance. Graphmarts can combine any subset of data in Anzo for analysis.
Hi-Res Analytics	Anzo Hi-Res Analytics enable users to explore and ask questions across all of their data. Using model-guided dashboards, users can perform computations across multi-dimensional data. Hi-Res Analytics dashboards generate complex graph queries dynamically based on user input.
IRI	An Internationalized Resource Identifier (IRI) is similar to URI but allows a greater range of characters. URI and IRI are often used interchangeably.
Journal or Volume	A journal, also known as a volume, refers to data that is stored in Anzo's embedded graph store. The graph store is transactional and is used to persist metadata, which is written to disk in a .jnl file. The system volume (or system data source) is the default, required volume where Anzo stores ontologies as well as system configuration, data set, catalog, registry, and access control metadata. Users can create secondary local volumes that are used for more compartmentalized data and can be created and deleted without affecting the core system.
Linked Data Set	A linked data set (LDS) is a fundamental concept. Anzo organizes all data, including system data, into linked data sets. An LDS is associated with a data model and can be searched, discovered, shared, and protected with access control. For example, graphmarts are organized in a linked data set or registry of graphmarts, pipelines are organized in a linked data set, the Activity Log is a linked data set, data source configurations exist in a linked data set, and so on.

Phrase	Description
NLP	Anzo onboards unstructured data using natural language processing (NLP) to find and extract data.
OData	Open Data Protocol (OData) facilitates the creation of interoperable RESTful APIs. The Anzo Data on Demand service provides OData-based feeds that can be used to query graphmart data from third-party business intelligence tools.
OSGi	The Open Service Gateway Initiative (OSGi) is the open-standard architecture upon which Anzo is built. It is a Java framework for developing and deploying software programs and libraries. OSGi enables Cambridge Semantics to compartmentalize Anzo into "bundles" that can be deployed, activated, and removed independently without affecting other bundles in the system.
Provenance	Anzo retains and displays the provenance of all onboarded structured data. The provenance explorer provides an overview of the relationships across various sources and models. Users can search for data entities and view associated pipelines, data sources, models, and schemas.
Registry	Anzo manages configurations in system-level registries. Each registry is a collection of application and system component configurations of the same type. Like data, registries are stored and managed with RDF named graphs according to ontologies. Technically, a registry is a Linked Data Set.
URI	A Uniform Resource Identifier (URI) is a globally unique identifier for a piece of information. A URL (Uniform Resource Locator) is a URI that specifies a location, such as a web address.

Related Topics

Introduction to the User Interface

User Guide

Deployment Guide

The Deployment Guide provides information for users who will deploy the Anzo and AnzoGraph infrastructure and install the software. The topics in this section describe the Anzo and AnzoGraph requirements and provide instructions for deploying the components.

Anzo and AnzoGraph Requirements

The topics in this section provide details about the Anzo and AnzoGraph server requirements.

- Anzo Requirements
- AnzoGraph Requirements
- AnzoGraph Server Sizing Guidelines

Anzo Requirements

This page provides important guidelines to follow when choosing the hardware and software for servers that host Anzo.

- Hardware Requirements
- Software Requirements
- Firewall Requirements
- File Storage Requirements
- Standalone Ingestion Server Requirements

For information about Anzo Unstructured architecture requirements, see Anzo Unstructured Requirements and Recommendations.

Hardware Requirements

Cambridge Semantics lists above average production system hardware requirements as a guideline. These specifications are similar to what Cambridge Semantics currently provisions as the standard hosted environment. Larger production data sets running interactive queries may require significantly more powerful hardware and RAM configurations. Keep in mind that you are installing both a high performance graph database server as well as a fully featured application server. Provision production server hardware accordingly to avoid performance issues.

The table below provides a summary of the recommended hardware for production servers and the minimum requirements for test servers.

Component	Minimum	Recommended	Guidelines
Available RAM	8 GB	**32 GB or more**	The Anzo system data source is a disk-based graph store (called a Journal or Volume). When the system source is queried, Anzo swaps the data from disk to memory on demand. Choosing a host server with more RAM increases the performance of system queries because the OS can store the journal data in its file cache, avoiding the need for Anzo to swap data from disk to memory. In addition, RAM is required to hold intermediate results for join queries.
Disk space and type	10 GB (Anzo Server) 100 GB (Data)	**100 GB (Anzo Server)** **1+ TB (Data)**	See File Storage Requirements below.
CPU	4 core 2.2GHz	**8 core 3GHz+**	Once you provision sufficient RAM and a high-performing I/O subsystem, performance depends on CPU capabilities. Keep in mind that you are provisioning for both a production database and a busy application server. A greater number of cores and high clock speed can make a dramatic difference in the performance of interactive queries.
Architecture	64-bit	**64-bit**	Cambridge Semantics only supports the 64-bit versions of the server for production use.

Software Requirements

This section lists the software requirements for Anzo servers and client workstations. It also includes user account information and lists the supported single sign-on providers.

Component	Minimum	Recommended	Guidelines
Operating System (Anzo Server)	RHEL/CentOS 6	RHEL/CentOS 7	Cambridge Semantics recommends that you tune the ulimits for your Linux distribution to increase the limits for certain resources. See Configure User Resource Limits for more information.
Microsoft Excel (Client Workstation)	Excel 2003	Excel 2007+	The Anzo for Office data integration mapping tool plugin requires Microsoft Excel.
Web Browser (Client Workstation)	Firefox 62+ Chrome 74+ Safari 12+	Chrome 80+	Use the latest versions of web browsers as some older versions will not work with the Anzo user interface components.
Enterprise-Level Anzo Service User Account	N/A	N/A	It is important to work with your IT organization to create an Anzo service user account at the enterprise level. The service user account needs to be associated with a central directory server (LDAP) so that it is available across Anzo environments and is managed in accordance with the permissions policies of your company. For more information, see Anzo Service User Account Requirements below.

Anzo Service User Account Requirements

For consistent and appropriate access management across current and future Anzo environments, it is important for the IT organization to create an enterprise-level, LDAP-managed Anzo service user account. The service user account should be used when installing and running Anzo and all component environments, such as AnzoGraph, Spark, Elasticsearch, and Anzo Unstructured clusters. The service account should not have root user privileges but does need the following access:

- The account must have read and write permissions for the Anzo component installation directories. The default Anzo server installation directory is `/opt/Anzo`.
- The account must have read and write access to the shared file store, such as the NFS mount location, where all Anzo components will read and write files during the data onboarding processes.
- The account must have a home directory on the Anzo host server.

Supported Single Sign-On Providers

Anzo supports the following single sign-on (SSO) protocols:

- Basic SSO
- Facebook OAuth
- JSON Web Tokens (JWT)
- Kerberos
- OpenID Connect (OIDC)
- Security Assertion Markup Language (SAML)
- Spring Security OAuth2

For information about configuring SSO access, see Configuring SSO Access.

Firewall Requirements

The table below lists the TCP ports to open on the Anzo host.

Port	Description	Access Needed...
61616	Anzo port used by the software development kit (SDK) and command line interface (CLI)	• Between Anzo and users
61617	Anzo SSL port used by the SDK and CLI	• Between Anzo and users
8022	Anzo SSH service port	• Between Anzo and users
8945	Anzo Administration service port	• Between Anzo and users
8946	Anzo Administration service SSL port	• Between Anzo and users
80	Application HTTP port	• Between Anzo and users
443	Application HTTPS port.	• Between Anzo and users
3389	LDAP port	• Between Anzo and the LDAP server
9393 (optional)	Optional Java Management Extensions (JMX) port. Enable this port if you want to connect to Anzo from a JMX client.	• Between Anzo and the JMX client

Port	Description	Access Needed...
9394 (optional)	Optional JMX SSL port. Enable this port if you want to make a secure connection to Anzo from a JMX client.	• Between Anzo and the JMX client
5700	The Anzo protocol (gRPC) port for secure communication between AnzoGraph and Anzo For more information about the communication between Anzo and AnzoGraph, see Firewall Requirements in AnzoGraph Server Requirements.	• Between Anzo and the AnzoGraph leader server
5600	AnzoGraph's SSL system management port	• Between Anzo and the AnzoGraph leader server

File Storage Requirements

Anzo supports reading from and writing to storage systems such as a mounted NFS, Hadoop Distributed File Systems (HDFS), File Transfer Protocol (FTP or FTPS) systems, Google Cloud Platform (GCP) storage, and Amazon Simple Cloud Storage Service (S3).

Set up a storage system that is accessible by both Anzo and AnzoGraph. Depending on your infrastructure and use case, you might need to have enough storage space available for storing source data files, RDF load files, ETL job files, and other supporting files.

For more information about connecting to file storage, see Connecting to a File Store.

Standalone Ingestion Server Requirements

Anzo includes an embedded Spark ETL engine to integrate data from various sources. Depending on your server configuration, the embedded engine might not be sufficient for ingesting very large amounts of data. To support ingestion of large data sets, you can install standalone ingestion servers. This page lists the recommended configuration for standalone data ingestion servers.

Component	Recommendation
Available RAM	100+ GB
Disk Space	200+ GB
CPU	16+ cores

Related Topics

Installing Anzo

AnzoGraph Requirements

This topic lists the minimum requirements and recommendations to follow for setting up static AnzoGraph host servers and cluster environments.

Hardware Requirements

Cambridge Semantics lists above average production system hardware requirements as a guideline. Large production data sets running interactive queries may require significantly more powerful hardware and RAM configurations. Provision production server hardware accordingly to avoid performance issues.

Component	Minimum	Recommended	Guidelines
Available RAM	16 GB (for small-scale testing only)	**200 GB or more**	AnzoGraph needs enough RAM to store data, intermediate query results, and run the server processes. Cambridge Semantics recommends that you allocate 3 to 4 times as much RAM as the planned data size. Do not overcommit RAM on a VM or on the hypervisor/container host. Avoid memory paging to disk (swapping) to achieve the highest possible level of performance. For more information about determining the server and cluster size that is ideal for hosting AnzoGraph, see AnzoGraph Server Sizing Guidelines.
Disk space and type	10 GB HDD	**200+ GB SSD**	AnzoGraph requires 10 GB for internal requirements. The amount of additional disk space required for load file staging, data persistence, or backups depends on the size of the data to be loaded. For persistence, Cambridge Semantics recommends that you have twice as much disk space on the local AnzoGraph file system as RAM on the server.

Component	Minimum	Recommended	Guidelines
CPU (Intel only)	2 cores	**32 cores**	Once you provision sufficient RAM and a high-performing I/O subsystem, performance depends on raw CPU capabilities. A greater number of cores can make a dramatic difference in the performance of interactive queries. Do not overcommit CPU cores so that the sum of vCores becomes greater than the number of real CPU cores. **AnzoGraph does not support AMD processors at this time**.
Networking	10gbE	**20+gbE**	Not applicable for single server installations. Since AnzoGraph is high performance computing (HPC) Massively Parallel Processing (MPP) OLAP engine, inter-cluster communications bandwidth dramatically affects performance. AnzoGraph clusters require optimal network bandwidth. **Important** All servers in a cluster must be in the same network. Make sure that all instances are in the same VLAN, security group, or placement group. In a switched network, make sure that all NICs link to the same Top Of Rack or Full-Crossbar Modular switch. If possible, enable SR-IOV and other HW acceleration methods and dedicated layer 2 networking that guarantees bandwidth.

Clusters and Virtual Environments

AnzoGraph requires that all elements of the infrastructure provide the same quality of service (QoS). Do not run AnzoGraph on the same server as any other software except when in single-server mode and with an expectation of lowered performance. Providing the same QoS is especially important when using AnzoGraph in a clustered configuration. If any of the servers in the cluster perform additional processing, the cluster becomes unbalanced and may perform poorly. A single poor performing server degrades the other servers to the same performance level. **All nodes require the same hardware specification and configuration**. Also use static IP addresses or make sure that DHCP leases are persistent.

To ensure the maximum and most reliable QoS for CPU, memory, and network bandwidth, do not co-locate other virtual machines or containers (such as Docker containers) on the same hypervisor or container host. For hypervisor-managed VMs, configure the hypervisor to reserve the available memory for the AnzoGraph server. For clusters, make sure there is enough physical RAM to support all of the AnzoGraph servers, and reserve the memory via the hypervisor.

In addition, running memory compacting services such as Kernel Same-page Merging (KSM) impacts CPU QoS significantly and does not benefit AnzoGraph. Live migrations also impact the performance of VMs while they get migrated. While live migration can provide value for planned host maintenance, AnzoGraph performance may be impacted if live migrations occur frequently. For more information about Kernel Same-page Merging, see https://en.wikipedia.org/wiki/Kernel_same-page_merging.

Note Advanced configurations may benefit from CPU pinning on the hypervisor host and disabling CPU hyper-threading. For more information about CPU pinning, see https://en.wikipedia.org/wiki/Processor_affinity. For information about hyper-threading, see https://en.wikipedia.org/wiki/Hyper-threading.

Cambridge Semantics can provide benchmarks to establish relative cluster performance metrics and validate the environment.

Software Requirements

The table below lists the software requirements for AnzoGraph servers.

Component	Minimum	Recommended	Guidelines
Operating System	RHEL6, CentOS6	**RHEL7+, CentOS7+**	See Tuning User Resource Limitations (ulimits) below for information about setting ulimits. **Note** AnzoGraph is not supported on RHEL/CentOS 8 at this time.
GNU Compiler Collection	N/A	**Installed**	Install the latest version of the GCC tools for your operating system.
bzip2	N/A	**Installed**	Required for unpacking the AnzoGraph tool set during installation.

Tuning User Resource Limitations (ulimits)

Cambridge Semantics recommends that you tune the user resource limits (ulimits) for your Linux distribution to increase the limits for the following resources:

- Increase the **open files** limit to at least **4096**.
- Increase the limit for the following resources to **unlimited**:
  - cpu time
  - file locks
  - file size
  - max memory size
  - max user processes
  - virtual memory

To view the current ulimits, run `ulimit -a`. To permanently change ulimits, modify the `/etc/security/limits.conf` file. For more information, see How to set ulimit values in the RHEL support documentation.

Note Also make sure that the Linux environment variables **http_proxy** and **https_proxy** are not set. The Anzo gRPC protocol cannot make connections to the database when proxies are enabled.

Optional Software

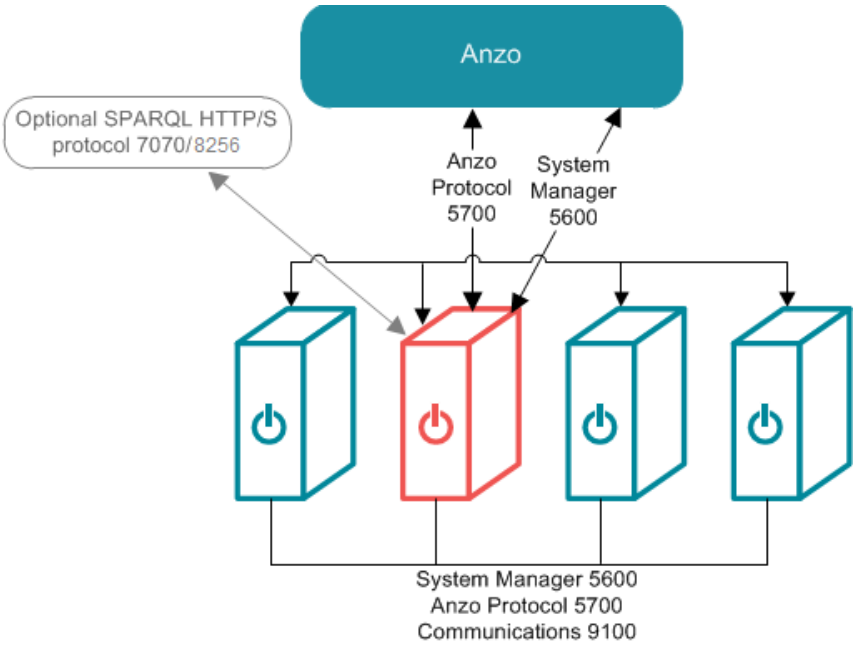
Program	Description
vim	Editor for creating or changing files.
sudo	Enables users to run programs with alternate security privileges, such as the azg_system_config utility that configures the required AnzoGraph kernel settings.
net-tools	Networking utilities.
psutil	Python system and process utilities for retrieving information on running processes and system usage.
tuned	Linux system service to apply tunables.
wget	Utility for downloading files over a network.
Google SDK	For virtual servers on Google Cloud Engine (GCE). Command line tool to enable syncing of data from Google storage. You can download the latest version from Google: https://cloud.google.com/sdk/.

Firewall Requirements

AnzoGraph servers communicate via TCP/IP sockets. AnzoGraph communicates with Anzo via the secure, encrypted, gRPC-based Anzo protocol. Since AnzoGraph is SPARQL-compliant, you also have the option to use standard SPARQL HTTP/S protocol for communication.

Important For AnzoGraph clusters, all servers in the cluster must be in the same network. Make sure that all instances are in the same VLAN, security group, or placement group.

Open the TCP ports listed in the table below. This image shows a visual representation of the communication ports:



Port	Description	Access Needed...
5700	The Anzo protocol (gRPC) port for secure communication between AnzoGraph and Anzo	• Between Anzo and the AnzoGraph leader server • Between all AnzoGraph servers in the cluster • Available for AnzoGraph on single node installations
5600	AnzoGraph's SSL system management port	• Between Anzo and the AnzoGraph leader server • Between all AnzoGraph servers in the cluster • Available for AnzoGraph on single node installations

Port	Description	Access Needed...
9100	AnzoGraph's internal fabric communications port	• Between all AnzoGraph servers in a cluster • Available for AnzoGraph on single node installations
7070 (optional)	Optional SPARQL service HTTP port to enable if you want to give external applications access to AnzoGraph	• Between external applications and the AnzoGraph leader server
8256 (optional)	Optional SPARQL service HTTPS port to enable if you want to give external applications access to AnzoGraph	• Between external applications and the AnzoGraph leader server

Related Topics

AnzoGraph Server Sizing Guidelines

AnzoGraph Server Sizing Guidelines

This topic provides guidance on determining the server and cluster size that is ideal for hosting AnzoGraph, depending on the characteristics of your data.

- Memory Sizing Guidelines
- Analyzing Data Characteristics in Load Files
- Cluster Sizing Guidelines

Memory Sizing Guidelines

Since AnzoGraph is a high-performance, in-memory database, it is important to consider the amount of memory needed to store the data that you plan to load. Estimating the amount of memory your workload requires can help you decide what size server to use and whether to use multiple servers. The sections below describe the key points to consider about memory usage and AnzoGraph.

- Data at rest should remain below 50% of the total memory
- AnzoGraph reserves 20% of the memory for the OS
- Memory usage can be high during loads
- Memory usage depends on data characteristics

Data at rest should remain below 50% of the total memory

The data loaded into memory should not consume more than 50% of the total available memory on the instance or across a cluster. Ideally, the data at rest should use only 25%-30% of the available memory because query

processing and intermediate results can temporarily consume a very large amount of RAM.

AnzoGraph reserves 20% of the memory for the OS

To avoid unexpected shutdowns by the Linux operating system, the default AnzoGraph configuration leaves 20% of memory available for the OS; AnzoGraph will not use more than 80% of the total available memory. Account for this memory buffer in sizing calculations.

Memory usage can be high during loads

During the load streaming process, before duplicates are pruned and triples are moved to their final storage blocks, memory usage temporarily increases and potentially doubles, particularly if the data includes many string values.

Memory usage depends on data characteristics

Memory usage varies significantly depending on the makeup of the data, such as the data types and sizes of literal values, and the complexity of the queries that you run. Triple storage ranges anywhere from 12 bytes per triple to 1 megabyte for a triple that stores pages of text from an unstructured document. For example:

- Triples with integer objects like the following example require about 16 bytes to store in memory.

```
<http://csi.com/resource/person1> <http://csi.com/resource/age> 50
```

- Triples made up of URIs like the following example require about 18 bytes to store in memory.

```
<http://csi.com/resource/person1> <http://csi.com/resource/friend>
<http://csi.com/resource/person100>
```

- Triples with user-defined data types (UDTs) like the following example also require about 18 bytes to store in memory.

```
<http://csi.com/resource/person1> <http://csi.com/resource/height>
"5'8""^^height
```

- Triples with dateTime values like the following example require about 20 bytes to store in memory.

```
<http://www.wikidata.org/entity/Q65949130>
<http://www.wikidata.org/prop/direct/P585>
"1995-01-01T00:00:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTime> .
```

- Triples with long strings like the following example require about 700 bytes to store in memory.

```
<http://dbpedia.org/resource/Keanu_Reeves> <http://dbpedia.org/ontology/abstract>
"Keanu Charles Reeves
(/kɛrˈɑːnuː/ kay-AH-noo; born September 2, 1964) is a Canadian actor, producer,
director and musician.
Reeves is best known for his acting career, beginning in 1985 and spanning more than
```

```
three decades.
He gained fame for his starring role performances in several blockbuster films
including comedies
from the Bill and Ted franchise (1989–1991), action thrillers Point Break (1991) and
Speed (1994),
and the science fiction-action trilogy The Matrix (1999–2003). He has also appeared
in dramatic
films such as Dangerous Liaisons (1988), My Own Private Idaho (1991), and Little
Buddha (1993),
as well as the romantic horror Bram Stoker's Dracula (1992)."
```

The table below provides estimates for the number of triples that you can load and query with commonly configured amounts of available RAM. The table also lists the number of triples that could be stored if a data set comprised the example triples above.

Note The examples below show the number of triples at rest and consider that the data should not consume more than 50% of the available RAM.

Available RAM	General Estimate	Examples
16 GB	Up to about 100 million triples	Considering that the data at rest should use less than 8 GB RAM, a server with 16 GB total RAM could store: • About 12 million 700-byte triples like the Keanu Reeves example above. • About 475 million 18-byte URI triples like the example above.
32 GB	Up to about 200 million triples	Considering that the data at rest should use less than 16 GB RAM, a server with 32 GB total RAM could store: • About 24 million 700-byte triples like the Keanu Reeves example above. • About 850 million 20-byte triples like the dateTime example above.

Available RAM	General Estimate	Examples
64 GB	Up to about 400 million triples	Considering that the data at rest should use less than 32 GB RAM, a server with 64 GB total RAM could store: • About 48 million 700-byte triples like the Keanu Reeves example above. • About 1.7 billion 20-byte triples.
128 GB	Up to about 800 million triples	Considering that the data at rest should use less than 64 GB RAM, a server with 128 GB total RAM could store: • About 96 million 700-byte triples like the Keanu Reeves example above. • About 3.4 billion 20-byte triples.
256 GB	Up to about 1.5 billion triples	Considering that the data at rest should use less than 128 GB RAM, a server with 256 GB total RAM could store: • About 192 million 700-byte triples like the Keanu Reeves example above. • About 6.8 billion 20-byte triples.
480 GB	Up to about 3 billion triples	Considering that the data at rest should use less than 240 GB RAM, a server with 480 GB total RAM could store: • About 368 million 700-byte triples like the Keanu Reeves example above. • About 12 billion 20-byte triples.

Analyzing Data Characteristics in Load Files

AnzoGraph enables you to perform pre-load analysis on file-based linked data sets without actually loading the data into memory. You can use this method to run statistical queries, such as counting the number of triples or returning a list of the unique subjects and predicates. Performing a "dry run" of a data load enables you to analyze data set characteristics to help with tasks such as memory sizing. Since the data remains on disk, you can use this method to capture statistics about a large data set without having to deploy an AnzoGraph cluster that has enough memory to store all of the data.

Important Considerations for Analyzing Load Files

- Since AnzoGraph scans the files on disk, queries run much slower than they do when run against data in memory. Consider performance when deciding how many files to query at once and how complex to make the queries.
- Though the pre-load feature does not use memory for storing data, queries that you run against files do consume memory. The server must have sufficient memory available to use for these intermediate query results.
- Unlike loads into the database, pre-load analysis does not prune duplicate triples. Statistics returned for load file queries may differ somewhat from the statistics returned after the data is loaded.

Analysis Query Syntax

Use the following query syntax to analyze load files :

```
SELECT expression
FROM EXTERNAL <URI>
[ FROM EXTERNAL <URI> ]
WHERE { triple_patterns }
```

Option	Description
`SELECT` *`expression`*	The SELECT clause specifies an expression that returns statistical results such as a count of the total number of triples or the number of distinct predicates. Queries that return values for a specific property may return an error.

`FROM EXTERNAL` `<`*`URI`*`>`	The URI in the FROM clause specifies the location of the load file or directory of files. For example, this URI specifies a single file:
	`<file:/data/load/values.ttl>`
	This example specifies a directory of files:
	`<dir:/data/store/LoadDBNorthwind/rdf.ttl.gz>`

For example, the following query analyzes the files in the rdf.ttl.gz directory for an FLDS. The query counts the total number of triples in the files:

```
SELECT (count (*) as ?triples)
FROM EXTERNAL <dir:/nfs/data/store/LoadGHIB_f5886/rdf.ttl.gz>
WHERE { ?s ?p ?o . }
```

```
triples
-----------
```

```
143704445
1 rows
```

Assessing Memory Requirements Based on File Analysis

Although the memory required to load and perform queries on specific data sets will vary based on the size and type of data contained in a data set as well as the type of queries run, you can still obtain a reasonable estimate for the amount of memory you will need to store data set by using the equation below:

```
total_triples x avg_triple_size + total_chars = size_estimate(bytes)
```

Follow the steps below to calculate the values to use in the equation:

1. Count the total number of triples in the files
2. Determine the average triple size
3. Count the number of characters for all strings
4. Calculate the size estimate

Count the total number of triples in the files

As shown in the example above, the following query counts the total number of triples in FLDS load files:

```
SELECT (count (*) as ?triples)
FROM EXTERNAL <dir:/nfs/data/store/LoadGHIB_f5886/rdf.ttl.gz>
WHERE { ?s ?p ?o . }
```

```
triples
-----------
143704445
1 rows
```

Determine the average triple size

The Memory usage depends on data characteristics section above shows some example triples and their estimated size. If you are familiar with the data in the files, you may be able to determine the average size based on the examples. Otherwise, Cambridge Semantics recommends using 30 bytes as the average triple size.

Count the number of characters for all strings

For ASCII characters, AnzoGraph uses about 1-byte of memory to store each character. Counting the number of characters in the load files provides a good estimate of the number of bytes required to store the strings in your data.

```
SELECT (SUM(IF(DATATYPE(?o)=<http://www.w3.org/2001/XMLSchema#string>,
       (STRLEN(?o)),0)) as ?char_count)
FROM EXTERNAL <URI>
WHERE {?s ?p ?o}
```

For example, the following query returns the number of characters in the strings for the FLDS referenced above:

```
SELECT (SUM(IF(DATATYPE(?o)=<http://www.w3.org/2001/XMLSchema#string>,
       (STRLEN(?o)),0)) as ?char_count)
FROM EXTERNAL <dir:/nfs/data/store/LoadGHIB_f5886/rdf.ttl.gz>
WHERE {?s ?p ?o}
```

```
char_count
------------
684348190
1 rows
```

Calculate the size estimate

Once you have counted the triples, determined the average triple size, and counted the characters, use the formula below to estimate the amount of memory needed to store the data at rest:

```
total_triples x avg_triple_size + total_chars = size_estimate(bytes)
```

For example:

```
143,704,445 x 30 + 684,348,190 = 4,995,481,540 bytes
```

This example FLDS requires roughly 5 GB of memory to store the data.

Cluster Sizing Guidelines

When your workload size requires using a cluster, do not create clusters with fewer than 4 nodes. When using a single node, data gets redistributed in memory without using the network. If you add 1 or 2 more nodes to create a 2- or 3-node cluster, data then gets distributed over the network. The CPU gain from the additional 1 or 2 nodes does not outweigh the performance degradation from the network. Using at least 4 nodes significantly reduces the network degradation and provides a near-linear performance benefit when compared to a single node.

Related Topics

AnzoGraph Requirements

Installing AnzoGraph

Installing Anzo and AnzoGraph

The topics in this section provide instructions for installing Anzo, AnzoGraph, and the Anzo plugins.

- Installing Anzo
- Installing AnzoGraph
- Installing the Anzo for Excel and Office Plugins

Installing Anzo

This topic provides instructions for installing Anzo. For information about server requirements, see Anzo Requirements.

1. Complete the Pre-Installation Configuration
2. Install and Configure Anzo
3. Complete the Post-Installation Configuration

Complete the Pre-Installation Configuration

- Make Sure the Anzo Service User Account is Created
- Configure User Resource Limits

Make Sure the Anzo Service User Account is Created

Important It is important to work with your IT organization to ensure that an Anzo service user account is created at the enterprise level. The user account needs to be associated with a central directory server (LDAP) so that it is available for installing and running Anzo components across environments. For more information, see Anzo Service User Account Requirements.

If necessary, you can create a temporary user account on the Anzo host server. Note that creating the account locally can cause issues when migrating Anzo or integrating with a central LDAP server. The service account should meet the following requirements:

- The service account should not have root-user privileges.
- The account must have read and write permissions for the Anzo installation directory. The default installation directory is `/opt/Anzo`.
- The account must have read and write access to the shared file store, such as the NFS mount location, where Anzo will read and write files during the data onboarding processes.

  Note If your organization will use Anzo Unstructured with Elasticsearch to onboard unstructured data, it is especially important to install and run Anzo as a non-root user. Elasticsearch cannot be run by a root user, but it must have access to the data that Anzo writes on the shared file store. When Anzo is run as root the data that it generates is owned by root and Elasticsearch cannot access it.

Configure User Resource Limits

Cambridge Semantics recommends that you tune the user resource limits (ulimits) for your Linux distribution to increase the limits for the following resources:

- Increase the **open files** limit to at least **65535**.
- Increase the limit for the following resources to **unlimited**:
  - cpu time
  - file locks
  - file size
  - max memory size
  - max user processes
  - virtual memory

To view the current ulimits, run `ulimit -a`. To permanently change ulimits, modify the `/etc/security/limits.conf` file. For more information, see How to set ulimit values in the RHEL support documentation.

Install and Configure Anzo

Follow the instructions below to install Anzo. These instructions assume that you have copied the Anzo installation script to the server.

Important Complete the steps below as the Anzo service user.

1. If necessary, run the following command to become the Anzo service user:

   ```
   # su name
   ```

   Where *name* is the name of the service user. For example:

   ```
   # su anzo
   ```

2. If necessary, run the following command to make the Anzo installation script executable:

   ```
   chmod +x script_name
   ```

3. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

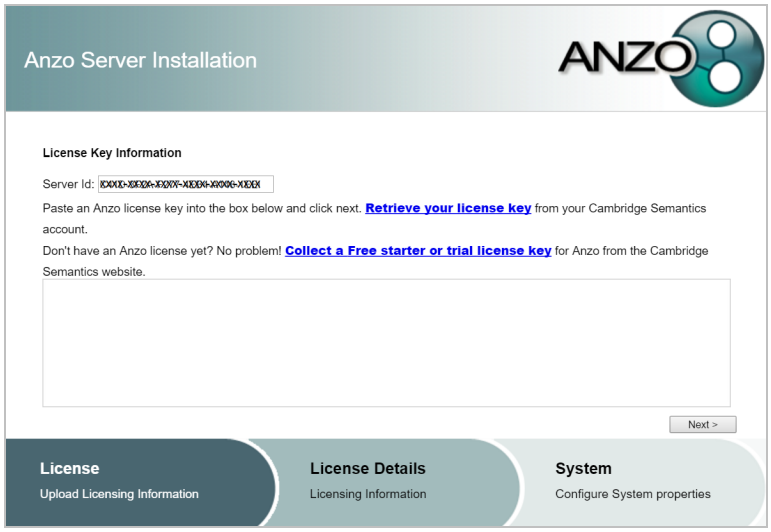
4. Press **Enter** to start the installation.

5. Review the software license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** to accept the terms or type **2** to disagree and stop the installation.
6. Specify the components to install. Item 1 is the Anzo server; item 2 is the Anzo command line client. To install both components, accept the default value by pressing **Enter**. Or type **1** to install only the server or **2** to install only the command line client, then press **Enter**.
7. Specify the path and directory for the Anzo installation. Press **Enter** to accept the default installation path or type an alternate path and then press **Enter**.
8. Indicate whether you want the installer to create symlinks. Press **Enter** for yes or type **n** and press **Enter** for no.
9. If you chose to let the installer create symlinks, specify the directory to create the symlinks in. Press **Enter** to accept the default path or type an alternate path and then press **Enter**.
10. Specify the maximum amount of memory (in MB) that the server can use and then press **Enter**. The installation wizard lists the total RAM available. To meet the minimum memory requirement, the wizard chooses 1/4 of the total memory as the default value. Cambridge Semantics recommends that you allocate at least 1/2 of the total memory to Anzo.

    The wizard installs the components that you selected and then asks if you want to start the Anzo services.
11. Press **Enter** to start the Anzo services. When prompted, open a browser and go to the following URL to open the license administration wizard.

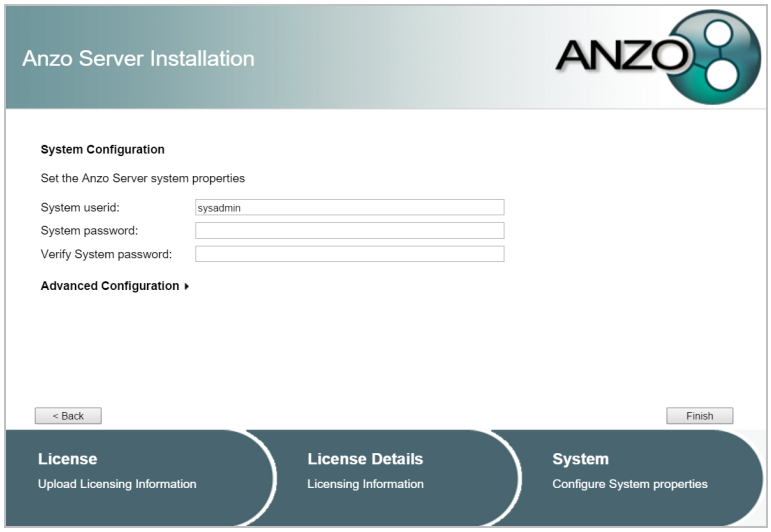
    ```
    http://Anzo_server:8945/
    ```

    Where *Anzo_server* is the Anzo server DNS name or IP address. The License Key Information screen appears.

12. Paste your license key into the box provided and then click **Next**. If necessary, you can obtain the license key by clicking **Retrieve your license key** and logging in to your Cambridge Semantics account.

13. The wizard displays your license details. Review the details and then click **Next**. The wizard displays the System Configuration screen.



14. Specify the user ID and password for the system administrator. The default system user ID is **sysadmin**. You can click **Advanced Configuration** to view the advanced **Storage Directory** setting. This setting configures the binary store location. By default Anzo stores binary data in `/opt/Anzo/Server/data`. You can change the location by typing a new path and directory.
15. Click **Finish**. The wizard starts configures and restarts the server. The process may take several minutes. Once the server is running, the browser displays the Anzo login screen. Before logging in, there is one more configuration step to complete. Some of the Anzo services will not have started properly because they could not bind to the default HTTP/S ports. The default Anzo HTTP port is 80 and the HTTPS port is 443. Since non-root users cannot access ports below 1000, Anzo services will not be able to access the default ports when Anzo is run by the new service user. The Anzo port settings need to be changed to the non-root ports 8080 and 8443:
    a. On the Anzo server, run the following command to make an SSH connection to the Anzo Command Console as the **sysadmin** user:

    ```
    ssh sysadmin@localhost -p 8022
    ```

    b. When prompted, specify the password for the sysadmin user and log in to the Anzo OSGI Command Console.
    c. At the OSGI prompt, run the commands below, followed by exit to exit the console:

    ```
    osgi> httpPort 8080
    ```

    ```
    osgi> httpsPort 8443
    ```

```
osgi> exit
```

16. Run the following command to restart Anzo and complete the port configuration:

    ```
    ./install_path/Anzo_root_dir/Server/AnzoServer restart
    ```

17. When Anzo starts, open the Anzo user interface by going to the following URL in your browser:

    ```
    https://hostname
    ```

    Where *hostname* is the Anzo server DNS name or IP address.

Complete the Post-Installation Configuration

This section provides instructions for completing post-installation tasks.

- Route Anzo HTTP/S Ports to Non-Root Ports for User Access
- Change the Local Spark Engine Callback URL to the Non-Root Port
- Configure an Anzo Service Manager

Route Anzo HTTP/S Ports to Non-Root Ports for User Access

This section provides instructions for configuring the firewall to forward HTTP requests to port 8080 and HTTPS requests to port 8443 so that users can access Anzo without having to specify the new HTTP/S ports.

Note Root user privileges are required to complete this task.

To re-route Anzo ports using the iptables interface

Run the following commands to route the Anzo ports via the iptables interface:

```
# iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
# iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 8443
# iptables-save > /etc/sysconfig/iptables
```

To re-route Anzo ports using the firewalld interface

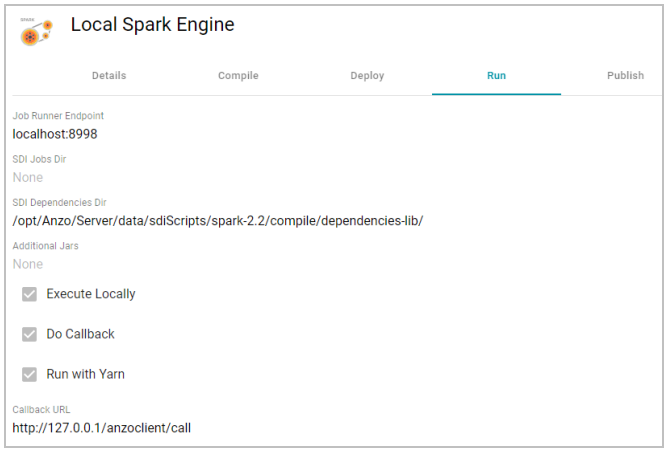
Run the following commands to route the Anzo ports via the firewalld interface:

```
# firewall-cmd --permanent --add-forward-port=port=443:proto=tcp:toport=8443
# firewall-cmd --permanent --add-forward-port=port=80:proto=tcp:toport=8080
# firewall-cmd --reload
```

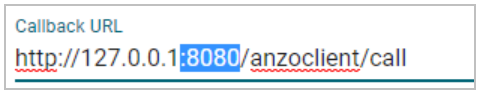
Change the Local Spark Engine Callback URL to the Non-Root Port

If you plan to use the pre-configured local Anzo Spark ETL engine to run pipelines, the callback URL for the engine must be configured to bind to the new Anzo HTTP port. Follow the instructions below to change the callback URL.

1. In the Anzo console, expand the **Administration** menu and click **ETL Engine Config**.
2. On the ETL Engine Config screen, click the **Local Spark Engine** to view the configuration details for the engine.
3. Click the **Run** tab. Anzo displays the Run screen. For example:



4. At the bottom of the screen, click the edit icon (✎) next to the **Callback URL** field (hover your pointer over the field to display the edit icon). Then edit the callback URL value to specify the HTTP port at the end of the IP address. For example:



5. Click the check mark icon (✓) to save the change.

Configure an Anzo Service Manager

Cambridge Semantics recommends that you configure an Anzo service for starting Anzo automatically as the service user. Follow the instructions below to implement the service.

Note Root user privileges are required to complete this task.

1. Create a file called **anzo-server.service** in the `/usr/lib/systemd/system` directory. For example:

```
# vi /usr/lib/systemd/system/anzo-server.service
```

2. Add the following contents to anzo-server.service:

```
[Unit]
Description=Service for Anzo server.
After=syslog.target network.target local-fs.target remote-fs.target nss-lookup.target
[Service]
Type=simple
RemainAfterExit=yes
```

```
ExecStart=/install_path/Server/AnzoServer start
ExecStop=/install_path/Server/AnzoServer stop
User=service_user_name
Group=service_user_name
[Install]
WantedBy=default.target
```

Where *install_path* is the installation path and directory and *service_user_name* is the name of the service user that you created in Make Sure the Anzo Service User Account is Created above. For example:

```
[Unit]
Description=Service for Anzo server.
After=syslog.target network.target local-fs.target remote-fs.target nss-lookup.target
[Service]
Type=simple
RemainAfterExit=yes
ExecStart/opt/Anzo/Server/AnzoServer start
ExecStop=/opt/Anzo/Server/AnzoServer stop
User=anzo
Group=anzo
[Install]
WantedBy=default.target
```

3. Save and close the file, and then run the following commands to start and enable the new service:

```
# systemctl start anzo-server.service
```

```
# systemctl enable anzo-server.service
```

The client displays a message such as the following:

```
Created symlink from /etc/systemd/system/default.target.wants/anzo-server.service to
/usr/lib/systemd/system/anzo-server.service.
```

Once the service is enabled, Anzo should be running. Any time you start and stop Anzo, run the following systemctl commands: `sudo systemctl stop anzo-server` and `sudo systemctl start anzo-server`.

See the Getting Started Guide for an introduction to the user interface, basic setup, and instructions for building a sample solution from scratch.

Related Topics

Upgrading Anzo

Installing the Anzo for Excel and Office Plugins

Upgrading Anzo

Before you upgrade Anzo, Cambridge Semantics recommends that you make a backup copy of the current Anzo installation in case you have issues and need to revert to the original version. There are three commonly used methods for backing up Anzo:

- Some users choose to make a copy of the Anzo system volume or journal, `install_path/Server-/data/journal/anzo.jnl`. If you keep a copy of anzo.jnl, you can restore the original Anzo version by reinstalling that release and then copying the backed up journal file into the installation.
- Some users choose to copy or create a tarball of the entire Anzo installation directory, `install_path/Anzo`. A backup of the directory can be large, however, and you might want to remove log files to reduce the overall size of the directory before copying or compressing it. If you keep a copy of `install_path/Anzo`, you can restore that version by uninstalling the new version and moving the backed up directory to the original installation location.
- Some users choose to take a snapshot of the application disk.

Follow the instructions below to upgrade Anzo.

Important Complete the steps below as the Anzo service user. When Anzo is initially installed, a server ID is generated based on a number of system properties, including the user account that runs the installation script. The Anzo server license is tied to that server ID. If Anzo is re-installed (for instance, during an upgrade) by a different user account, a new server ID is generated and the existing license will no longer be valid for the installation. For more information, see Licensing and User Account Best Practices.

1. Stop the existing Anzo server if it is running. Then copy the new Anzo installation script to the server and run the following command to make the script executable:

   ```
   chmod +x script_file_name
   ```

2. Run the following command to start the installation wizard and perform the upgrade:

   ```
   ./script_file_name
   ```

   The wizard unpacks the JRE and then waits for input before starting the upgrade.

3. Press **Enter** to start the upgrade. The wizard detects the existing installation and asks if you want to update it.
4. Press **Enter** to update the existing installation.
5. Review the software license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** and press **Enter** to accept the terms or type **2** and press **Enter** to disagree and stop the update.
6. Specify the components to install. Accept the default entry by pressing **Enter**. Or type **1** to install only the server components or **2** to install only the command line client, then press **Enter**.

7. Specify the maximum amount of memory (in MB) that the server can use and then press **Enter**. The wizard lists the amount of memory you have dedicated to the existing Anzo installation. You can type a different value if necessary, and then press **Enter**. The wizard starts the upgrade and then asks if you want to start the server automatically when the upgrade completes.
8. Press **Enter** to start Anzo when the upgrade completes. If you do not want to start the server, type **n** and then press **Enter**. The setup wizard completes the upgrade process.

Related Topics

Installing Anzo

Uninstalling Anzo

This topic provides instructions for uninstalling Anzo.

1. Run the following command to begin the uninstall process:

   ```
   ./install_path/Anzo/uninstall
   ```

2. Press **Enter** to confirm that you want to uninstall Anzo. The wizard asks if you want to clear the Anzo installation directory and user and configuration files.
3. Press **Enter** if you want the wizard to remove the entire Anzo installation directory as well as all configuration and user files. Type **n** and then press **Enter** if you do not want the wizard to remove the installation directory.

   The wizard uninstalls Anzo.

Installing AnzoGraph

The topics in this section provide instructions for installing AnzoGraph on a single server or cluster configuration.

- Installing AnzoGraph on a Single Server
- Installing AnzoGraph on a Cluster
- Upgrading AnzoGraph

Installing AnzoGraph on a Single Server

This topic provides instructions for installing AnzoGraph on a single server. For information about server requirements, see AnzoGraph Requirements.

Important Because AnzoGraph offers features such as user-defined extensions, it is not secure software certified and should not be installed or run as the root user. In addition, since AnzoGraph will access the data that Anzo writes on the shared file store, it is important to install and run AnzoGraph with the same service account that runs Anzo. For more information, see Anzo Service User Account Requirements.

For instructions on setting up an AnzoGraph cluster, see Installing AnzoGraph on a Cluster.

1. Complete the Pre-Installation Configuration
2. Install AnzoGraph
3. Complete the Post-Installation Configuration

Complete the Pre-Installation Configuration

- Install the Required Software
- Configure User Resource Limits

Install the Required Software

- Install GCC and BZIP2 (Required for all Deployments)
- Install OpenJDK 11 (Required for Anzo Unstructured and Data Toolkit Service Deployments)

Install GCC and BZIP2 (Required for all Deployments)

Make sure that the host server has the following software packages installed. These packages are required for all deployments:

- **GNU Compiler Collection (GCC)**: AnzoGraph requires the latest version of the GCC tools for your operating system. Run the following command to install GCC:

  ```
  sudo yum install gcc
  ```

  Note Specifically, AnzoGraph requires the **glibc**, **glibc-devel**, and **gcc-c++** libraries. Typically, when you install GCC by running `yum install gcc`, those libraries are included as part of the package. In rare cases, depending on the host server configuration, installing GCC excludes certain libraries. If AnzoGraph fails to start and you receive a "Compilation failed" message, it may indicate that some of the required libraries are missing. To install the missing libraries, run the following command:

  ```
  sudo yum install glibc glibc-devel gcc-c++
  ```

- **bzip2**: Required for unpacking the AnzoGraph tool set during installation. Run the following command to install bzip2:

  ```
  sudo yum install bzip2
  ```

Install OpenJDK 11 (Required for Anzo Unstructured and Data Toolkit Service Deployments)

AnzoGraph uses a Java client interface (datatoolkit-<version>.jar) to communicate with data sources when the Data Toolkit service is used to incorporate data from remote endpoints into graphmarts. AnzoGraph also uses the Java client to communicate with Elasticsearch when Anzo Unstructured graphmarts are deployed. Java Development Kit version 11 is required for using the Java client. Follow the instructions below to install OpenJDK.

1. Run the following command to install OpenJDK 11:

   ```
   sudo yum install java-11-openjdk
   ```

   Note Do not set the $JAVA_HOME variable to use the JDK installation at this time. AnzoGraph's system management daemon requires JAVA_HOME, and it is set as part of the post-installation configuration. In addition, the Elasticsearch plugin is deployed after AnzoGraph is installed.

2. If your organization uses Anzo Unstructured, test the connection between the AnzoGraph server and Elasticsearch. Make sure that Elasticsearch is running and then run the following telnet command:

   ```
   telnet <Elasticsearch_server_IP> <port>
   ```

   By default, the port range for Elasticsearch requests (http.port) is 9200-9300. If port 9200 is not available when Elasticsearch is started, Elasticsearch tries 9201 and so on until it finds an accessible port. Specify the HTTP request port that Elasticsearch is using.

Configure User Resource Limits

Cambridge Semantics recommends that you tune the user resource limits (ulimits) for your Linux distribution to increase the limits for the following resources:

- Increase the **open files** limit to at least **4096**.
- Increase the limit for the following resources to **unlimited**:
  - cpu time
  - file locks
  - file size
  - max memory size
  - max user processes
  - virtual memory

To view the current ulimits, run `ulimit -a`. To permanently change ulimits, modify the `/etc/security/limits.conf` file. For more information, see How to set ulimit values in the RHEL support documentation.

Note Also make sure that the Linux environment variables **http_proxy** and **https_proxy** are not set. The Anzo gRPC protocol cannot make connections to the database when proxies are enabled.

Install AnzoGraph

Follow the instructions below to install AnzoGraph.

Important Complete the steps below as the Anzo service user.

1. If necessary, run the following command to become the Anzo service user:

   ```
   # su name
   ```

   Where *name* is the name of the service user. For example:

   ```
   # su anzo
   ```

2. If necessary, run the following command to make the AnzoGraph installation script executable:

   ```
   chmod +x script_name
   ```

3. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

4. Press **Enter** to proceed with the installation. The wizard displays the AnzoGraph license agreement.
5. Review the license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** to accept the terms or type **2** to disagree and stop the installation.
6. The wizard prompts you to specify which components to install. Specify **1** (AnzoGraph) and press **Enter**.
7. Specify the path and directory for the AnzoGraph installation. Press **Enter** to accept the default installation path or type an alternate path and then press **Enter**.
8. At the server installation type prompt, accept the default option **1** (**Standalone**) and press **Enter**.
9. Indicate whether this installation is for use with Anzo. Press **Enter** for **Yes**. Answering yes configures AnzoGraph to use the settings that are optimal for Anzo. Answering no configures the settings that are optimal for AnzoGraph standalone use.
10. Set up the AnzoGraph admin user. Type a username to use for authentication. Anzo will use this username to connect to AnzoGraph. Then press **Enter**.
11. Type a password for the Anzo username and press **Enter**. **Note**: Some special characters, such as $ and *, are treated as parameters in bash. When typing a password, avoid or escape special characters to remove their special meaning to the command line. For more information, see Quoting in the Bash Reference Manual.
12. Configure any additional AnzoGraph settings. If Cambridge Semantics Support provided custom settings to use for your configuration, type the supplied values and then press **Enter**. Separate multiple settings with the new line escape sequence, **\n**. For example, the following entry sets two custom settings: `truncate_clob-b=true\npersistence_directory=/data/`.

    Note If you are installing AnzoGraph as the root user, add the following value to this prompt:

    ```
    enable_root_user=true
    ```

13. The wizard extracts the AnzoGraph files and completes the installation. Proceed to Installing AnzoGraph on a Single Server below to complete the initial configuration and start the database.

Complete the Post-Installation Configuration

- Deploy the Data Toolkit Plugin (Anzo Unstructured and Data Toolkit Service Environments)
- Configure and Start the AnzoGraph Services (All Environments)

Deploy the Data Toolkit Plugin (Anzo Unstructured and Data Toolkit Service Environments)

If your organization uses Anzo Unstructured or the Anzo Data Toolkit Service, copy the **datatoolkit-1.0.0.jar** plugin provided by Cambridge Semantics to the `<install_path>/lib/udx` directory on the AnzoGraph server.

Configure and Start the AnzoGraph Services (All Environments)

There are three processes involved in the initial AnzoGraph startup. And subsequent starts involve one or more of these steps depending on the state of AnzoGraph and the server:

1. The first process involves the configuration of the Linux kernel. The default kernel configuration for the following settings is not optimal for AnzoGraph:
   - **transparent_hugepage**: Transparent Huge Pages (THP) are enabled by default and can severely degrade AnzoGraph performance. THP should be disabled for AnzoGraph.
   - **max_map_count**: By default, the maximum number of memory map areas that a process can use is 65535. Since AnzoGraph is memory intensive, it may reach the maximum map count and be shut down by the operating system. AnzoGraph requires a value of **2097152**.

   At startup, AnzoGraph checks these settings and returns a warning if the values are not suitable. You are required to make the kernel changes or configure AnzoGraph to start with non-optimal configurations. The AnzoGraph deployment includes a script (`<install_path>/bin/azg_system_config`) that makes the required kernel configuration changes. Superuser privileges are required to make the changes, however, and each time the host server is rebooted the script must be run again because the kernel configuration reverts to the defaults.

   #### Show me what azg_system_config does

   The script runs the following commands to disable THP:

   ```
   echo never > /sys/kernel/mm/transparent_hugepage/enabled
   ```

   ```
   echo never > /sys/kernel/mm/transparent_hugepage/defrag
   ```

   The script runs this command to increase the max_map_count value:

   ```
   sysctl -w vm.max_map_count=2097152
   ```

2. The second process involves the AnzoGraph system management daemon, **azgmgrd**. This very lightweight program manages AnzoGraph communication. Though azgmgrd is especially important for managing connections between servers in a cluster, it is also required for single server installations. It must be running to start the database, but it typically does not need to be restarted unless you are upgrading AnzoGraph or the host server is rebooted. It does not need to be stopped and started each time the database is restarted.
3. The third process involves starting the database with the system manager.

To ensure that the right account/permissions are used to perform the three steps above (i.e., the root user makes the kernel changes and the Anzo service account starts the system management daemon and the database) whenever the host server is rebooted, Cambridge Semantics recommends that you configure services to run the startup steps. This section provides instructions for configuring the three services.

- Configure the Linux Kernel Configuration Service
- Configure the AnzoGraph System Management Service
- Configure the AnzoGraph Database Service

Note Root user privileges are required to complete the tasks below.

Configure the Linux Kernel Configuration Service

Follow the instructions below to set up a service to apply the Linux kernel configuration changes any time the AnzoGraph host server is restarted.

Note If making the kernel changes is not possible, you can set the `os_allow_alternate_vm_config` value to **true** in the AnzoGraph settings file. This setting enables AnzoGraph to start with non-optimal Linux configurations. See Changing AnzoGraph Configuration Settings for instructions.

1. Run the following command to copy the AnzoGraph system configuration script, azg_system_config, to the root directory:

   ```
   # cp /install_path/bin/azg_system_config /root/
   ```

   For example:

   ```
   # cp /opt/anzograph/bin/azg_system_config /root/
   ```

2. Run the following command to remove "sudo" from the azg_system_config script:

   ```
   # sed -i 's/sudo//g' /root/azg_system_config
   ```

3. Create a file called **azg_system_config.service** in the `/usr/lib/systemd/system` directory. For example:

   ```
   # vi /usr/lib/systemd/system/azg_system_config.service
   ```

4. Add the following contents to azg_system_config.service:

```
[Unit]
Description=Configure Linux for AnzoGraph
[Service]
Type=oneshot
ExecStart=/root/azg_system_config
[Install]
WantedBy=multi-user.target
```

5. Save and close the file.
6. Run the following commands to start and enable the new service:

```
# systemctl start azg_system_config.service
```

```
# systemctl enable azg_system_config.service
```

Configure the AnzoGraph System Management Service

Follow the instructions below to set up a service that starts the AnzoGraph system management daemon (azgmgrd) as the Anzo service user if the host server is restarted.

1. Create a file called **azgmgrd.service** in the `/usr/lib/systemd/system` directory. For example:

```
# vi /usr/lib/systemd/system/azgmgrd.service
```

2. Add the following contents to azgmgrd.service:

```
[Unit]
Description=AnzoGraph communication service
# depends on NetworkManager-wait-online.service enabled
Wants=network-online.target
After=network-online.target
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/azgmgrd.pid
WorkingDirectory=install_path
StandardOutput=syslog
StandardError=syslog
User=Anzo_service_user
UMask=0022
Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/install_path/bin:/install_
path/tools/bin
# Uncomment the following JAVA_HOME line for Anzo Unstructured and/or
```

```
# Data Toolkit Service environments
# ENVIRONMENT=JAVA_HOME=/usr/lib/jvm/jre-11
ExecStart=/install_path/bin/azgmgrd /install_path/
CPUAccounting=false
MemoryAccounting=false
[Install]
WantedBy=multi-user.target
```

For example:

```
[Unit]
Description=AnzoGraph communication service
# depends on NetworkManager-wait-online.service enabled
Wants=network-online.target
After=network-online.target
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/azgmgrd.pid
WorkingDirectory=/opt/anzograph
StandardOutput=syslog
StandardError=syslog
User=anzo
UMask=0022

Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/opt/anzograph/bin:/opt/anzog
raph/tools/bin
# Uncomment the following JAVA_HOME line for Anzo Unstructured and/or
# Data Toolkit Service environments
ENVIRONMENT=JAVA_HOME=/usr/lib/jvm/jre-11
ExecStart=/opt/anzograph/bin/azgmgrd /opt/anzograph/
CPUAccounting=false
MemoryAccounting=false
[Install]
WantedBy=multi-user.target
```

3. Save and close the file.
4. Run the following commands to start and enable the new service:

```
# systemctl start azgmgrd.service
```

```
# systemctl enable azgmgrd.service
```

Configure the AnzoGraph Database Service

Follow the instructions below to set up a service that will start AnzoGraph as the Anzo service user. This service is configured to run after the system management daemon is started.

1. Create a file called **anzograph.service** in the `/usr/lib/systemd/system` directory. For example:

```
# vi /usr/lib/systemd/system/anzograph.service
```

2. Add the following contents to anzograph.service:

```
[Unit]
Description=AnzoGraph database service
After=azgmgrd.service
Wants=azgmgrd.service
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/anzograph.pid
WorkingDirectory=install_path
StandardOutput=syslog
StandardError=syslog
RemainAfterExit=no
Restart=on-failure
RestartSec=60s
User=Anzo_service_user
UMask=0022
Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/install_path/bin:/install_
path/tools/bin
ExecStart=/install_path/bin/azgctl -start
[Install]
WantedBy=multi-user.target
```

For example:

```
[Unit]
Description=AnzoGraph database service
After=azgmgrd.service
Wants=azgmgrd.service
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/anzograph.pid
WorkingDirectory=/opt/anzograph
StandardOutput=syslog
```

```
StandardError=syslog
RemainAfterExit=no
Restart=on-failure
RestartSec=60s
User=anzo
UMask=0022

Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/opt/anzograph/bin:/opt/anzog
raph/tools/bin
ExecStart=/opt/anzograph/bin/azgctl -start
[Install]
WantedBy=multi-user.target
```

3. Save and close the file.
4. Run the following commands to start and enable the new service:

```
# systemctl start anzograph.service
```

```
# systemctl enable anzograph.service
```

Once the services are in place and enabled, AnzoGraph should be running. Any time you start and stop the database, run the following systemctl commands: `sudo systemctl stop anzograph` and `sudo systemctl start anzograph`. You do not need to stop and start azgmgrd.

For instructions on configuring the connection to AnzoGraph in the Anzo console, see Connecting to AnzoGraph.

Related Topics

Installing AnzoGraph on a Cluster

Upgrading AnzoGraph

Connecting to AnzoGraph

Installing AnzoGraph on a Cluster

This topic provides instructions for installing AnzoGraph on a cluster. For information about server requirements, see AnzoGraph Requirements.

Important Because AnzoGraph offers features such as user-defined extensions, it is not secure software certified and should not be installed or run as the root user. In addition, since AnzoGraph will access the data that Anzo writes on the shared file store, it is important to install and run AnzoGraph with the same service account that runs Anzo. For more information, see Anzo Service User Account Requirements.

For instructions on installing AnzoGraph on a single server, see Installing AnzoGraph on a Single Server.

1. Complete the Pre-Installation Configuration
2. Install AnzoGraph on the Compute Servers
3. Install AnzoGraph on the Leader Server
4. Complete the Post-Installation Configuration

Complete the Pre-Installation Configuration

- Install the Required Software
- Configure User Resource Limits

Install the Required Software

- Install GCC and BZIP2 (Required for all Deployments)
- Install OpenJDK 11 (Required for Anzo Unstructured and Data Toolkit Service Deployments)

Install GCC and BZIP2 (Required for all Deployments)

Make sure that each server in the cluster has the following software packages installed. These packages are required for all deployments:

- **GNU Compiler Collection (GCC)**: AnzoGraph requires the latest version of the GCC tools for your operating system. Run the following command to install GCC:

  ```
  sudo yum install gcc
  ```

  Note Specifically, AnzoGraph requires the **glibc**, **glibc-devel**, and **gcc-c++** libraries. Typically, when you install GCC by running `yum install gcc`, those libraries are included as part of the package. In rare cases, depending on the host server configuration, installing GCC excludes certain libraries. If AnzoGraph fails to start and you receive a "Compilation failed" message, it may indicate that some of the required libraries are missing. To install the missing libraries, run the following command:

  ```
  sudo yum install glibc glibc-devel gcc-c++
  ```

- **bzip2**: Required for unpacking the AnzoGraph tool set during installation. Run the following command to install bzip2:

  ```
  sudo yum install bzip2
  ```

Install OpenJDK 11 (Required for Anzo Unstructured and Data Toolkit Service Deployments)

AnzoGraph uses a Java client interface (datatoolkit-<version>.jar) to communicate with data sources when the Data Toolkit service is used to incorporate data from remote endpoints into graphmarts. AnzoGraph also uses the Java client to communicate with Elasticsearch when Anzo Unstructured graphmarts are deployed. Java Development Kit

version 11 is required for using the Java client. Follow the instructions below to install OpenJDK on each server in the cluster.

1. Run the following command to install OpenJDK 11:

   ```
   sudo yum install java-11-openjdk
   ```

   Note Do not set the $JAVA_HOME variable to use the JDK installation at this time. AnzoGraph's system management daemon requires JAVA_HOME, and it is set as part of the post-installation configuration. In addition, the Elasticsearch plugin is deployed after AnzoGraph is installed.

2. If your organization uses Anzo Unstructured, test the connection between the AnzoGraph leader server and Elasticsearch. Make sure that Elasticsearch is running and then run the following telnet command:

   ```
   telnet <Elasticsearch_server_IP> <port>
   ```

   By default, the port range for Elasticsearch requests (http.port) is 9200-9300. If port 9200 is not available when Elasticsearch is started, Elasticsearch tries 9201 and so on until it finds an accessible port. Specify the HTTP request port that Elasticsearch is using.

Configure User Resource Limits

Cambridge Semantics recommends that you tune the user resource limits (ulimits) for your Linux distribution to increase the limits for the following resources:

- Increase the **open files** limit to at least **4096**.
- Increase the limit for the following resources to **unlimited**:
  - cpu time
  - file locks
  - file size
  - max memory size
  - max user processes
  - virtual memory

To view the current ulimits, run `ulimit -a`. To permanently change ulimits, modify the `/etc/security/limits.conf` file. For more information, see How to set ulimit values in the RHEL support documentation.

Note Also make sure that the Linux environment variables **http_proxy** and **https_proxy** are not set. The Anzo gRPC protocol cannot make connections to the database when proxies are enabled.

Make note of the IP addresses for each of the servers in the cluster. The installation wizard will prompt you to enter the IP addresses during the installation. In addition, choose one server to be the leader server. Then proceed to the instructions below and install AnzoGraph on the compute servers.

Install AnzoGraph on the Compute Servers

Follow the instructions below to install AnzoGraph on each compute server.

Important Complete the steps below as the Anzo service user.

1. If necessary, run the following command to become the Anzo service user:

   ```
   # su name
   ```

   Where *name* is the name of the service user. For example:

   ```
   # su anzo
   ```

2. If necessary, run the following command to make the AnzoGraph installation script executable:

   ```
   chmod +x script_name
   ```

3. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

4. Press **Enter** to proceed with the installation. The wizard displays the AnzoGraph license agreement.
5. Review the license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** to accept the terms or type **2** to disagree and stop the installation.
6. The wizard prompts you to specify which components to install. Specify **1** (AnzoGraph) and press **Enter**.
7. Specify the path and directory for the AnzoGraph installation. Specify the same location on each server. Press **Enter** to accept the default installation path or type an alternate path and then press **Enter**.
8. At the server installation type prompt, specify option **3** (**Cluster Slave**) and press **Enter**.
9. Indicate whether this installation is for use with Anzo. Press **Enter** for **Yes**. Answering yes configures AnzoGraph to use the settings that are optimal for Anzo. Answering no configures the settings that are optimal for AnzoGraph standalone use.
10. Type a comma-separated list of the IP addresses for each server in the cluster. Type the leader server IP address first, followed by each compute IP address. For example, on a cluster with 4 servers where 192.168.2.1 is the leader server:

    ```
    192.168.2.1,192.168.2.2,192.168.2.3,192.168.2.4
    ```

    Important Make sure that you enter this value exactly the same, with IP addresses in the same order, during the installation on each server.

11. After typing the list of IP addresses, press **Enter**. The wizard extracts the AnzoGraph files and completes the installation.
12. Repeat the steps above to install AnzoGraph on each compute server. Then proceed to Install AnzoGraph on the Leader Server below.

Install AnzoGraph on the Leader Server

Follow the instructions below to install AnzoGraph on the leader server.

Important Complete the steps below as the Anzo service user.

1. If necessary, run the following command to become the Anzo service user:

   ```
   # su name
   ```

   Where *name* is the name of the service user. For example:

   ```
   # su anzo
   ```

2. If necessary, run the following command to make the AnzoGraph installation script executable:

   ```
   chmod +x script_name
   ```

3. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

4. Press **Enter** to proceed with the installation. The wizard displays the AnzoGraph license agreement.
5. Review the license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** to accept the terms or type **2** to disagree and stop the installation.
6. The wizard prompts you to specify which components to install. Specify **1** (AnzoGraph) and press **Enter**.
7. Specify the path and directory for the AnzoGraph installation. Specify the same location as the compute server installations. Press **Enter** to accept the default installation path or type an alternate path and then press **Enter**.
8. At the server installation type prompt, specify option **2** (**Cluster Leader**) and press **Enter**.
9. Indicate whether this installation is for use with Anzo. Press **Enter** for **Yes**. Answering yes configures AnzoGraph to use the settings that are optimal for Anzo. Answering no configures the settings that are optimal for AnzoGraph standalone use.
10. Set up the AnzoGraph admin user. Type a username to use for authentication. Anzo will use this username to connect to AnzoGraph. Then press **Enter**.

11. Type a password for the Anzo username and press **Enter**. **Note**: Some special characters, such as $ and *, are treated as parameters in bash. When typing a password, avoid or escape special characters to remove their special meaning to the command line. For more information, see Quoting in the Bash Reference Manual.
12. Type a comma-separated list of the IP addresses for each server in the cluster. Type the leader server IP address first, followed by each compute IP address. For example, on a cluster with 4 servers where 192.168.2.1 is the leader server:

    ```
    192.168.2.1,192.168.2.2,192.168.2.3,192.168.2.4
    ```

    Important Make sure that you enter this value exactly the same, with IP addresses in the same order, as the compute servers.

13. After typing the list of IP addresses, press **Enter**. Configure any additional AnzoGraph settings. If Cambridge Semantics Support provided custom settings to use for your configuration, type the supplied values and then press **Enter**. Separate multiple settings with the new line escape sequence, **\n**. For example, the following entry sets two custom settings: `truncate_clob=true\npersistence_directory=/data/`.

    Note If you are installing AnzoGraph as the root user, add the following value to this prompt:

    ```
    enable_root_user=true
    ```

14. The wizard extracts the AnzoGraph files and completes the installation. Proceed to Installing AnzoGraph on a Cluster below to complete the initial cluster configuration and start AnzoGraph.

Complete the Post-Installation Configuration

- Deploy the Data Toolkit Plugin (Anzo Unstructured and Data Toolkit Service Environments)
- Configure and Start the AnzoGraph Services (All Environments)

Deploy the Data Toolkit Plugin (Anzo Unstructured and Data Toolkit Service Environments)

If your organization uses Anzo Unstructured or the Data Toolkit Service, copy the **datatoolkit-1.0.0.jar** plugin provided by Cambridge Semantics to the `<install_path>/lib/udx` directory on the AnzoGraph leader server.

Configure and Start the AnzoGraph Services (All Environments)

There are three processes involved in the initial startup of AnzoGraph on a cluster. And subsequent starts involve one or more of these steps depending on the state of AnzoGraph and the servers:

1. The first process involves the configuration of the Linux kernel and it applies to all servers in the cluster. The default kernel configuration for the following settings is not optimal for AnzoGraph:
    - **transparent_hugepage**: Transparent Huge Pages (THP) are enabled by default and can severely degrade AnzoGraph performance. THP should be disabled for AnzoGraph.

- **max_map_count**: By default, the maximum number of memory map areas that a process can use is 65535. Since AnzoGraph is memory intensive, it may reach the maximum map count and be shut down by the operating system. AnzoGraph requires a value of **2097152**.

At startup, AnzoGraph checks these settings and returns a warning if the values are not suitable. You are required to make the kernel changes or configure AnzoGraph to start with non-optimal configurations. The AnzoGraph deployment includes a script (`<install_path>/bin/azg_system_config`) that makes the required kernel configuration changes. Superuser privileges are required to make the changes, however, and each time the host server is rebooted the script must be run again because the kernel configuration reverts to the defaults.

Show me what azg_system_config does

The script runs the following commands to disable THP:

```
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

```
echo never > /sys/kernel/mm/transparent_hugepage/defrag
```

The script runs this command to increase the max_map_count value:

```
sysctl -w vm.max_map_count=2097152
```

2. The second process involves the AnzoGraph system management daemon, **azgmgrd**. This very lightweight program runs on all servers in the cluster and manages AnzoGraph communication between the nodes. It must be running to start the database, but it typically does not need to be restarted unless you are upgrading AnzoGraph or the host servers are rebooted. It does not need to be stopped and started each time the database is restarted.
3. The third process involves starting the database with the system manager. Starting the database is done only on the leader server. The leader connects to the system managers on the compute servers and starts the database across the cluster.

To ensure that the right account/permissions are used to perform the three steps above (i.e., the root user makes the kernel changes and the Anzo service account starts the system management daemon and the database) whenever the host server is rebooted, Cambridge Semantics recommends that you configure services to run the startup steps. This section provides instructions for configuring the three services.

Important Configure the first two services, the Linux kernel configuration service and the AnzoGraph system management service, on all servers in the cluster. Configure the AnzoGraph database service only on the leader node.

- Configure the Linux Kernel Configuration Service
- Configure the AnzoGraph System Management Service
- Configure the AnzoGraph Database Service

Note Root user privileges are required to complete the tasks below.

Configure the Linux Kernel Configuration Service

On each server in the cluster, follow the instructions below to set up a service to apply the Linux kernel configuration changes any time the AnzoGraph host server is restarted.

Note If making the kernel changes is not possible, you can set the `os_allow_alternate_vm_config` value to **true** in the AnzoGraph settings file. This setting enables AnzoGraph to start with non-optimal Linux configurations. See Changing AnzoGraph Configuration Settings for instructions.

1. Run the following command to copy the AnzoGraph system configuration script, azg_system_config, to the root directory:

   ```
   # cp /install_path/bin/azg_system_config /root/
   ```

   For example:

   ```
   # cp /opt/anzograph/bin/azg_system_config /root/
   ```

2. Run the following command to remove "sudo" from the azg_system_config script:

   ```
   # sed -i 's/sudo//g' /root/azg_system_config
   ```

3. Create a file called **azg_system_config.service** in the `/usr/lib/systemd/system` directory. For example:

   ```
   # vi /usr/lib/systemd/system/azg_system_config.service
   ```

4. Add the following contents to azg_system_config.service:

   ```
   [Unit]
   Description=Configure Linux for AnzoGraph
   [Service]
   Type=oneshot
   ExecStart=/root/azg_system_config
   [Install]
   WantedBy=multi-user.target
   ```

5. Save and close the file.
6. Run the following commands to start and enable the new service:

   ```
   # systemctl start azg_system_config.service
   ```

```
# systemctl enable azg_system_config.service
```

7. Repeat this process on all of the compute servers and the leader server.

Configure the AnzoGraph System Management Service

On each server in the cluster, follow the instructions below to set up a service that starts the AnzoGraph system management daemon (azgmgrd) as the Anzo service user if the host server is restarted.

1. Create a file called **azgmgrd.service** in the `/usr/lib/systemd/system` directory. For example:

```
# vi /usr/lib/systemd/system/azgmgrd.service
```

2. Add the following contents to azgmgrd.service:

```
[Unit]
Description=AnzoGraph communication service
# depends on NetworkManager-wait-online.service enabled
Wants=network-online.target
After=network-online.target
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/azgmgrd.pid
WorkingDirectory=install_path
StandardOutput=syslog
StandardError=syslog
User=Anzo_service_user
UMask=0022
Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/install_path/bin:/install_
path/tools/bin
# Uncomment the following JAVA_HOME line for Anzo Unstructured and/or
# Data Toolkit Service environments
# ENVIRONMENT=JAVA_HOME=/usr/lib/jvm/jre-11
ExecStart=/install_path/bin/azgmgrd /install_path/
CPUAccounting=false
MemoryAccounting=false
[Install]
WantedBy=multi-user.target
```

For example:

```
[Unit]
Description=AnzoGraph communication service
# depends on NetworkManager-wait-online.service enabled
```

```
Wants=network-online.target
After=network-online.target
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/azgmgrd.pid
WorkingDirectory=/opt/anzograph
StandardOutput=syslog
StandardError=syslog
User=anzo
UMask=0022

Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/opt/anzograph/bin:/opt/anzog
raph/tools/bin
# Uncomment the following JAVA_HOME line for Anzo Unstructured and/or
# Data Toolkit Service environments
ENVIRONMENT=JAVA_HOME=/usr/lib/jvm/jre-11
ExecStart=/opt/anzograph/bin/azgmgrd /opt/anzograph/
CPUAccounting=false
MemoryAccounting=false
[Install]
WantedBy=multi-user.target
```

3. Save and close the file.
4. Run the following commands to start and enable the new service:

```
# systemctl start azgmgrd.service
```

```
# systemctl enable azgmgrd.service
```

5. Repeat this process on all of the compute servers and the leader server.

Configure the AnzoGraph Database Service

On the leader server only, follow the instructions below to set up a service that will start AnzoGraph as the Anzo service user. This service is configured to run after the system management daemon is started.

1. Create a file called **anzograph.service** in the `/usr/lib/systemd/system` directory. For example:

```
# vi /usr/lib/systemd/system/anzograph.service
```

2. Add the following contents to anzograph.service:

```
[Unit]
Description=AnzoGraph database service
After=azgmgrd.service
Wants=azgmgrd.service
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/anzograph.pid
WorkingDirectory=install_path
StandardOutput=syslog
StandardError=syslog
RemainAfterExit=no
Restart=on-failure
RestartSec=60s
User=Anzo_service_user
UMask=0022
Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/install_path/bin:/install_
path/tools/bin
ExecStart=/install_path/bin/azgctl -start
[Install]
WantedBy=multi-user.target
```

For example:

```
[Unit]
Description=AnzoGraph database service
After=azgmgrd.service
Wants=azgmgrd.service
[Service]
Type=forking
# The PID file is optional but recommended so that systemd
# can identify the main process of the daemon
# PIDFile=/var/run/anzograph.pid
WorkingDirectory=/opt/anzograph
StandardOutput=syslog
StandardError=syslog
RemainAfterExit=no
Restart=on-failure
RestartSec=60s
User=anzo
UMask=0022
```

```
Environment=PATH=/sbin:/bin:/usr/sbin:/usr/bin:/opt/anzograph/bin:/opt/anzog
raph/tools/bin
ExecStart=/opt/anzograph/bin/azgctl -start
[Install]
WantedBy=multi-user.target
```

3. Save and close the file.
4. Run the following commands to start and enable the new service:

```
# systemctl start anzograph.service
```

```
# systemctl enable anzograph.service
```

Once the services are in place and enabled, AnzoGraph should be running. Any time you start and stop the database, run the following systemctl commands on the leader node: `sudo systemctl stop anzograph` and `sudo systemctl start anzograph`. You do not need to stop and start azgmgrd.

For instructions on configuring the connection to AnzoGraph in the Anzo console, see Connecting to AnzoGraph.

Related Topics

Installing AnzoGraph on a Single Server

Upgrading AnzoGraph

Connecting to AnzoGraph

Upgrading AnzoGraph

Follow the appropriate instructions below to upgrade AnzoGraph on a single server or a cluster.

- Upgrading AnzoGraph on a Single Server
- Upgrading AnzoGraph on a Cluster

Upgrading AnzoGraph on a Single Server

1. If necessary, run the following command to become the Anzo service user:

```
# su name
```

   Where *name* is the name of the service user. For example:

```
# su anzo
```

2. If necessary, run the following command to make the AnzoGraph installation script executable:

```
chmod +x script_name
```

3. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

4. Press **Enter** to start the upgrade. The wizard detects the existing installation and asks if you want to update it.
5. Press **Enter** to update the existing installation.
6. The wizard prompts you to specify which components to upgrade. Specify **1** (AnzoGraph) and press **Enter**.
7. The wizard asks if you want to update the AnzoGraph configuration settings. Cambridge Semantics recommends that you choose to update the settings whenever possible so that your installation stays up-to-date with the latest default configuration setting values. Type **y** (**yes**) and press **Enter** to update the settings. Or press **Enter** to preserve the current settings.

   The installer saves a backup copy of the existing settings file before creating the new file. After the upgrade, you can revert AnzoGraph to a previous configuration any time by using a backup file. For more information, see Changing AnzoGraph Configuration Settings.

8. If you chose not to update settings, the wizard proceeds with the upgrade. When the upgrade completes, skip to the last step and start AnzoGraph. If you chose to update the settings, proceed to the next step.
9. At the server installation type prompt, accept the default option **1** (**Standalone**) and press **Enter**.
10. Indicate whether this installation is for use with Anzo. Press **Enter** for **Yes**. Answering yes configures AnzoGraph to use the settings that are optimal for Anzo. Answering no configures the settings that are optimal for AnzoGraph standalone use.
11. Set up the AnzoGraph admin user. Type a username to use for authentication. Anzo will use this username to connect to AnzoGraph. Then press **Enter**.
12. Type a password for the Anzo username and press **Enter**. **Note**: Some special characters, such as $ and *, are treated as parameters in bash. When typing a password, avoid or escape special characters to remove their special meaning to the command line. For more information, see Quoting in the Bash Reference Manual.
13. Configure any AnzoGraph settings. If Cambridge Semantics Support provided custom settings to use for your configuration, type the supplied values and then press **Enter**. Separate multiple settings with the new line escape sequence, **\n**. For example, the following entry sets two custom settings: `truncate_clob-b=true\npersistence_directory=/data/`.
14. The wizard extracts the AnzoGraph files and completes the upgrade.
15. Run the appropriate commands to start the system management daemon and database:
    - If services are set up as described in Installing AnzoGraph on a Cluster, run the following commands:

      ```
      sudo systemctl start azgmgrd
      ```

      ```
      sudo systemctl start anzograph
      ```

- If services are not set up, run the following commands:

```
/install_path/bin/azgmgrd
```

```
/install_path/bin/azgctl -start
```

Upgrading AnzoGraph on a Cluster

Follow these steps to upgrade AnzoGraph on a cluster. The steps guide you through upgrading the leader server using the installation script and then cloning the updated AnzoGraph files to the compute nodes.

1. Log on to the leader server. If necessary, run the following command to become the Anzo service user:

   ```
   # su name
   ```

   Where *name* is the name of the service user. For example:

   ```
   # su anzo
   ```

2. If necessary, run the following command to make the AnzoGraph installation script executable:

   ```
   chmod +x script_name
   ```

3. Run the appropriate commands to stop the database and the system manager daemons for the existing installation:
   - If services are set up as described in Installing AnzoGraph on a Cluster, run the following commands on the leader server to stop the database and daemon on all servers in the cluster:

     ```
     sudo systemctl stop anzograph
     ```

     ```
     sudo systemctl stop azgmgrd
     ```

   - If services are not set up, run the following commands on the leader server to stop the database and daemon on all servers in the cluster:

     ```
     /install_path/bin/azgctl -stop
     ```

     ```
     /install_path/bin/azgctl -stopdaemon
     ```

4. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

5. Press **Enter** to start the upgrade. The wizard detects the existing installation and asks if you want to update it.

6. Press **Enter** to update the existing installation.
7. The wizard prompts you to specify which components to upgrade. Specify **1** (AnzoGraph) and press **Enter**.
8. The wizard asks if you want to update the AnzoGraph configuration settings. Cambridge Semantics recommends that you choose to update the settings whenever possible so that your installation stays up-to-date with the latest default configuration setting values. Type **y** (**yes**) and press **Enter** to update the settings. Or press **Enter** to preserve the current settings.

   The installer saves a backup copy of the existing settings file before creating the new file. After the upgrade, you can revert AnzoGraph to a previous configuration any time by using a backup file. For more information, see Changing AnzoGraph Configuration Settings.
9. If you chose not to update settings, the wizard proceeds with the upgrade. When the upgrade completes, skip to the last step and start AnzoGraph. If you chose to update the settings, proceed to the next step.
10. At the server installation type prompt, accept the default option **2** (**Cluster Leader**) and press **Enter**.
11. Indicate whether this installation is for use with Anzo. Press **Enter** for **Yes**. Answering yes configures AnzoGraph to use the settings that are optimal for Anzo. Answering no configures the settings that are optimal for AnzoGraph standalone use.
12. Set up the AnzoGraph admin user. Type a username to use for authentication. Anzo will use this username to connect to AnzoGraph. Then press **Enter**.
13. Type a password for the Anzo username and press **Enter**. **Note**: Some special characters, such as $ and *, are treated as parameters in bash. When typing a password, avoid or escape special characters to remove their special meaning to the command line. For more information, see Quoting in the Bash Reference Manual.
14. Configure any AnzoGraph settings. If Cambridge Semantics Support provided custom settings to use for your configuration, type the supplied values and then press **Enter**. Separate multiple settings with the new line escape sequence, **\n**. For example, the following entry sets two custom settings: `truncate_clob-b=true\npersistence_directory=/data/`.
15. The wizard extracts the AnzoGraph files and completes the upgrade.
16. On the leader server and each of the compute servers, run the appropriate command to start the system management daemon:
    - If services are set up, run the following command:

      ```
      sudo systemctl start azgmgrd
      ```

    - If services are not set up, run the following command:

      ```
      /install_path/bin/azgmgrd
      ```

17. On the leader server, run the following command to clone the updated leader installation to the compute nodes:

    ```
    /install_path/bin/azgctl -clone
    ```

Depending on the number of nodes in the cluster, this step can take a few minutes.

18. When the clone process finishes and the prompt returns, run the appropriate command on the leader server to stop the system management daemon on all servers:
    - If services are set up, run the following command:

      ```
      sudo systemctl stop azgmgrd
      ```

    - If services are not set up, run the following command:

      ```
      /install_path/bin/azgctl -stopdaemon
      ```

19. On the leader server and each of the compute servers, restart the system management daemon:
    - If services are set up, run the following command:

      ```
      sudo systemctl start azgmgrd
      ```

    - If services are not set up, run the following command:

      ```
      /install_path/bin/azgmgrd
      ```

    Important Failing to restart azgmgrd on each server after the clone process can result in an incomplete upgrade.

20. On the leader server, run the appropriate command to start the database:
    - If services are set up, run the following command:

      ```
      sudo systemctl start anzograph
      ```

    - If services are not set up, run the following command:

      ```
      /install_path/bin/azgctl -start
      ```

Related Topics

Installing AnzoGraph on a Single Server

Installing AnzoGraph on a Cluster

Installing the Anzo for Excel and Office Plugins

After installing Anzo, you can access the installation packages for the Anzo for Excel and Anzo for Office plugins. Anzo for Office includes the data integration mapping tool which enables you to map relationships between schemas and models as well as apply various transformations to the source data. Anzo for Excel enables users to collect, combine, share, upload, and reuse the data in Microsoft Excel worksheets.

To access the installations that are included with your license, go to the following URL:

```
http://Anzo_server/installs
```

Where *Anzo_server* is the Anzo server DNS name or IP address. Follow the instructions onscreen to download and install the plugins.

Deploying the Anzo Unstructured Infrastructure

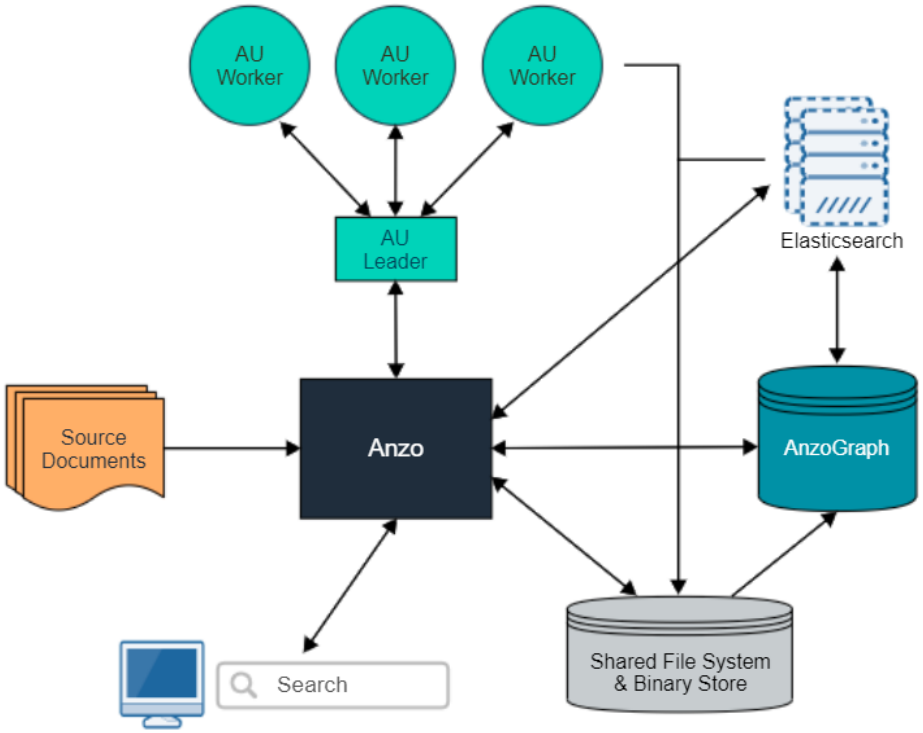
If your organization plans to onboard unstructured data to Anzo, additional infrastructure is required for running unstructured pipelines. The topics in this section provide an overview of the Anzo Unstructured (AU) infrastructure, information about the requirements and recommendations for deploying the architecture, and instructions for installing the software components.

Note The Anzo Unstructured architecture is not required if you do not work with unstructured data.

Anzo Unstructured Architecture and Process Overview

Anzo onboards unstructured data through pipelines that run in a distributed environment where a cluster of worker nodes process the incoming documents and generate output artifacts for Anzo. This topic provides an overview of the Anzo Unstructured (AU) pipeline process and infrastructure.

The diagram below provides a high level overview of the architecture. The description below the diagram describes the unstructured data onboarding process and resulting artifacts.



When an unstructured pipeline is run, an Anzo crawler service streams data to a pipeline service. The pipeline service reads the stream of files and constructs the appropriate request payloads—one request per document to process.

Anzo sends the requests to the AU leader instance, and the leader queues the requests and distributes them to the AU worker server instances to process in parallel. When each worker instance processes a document, it creates a temporary output artifact on the shared file system. The artifact includes:

- An RDF file that describes the text annotations and general metadata about the processed document.
- A binary store artifact for Anzo.
- A JSON artifact that contains a reference to the extracted text of the document. Elasticsearch uses this artifact to generate the document index.

When the AU workers have processed all of the documents, Anzo completes the following post-processing steps:

- Consolidate the RDF artifacts from the workers and create a file-based linked data set (FLDS) for loading to AnzoGraph.
- Read the JSON artifacts and instruct the Elasticsearch server to build an index with the text extracted from the documents. A snaphsot of the index is saved on the file system with the FLDS. Any time a graphmart that includes that FLDS is loaded to an AnzoGraph instance, Anzo loads the corresponding snapshot into the Elasticsearch server that is associated with the AnzoGraph connection.

When the post-processing is finished, the pipeline service finalizes the FLDS metadata to store in its catalog. The new unstructured data set becomes available in the Dataset catalog, and it can be added to a graphmart and loaded to AnzoGraph for use in Hi-Res Analytics dashboards.

Related Topics

Anzo Unstructured Requirements and Recommendations

Deploying an Anzo Unstructured Cluster

Installing and Configuring Elasticsearch

Anzo Unstructured Requirements and Recommendations

The Anzo Unstructured (AU) infrastructure is highly customizable and scalable. The number, size, and configuration of the servers in the environment depends on your unstructured data size, pipeline workload, and performance expectations. This topic provides guidance on determining the infrastructure to deploy as well as the requirements for each of the AU components. For an introduction to the AU architecture and pipeline process, see Anzo Unstructured Architecture and Process Overview.

AU requires two programs that are installed separately from Anzo:

- An Anzo Unstructured cluster for processing the incoming data. See Anzo Unstructured Cluster Requirements and Recommendations.
- Elasticsearch for indexing and searching unstructured document contents. See Elasticsearch Requirements and Recommendations.

Anzo Unstructured Cluster Requirements and Recommendations

An Anzo Unstructured (AU) cluster consists of one leader instance and one or more worker instances. Cambridge Semantics provides an installation script for installing the AU software. In an AU cluster:

- The **leader** instance is a lightweight program and is typically installed on the Anzo host server.
- The **worker** instances require significant resources to process the unstructured documents and are typically installed on dedicated servers.

Consider the size of your unstructured data workload when deploying worker host servers. Each worker instance can have multiple server instances to process documents. The table below lists the requirements for Anzo Unstructured worker servers:

Component	Requirement
Operating System	RHEL/CentOS 7+
CPU	4+ cores
RAM	16+ GB
Disk Space	10+ GB
File System	The Anzo file store (shared file system) must be accessible from each AU server in the cluster. For more information about file stores, see Connecting to a File Store.

For instructions on installing Anzo Unstructured, see Deploying an Anzo Unstructured Cluster.

Elasticsearch Requirements and Recommendations

Anzo Unstructured uses the Elasticsearch engine to build an index after an unstructured pipeline runs and for running searches on unstructured data that is onboarded to Anzo. When choosing an Elasticsearch host server, consider the following information:

- Generating the index is a lightweight operation compared to document search operations. If you have a light unstructured data workload and do not perform text searches on large amounts of data, installing an Elasticsearch engine on the Anzo host server might be sufficient.
- If you onboard a large number of unstructured documents and plan to perform text searches across a large amount of data, Cambridge Semantics recommends that you install Elasticsearch on a dedicated server.

The table below list the Elasticsearch server requirements:

Component	Requirement
Elasticsearch Version	7.1.1
CPU	8+ cores
RAM	64+ GB
Disk Space	100+ GB
Ports	By default, the port range for Elasticsearch requests (http.port) is **9200-9300**. If port 9200 is not available when Elasticsearch is started, Elasticsearch tries 9201 and so on until it finds an accessible port. The Anzo server and the AnzoGraph leader server need to be able to access Elasticsearch on the HTTP request port that Elasticsearch uses.
File System	The Anzo file store (shared file system) must be accessible from each Elasticsearch server. For more information about file stores, see Connecting to a File Store.

For instructions on installing Elasticsearch, see Installing and Configuring Elasticsearch.

Related Topics

Anzo Unstructured Architecture and Process Overview

Deploying an Anzo Unstructured Cluster

Installing and Configuring Elasticsearch

Deploying an Anzo Unstructured Cluster

This topic provides instructions for deploying an Anzo Unstructured (AU) cluster. See Anzo Unstructured Requirements and Recommendations for information about cluster requirements.

Important Since the Anzo Unstructured cluster will access the shared file store, it is important to install and run the software with the same service account that runs Anzo. For more information, see Anzo Service User Account Requirements.

1. Deploy the Leader Node
2. Deploy the Worker Nodes
3. Anzo Unstructured Cluster Configuration

AU Cluster Upgrade Notes

The steps to upgrade the Anzo Unstructured software are the same as the deployment instructions below. When you update the existing installation, each prompt defaults to the value that is specified for the current deployment. You can press **Enter** through the prompts to retain the existing settings. The last step in the process, however, asks if you want to overwrite files in the `<AnzoDU_install_dir>/etc` directory that have been modified. Cambridge Semantics recommends that you choose **ya (Yes To All)** to overwrite all files in that directory so that important options from the version you are upgrading to are deployed to your environment. If you have customized files in the etc directory, create a backup copy of the directory before starting the upgrade so that you can refer to the backup files when customizing the new version.

Deploy the Leader Node

Follow the instructions below to deploy the Anzo Unstructured leader node.

Important Complete the steps below as the Anzo service user.

1. Copy the Anzo DU installation script to the leader host server and then run the following command to make the script executable:

   ```
   chmod +x script_name
   ```

2. Run the following command to start the installation wizard:

   ```
   ./script_name
   ```

   The script unpacks the JRE and then waits for input before starting the installation.

3. Press **Enter** to start the installation.
4. Review the software license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** to accept the terms or type **2** to disagree and stop the installation.
5. At the prompt that asks which components to install, type **1** (Leader) and then press **Enter**.
6. Specify the directory to install Anzo DU. Press **Enter** to accept the default installation path or type an alternate path and then press **Enter**.
7. The wizard prompts for the IP address of this leader instance. The wizard defaults to the IP address of the server. Press **Enter** to accept the default value. If necessary, type a different IP address, and then press **Enter**.
8. The wizard prompts for any additional leader node IP addresses. Typically there is one leader node and this value is specified as the same IP address as the previous step. If you set up additional leader nodes for redundancy, however, enter a comma separated list of the alternate nodes. Otherwise, accept the default value and press **Enter**.
9. Specify the maximum amount of memory (in MB) that this leader instance can use. The install wizard lists the total RAM available and chooses 1/2 of the total memory as the default value. Adjust the value as needed or

accept the default value and then press **Enter**.

The wizard proceeds to install Anzo DU according to the values that you specified.

10. When the installation is complete, run the following command to start the leader instance:

```
./install_path/AnzoDU_root_dir/leader start
```

For example:

```
./opt/AnzoDU/leader start
```

Deploy the Worker Nodes

Follow the instructions below to deploy the Anzo Distributed Unstructured worker nodes.

Important Complete the steps below as the Anzo service user.

1. Make sure that the worker host servers have access to the Anzo shared file system and meet the requirements in Anzo Unstructured Cluster Requirements and Recommendations.
2. Copy the Anzo DU installation script to each of the worker host servers and then run the following command to make the script executable:

```
chmod +x script_name
```

3. Run the following command to start the installation wizard:

```
./script_name
```

The script unpacks the JRE and then waits for input before starting the installation.

4. Press **Enter** to start the installation.
5. Review the software license agreement. Press **Enter** to scroll through the terms. At the end of the agreement, type **1** to accept the terms or type **2** to disagree and stop the installation.
6. At the prompt that asks which components to install, type **2** (Worker) and then press **Enter**.
7. Specify the directory to install Anzo DU. Press **Enter** to accept the default installation path or type an alternate path and then press **Enter**.
8. The wizard prompts for the IP address to use for this worker node. The wizard defaults to the IP address of the server. Press **Enter** to accept the default value. If necessary, type a different IP address, and then press **Enter**.
9. The wizard prompts you to specify the maximum number of service instances for this worker node. Each service instance processes one unstructured document at a time. The default value is 2 instances. Press **Enter** to accept the default or specify another value and then press **Enter**.
10. Specify the port to use for this worker. The wizard defaults to port **2552**. Press **Enter** to accept the default value or type a different port and then press **Enter**.

11. The wizard prompts you to enter the IP address of the leader node. Specify the IP address for the leader instance that you deployed in the procedure above. If you deployed multiple leader nodes, specify each leader's IP address in a comma separated list.
12. Specify the maximum amount of memory (in MB) that this worker instance can use. The install wizard lists the total RAM available and chooses 1/2 of the total memory as the default value. Adjust the value as needed or accept the default value and then press **Enter**.

    The wizard proceeds to install Anzo DU according to the values that you specified.
13. When the installation is complete, run the following command to start the worker instance:

    ```
    ./install_path/AnzoDU_root_dir/worker start
    ```

    For example:

    ```
    ./opt/AnzoDU/worker start
    ```
14. Repeat the steps above for each worker instance in the cluster.

Anzo Unstructured Cluster Configuration

You do not need to perform additional configuration after the initial deployment of an Anzo Unstructured cluster. To review the configuration that Anzo creates based on the values specified during the installation, view the Distributed Pipeline options in **Server Settings** in the Administration menu. For more information, see Changing the Distributed Pipeline Configuration.

Important Any time the AU leader instance is restarted, the following two services must be restarted in Anzo:

- Anzo Server Akka Cluster Integration
- Anzo Unstructured Distributed

To restart a service:

1. Log in to the Anzo console, expand the Administration menu, and click **Advanced Configuration**.
2. On the Advanced Configuration screen, click the **I understand and accept the risk** link to view the Anzo bundles.
3. In the **Search** field at the top of the screen, start typing the name of the service that you want to restart. When the service appears in the list onscreen, click the service name to view the details.
4. At the top of the screen, click **Stop Bundle**. Then click **Start Bundle** when the start option becomes available.

Related Topics

Anzo Unstructured Architecture and Process Overview

Anzo Unstructured Requirements and Recommendations

Installing and Configuring Elasticsearch

Installing and Configuring Elasticsearch

This topic provides instructions for deploying Elasticsearch for use in the Anzo Unstructured environment.

Important Elasticsearch cannot be run as the root user and must have read and write access to the Anzo file store. Therefore, it is important to install and run Elasticsearch as the Anzo service user, otherwise unstructured pipelines will fail due to permissions errors. For more information, see Anzo Service User Account Requirements.

1. Make sure that the Elasticsearch host server has access to the Anzo shared file system and meets the requirements in Elasticsearch Requirements and Recommendations.
2. Become the Anzo user before proceeding. If necessary, create the user on the server. For more information, see Installing Anzo.
3. Download Elasticsearch version 7.1.1 from the Elasticsearch Past Releases website. Version 7.1.1 Docker images are also available from the Docker @ Elastic website. Follow the Elasticsearch documentation to install the software.
4. Configure Elasticsearch to save snapshots to the Anzo shared file system.
   - For a mounted file system, such as NFS, uncomment the Path setting, **path.repo**, in *elasticsearch_install_dir*`/config/elasticsearch.yml` and specify the path and directory for the mounted file system:

     ```
     path.repo: /path/directory
     ```

     For example:

     ```
     path.repo: /opt/anzoshare
     ```

   - For S3, see S3 Repository Plugin in the Elasticsearch documentation for information about installing the S3 repository plugin. Then see Client Settings for instructions on configuring the S3 client.
   - For HDFS, see Hadoop HDFS Repository Plugin in the Elasticsearch documentation for information about installing the HDFS repository plugin. Then see Hadoop Security for information about configuring Kerberos authentication.
5. Configure the amount of memory that Elasticsearch can use. By default, Elasticsearch is configured to use a maximum heap size of 1 GB. Cambridge Semantics recommends that you increase the amount to 50% of the memory that is available on the server. To change the configuration, open the *elasticsearch_install_dir*`/config/jvm.options` file in an editor. At the top of the file, modify the **Xms** and **Xmx** values to replace the **1** with the new value. For example:

   ```
   # Xms represents the initial size of total heap space
   # Xmx represents the maximum size of total heap space
   ```

```
-Xms15g
-Xmx15g
```

6. If you want to secure the Elasticsearch instance, follow the instructions in Configuring security in Elasticsearch in the Elasticsearch documentation.

   Important: If you set up SSL authentication with a trusted certificate, make sure that you add the certificate to the Anzo trust store. For instructions, see Uploading a Certificate to the Anzo Trust Store.

7. When the configuration is complete, run the following command to start Elasticsearch:

   ```
   ./elasticsearch_install_dir/bin/elasticsearch
   ```

   For more information about starting Elasticsearch, see Starting Elasticsearch in the Elasticsearch documentation. For information about configuring Elasticsearch to start automatically as the Anzo user, see Configuring an Elasticsearch Service below.

Once this Elasticsearch instance is configured and running, follow the instructions in Connecting to Elasticsearch to connect Anzo to this instance.

Configuring an Elasticsearch Service

Cambridge Semantics recommends that you configure an Elasticsearch service for starting Elasticsearch automatically as the Anzo service user. Follow the instructions below to implement the service.

Note: Root user privileges are required to complete this task.

1. Create a file called **es.service** in the `/usr/lib/systemd/system` directory. For example:

   ```
   # vi /usr/lib/systemd/system/es.service
   ```

2. Add the following contents to es.service:

   ```
   [Unit]
   Description=elasticsearch
   Wants=network-online.target
   After=network-online.target
   [Service]
   Type=oneshot
   ExecStart=/sbin/runuser -l Anzo_user /install_path/elasticsearch-
   7.1.1/bin/elasticsearch
   [Install]
   WantedBy=multi-user.target
   ```

   Where *install_path* is the path to elasticsearch and *Anzo_user* is the Anzo service user. For example:

```
[Unit]
Description=elasticsearch
Wants=network-online.target
After=network-online.target
[Service]
Type=oneshot
ExecStart=/sbin/runuser -l anzo /opt/elasticsearch-7.1.1/bin/elasticsearch
[Install]
WantedBy=multi-user.target
```

3. Save and close the file, and then run the following commands to start and enable the new service:

```
# systemctl enable es.service
```

```
# systemctl status es.service
```

```
# systemctl start es.service
```

Once the service is in place, Elasticsearch should be stopped and started via systemctl. For example, `systemctl stop es` and `systemctl start es`.

Related Topics

Anzo Unstructured Architecture and Process Overview

Anzo Unstructured Requirements and Recommendations

Deploying an Anzo Unstructured Cluster

Connecting to Elasticsearch

Setting Up Cloud Infrastructure for Dynamic Deployments

Anzo integrates with Amazon Web Services, Google Cloud, and Microsoft Azure services to offer dynamic, Kubernetes-based deployments of AnzoGraph, Spark, and Elasticsearch. The Kubernetes integration automates the scaling and management of the computing resources that support Anzo's data onboarding, modeling, blending, and access capabilities.

When dynamic deployments are enabled:

- Anzo users activate pre-configured Kubernetes environments on-demand without needing specific technical, cloud platform, or infrastructure provisioning skills.
- Right-sized clusters are automatically created and deleted in response to users' real-time requests, avoiding the need to keep instances running indefinitely and reducing the overall cost of maintaining the applications.
- Anzo provides a unified user interface for administrators to create and view all of the dynamic deployment options. Multiple cloud service providers can be managed through the same user interface.

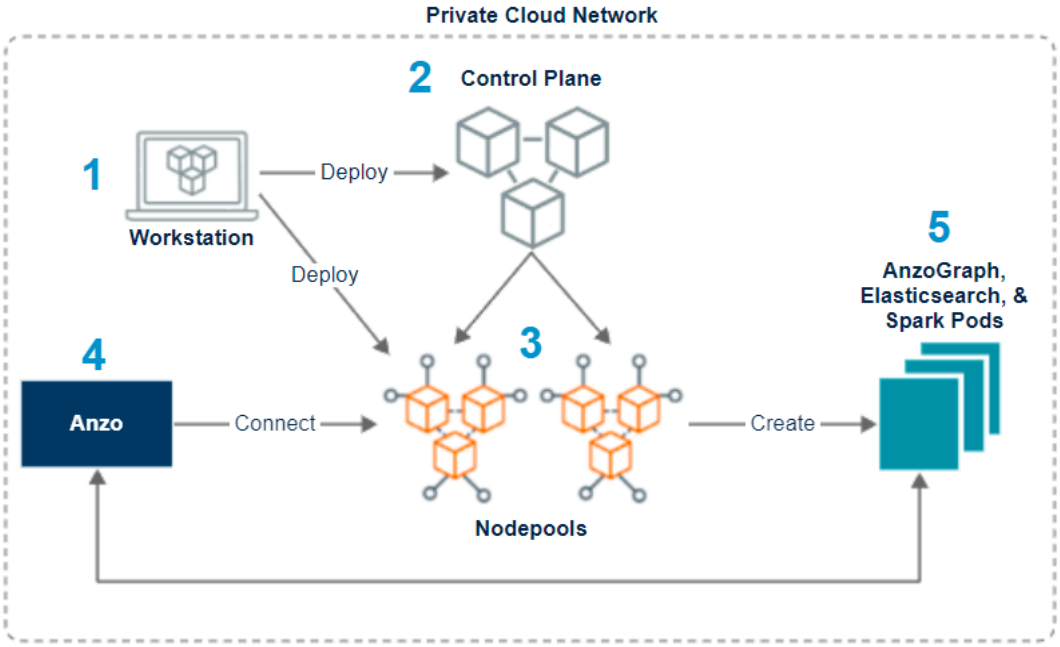
The topics in this section provide an overview of the cloud architecture and include instructions for provisioning a K8s cluster to enable dynamic deployments.

Dynamic Deployment Architecture and Process Overview

Anzo supports cloud-based dynamic deployments using Amazon Elastic Kubernetes Service (EKS), Google Kubernetes Engine (GKE), and Azure Kubernetes Service (AKS). These managed services enable enterprises to deploy Kubernetes (K8s) based applications on-demand without needing to maintain the K8s control plane. The K8s services interact with other services on the platform to provide

- User role management and authentication.
- Instance provisioning and deprovisioning.
- Isolated networking for deploying and maintaining clusters.
- A container registry for maintaining container images.
- Network load balancing for the hosted applications.

The process for setting up the infrastructure that will enable users to deploy applications on-demand with Anzo is the same on each of the platforms. Provision a K8s cluster on your preferred platform and then register the cluster in Anzo. Cambridge Semantics supplies scripts and configuration files that assist in creating clusters and make it straightforward to customize the K8s worker nodes for hosting a variety of applications and controllers to manage the applications. The diagram below shows a simplified view of the process and architecture.



1. First, you configure a workstation to use for creating and managing the K8s infrastructure. The workstation needs to have the required cloud provider software packages as well as the deployment scripts and configuration files supplied by Cambridge Semantics. This workstation will be used to connect to the K8s API endpoint and provision the K8s cluster and its nodepools.
2. Next, you deploy the K8s control plane, the master nodes that manage the cluster.
3. Then you create any number of nodegroups or nodepools in the K8s cluster. These are the worker instances that are configured according to the requirements of the applications that will be deployed, i.e., an AnzoGraph nodepool, an Elasticsearch nodepool, or a Spark nodepool. The nodepool configuration also includes any restrictions that you want to employ, such as the size and number of instances that can be used for the application and whether the instances remain deployed or are destroyed when the application is not in use.
4. Once the K8s infrastructure is in place, you configure a "Cloud Location" in Anzo. The K8s cluster and credentials are registered in Anzo so that Anzo can connect to the nodepools and the configured services become available for users to deploy on-demand.
5. Users deploy applications as needed through Anzo. Cambridge Semantics hosts all pods/containers in a public repository. You can choose to deploy pods from that repository or maintain an internal elastic container registry and deploy images from there.

Related Topics

Creating a K8s Cluster with Amazon EKS

Creating a K8s Cluster with Google Kubernetes Engine

Creating a K8s Cluster with Amazon EKS

This topic provides instructions for setting up a Kubernetes cluster with Amazon Elastic Kubernetes Service (EKS) using the AWS command line tools, **aws-cli** and **eksctl**.

1. Configure Your Workstation
2. Create AWS IAM Policies for EKS Permissions
3. Configure the VPC for the EKS Cluster
4. Create the EKS Cluster
5. Create Nodepools in the EKS Cluster

Configure Your Workstation

This section describes the requirements for configuring a workstation to use for creating and managing the EKS cluster. This workstation will be used to launch the EKS cluster and its nodepools and connect to the cluster API endpoint.

Component	Requirement
Operating System	The operating system for the workstation must be **CentOS 7.6 or higher**.
Networking	The workstation should be in the same VPC as the EKS cluster or on a network that is routable from the VPC. For information about the EKS cluster network requirements, see Configure the VPC for the EKS Cluster.
Software	See Required Software Packages below.
CSI-Supplied EKS Scripts	Cambridge Semantics provides scripts and configuration files to use for provisioning the EKS cluster and its nodepools. Download the files to the workstation. Details about the files are provided in EKS Cluster Creation Scripts and Configuration Files below.

Required Software Packages

Make sure that the workstation has the following software packages installed. In order to download the packages, the workstation needs outbound internet connectivity.

python-pip

PIP is a package manager for Python packages. Run the following commands to install PIP:

```
sudo yum install epel-release
sudo yum install python-pip
```

aws-cli

AWS CLI is the AWS command line interface. Version 1.16.156 or higher is required. Run the following command to install the aws-cli package:

```
pip install awscli --upgrade --user
```

Use the `aws configure` command to set the region and AWS secret and access keys for the IAM user who is responsible for EKS cluster deployment and management.

eksctl

EKSCTL is the AWS EKS command line interface. Version 0.3.1 or higher is required. Run the following commands to download and unpack the eksctl package and place it in the `/usr/local/bin` directory:

```
curl --silent --location
"https://github.com/weaveworks/eksctl/releases/download/latest_release/eksctl_
$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
sudo mv /tmp/eksctl /usr/local/bin
```

kubectl

KUBECTL is the Kubernetes command line interface. The workstation must either use the same kubectl version as the version on the EKS cluster or a version that is within one minor version. For example, a v1.2 client should work with v1.1, v1.2, and v1.3. Run the following commands to configure access to the Kubernetes repository and install kubectl:

```
sudo bash -c "cat >/etc/yum.repos.d/kubernetes.repo" << 'EOF'
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF
```

```
sudo yum install -y kubectl-version
```

For example:

```
sudo yum install -y kubectl-1.13.0
```

EKS Cluster Creation Scripts and Configuration Files

Cambridge Semantics provides a package of files that enable users to manage the configuration, creation, and deletion of EKS clusters and nodepools. The files and their directory structure are shown below.

```
├── aws_cli_common.sh
├── common.sh
├── conf.d
│   ├── ca_autodiscover.yaml
│   ├── k8s_cluster.conf
│   ├── nodepool_anzograph.yaml
│   ├── nodepool_common.yaml
│   ├── nodepool_dynamic.yaml
│   ├── nodepool_operator.yaml
│   ├── nodepool.yaml
│   └── warm_ip_target.yaml
├── create_k8s.sh
├── create_nodepools.sh
├── delete_k8s.sh
├── delete_nodepools.sh
└── README.md
```

- The **create_k8s.sh** and **create_nodepools.sh** scripts are used to provision the EKS cluster and any number of nodepools in the cluster. And the **delete_k8s.sh** and **delete_nodepools.sh** scripts are used to remove the cluster or nodepools.
- The configuration file, **k8s_cluster.conf** in the **conf.d** directory, is used with the create_k8s and delete_k8s scripts and supplies the cluster-wide configuration parameters such as network, region, and timeout values.
- The configuration files, **nodepool*.yaml** in the **conf.d** directory, contain nodepool settings. The files include sample configuration values for different types of nodepools, such as for Spark, Elasticsearch, or AnzoGraph. Each file is used to create one type of nodepool.
- The **aws-cli-common.sh** and **common.sh** scripts are used by the create and delete cluster and nodepool scripts.

The instructions for Create the EKS Cluster and Create Nodepools in the EKS Cluster provide more details about the scripts and configuration files.

Create AWS IAM Policies for EKS Permissions

Two AWS Identity and Access Management (IAM) policies need to be created to grant the appropriate permissions to AWS users who will set up and manage the EKS cluster and/or deploy applications using the K8s service:

- **EKS Cluster Admin**: The cluster administrator is responsible for EKS cluster management tasks such as creating and deleting clusters and nodepools and adding and removing containers in the Elastic Container Registry (ECR).
- **EKS Cluster User**: The cluster user uses the EKS cluster that the admin creates. This user manages namespaces and creates K8s objects like pods and services. This user needs read access to the ECR for

deploying applications on K8s clusters. This user should also be added to aws-auth configmap of the K8s cluster.

The policies to add to AWS are provided below. For information about creating policies, see Creating IAM Policies in the AWS documentation.

- EKS Cluster Admin Policy
- EKS Cluster User Policy

EKS Cluster Admin Policy

The following IAM policy applies the minimum permissions needed for a cluster administrator who will create and manage the EKS cluster.

```
{
        "Version": "2012-10-17",
        "Statement": [
                {
                  "Sid": "IAMPermissions",
                  "Effect": "Allow",
                  "Action": [
                        "iam:GetInstanceProfile",
                        "iam:CreateInstanceProfile",
                        "iam:AddRoleToInstanceProfile",
                        "iam:RemoveRoleFromInstanceProfile",
                        "iam:DeleteInstanceProfile",
                        "iam:GetRole",
                        "iam:CreateRole",
                        "iam:TagRole",
                        "iam:PassRole",
                        "iam:GetRolePolicy",
                        "iam:AttachRolePolicy",
                        "iam:PutRolePolicy",
                        "iam:DetachRolePolicy",
                        "iam:DeleteRolePolicy",
                        "iam:UntagRole",
                        "iam:DeleteRole"
                  ],
                  "Resource": "*"
                },
                {
                  "Sid": "ComputeAndEKS",
                  "Effect": "Allow",
                  "Action": [
                        "autoscaling:*",
                        "cloudformation:*",
```

```
                        "elasticloadbalancing:*",
                        "ec2:*",
                        "eks:*"
                    ],
                    "Resource": "*"
                },
                {
                    "Sid": "ECRPushPull",
                    "Effect": "Allow",
                    "Action": [
                        "ecr:CompleteLayerUpload",
                        "ecr:DescribeImages",
                        "ecr:GetAuthorizationToken",
                        "ecr:DescribeRepositories",
                        "ecr:UploadLayerPart",
                        "ecr:InitiateLayerUpload",
                        "ecr:BatchCheckLayerAvailability",
                        "ecr:PutImage"
                    ],
                    "Resource": "*"
                }
        ]
}
```

EKS Cluster User Policy

The following IAM policy applies the minimum permissions needed for an EKS cluster user.

```
{
        "Version": "2012-10-17",
        "Statement": [
                {
                    "Sid": "Compute",
                    "Effect": "Allow",
                    "Action": [
                        "ec2:*",
                        "elasticloadbalancing:*",
                        "autoscaling:*"
                    ],
                    "Resource": "*"
                },
                {
                    "Sid": "Pricing",
                    "Effect": "Allow",
                    "Action": [
                        "pricing:GetProducts"
```

```
                ],
                "Resource": "*"
            },
            {
                "Sid": "EKSListAndDescribe",
                "Effect": "Allow",
                "Action": [
                      "eks:ListUpdates",
                      "eks:DescribeCluster",
                      "eks:ListClusters"
                ],
                "Resource": "arn:aws:eks:*:*:cluster/*"
            },
            {
                "Sid": "ECRPull",
                "Effect": "Allow",
                "Action": [
                      "ecr:GetDownloadUrlForLayer",
                      "ecr:GetAuthorizationToken",
                      "ecr:BatchGetImage",
                      "ecr:BatchCheckLayerAvailability"
                ],
                "Resource": "*"
            }
        ]
}
```

Configure the VPC for the EKS Cluster

The EKS cluster must be deployed in a non-default VPC that meets the following requirements.

Note You can choose to let the AWS EKS service create the VPC when the cluster is provisioned. However, if Anzo is currently deployed, Cambridge Semantics recommends that you provision the EKS cluster in the same VPC as Anzo.

- The VPC needs pairs of public and private subnets in at least two availability zones. The two availability zones are required by the AWS EKS service to maintain high availability.

  The subnets should specify CIDRs that can contain a large number of IP addresses. For example, a subnet mask of /19 is ideal for private subnets for pod IP addresses, and a subnet mask of /21 is ideal for public subnets that host elastic load balancers.

- Nodes in the nodegroup need outbound internet connectivity for registering to EKS for the first time.
- The DNS hostname and DNS resolution properties must be enabled.
- Tag the private subnet with the following key and value so that K8s knows the subnet can be used for internal load balancers: `kubernetes.io/role/internal-elb: 1`

- Tag the public subnet with the following key and value so that K8s knows the subnet can be used for external load balancers: `kubernetes.io/role/elb: 1`

For more information, see Creating a VPC for Your Amazon EKS Cluster in the AWS documentation.

Create the EKS Cluster

Follow the steps below to configure and then provision the K8s control plane that will manage the nodepools that will be used with Anzo to deploy K8s applications.

1. Configure the Cluster
2. Create the Cluster

Configure the Cluster

Before creating the EKS cluster, specify the networking and control plane configuration to use by creating a configuration file based on the sample **k8s_cluster.conf** file in the **conf.d** directory. Follow the instructions below to configure the cluster.

1. In the **conf.d** directory, copy **k8s_cluster.conf** to create a file to customize. You can specify any name for the .conf file.

   Note: If you want to create multiple EKS clusters, you can create multiple .conf files. When you create a cluster (using create_k8s.sh), you specify the name of the .conf file to use for that cluster.

2. Open the new .conf file for editing. The file includes the list of parameters and sample values. Modify the values as needed. The table below describes each setting.

   View the k8s_cluster.conf sample file

Parameter	Description	Sample Value
REGION	The AWS region for the EKS cluster.	us-east-1
AvailabilityZones	List of Availability Zones. Separate zone names with a space.	us-east-1a us-east-1b
TAGS	A comma-separated list of tags to add to the EKS cluster.	Owner=User1,Description=EKS Cluster desc

Parameter	Description	Sample Value
VPC_ID	The ID of the VPC to provision the cluster into. This is the VPC described in Configure the VPC for the EKS Cluster. If you want EKSCTL to create the VPC, leave this value blank.	Not set
VPC_CIDR	The CIDR to use for the VPC. Provide this value even if VPC_ID is not set.	10.107.0.0/16
NAT_SUBNET_CIDR	The CIDR for the public subnet to be used by the NAT gateway.	10.107.0.0/24
PUBLIC_SUBNET_ CIDRS	CIDRs for the public subnets. Separate multiple CIDRs with a space.	10.107.1.0/24 10.107.2.0/24
PRIVATE_SUBNET_ CIDRS	CIDRs for the private subnets. Separate multiple CIDRs with a space.	10.107.3.0/24 10.107.4.0/24
VPC_NAT_MODE	The NAT mode for the VPC. Valid values are "HighlyAvailable," "Single," or "Disable."	Single
WARM_IP_TARGET	Specifies the number of free IP addresses to keep available for pod assignment on the node.	8
CUSTOMER_CIDRS	Comma-separated list of the CIDRs that can access the K8s API over port 443.	10.107.0.0/16 10.108.0.0/16
CLUSTER_NAME	Name for the EKS cluster.	testcluster
CLUSTER_VERSION	Version of the EKS cluster.	1.13

Parameter	Description	Sample Value
ENABLE_PRIVATE_ ACCESS	Whether to enable private (VPC-only) access to EKS cluster endpoint.	true
ENABLE_PUBLIC_ ACCESS	Whether to enable public access to EKS cluster endpoint.	false
CNI_VERSION	Version of the VPC CNI plugin.	1.5
LOGGING_TYPES	Comma-separated list of the logging types to enable for the cluster. Valid values are "api," "audit," "authenticator," "controllerManager," and "scheduler."	api,audit
WAIT_DURATION	The number of seconds to wait before timing out during AWS resource creation.	1200
WAIT_INTERVAL	The number of seconds to wait before polling for resource state information.	10
STACK_CREATION_ TIMEOUT	Time in minutes to wait for EKS cluster state changes before timing out. For example the time to wait for creation or update to complete.	30m
EKSCTL_VERSION	The version of eksctl on the workstation.	0.3.0
AWS_CLI_VERSION	The version of the aws-cli on the workstation.	1.16.156

3. Save and close the configuration file, and then proceed to Create the Cluster.

Create the Cluster

To create an EKS cluster that is configured according to the file you created in Configure the Cluster, run the **create_ k8s.sh** script with the following command:

```
$ ./create_k8s.sh cluster_config_filename [ -f ] [ -h ]
```

Argument	Description
cluster_config_filename	**Required** name and extension of the configuration file that supplies the parameter values for this cluster. You do not need to specify the path to the conf.d directory.
-f	**Optional** flag. If specified, the script will prompt for confirmation before proceeding with each stage involved in creating the cluster and its components.
-h	**Optional** flag. If specified, the help will be displayed.

For example:

```
$ ./create_k8s.sh k8s_cluster.conf
```

This script creates an EKS cluster along with the networking specifications described by the .conf file. The script deploys the infrastructure, such as a new VPC if no VPC_ID was specified and pairs of public and private subnets in multiple availability zones, using the CIDRs specified in the .conf file. It also creates an Internet and NAT gateway and route tables.

Note The user who creates the cluster will be registered as a **system:master** for the cluster. For information about adding other IAM users, see Managing Users or IAM Roles for your Cluster in the Amazon EKS documentation.

When cluster creation is complete, proceed to Create Nodepools in the EKS Cluster to add one or more nodepools to the cluster.

Create Nodepools in the EKS Cluster

Follow the steps below to configure and then provision the nodepools in the EKS cluster that will be used to deploy K8s applications with Anzo.

1. Configure a Nodepool
2. Create a Nodepool

Configure a Nodepool

Before creating a nodepool, supply the configuration values to use for that nodepool based on a sample **nodepool*.yaml** file in the **conf.d** directory. Follow the instructions below to configure a nodepool.

1. Depending on the application that you want to deploy with this nodepool, determine the configuration file to copy and customize from the **nodepool*.yaml** sample files in the **conf.d** directory:

- **nodepool_anzograph.yaml**: The sample values in this file configure an autoscaling group for an AnzoGraph nodepool. The specified EC2 instance type offers high CPU and memory resources, and the file includes Linux kernel configuration settings that are ideal for AnzoGraph.
- **nodepool_common.yaml**: The sample values in this file configure a common nodepool, with a relatively small EC2 instance type that can be used for smaller workloads. This type of configuration is typically used to deploy K8s services that manage the EC2 instances that connect to the cluster.
- **nodepool_dynamic.yaml**: The sample values in this file configure an autoscaling group for a dynamic nodepool, where the EC2 instance type offers medium to large CPU and memory resources that are reasonable for deploying Elasticsearch and Spark applications. The file also includes system tuning bootstrap values that are required by Elasticsearch and/or Spark.
- **nodepool_operator.yaml**: The sample values in this file configure an autoscaling group for an operator nodepool that can be used to deploy controllers for managing applications deployed on K8s clusters.
- **nodepool.yaml**:
- **warm_ip_target.yaml**:

2. Copy the appropriate configuration file to create the version to customize. You can specify any name for the .yaml file.
3. Open the new .yaml file for editing. The file includes the list of parameters and sample values. Modify the values as needed. The table below describes the common settings.

View a nodepool.yaml sample file

Parameter	Description	Sample Value
apiVersion	The eksctl API version used to deploy EKS objects.	eksctl.io/v1alpha5
kind	The type of object to be created.	ClusterConfig
metadata parameters		
name	The name of the EKS cluster.	cluster01
region	The region to deploy the EKS cluster in.	us-east-1

Parameter	Description	Sample Value
nodeGroups parameters		
name	The prefix to give to the names of the worker nodes that will be deployed.	anzograph
labels	Labels to attach to the nodes in the nodepool. Once deployed, labels can be viewed by running `kubectl get node <node_name> -o yaml`	{'csi.com/node-purpose': 'anzograph'}
instanceType	The EC2 instance type to use when deploying the nodes.	m5.large
desiredCapacity	The number of Amazon EC2 instances that the auto-scaling group should maintain.	0
availabilityZones	The list of availability zones to use when deploying auto scaling group instances.	us-east-1a
minSize	The minimum number of instances that can exist in the auto scaling group.	0
maxSize	The maximum number of instances that can exist in the auto scaling group.	5

Parameter	Description	Sample Value
volumeSize	The storage size (in GB) to use for Amazon EBS volumes.	50
maxPodsPerNode	The maximum number of pods that can be hosted on each EC2 instance.	8
iam:withAddonPolicies: autoScaler	Indicates whether this nodepool should be part of the cluster auto-scaler.	true
iam:withAddonPolicies: imageBuilder	Indicates whether to allow this nodepool to access the full Elastic Container Registry (ECR).	true
volumeType	The type of Amazon EBS volume to use.	gp2
privateNetworking	Indicates whether to isolate the nodepool from the public internet.	true
securityGroups: withShared	Indicates whether to create a shared security group for this nodepool to allow communication with other nodepools.	true
securityGroups: withLocal	Indicates whether to create a separate security group for this nodepool.	true

Parameter	Description	Sample Value
ssh: allow	Indicates whether to allow SSH access for the EC2 instances in this nodepool.	true
ssh: publicKeyName	The key pair name to use for SSH access.	
ssh: publicKey	The public key string for the specified publicKeyName.	
taints	Indicates the schedule setting for the nodes.	'cambridgesemantics.com/dedicated': 'anzograph:NoSchedule'
tags	A list of tags to attach to each EC2/K8s instance.	
preBootstrapCommands	A list of commands to be run on the EC2 instance during the deployment.	

4. Save and close the configuration file, and then proceed to Create a Nodepool. Repeat the steps above to configure additional nodepools.

Create a Nodepool

To create a nodepool that is configured according to a file you created in Configure a Nodepool, run the **create_nodepools.sh** script with the following command:

```
$ ./create_nodepools.sh nodepool_config_filename [ -f ] [ -h ]
```

Argument	Description
nodepool_config_ filename	**Required** name and extension of the configuration file that supplies the parameter values for this nodepool. You do not need to specify the path to the conf.d directory.
-f	**Optional** flag. If specified, the script will prompt for confirmation before proceeding with each stage involved in creating the nodepool and its components.

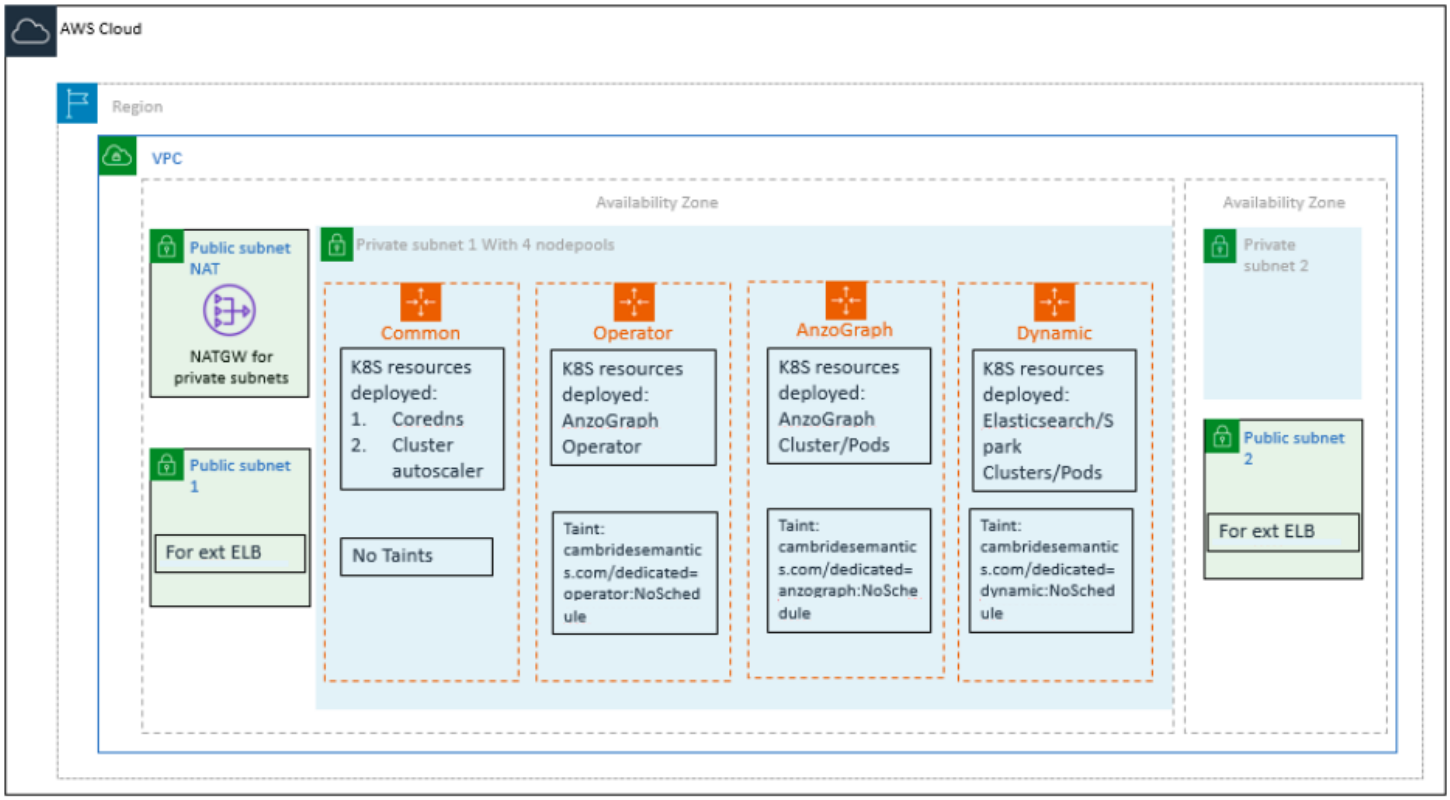
Argument	Description
-h	**Optional** flag. If specified, the help will be displayed.

For example:

```
$ ./create_nodepools.sh anzograph_nodepool.yaml
```

This script creates the nodes in the nodepool according to the specifications provided in the .yaml file. Repeat the process to create additional nodepools.

The diagram below shows the architecture for a sample deployment with four nodepools.



The next step is to create a Cloud Location in Anzo so that Anzo can connect to the EKS cluster and deploy applications. See Managing Cloud Locations.

Related Topics

Dynamic Deployment Architecture and Process Overview

Managing Cloud Locations

Creating a K8s Cluster with Google Kubernetes Engine

This topic provides instructions for setting up a Kubernetes cluster with Google Kubernetes Engine (GKE) using the Google command line tool, **gcloud**.

1. Configure Your Workstation
2. Create IAM Roles for GKE Permissions
3. Configure the Networking for the GKE Cluster
4. Create the GKE Cluster
5. Create Nodepools in the GKE Cluster

Configure Your Workstation

This section describes the requirements for configuring a workstation to use for creating and managing the GKE cluster. This workstation will be used to launch the GKE cluster and its nodepools and connect to the cluster API endpoint.

Component	Requirement
Operating System	The operating system for the workstation must be **CentOS 7.6 or higher**.
Networking	The workstation should be in the same VPC network as the GKE cluster or on a network that is routable from the VPC. For information the GKE cluster network, see Configure the Networking for the GKE Cluster.
Software	• **Python 2.7.9 or higher** is required. • **Google Cloud SDK** is required. For installation instructions, see Installing Google Cloud SDK below.
CSI-Supplied GKE Scripts	Cambridge Semantics provides scripts and configuration files to use for provisioning the GKE cluster and its nodepools. Download the files to the workstation. Details about the files are provided in GKE Cluster Creation Scripts and Configuration Files below.

Installing Google Cloud SDK

Follow the instructions below to install Google Cloud SDK on your workstation.

1. Run the following command to configure access to the Google Cloud repository:

```
sudo tee -a /etc/yum.repos.d/google-cloud-sdk.repo << EOM
[google-cloud-sdk]
name=Google Cloud SDK
baseurl=https://packages.cloud.google.com/yum/repos/cloud-sdk-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
```

```
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOM
```

2. Run the following command to install Cloud SDK:

   ```
   sudo yum install kubectl google-cloud-sdk
   ```

   The following packages are installed:

   ```
   google-cloud-sdk-app-engine-grpc
   google-cloud-sdk-pubsub-emulator
   google-cloud-sdk-app-engine-go
   google-cloud-sdk-cloud-build-local
   google-cloud-sdk-datastore-emulator
   google-cloud-sdk-app-engine-python
   google-cloud-sdk-cbt
   google-cloud-sdk-bigtable-emulator
   google-cloud-sdk-datalab
   google-cloud-sdk-app-engine-java
   ```

3. Next, configure the default project and region settings for the Cloud SDK:
   a. Run the following command to set the default project for the GKE cluster:

      ```
      gcloud config set project project_ID
      ```

      Where *project_ID* is the Project ID for the project in which the GKE cluster will be provisioned.

   b. If you work with zonal clusters, run the following command to set the default compute zone for the GKE cluster:

      ```
      gcloud config set compute/zone compute_zone
      ```

      Where *compute_zone* is the default compute zone for the GKE cluster. For example:

      ```
      gcloud config set compute/zone us-central1-a
      ```

   c. If you work with regional clusters, run the following command to set the default region for the GKE cluster:

      ```
      gcloud config set compute/region compute_region
      ```

      Where *compute_region* is the default region for the GKE cluster. For example:

      ```
      gcloud config set compute/region us-east1
      ```

d. To make sure that you are using the latest version of the Cloud SDK, run the following command to check for updates:

```
gcloud components update
```

GKE Cluster Creation Scripts and Configuration Files

Cambridge Semantics provides a package of files that enable users to manage the configuration, creation, and deletion of GKE clusters and nodepools. The files and their directory structure are shown below.

```
├── common.sh
├── conf.d
│   ├── k8s_cluster.conf
│   ├── nodepool_anzograph.conf
│   ├── nodepool_anzograph.tuner.conf
│   ├── nodepool_common.conf
│   ├── nodepool.conf
│   ├── nodepool_dynamic.conf
│   ├── nodepool_dynamic.tuner.conf
│   └── nodepool_operator.conf
├── create_k8s.sh
├── create_nodepools.sh
├── delete_k8s.sh
└── delete_nodepools.sh
```

- The **create_k8s.sh** and **create_nodepools.sh** scripts are used to provision the GKE cluster and any number of nodepools in the cluster. And the **delete_k8s.sh** and **delete_nodepools.sh** scripts are used to remove the cluster or nodepools.
- The configuration file, **k8s_cluster.conf** in the **conf.d** directory, is used with the create_k8s and delete_k8s scripts and supplies the cluster-wide configuration parameters such as network, region, and timeout values.
- The configuration files, **nodepool*.conf** in the **conf.d** directory, contain nodepool settings. The files include sample configuration values for different types of nodepools, such as for Spark, Elasticsearch, or AnzoGraph. Each file is used to create one type of nodepool.
- The **common.sh** script is used by the create and delete cluster and nodepool scripts.

The instructions for Create the GKE Cluster and Create Nodepools in the GKE Cluster provide more details about the scripts and configuration files.

Create IAM Roles for GKE Permissions

Two Google Identity and Access Management (IAM) custom roles need to be created to grant the appropriate permissions to users who will set up and manage the GKE cluster and/or deploy applications using the K8s service:

- **GKE Cluster Admin**: The cluster administrator is responsible for GKE cluster management tasks such as creating and deleting clusters and nodepools and adding and removing containers in the Google Container Registry (GCR).
- **GKE Cluster Developer**: The cluster developer uses the GKE cluster that the admin creates. The user needs full access to k8s API objects to manage namespaces and creates K8s objects like pods and services.

The roles to add to Google Cloud are provided below. For information about creating roles, see Creating and Managing Custom Roles in the Google Cloud documentation. For information about the Kubernetes role-based access control (RBAC) system, see Role-Based Access Control in the Kubernetes Engine documentation.

GKE Cluster Admin Role

The following IAM role applies the minimum permissions needed for a cluster administrator who will create and manage the GKE cluster. Save the contents below as a JSON file. For example, `/home/centos/gke-cluster-admin.json`, and then run the following command to create the role:

```
gcloud iam roles create --project project_name --file=/path/filename.json
```

```
{
        "name": "customClusterAdminRole",
        "title": "Custom Role for GKE Cluster Admin",
        "includedPermissions": [
        "compute.networks.delete",
        "compute.networks.get",
        "compute.networks.list",
        "compute.networks.updatePolicy",
        "compute.routers.create",
        "compute.routers.delete",
        "compute.routers.get",
        "compute.routers.list",
        "compute.routers.update",
        "iam.serviceAccounts.actAs",
        "iam.serviceAccounts.get",
        "iam.serviceAccounts.list",
        "resourcemanager.projects.get",
        "resourcemanager.projects.list",
        "container.clusters.create",
        "container.clusters.delete",
        "container.clusters.get",
        "container.clusters.list",
        "container.clusters.update",
```

```
        "container.operations.*",
        "resourcemanager.projects.get",
        "Resourcemanager.projects.list"
    ],
    "stage": "BETA"
}
```

GKE Cluster Developer Role

The following IAM role applies the minimum permissions needed for a GKE cluster developer. Save the contents below as a JSON file. For example, `/home/centos/gke-cluster-developer.json`, and then run the following command to create the role:

```
gcloud iam roles create --project project_name --file=/path/filename.json
```

```
{
        "name": "customClusterDeveloperRole",
        "title": "Custom Role for GKE Cluster Developer",
        "includedPermissions": [
                "container.*",
                "resourcemanager.projects.get",
                "resourcemanager.projects.list",
                "container.apiServices.*",
                "container.backendConfigs.*",
                "container.bindings.*",
                "container.certificateSigningRequests.create",
                "container.certificateSigningRequests.delete",
                "container.certificateSigningRequests.get",
                "container.certificateSigningRequests.list",
                "container.certificateSigningRequests.update",
                "container.certificateSigningRequests.updateStatus",
                "container.clusterRoleBindings.get",
                "container.clusterRoleBindings.list",
                "container.clusterRoles.get",
                "container.clusterRoles.list",
                "container.clusters.get",
                "container.clusters.list",
                "container.componentStatuses.*",
                "container.configMaps.*",
                "container.controllerRevisions.get",
                "container.controllerRevisions.list",
                "container.cronJobs.*",
                "container.csiDrivers.*",
                "container.csiNodes.*",
                "container.customResourceDefinitions.*",
                "container.daemonSets.*",
```

```
            "container.deployments.*",
            "container.endpoints.*",
            "container.events.*",
            "container.horizontalPodAutoscalers.*",
            "container.ingresses.*",
            "container.initializerConfigurations.*",
            "container.jobs.*",
            "container.limitRanges.*",
            "container.localSubjectAccessReviews.*",
            "container.namespaces.*",
            "container.networkPolicies.*",
            "container.nodes.*",
            "container.persistentVolumeClaims.*",
            "container.persistentVolumes.*",
            "container.petSets.*",
            "container.podDisruptionBudgets.*",
            "container.podPresets.*",
            "container.podSecurityPolicies.get",
            "container.podSecurityPolicies.list",
            "container.podTemplates.*",
            "container.pods.*",
            "container.replicaSets.*",
            "container.replicationControllers.*",
            "container.resourceQuotas.*",
            "container.roleBindings.get",
            "container.roleBindings.list",
            "container.roles.get",
            "container.roles.list",
            "container.runtimeClasses.*",
            "container.scheduledJobs.*",
            "container.secrets.*",
            "container.selfSubjectAccessReviews.*",
            "container.serviceAccounts.*",
            "container.services.*",
            "container.statefulSets.*",
            "container.storageClasses.*",
            "container.subjectAccessReviews.*",
            "container.thirdPartyObjects.*",
            "container.thirdPartyResources.*",
            "container.tokenReviews.*",
            "compute.machineTypes.list",
            "storage.buckets.list",
            "storage.objects.get",
            "storage.objects.list"
    ],
    "stage": "BETA"
```

```
}
```

Configure the Networking for the GKE Cluster

The GKE cluster can be deployed into a new or existing VPC. Any networking components that do not exist, such as subnets for hosting the nodepools and a NAT gateway for outbound internet access, are created by the GKE service when the K8s cluster is deployed. Cambridge Semantics recommends that you enable Network Policies. For information, see Configuring Network Policies for Applications in the Kubernetes Engine documentation.

Note If Anzo is currently deployed, Cambridge Semantics recommends that you provision the GKE cluster in the same network as Anzo.

Create the GKE Cluster

Follow the steps below to configure and then provision the K8s control plane that will manage the nodepools that will be used with Anzo to deploy K8s applications.

1. Follow the Google Cloud Best Practices
2. Configure the Cluster
3. Create the Cluster

Follow the Google Cloud Best Practices

Before provisioning the cluster, Cambridge Semantics recommends that you implement the following best practices:

- Disable the Kubernetes Dashboard if it is enabled.
- Enable Stackdriver Logging. For information, see Stackdriver Logging in the Google Cloud documentation.

Configure the Cluster

Before creating the GKE cluster, specify the networking and control plane configuration to use by creating a configuration file based on the sample **k8s_cluster.conf** file in the **conf.d** directory. Follow the instructions below to configure the cluster.

1. In the **conf.d** directory, copy **k8s_cluster.conf** to create a file to customize. You can specify any name for the .conf file.

   Note: If you want to create multiple GKE clusters, you can create multiple .conf files. When you create a cluster (using create_k8s.sh), you specify the name of the .conf file to use for that cluster.

2. Open the new .conf file for editing. The file includes the list of parameters and sample values. Modify the values as needed. The table below describes each setting.

   View the gke_k8s_cluster.conf sample file

Parameter	Description	Sample Value
NETWORK_BGP_ ROUTING	The Compute Engine resource type for the cluster. Valid values are "global," "regional," or "zonal."	regional
NETWORK_ SUBNET_MODE	Specifies the method to use to create subnets. For example, "automatically" or "manually."	custom
NETWORK_ ROUTER_NAME	The name to assign to the Cloud Router.	csi-cloudrouter
NETWORK_ ROUTER_MODE	The route advertisement mode for the Cloud Router.	custom
NETWORK_ ROUTER_ASN	The Border Gateway Protocol (BGP) autonomous system number.	64512
NETWORK_ ROUTER_DESC	A description of the Cloud Router.	Cloud router for K8S NAT.
NETWORK_NAT_ NAME	The name to assign to the NAT gateway.	csi-natgw
NETWORK_NAT_ UDP_IDLE_ TIMEOUT	The timeout value to use for UDP connections to the NAT gateway.	60s
NETWORK_NAT_ ICMP_IDLE_ TIMEOUT	The timeout value to use for ICMP connections to the NAT gateway.	60s
NETWORK_NAT_ TCP_ ESTABLISHED_ IDLE_TIMEOUT	The timeout value to use for TCP established connections to the NAT gateway.	60s

Parameter	Description	Sample Value
NETWORK_NAT_TCP_TRANSITORY_IDLE_TIMEOUT	The timeout value to use for TCP transitory connections to the NAT gateway.	60s
K8S_CLUSTER_NAME	The name to give to the cluster.	cloud-k8s-cluster
K8S_CLUSTER_PODS_PER_NODE	The maximum number of pods that can be hosted on each EC2 instance.	10
K8S_CLUSTER_ADDONS	A comma-separated list of the addons to enable for the cluster.	HttpLoadBalancing, HorizontalPodAutoscaling
GKE_MASTER_VERSION	The Kubernetes version to use to deploy the GKE cluster.	1.13.7-gke.8
GKE_MASTER_NODE_COUNT_PER_LOCATION	The number of master nodes to create in each of the cluster's zones.	1
GKE_NODE_VERSION	The Kubernetes version to use to deploy nodes.	1.13.7-gke.8
GKE_IMAGE_TYPE	The base operating system for the nodes in the cluster.	COS
GKE_MAINTENANCE_WINDOW	The time of day to perform maintenance on the cluster.	06:00
GKE_MASTER_ACCESS_CIDRS	The list of CIDR blocks (up to 50) that have access to the Kubernetes master via HTTPS.	10.128.0.0/9
K8S_PRIVATE_CIDR	The IP address range (in CIDR notation) for the pods in this cluster.	172.16.0.0/20

Parameter	Description	Sample Value
K8S_SERVICES_CIDR	The IP address range for the services.	172.17.0.0/20
GCLOUD_NODES_CIDR	The CIDR for new subnet that will be created for the K8s cluster.	192.168.0.0/20
K8S_API_CIDR	The IPv4 CIDR range to use for the master network. The range should have a subnet mask of /28 and should be used in conjunction with the `--enable-private-nodes` flag.	192.171.0.0/28
K8S_HOST_DISK_SIZE	The size of the boot disks on the nodes.	50GB
K8S_HOST_DISK_TYPE	The type of boot disk to use.	pd-standard
K8S_HOST_MIN_CPU_PLATFORM	The minimum CPU platform to use.	Not set
K8S_POOL_HOSTS_MAX	The maximum number of nodes to allocate for the default initial nodepool.	1000
K8S_METADATA	The compute engine metadata (in the format key=val,key=val) to make available to the guest operating system running on nodes in the nodepool.	disable-legacy-endpoints=true
K8S_MIN_NODES	The minimum number of nodes in the nodepool.	1
K8S_MAX_NODES	The maximum number of nodes in the nodepool.	3

Parameter	Description	Sample Value
GCLOUD_ RESOURCE_ LABELS	Labels to apply to the Google Cloud resources in use by the GKE cluster (unrelated to Kubernetes labels).	deleteafter=false, owner=user
GCLOUD_VM_ LABELS	Kubernetes labels to apply to all nodes in the nodepool.	deleteafter=false, description=k8s_cluster, owner=user, schedule=keep-alive
GCLOUD_VM_TAGS	A comma-separated list of tags to add to the cluster.	tag1,tag2
GCLOUD_VM_ MACHINE_TYPE	The machine type to use for the cluster master nodes.	n1-standard-1
GCLOUD_VM_SSD_ COUNT	The number of local SSD disks to provision on each node.	0
GCLOUD_ PROJECT_ID	The Project ID for the GKE cluster.	cloud-project-1592
GCLOUD_ NETWORK	The compute engine network for the GKE cluster.	devel-network
GCLOUD_NODES_ SUBNET_SUFFIX	The suffix to use for subnets.	nodes
GCLOUD_ CLUSTER_REGION	The region for the GKE cluster.	us-central1
GCLOUD_NODE_ LOCATIONS	The zones to replicate the nodes in.	us-central1-f
GCLOUD_NODE_ TAINTS	Indicates the schedule settings for the nodes.	key1=val1:NoSchedule, key2=val2:PreferNoSchedule

Parameter	Description	Sample Value
GCLOUD_NODE_SCOPE	The permissions or access scopes the nodes should have.	gke-default

3. Save and close the configuration file, and then proceed to Create the Cluster.

Create the Cluster

To create a GKE cluster that is configured according to the file you created in Configure the Cluster, run the **create_k8s.sh** script with the following command:

```
$ ./create_k8s.sh cluster_config_filename [ -f ] [ -h ]
```

Argument	Description
cluster_config_filename	**Required** name and extension of the configuration file that supplies the parameter values for this cluster. You do not need to specify the path to the conf.d directory.
-f	**Optional** flag. If specified, the script will prompt for confirmation before proceeding with each stage involved in creating the cluster and its components.
-h	**Optional** flag. If specified, the help will be displayed.

For example:

```
$ ./create_k8s.sh k8s_cluster.conf
```

This script creates a GKE cluster along with the networking specifications described by the .conf file. The script deploys the infrastructure, such as a new subnets, using the CIDRs specified in the .conf file. It also creates a cloud router and NAT gateway.

When cluster creation is complete, proceed to Create Nodepools in the GKE Cluster to add one or more nodepools to the cluster.

Create Nodepools in the GKE Cluster

Follow the steps below to configure and then provision the nodepools in the GKE cluster that will be used to deploy K8s applications with Anzo.

1. Configure a Nodepool
2. Create a Nodepool

Configure a Nodepool

Before creating a nodepool, supply the configuration values to use for that nodepool based on a sample **nodepool*.conf** file in the **conf.d** directory. Follow the instructions below to configure a nodepool.

1. Depending on the application that you want to deploy with this nodepool, determine the configuration file to copy and customize from the **nodepool*.conf** sample files in the **conf.d** directory:
   - **nodepool_anzograph.conf**: The sample values in this file configure an autoscaling group for an AnzoGraph nodepool. The specified instance type offers high CPU and memory resources.
   - **nodepool_anzograph.tuner.conf**: The sample values in this file include the Linux kernel configuration settings that are ideal for AnzoGraph. The parameters disable Transparent Huge Pages (THP) for best performance and increase the memory mapping (mmap) limit to help avoid out of memory situations.
   - **nodepool_common.conf**: The sample values in this file configure a common nodepool, with a relatively small instance type that can be used for smaller workloads. This type of configuration is typically used to deploy K8s services that manage the instances in the cluster.
   - **nodepool.conf**:
   - **nodepool_dynamic.conf**: The sample values in this file configure an autoscaling group for a dynamic nodepool, where the instance type offers medium to large CPU and memory resources that are reasonable for deploying Elasticsearch and Spark applications.
   - **nodepool_operator.conf**: The sample values in this file configure an autoscaling group for an operator nodepool that can be used to deploy services.
2. Copy the appropriate configuration file to create the version to customize. You can specify any name for the .conf file.
3. Open the new .conf file for editing. The file includes the list of parameters and sample values. Modify the values as needed. The table below describes the settings.

   View a nodepool.conf sample file

Parameter	Description	Sample Value
DOMAIN	Name of domain that hosts the K8s nodepool.	acme
KIND	The type of pods that will be hosted on this nodepool.	anzograph

Parameter	Description	Sample Value
GCLOUD_ CLUSTER_ REGION	The region that the GKE cluster is deployed in.	us-central1
GCLOUD_ NODE_ TAINTS	Indicates the schedule settings for the nodes.	cambridgesemantics.com/dedicated=anzograph:NoSchedule, cloud.google.com/gke-preemptible="false":PreferNoSchedule
GCLOUD_ PROJECT_ID	The Project ID for the nodepool.	cloud-project-1592
GKE_IMAGE_ TYPE	The base operating system for the nodes.	cos_containerd
K8S_ CLUSTER_ NAME	The name of GKE cluster to add the nodepool to.	acme-k8s-cluster
NODE_ LABELS	A list of Kubernetes labels to apply to all nodes in the nodepool.	cambridgesemantics.com/node-purpose=anzograph, cambridgesemantics.com/description=k8snode, schedule=keep-alive,owner=user1, deleteafter=no
MACHINE_ TYPES	A space-separated list of the instance types that can be used for the nodes.	n1-standard-16 n1-standard-32
TAGS	A comma-separated list of the Compute Engine tags to apply to all nodes in the nodepool.	csi-sdl

Parameter	Description	Sample Value
METADATA	The compute engine metadata (in the format key=val,key=val) to make available to the guest operating system running on nodes in the nodepool.	disable-legacy-endpoints=true
MAX_PODS_PER_NODE	The maximum number of pods that can be hosted on each instance.	8
MAX_NODES	The maximum number of instances that can be deployed in the nodepool.	8
MIN_NODES	The minimum number of instances in the nodepool.	0
NUM_NODES	The number of nodes in the nodepool in each of the cluster's zones.	1
DISK_SIZE	The size of the boot disks on the nodes.	200Gb

Parameter	Description	Sample Value
DISK_TYPE	The type of boot disk to use.	pd-standard

4. Save and close the configuration file, and then proceed to Create a Nodepool. Repeat the steps above to configure additional nodepools.

Create a Nodepool

To create a nodepool that is configured according to a file you created in Configure a Nodepool, run the **create_nodepools.sh** script with the following command:

```
$ ./create_nodepools.sh nodepool_config_filename [ -f ] [ -h ]
```

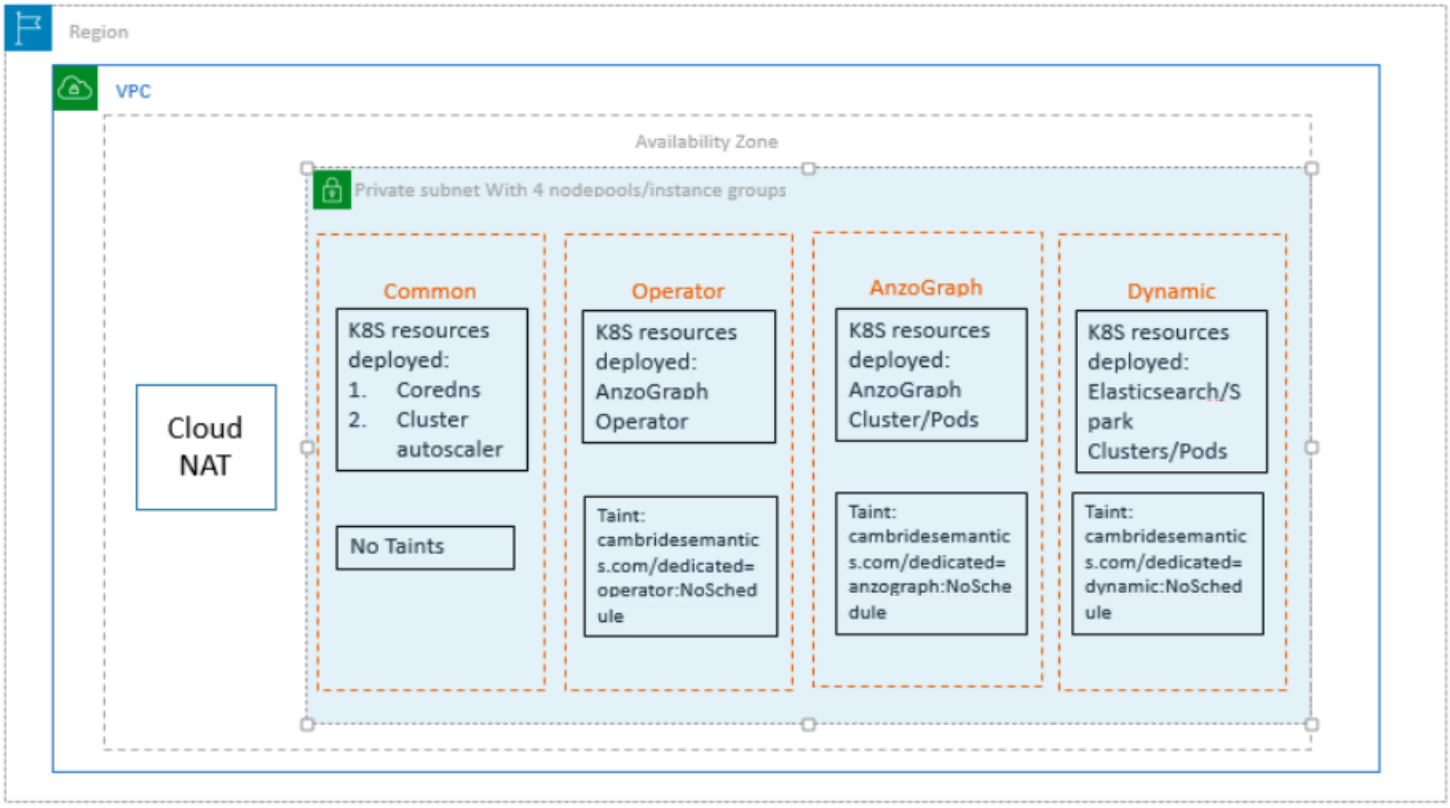
Argument	Description
nodepool_config_filename	**Required** name and extension of the configuration file that supplies the parameter values for this nodepool. You do not need to specify the path to the conf.d directory.
-f	**Optional** flag. If specified, the script will prompt for confirmation before proceeding with each stage involved in creating the nodepool and its components.
-h	**Optional** flag. If specified, the help will be displayed.

For example:

```
$ ./create_nodepools.sh anzograph_nodepool.conf
```

This script creates the nodes in the nodepool according to the specifications provided in the .conf file. Repeat the process to create additional nodepools.

The diagram below shows the architecture for a sample deployment with four nodepools.



The next step is to create a Cloud Location in Anzo so that Anzo can connect to the GKE cluster and deploy applications. See Managing Cloud Locations.

Related Topics

Dynamic Deployment Architecture and Process Overview

Managing Cloud Locations

Getting Started Guide

The Getting Started Guide helps users get started with Anzo. The topics in this section introduce the user interface, describe basic setup information, and provide a tutorial that guides users through the process of building a sample solution from start to finish.

Introduction to the User Interface

This topic provides instructions for accessing the Anzo user interface and gives a general overview of the design and layout. For information about the Anzo concepts and procedures that influence the user interface's design and use, see Anzo Concepts and Vocabulary.

To access the user interface:

1. Go to the following URL in your browser:

   ```
   https://hostname
   ```

   Where *hostname* is the Anzo server DNS name or IP address.

   Note: Your browser may warn you that the server's SSL certificate has not been signed by a trusted authority. This is normal behavior. To continue, click the **Advanced** link at the bottom of the page and then click the **Proceed** link. If you require a trusted site certificate, you can obtain one from a Certificate Authority and add it to the Anzo server.

2. On the Login screen, type your username and password and then click **Sign In**.

The user interface is organized and compartmentalized by the processes involved with building solutions. The design accommodates use cases where users with different permissions and responsibilities build various parts of a solution. The sections below introduce you to each of the user interface elements.

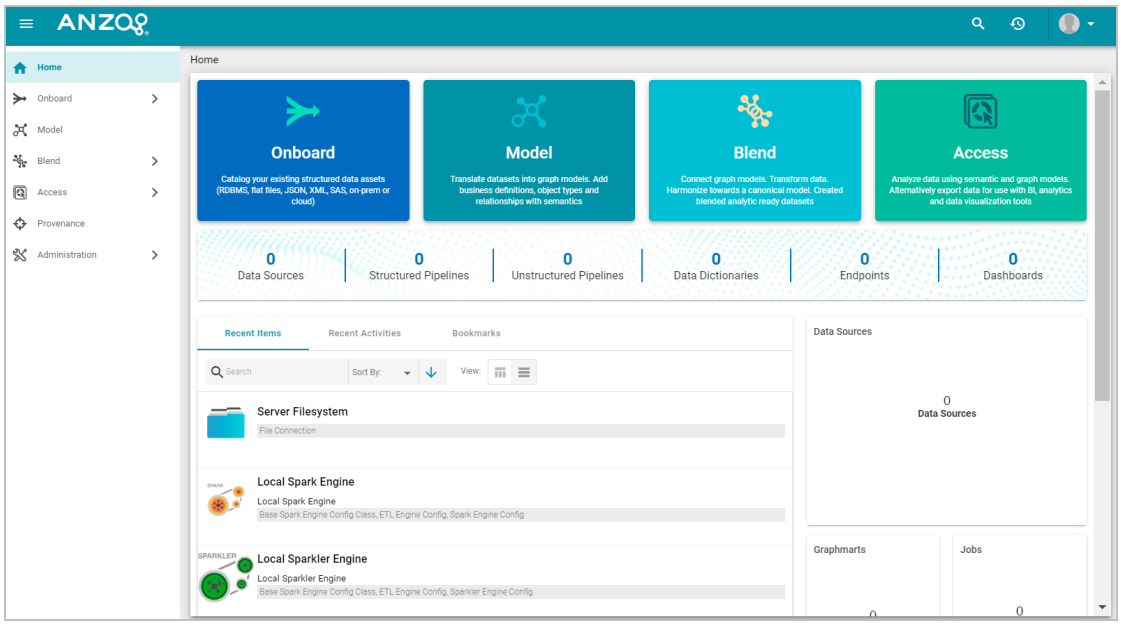
Note The following sections show an administrator view of the user interface. Depending on their assigned system role, some users might not see all components. For information about default Anzo roles and permissions, see Default Anzo Roles and Permissions.

- General Interface Elements
- Onboard
- Model
- Blend
- Access

- Provenance
- Administration

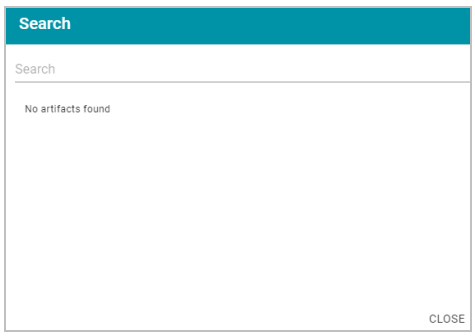
General Interface Elements

Each time you log into the console, the Home page is displayed and provides links to the features the logged in user has access to. Below the icons is a dashboard that displays an overview of the system artifacts and recently updated or bookmarked items. The dashboard becomes populated once you start onboarding data sources.



On the menu bar at the top, you can click the contents icon ( ) to expand and collapse the left navigation menu. The right side of the top menu bar includes the following options:

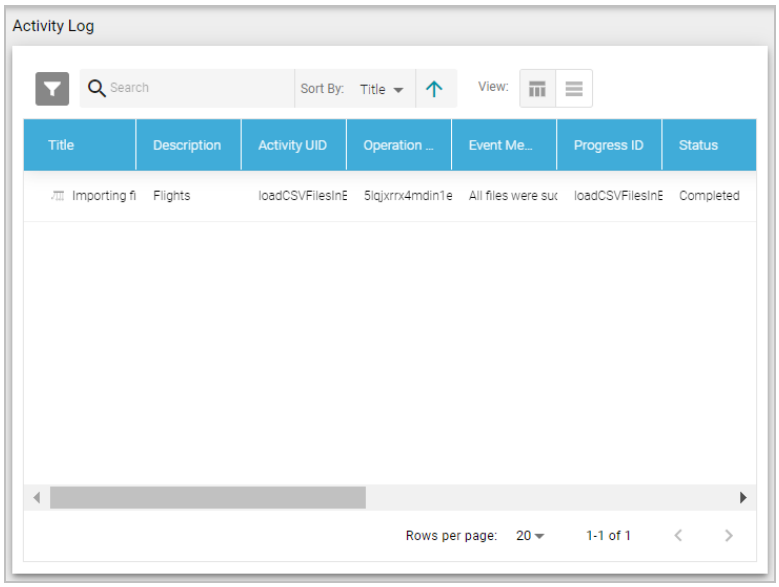
- The Search icon ( ) opens the Search dialog box where you can do a global search for a resource, such as a data source, data set, graphmart, or schema, by title. The Search field accepts wildcard characters.



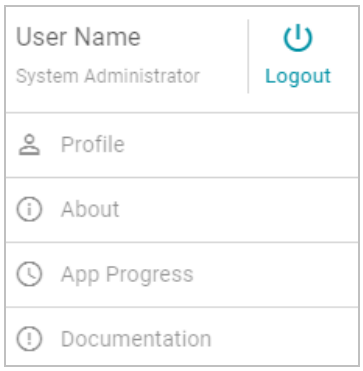
- The Activity Log icon ( ) opens the Activity Log, which offers a high-level view of individual activities, such as ETL engine and AnzoGraph usage and pipeline activity. The ability to filter activities based on their status and type enables users to drill down to activities of interest as well as profile user behavior. Additionally, the ability

to include system data expands the scope of monitoring and aids in troubleshooting.

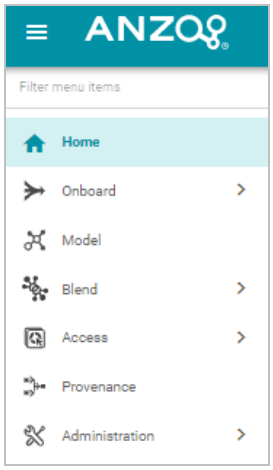
Once you start using Anzo and there are events that show up in the log, you can click **View All History** to open the full log for searching and filtering.



- The User menu ( ) provides access to your user profile, the About screen, and the Application Progress window, which lists recent application activity. It also includes the ability to log out of the console and a Documentation link that opens this guide.

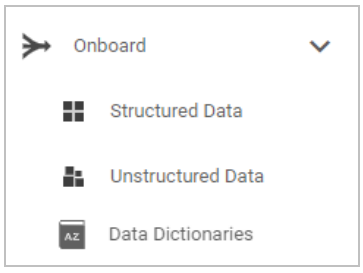


The navigation menu on the left provides access to all of the Anzo features.



Onboard

The Onboard menu provides access to the components that users configure to ingest data from various sources.

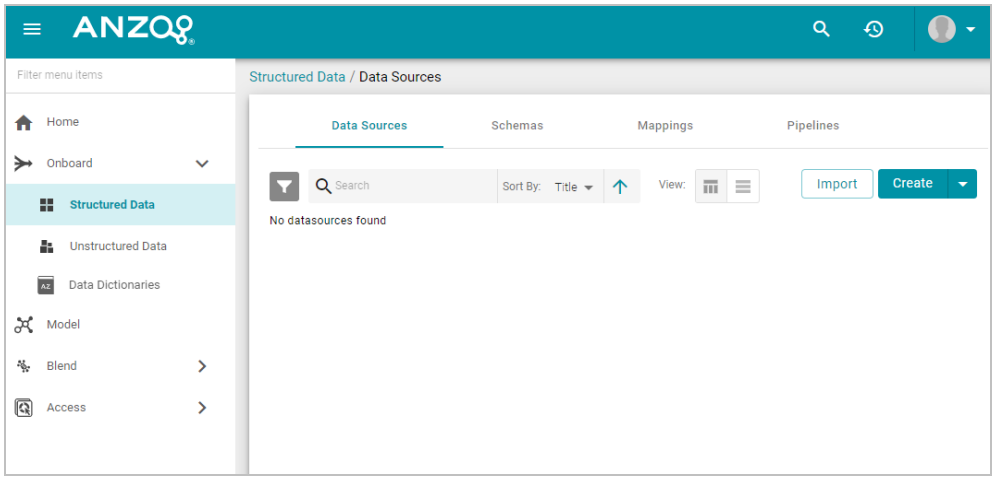


Structured Data

From **Structured Data**, users access data sources, schemas, mappings, and pipelines for structured data sources: database connections, CSV, JSON, XML, and SAS files.

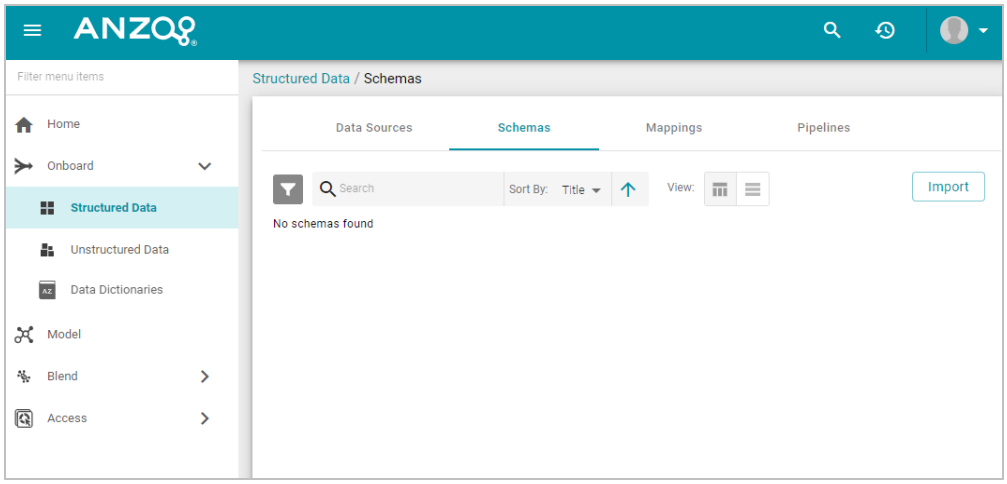
Data Sources

From the **Data Sources** tab, users connect to the files and databases that contain the data to onboard:



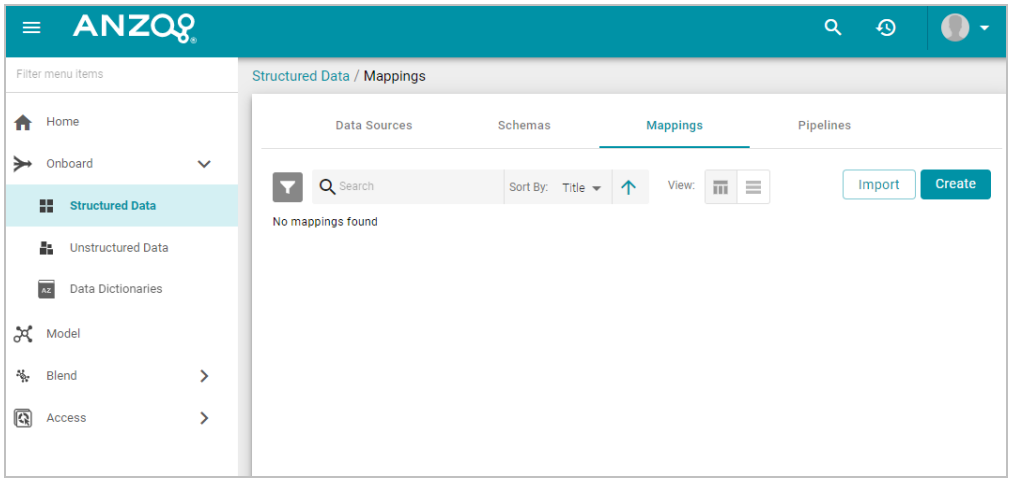
Schemas

From the **Schemas** tab, users view, create, and manage the schemas that define the data to onboard:



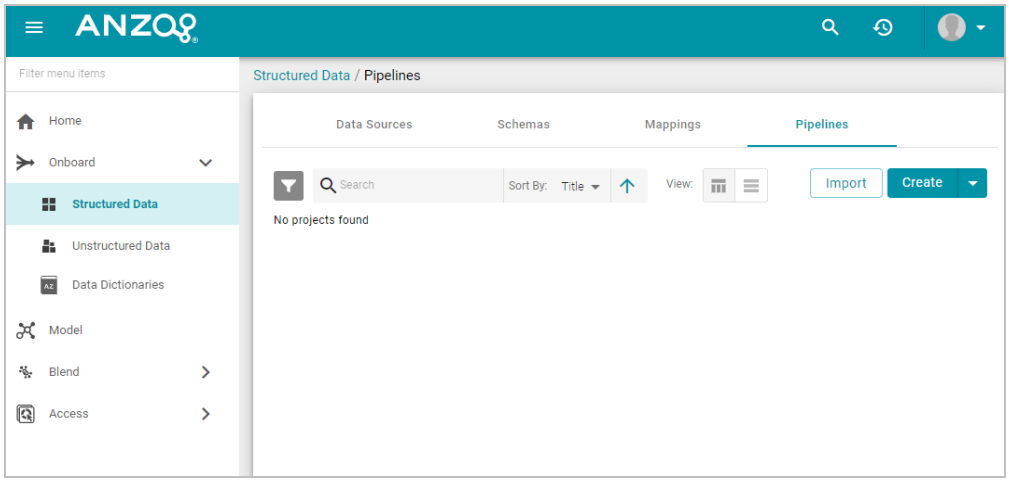
Mappings

From the **Mappings** tab, users view, create, and manage the mappings that describe the relationships between schemas and data models as well as perform optional transformations on the source data:



Pipelines

From the **Pipelines** tab, users view, create, and manage the pipelines that run ETL jobs to onboard data into Anzo:

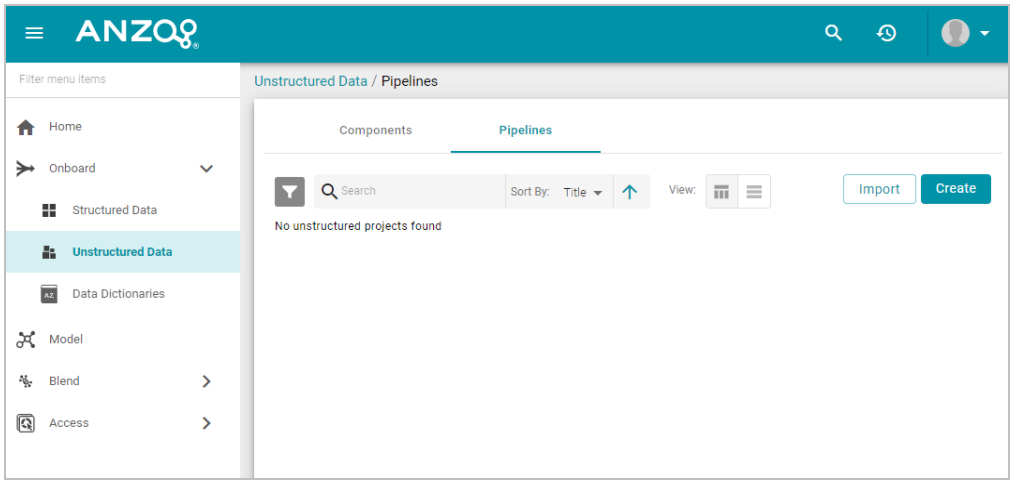


Unstructured Data

From **Unstructured Data**, users access and create pipelines for onboarding data from unstructured sources such as Office documents, PDFs, web pages, email messages, and knowledgebases.

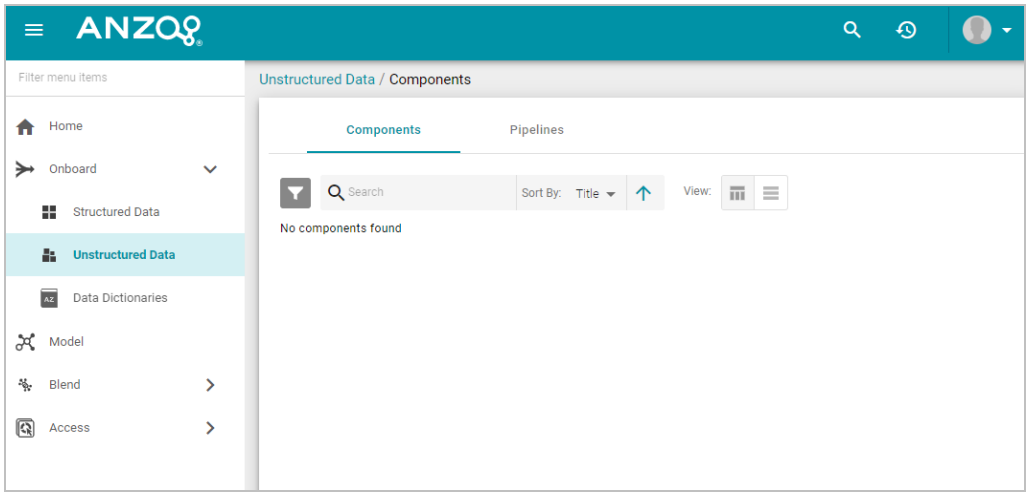
Pipelines

From the **Pipelines** tab, users can create and manage unstructured pipelines:



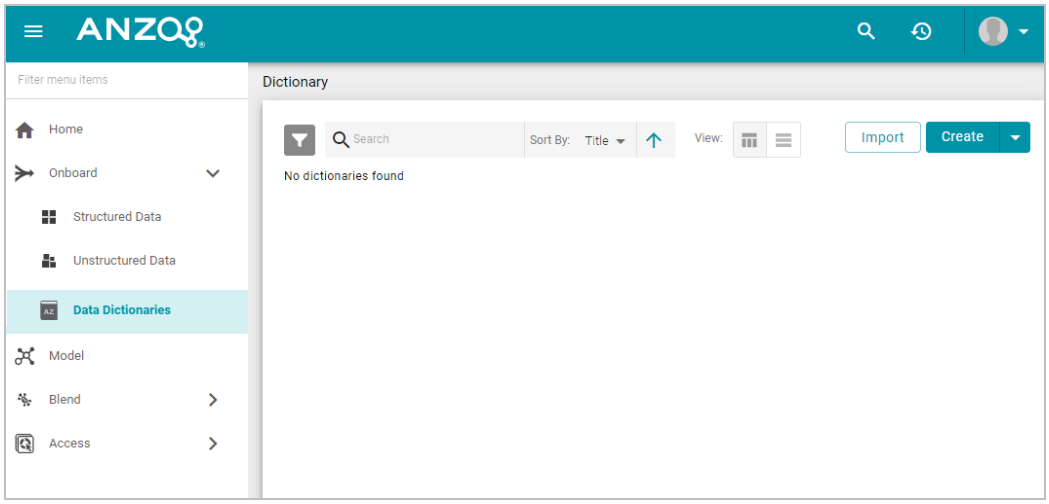
Components

From the **Components** tab, users can view and configure unstructured pipeline components:



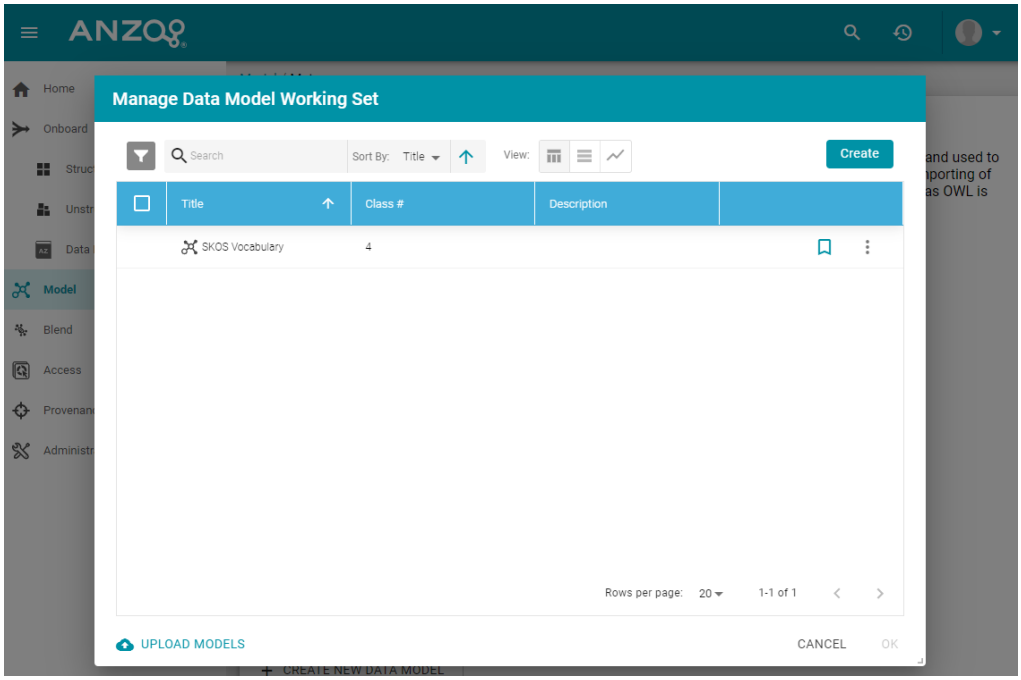
Data Dictionaries

From **Data Dictionaries**, users view, create, and manage metadata dictionaries, which define and normalize concepts across data sources.



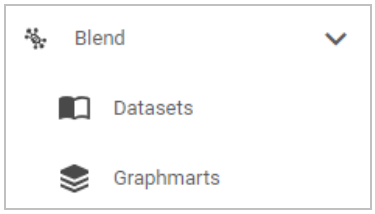
Model

The **Model** manager enables users to view, create, and manage the data models, which describe the concepts, attributes, and relationships in or across the data sets.



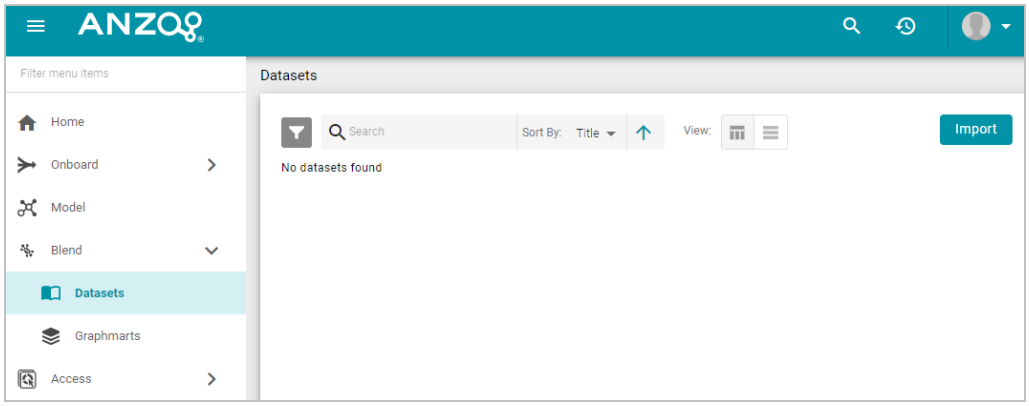
Blend

The **Blend** menu provides access to features that combine various datasets from different sources.



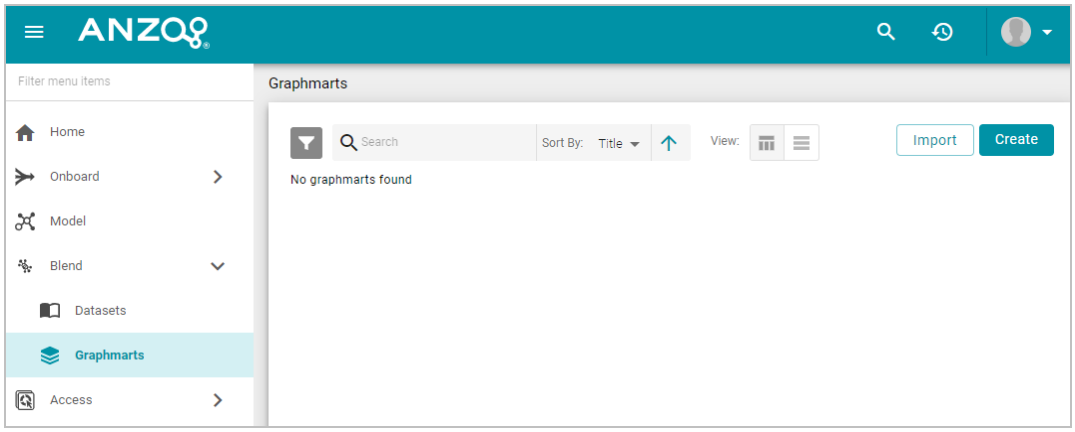
Datasets

The **Datasets** catalog is an inventory of all of the data in Anzo. Users can sort the list by tags, classes, and data set creators. Users can also add data sets to graphmarts for loading into AnzoGraph and then designing Hi-Res Analytics dashboards.



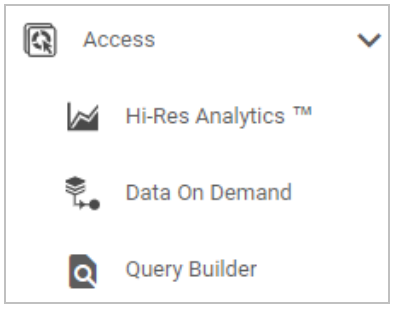
Graphmarts

The **Graphmarts** page lists all of the existing graphmarts. Users can designate favorites and sort the list by tags and graphmart creators. Users can click a graphmart in the list to view details such as the data sets included in the graphmart, data layer details, and associated dashboards.



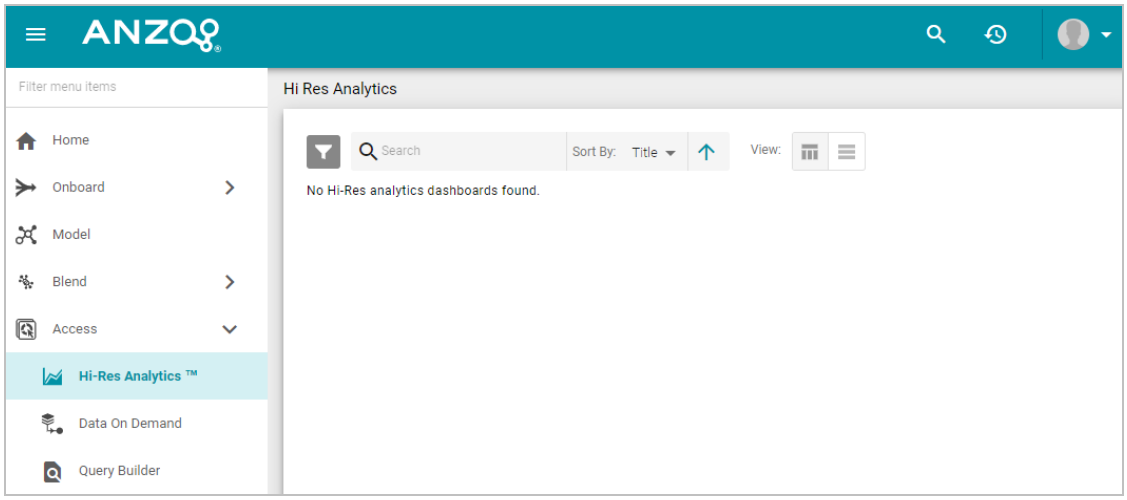
Access

The **Access** menu provides quick access to commonly used features.



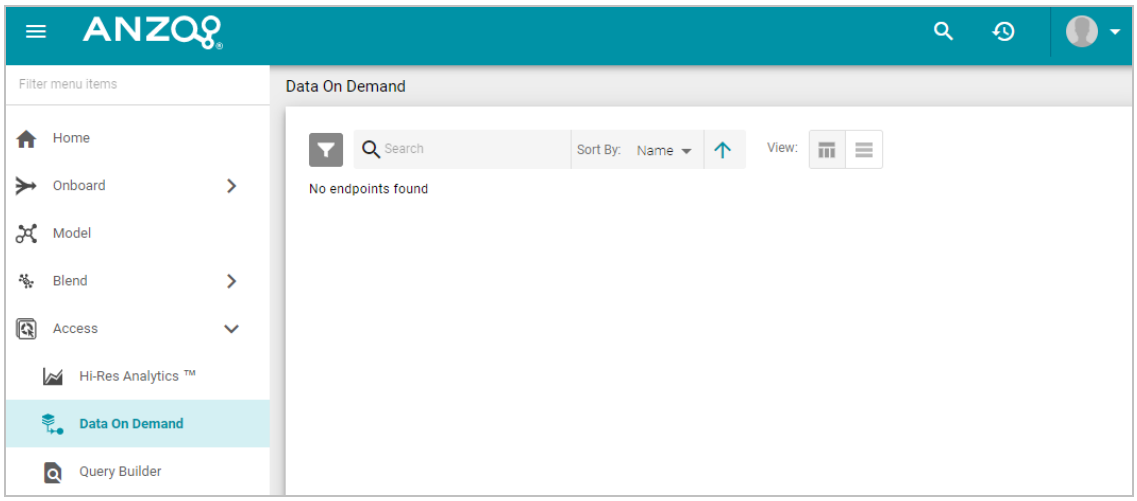
Hi-Res Analytics

The **Hi-Res Analytics** page lists all of the existing dashboards. Users can click a dashboard in the list to open it in the Hi-Res Analytics application. Users can designate favorites and sort the list by tags, graphmarts, and dashboard creators.



Data On Demand

The **Data On Demand** page provides a list of the data on demand endpoints that have been created.

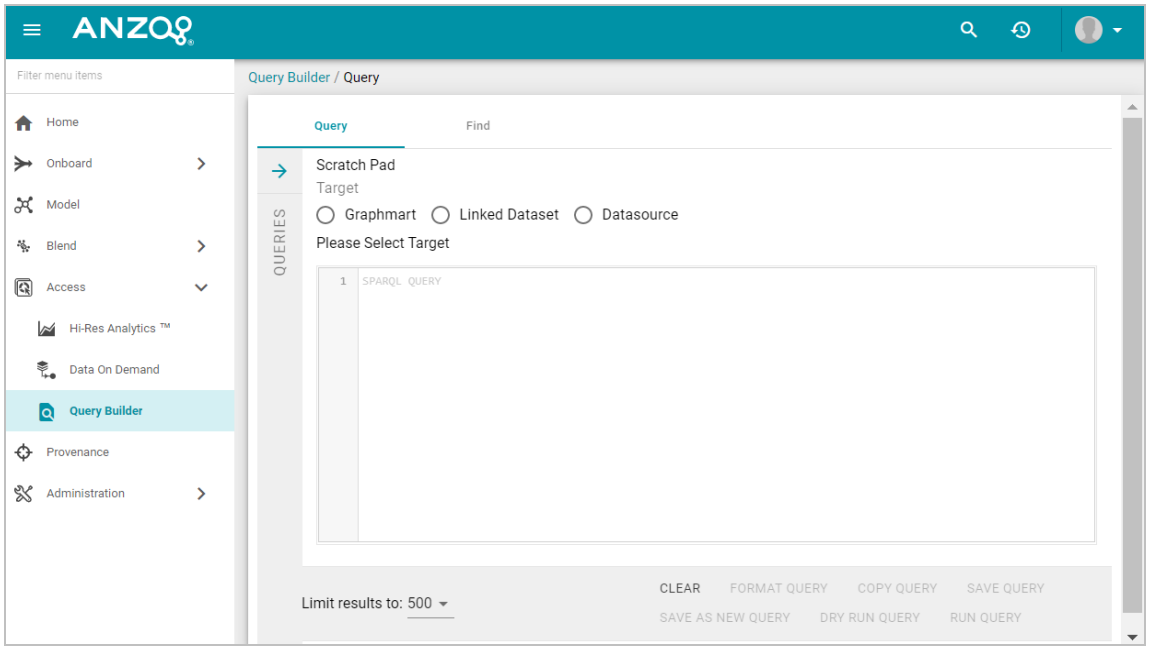


Query Builder

The **Query Builder** provides options to write and run SPARQL queries or find quads.

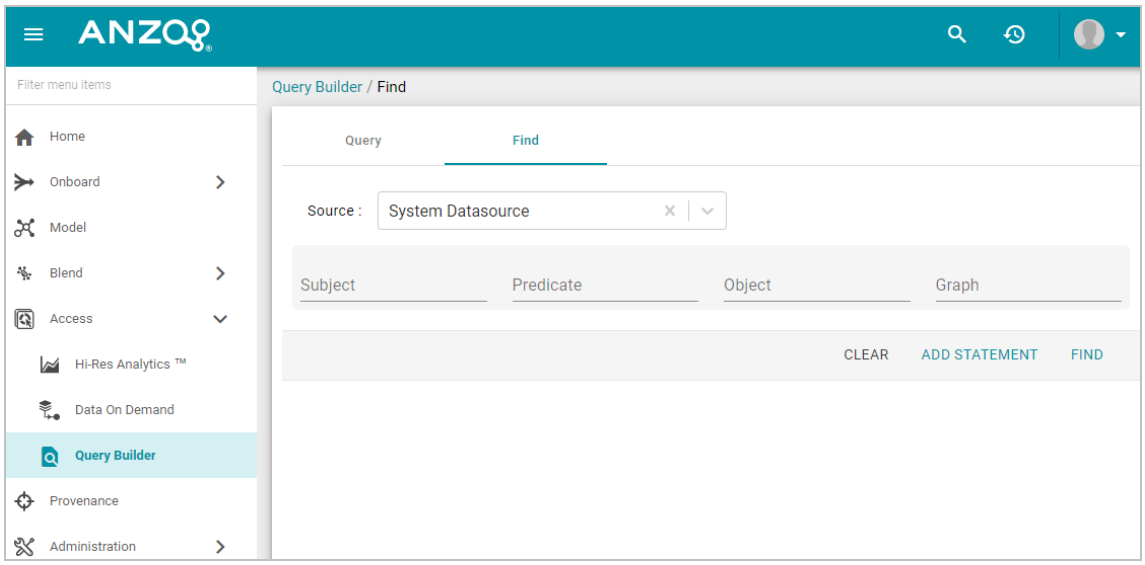
Query

The **Query** editor provides syntax assistance, type-ahead suggestions for model entity names, and automated prefix creation and query formatting for readability. It also includes the option to save queries for later use.



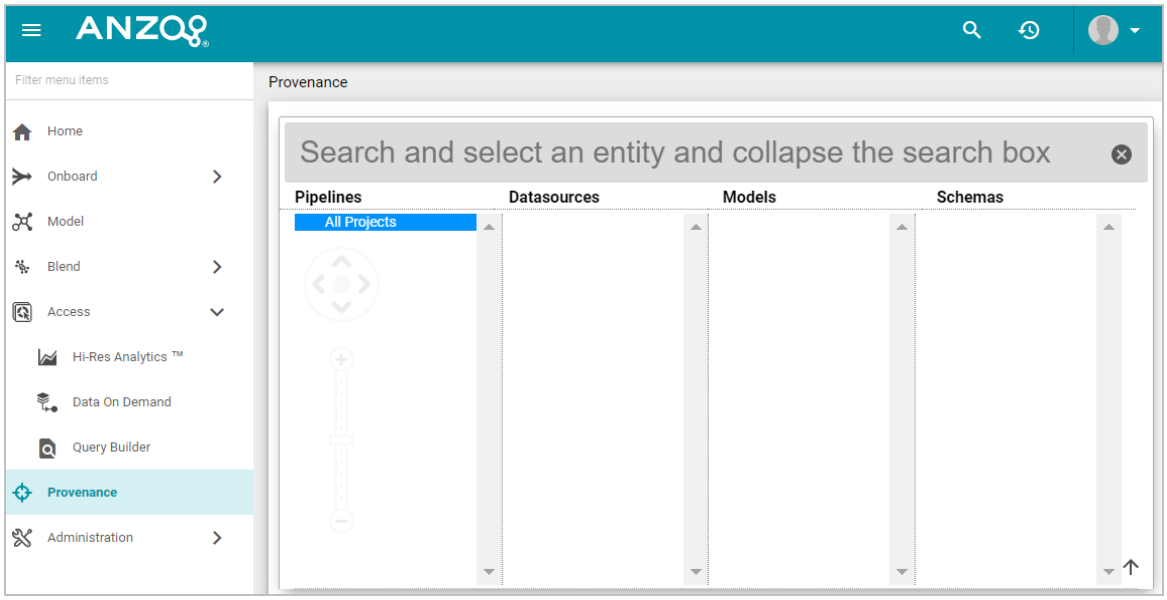
Find

The **Find** page enables users to search for quads by specifying a single subject, object, predicate, or graph name.



Provenance

The **Provenance** viewer enables users to view the lineage of all ingested structured data. The provenance explorer provides an overview of the relationships across various sources and models. Users can search for data entities and view associated pipelines, data sources, data models, and schemas.



Administration

The **Administration** menu provides access for administrators to complete the initial setup and management of Anzo components. See the Administration Guide for information about the options in the Administration menu.

Related Topics

Connecting to a File Store

Creating an Anzo Data Store

Making a Basic Connection to AnzoGraph

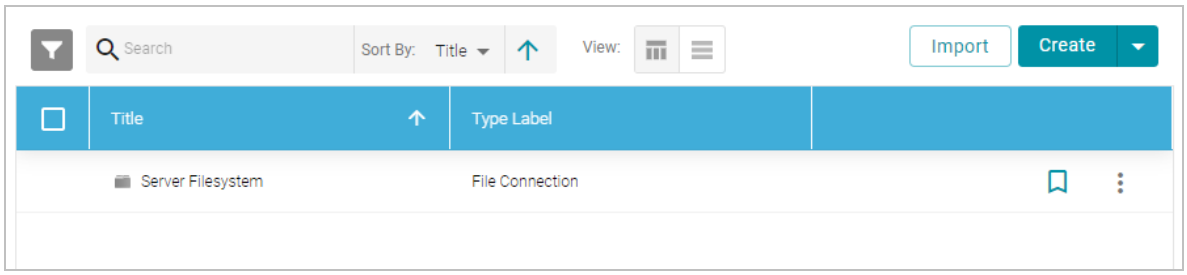
Tutorial: Building a Sample Solution from Scratch

Connecting to a File Store

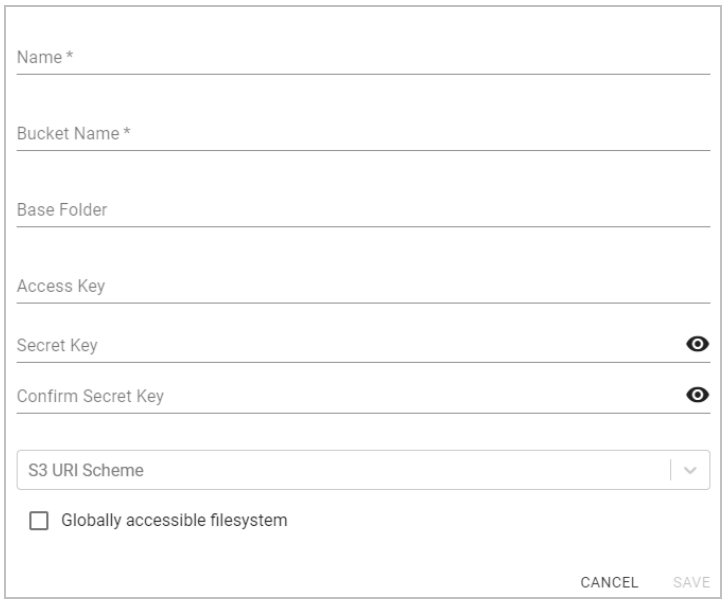
This topic provides instructions for defining a file system location for storing the files that Anzo ingests or creates. Anzo supports reading from and writing to local or mounted file systems (such as NFS), Hadoop Distributed File Systems (HDFS), File Transfer Protocol (FTP or FTPS) systems, Google Cloud Platform (GCP) storage, and Amazon Simple Cloud Storage Service (S3).

Tip The Anzo server file system location is configured and accessible by default. If you store files on a storage system that is mounted directly onto the Anzo, AnzoGraph, Elasticsearch, and Anzo Unstructured servers, you are not required to configure that location.

1. In the Anzo console, expand the **Administration** menu and click **File Store**. Anzo displays the File Store screen, which lists existing file store connections. For example:



2. Click the **Create** button and select the type of file connection that you want to create. For the local disk or mounted NFS, choose **File**. Anzo displays the create connection screen. For example:



3. On the connection screen, provide the file system details. The settings that display depend on the type of file connection that you chose. The list below describes the settings for each file connection type.

File

- **Name**: The name to use to describe this file connection within Anzo.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

HDFS

- **Name**: The name to use to describe this file connection within Anzo.
- **Nameservice IP or Name**: The IP address or host name for the storage system.
- **Port**: The port to access the server on.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **HDFS Configuration Path**: Enter the full path to the configuration files.
- **Keytab Path**: The full path to the keytab file.
- **Username**: The user name for the account used to access the server.
- **Password** and **Confirm Password**: The password for the account used to access the server.
- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

Important If you use Kerberos Authentication with HDFS, you must also configure your AnzoGraph cluster to authenticate with Kerberos. For instructions, see Configuring AnzoGraph for Kerberos Authentication.

FTP and FTPS

- **Name**: The name to use to describe this file connection within Anzo.
- **Server IP or Name**: The IP address or host name for the storage system.
- **Port**: The port to access the server on.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Username**: The user name for the account used to access the server.
- **Password** and **Confirm Password**: The password for the account used to access the server.
- **Keystore Path**: For FTPS connections, the full path to the keystore file.

- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

GCP

- **Name**: The name to use to describe this file connection within Anzo.
- **Bucket Name**: The name of the bucket to store files in.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Account Email**: The email address for the account used to access the storage.
- **Key File Location**: The full path to the keystore password file.
- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

S3

Important When using Amazon S3 for file storage, make sure that AWS server-side encryption is disabled. Anzo cannot read and write files on S3 if the location uses server-side encryption.

- **Name**: The name to use to describe this file connection within Anzo.
- **Bucket Name**: The name of the bucket to store files in.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Access Key**: The Access Key ID to use for accessing the S3 location.
- **Secret Key** and **Confirm Secret Key**: The Secret Key ID for the Access Key.
- **S3 URI Scheme**: Specifies whether the URI scheme is S3, S3 Native, or S3A.
- **Globally accessible filesystem**: **Required**. Enable this option for S3 file stores.

4. Click **Save** to save the configuration. The file store connection that you specified becomes available as a choice when you create graph data stores or select source files to onboard.

See Creating an Anzo Data Store for instructions on designating a directory on the file storage system where Anzo can save the AnzoGraph load files that are generated during the ETL process.

Related Topics

Introduction to the User Interface

Creating an Anzo Data Store

Making a Basic Connection to AnzoGraph

Tutorial: Building a Sample Solution from Scratch

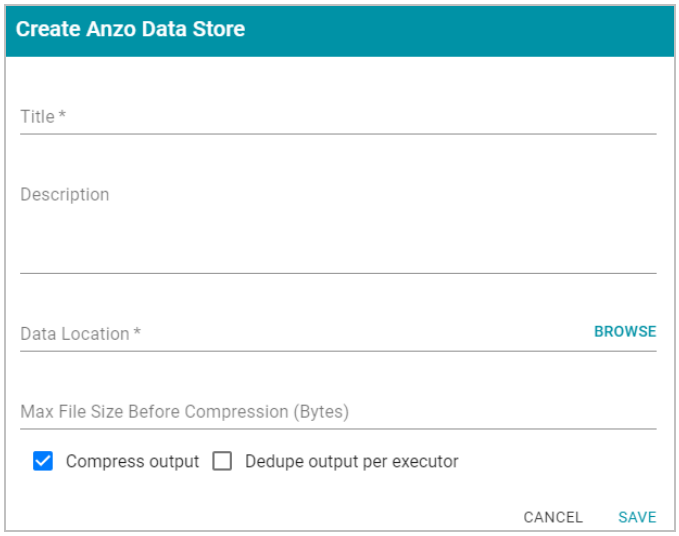
Creating an Anzo Data Store

This topic provides instructions for creating an Anzo data store, also known as a graph data source. Creating a data store means that you designate a directory on the file storage system where Anzo can save the AnzoGraph load files that are generated during the ETL process. All installations require at least one data store. You can create one graph data store and configure all pipelines to write to that store (each ETL run automatically creates a new sub-directory under the data store directory) or you can create multiple data stores to use for different data sets.

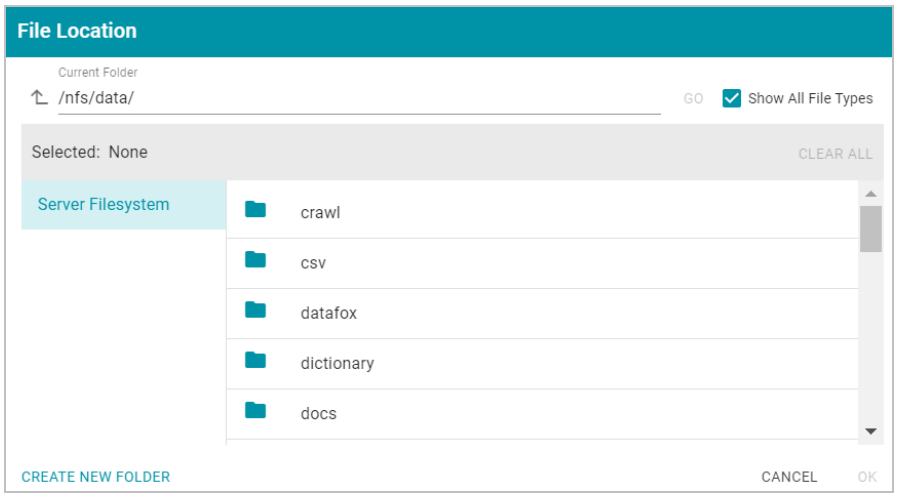
For information about setting up a file system or storage connection, see Connecting to a File Store.

Note Administrator privileges are required to complete this task.

1. In the Anzo console, expand the **Administration** menu and click **Anzo Data Store**.
2. On the Anzo Data Store screen, click the **Create** button. Anzo opens the Create Anzo Data Store screen.



3. Type a **Title** and optional **Description** for the data store.
4. Click in the **Data Location** field. Anzo opens the File Location dialog box.



5. On the left side of the screen, select the storage location where you want to create this data store. On the right side of the screen, navigate to the base directory where you want Anzo to save the data files for this data store. Select a directory, and then click **OK**. Each time ETL runs for this store, Anzo creates a new subdirectory under the base location that you specify.

   Note Ideally, the Data Location is a directory that the Anzo, AnzoGraph, and any Anzo Unstructured and Elasticsearch servers have access to, such as a mounted file system or cloud storage location. If you want Anzo to generate files for this data store in one location and load the files into AnzoGraph from another location, specify the file generation location in this field, and then specify the AnzoGraph load location in the **Alternate Data Location** field that is displayed on the details screen after you save the data store.

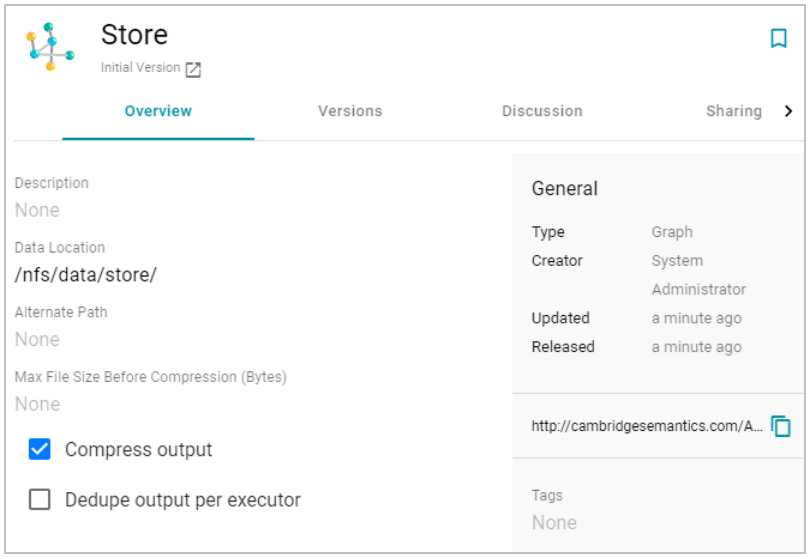
6. Specify whether to compress the generated load files. By default, the **Compress output** checkbox is selected, indicating that Anzo generates .ttl.gz files when writing to this graph data source. If you clear the checkbox, Anzo generates uncompressed .ttl files. To preserve disk space and reduce read times when loading data into memory, Cambridge Semantics recommends that you accept the default configuration and compress load files.
7. The Spark ETL engine does not remove duplicates by default when running pipelines. If the source contains a significant number of duplicate entities, you have two options for deduplicating the data:
   - **Deduplicate the data during the ETL process**: To deduplicate the data while running the jobs that will generate this graph source, select the **Dedupe output per executor** option. Enabling the dedupe option limits the number of duplicates to one duplicate per executor node. For example, if the Spark configuration has 10 executor nodes, the resulting data set can contain a maximum of 10 duplicate entities. Note that deduplication is based on primary keys and URI templates. **If the source does not employ templating, do not enable the dedupe option**.

     Important Enabling this option substantially increases the time it takes to run the jobs for this graph source.

   - **Deduplicate the data after loading it to AnzoGraph**: AnzoGraph deduplicates data during a "vacuum" process that runs automatically after data is loaded into memory. If you leave the Dedupe output per executor option disabled, duplicates will be removed by AnzoGraph.

     Note Deduplicating data with AnzoGraph streamlines the ETL process but can increase load time and temporary memory usage in AnzoGraph during the load.

8. Click **Save** to create the data store. Anzo saves the store and displays the data store overview. For example:



   You can click a field to edit a value. Click the check mark icon (✓) to save changes to an option, or click the X icon (✕) to clear the value for an option.

9. If you plan to load files into AnzoGraph from a location that is different than the **Data Location** that you specified, edit the **Alternate Data Location** field and select the location for AnzoGraph load files.

See Making a Basic Connection to AnzoGraph for instructions on connecting Anzo to AnzoGraph for loading the files that are generated for the new data store.

Related Topics

Introduction to the User Interface

Connecting to a File Store

Making a Basic Connection to AnzoGraph

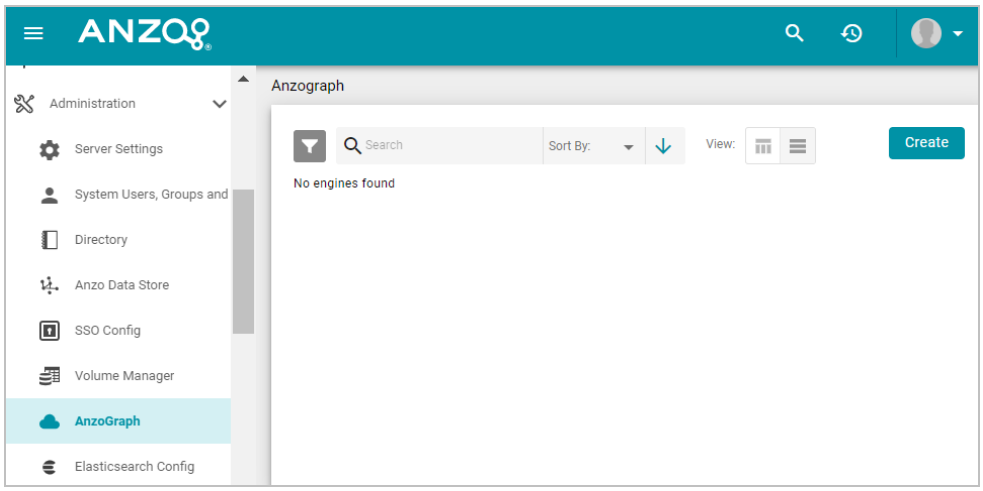
Tutorial: Building a Sample Solution from Scratch

Making a Basic Connection to AnzoGraph

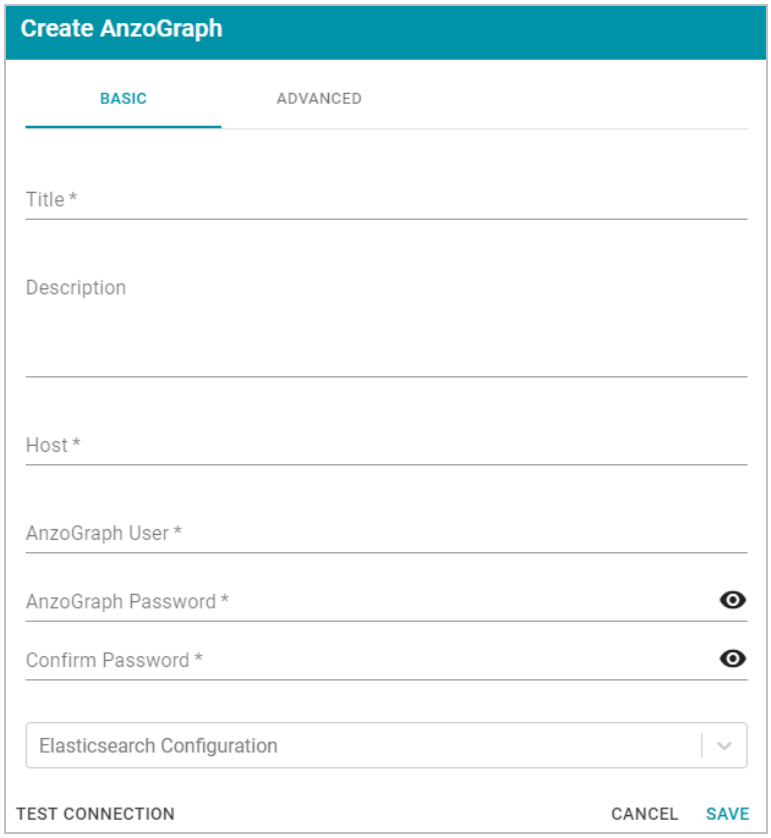
This topic provides instructions for quickly configuring a basic connection to AnzoGraph. For information about all of the AnzoGraph connection options, see Connecting to AnzoGraph.

Note Administrator privileges are required to complete this task.

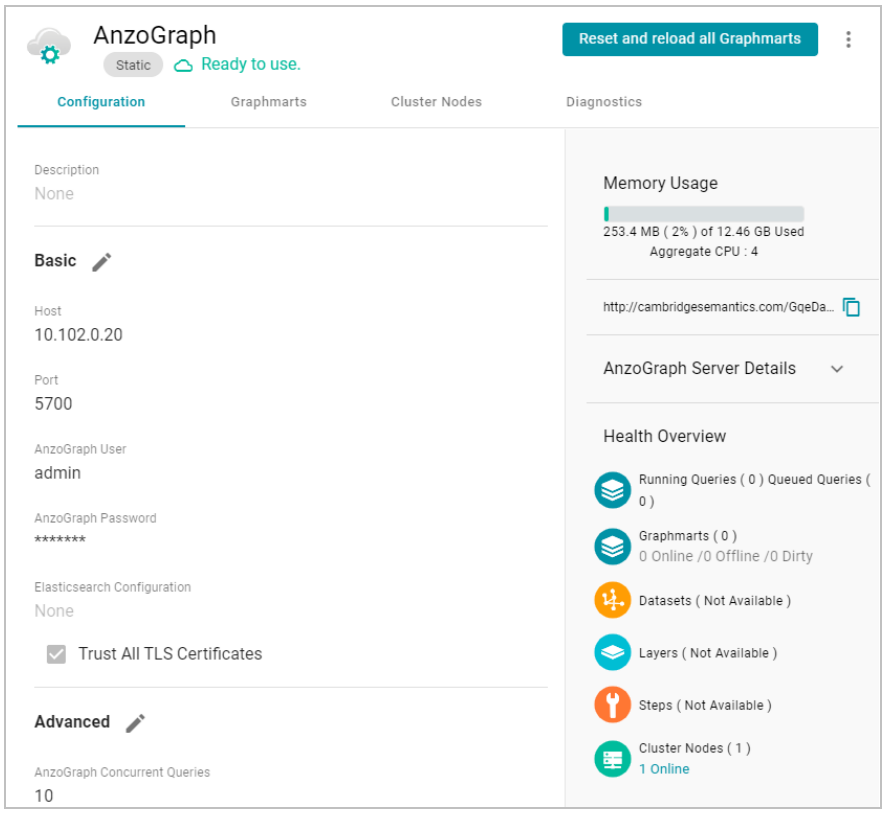
1. In the Anzo console, expand the **Administration** menu and click **AnzoGraph**. Anzo opens the AnzoGraph connection overview screen, which lists any existing connections. For example:



2. Click **Create** to add a connection. Anzo displays the Create AnzoGraph dialog box.



3. On the Basic tab, type a name for the engine in the **Title** field.
4. In the optional **Description** field, type a description for the graph query engine. If you leave this field blank, Anzo creates a description when you save the configuration.
5. In the **Host** field, type the AnzoGraph server host name or IP address. If you have a cluster, type the name or IP address of the leader server.
6. In the **AnzoGraph User** field, type the username that was created when AnzoGraph was installed.
7. Type the password for the AnzoGraph user in the **AnzoGraph Password** and **Confirm Password** fields.
8. Leave the **Elasticsearch Configuration** field unset. The Elasticsearch configuration is used with Anzo Unstructured. See Deploying the Anzo Unstructured Infrastructure for more information.
9. Click **Test Connection** to check if Anzo can connect to AnzoGraph. If the connection fails, make sure that AnzoGraph is running and that you typed the correct username and password.
10. Click **Save** to save the configuration. Anzo connects to AnzoGraph and returns to the AnzoGraph screen. Click the new AnzoGraph connection and then click the **Configuration** tab to view the connection status as well as memory usage details, overall data statistics, and graphmart details. For example:



See Tutorial: Building a Sample Solution from Scratch for next steps and instructions for creating a sample solution.

Related Topics

Introduction to the User Interface

Connecting to a File Store

Creating an Anzo Data Store

Tutorial: Building a Sample Solution from Scratch

Tutorial: Building a Sample Solution from Scratch

This topic helps get you started with Anzo by teaching you all of the basic steps to follow to create a complete solution. The instructions below provide a sample CSV file and guide you through all of the steps required to import the file and visualize the data in Hi-Res Analytics.

1. Prepare the Environment
2. Download the Sample Data
3. Create a CSV Data Source and Import the Sample Data
4. Auto-Ingest the Imported Data Set
5. Run the ETL Pipeline
6. Create a Graphmart and Load the Sample Data to AnzoGraph
7. Explore the Data in Hi-Res Analytics
8. Review the New Solution's Components

Prepare the Environment

To give the tutorial continuity and make it straightforward to follow, it helps to complete a few general setup steps before you start building the solution. This section provides background information and describes the general environment preparation.

This tutorial guides you through importing data from a file, running an ETL pipeline to create RDF files for AnzoGraph, and loading the data into AnzoGraph. Completing those steps requires a file store that Anzo and AnzoGraph can access, a designated location (Anzo data store) on that storage system to place the RDF files that Anzo generates, and a connection to AnzoGraph for loading the data.

Confirm that the following setup steps are complete before proceeding:

1. AnzoGraph and AnzoGraph have access to the same file store and its location is configured in Anzo. **Note**: The Anzo server file system is configured and accessible by default. If you store files on a file system that is mounted directly to the Anzo and AnzoGraph servers, you do not need to configure a new file system location. If you store files in the cloud, HDFS, or a location that is accessible via FTP, configure that storage location in Anzo.

   For instructions on configuring a file storage location, see Connecting to a File Store.
2. An Anzo data store has been created for the RDF files that Anzo generates during the ETL process. For instructions on creating a data store, see Creating an Anzo Data Store.
3. Anzo has a connection to AnzoGraph. For instructions on configuring the connection, see Making a Basic Connection to AnzoGraph.

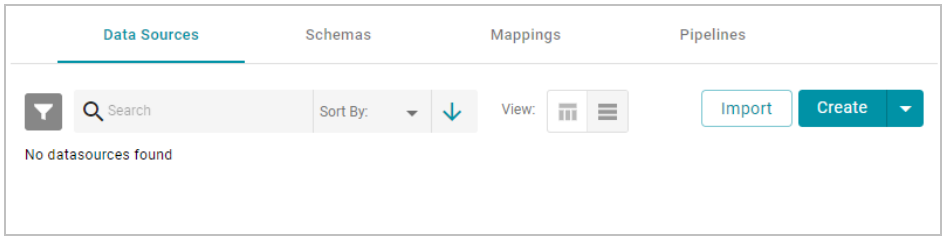
Download the Sample Data

This tutorial uses a small sample data set from Data World. The data is in CSV format and includes IMDB movie data from 2006 – 2016. Click the link below to download IMDB-Movie-Data.zip to your computer. Then extract the ZIP file to view **IMDB-Movie-Data.csv**.

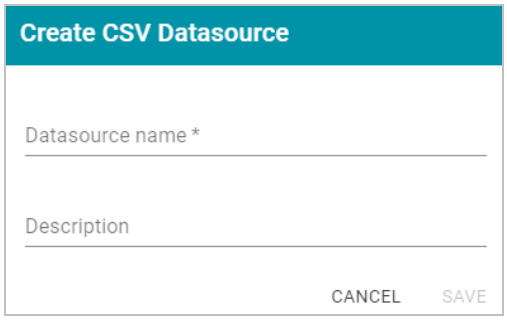
IMDB-Movie-Data.zip

Create a CSV Data Source and Import the Sample Data

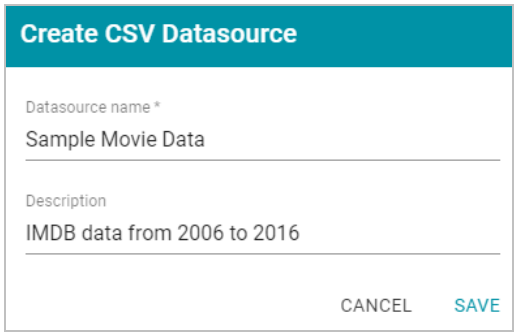
1. Log in to the Anzo console, expand the **Onboard** menu item, and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources:



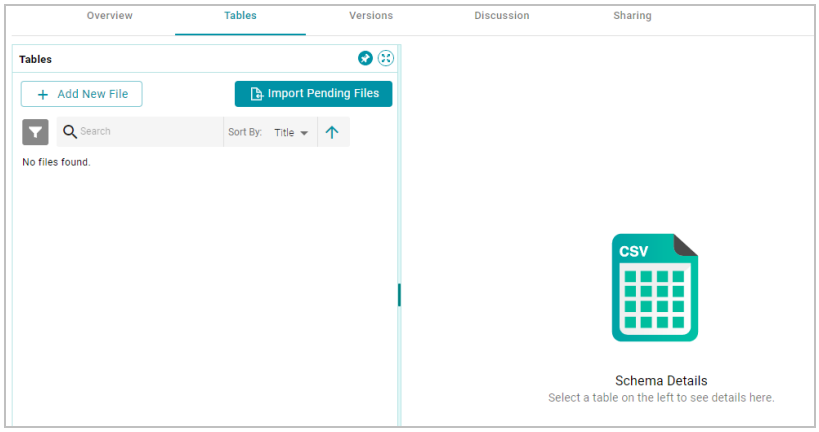
2. Click the **Create** button and select **CSV Data Source**. Anzo opens the Create CSV Datasource screen.



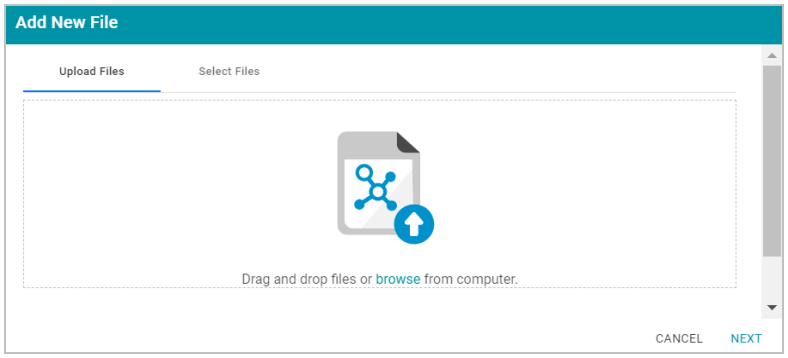
3. Type a name for the source in the **Datsource name** field. For example, type "Sample Movie Data." Then type an optional description in the **Description** field. For example:



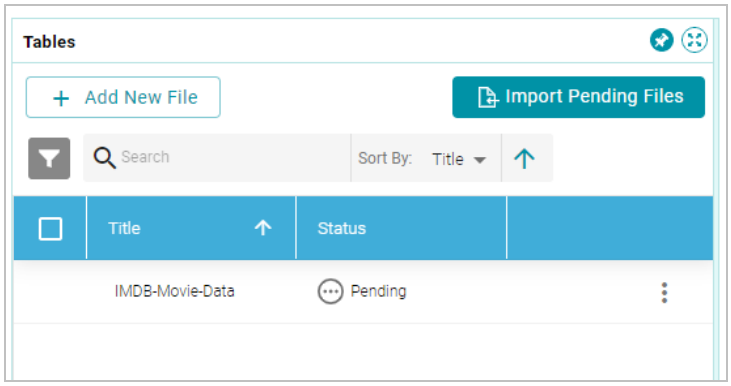
4. Click **Save**. Anzo saves the source and displays the Tables tab.



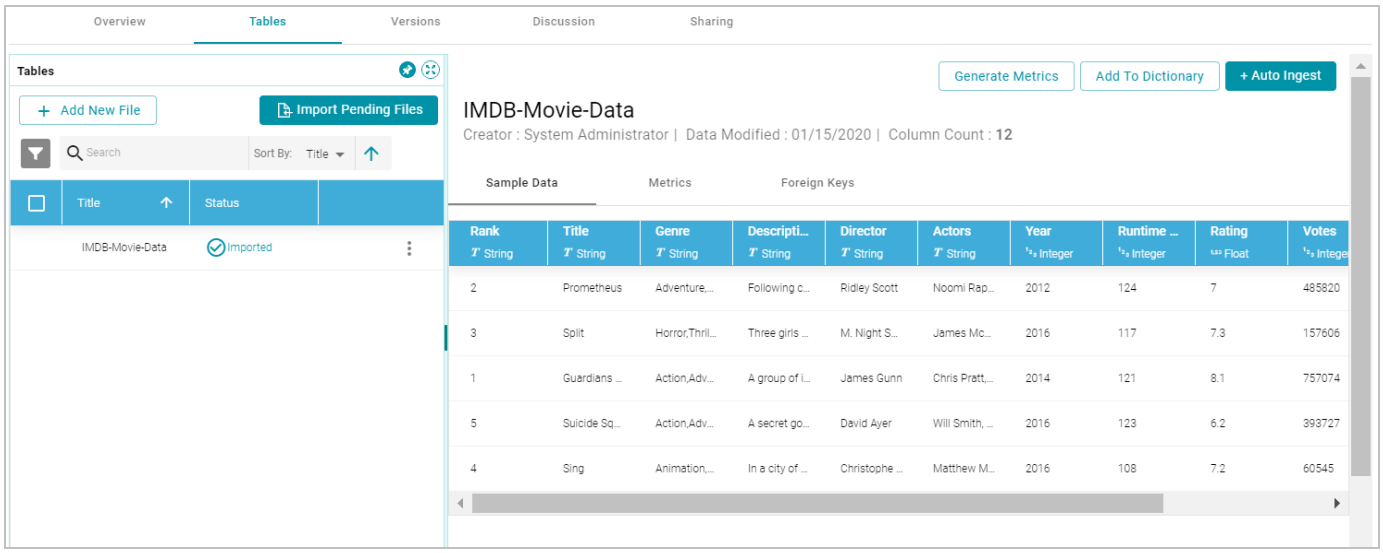
5. On the left side of the screen, click the **Add New File** button. Anzo displays the Add New File dialog box, and the **Upload Files** tab is selected.



6. On the **Upload Files** tab, drag and drop the IMDB-Movie-Data.csv file onto the screen or click **browse** and navigate to the file and double-click the file to select it. Anzo attaches the file and the Next button becomes active. Click **Next**. Anzo lists the file on the left side of the screen with a status of Pending. For example:



7. Click the **Import Pending Files** button to import the IMDB data to Anzo. Anzo imports the data and the status changes to **Imported**. Click the table row for IMDB-Movie-Data to display the schema on the right side of the screen and enable additional operations.

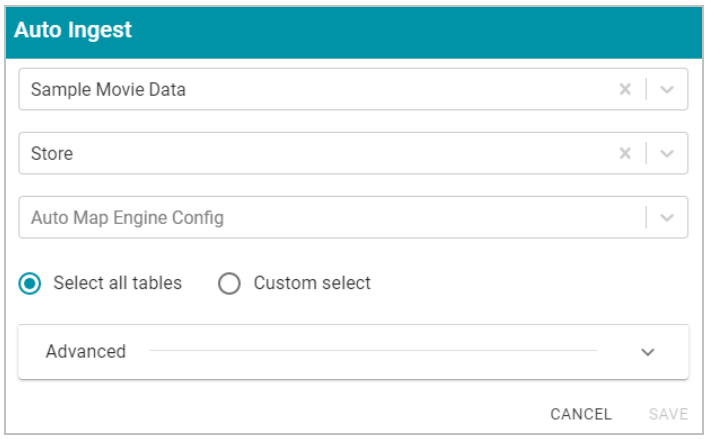


The data is now imported to Anzo as a CSV data set.

Proceed to the next step to ingest the data and automatically derive a data model, generate a mapping, and create an ETL pipeline for transforming the new data set to the RDF graph model.

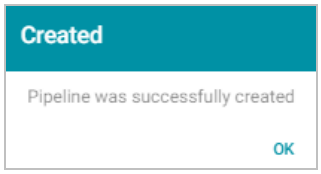
Auto-Ingest the Imported Data Set

1. On the Tables screen (from the last step of the procedure above), click the **Auto Ingest**. Anzo opens the Auto Ingest dialog box and automatically populates the data source and data store values. For example:



2. Click the **Auto Map Engine Config** field and select one of the ETL engines to use. This example uses the **Local Spark Engine**.
3. Leave the **Select all tables** radio button selected.

4. Click **Save**. Anzo creates a pipeline for the data set and displays a confirmation:

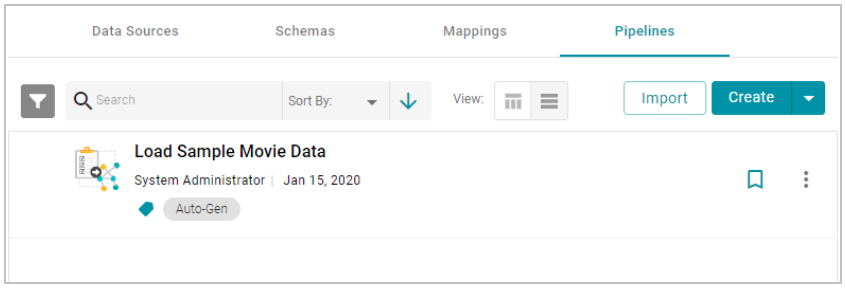


   Click **OK** to close the dialog box.

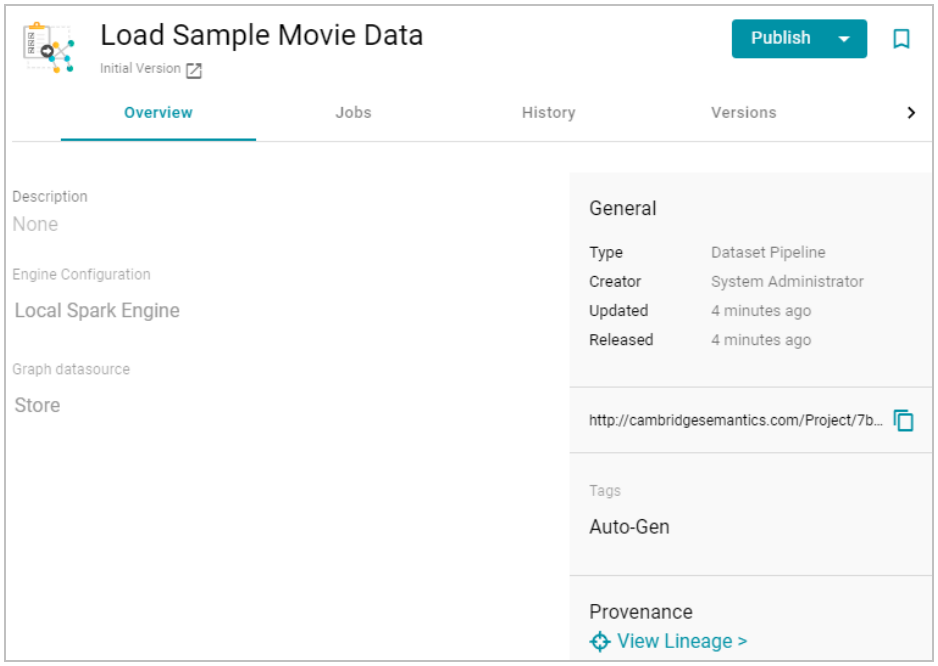
As part of creating the pipeline, Anzo also generates a data model and mapping for the sample data set. The last step in this tutorial describes each of the components. Proceed to the next step to run the pipeline and generate the AnzoGraph load files.

Run the ETL Pipeline

1. In the Onboard menu, click **Structured Data**. Then click the **Pipelines** tab. Anzo displays the Pipelines screen, which lists the pipeline for the sample movie data set. For example:



2. Click the pipeline title to view the pipeline overview. For example:

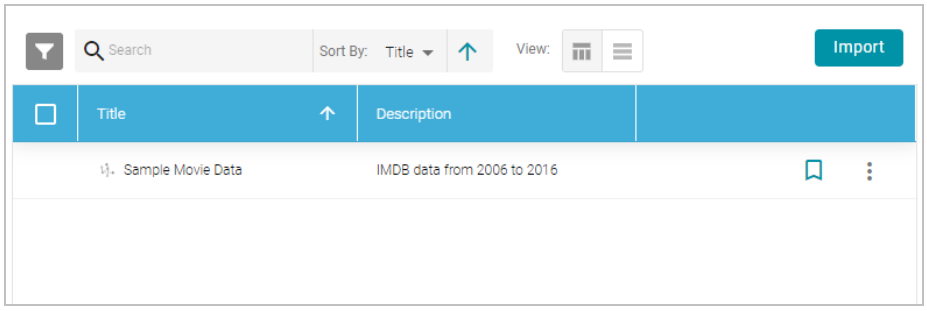


3. Click the **Publish** button to run the pipeline, transform the data to graph format, and generate the load files for AnzoGraph in the data store.

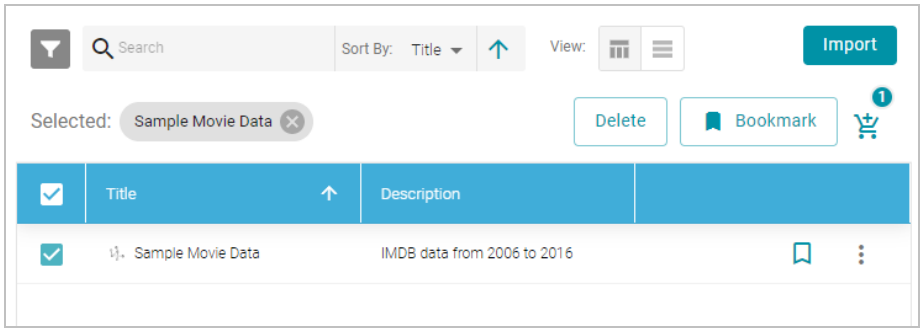
Anzo publishes the ETL files and displays a confirmation that job execution is in progress. Click **OK** to close the dialog box. Anzo completes the job execution and the sample movie data set becomes available in the **Dataset** catalog. Proceed to the next step to create a graphmart for the new data and load it into AnzoGraph.

Create a Graphmart and Load the Sample Data to AnzoGraph

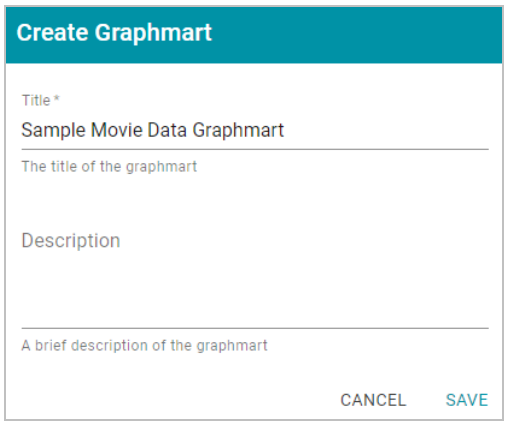
1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Datasets catalog, which shows an inventory of all of the data in Anzo. It lists the sample movie data set. For example:



2. Hover the pointer over the Sample Movie Data item to display a checkbox in the left column, and then select the checkbox. Anzo adds the data set to the shopping cart and additional icons become available at the top of the screen. For example:

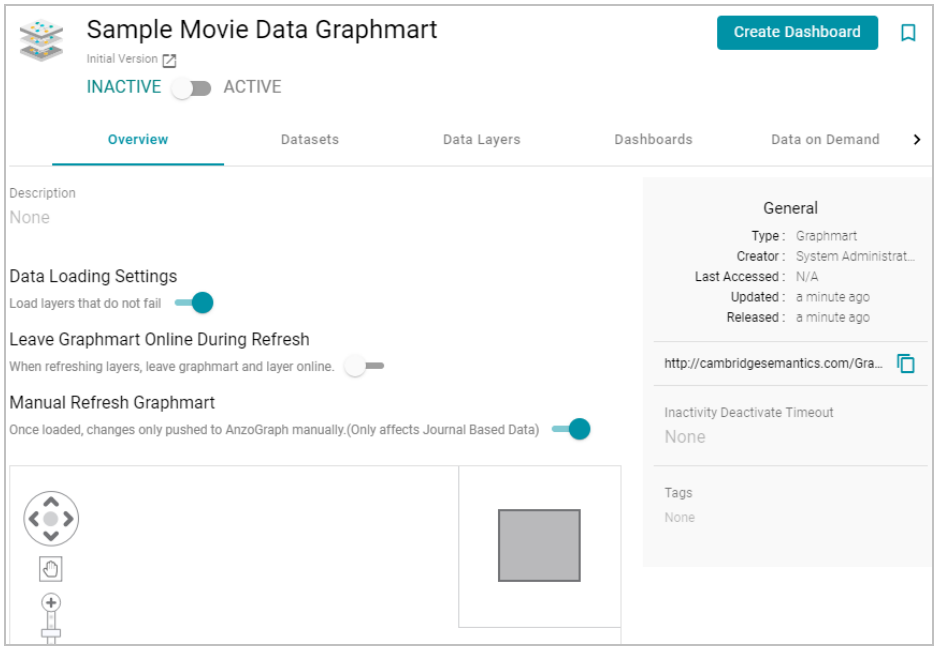


3. Click the shopping cart icon ( ) at the top of the screen. Anzo displays the Create Graphmart screen. For example:



Anzo populates the Title field by appending "Graphmart" to the data set name.

4. On the Create Graphmart screen, you have the option edit the title and type an optional description. Click **Save** when you are ready to create the graphmart. Anzo creates the graphmart and displays the Overview screen for the new graphmart. For example:



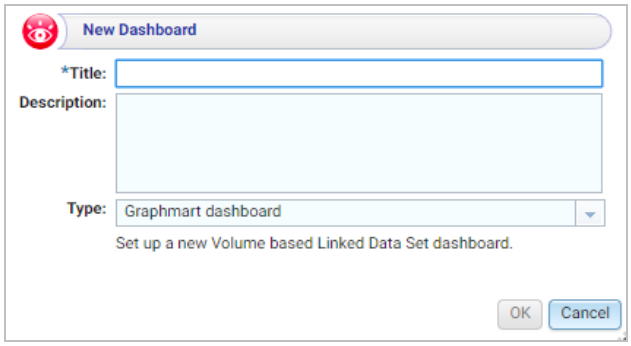
Since the source data is a single CSV file with 12 columns, the data model becomes a single class with 12 properties. The graphmart view at the bottom of the screen shows a single circle because there is one class. Graphmarts with multiple classes with relationships between them show a detailed graph view.

5. To load the graphmart to AnzoGraph, slide the slider at the top of the screen from Inactive to **Active**. Anzo loads the graphmart.

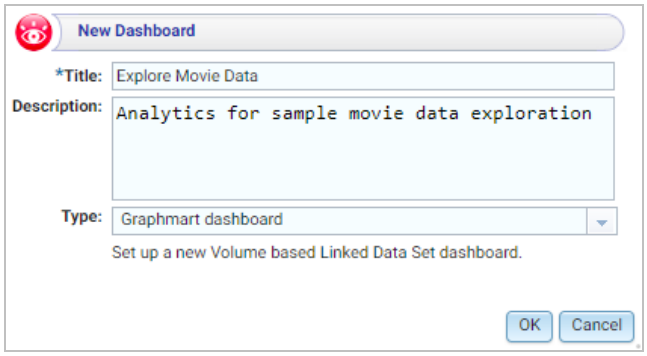
Now that the data is loaded into AnzoGraph, it is available to view in dashboards. Proceed to the next step to explore the sample data with Hi-Res Analytics.

Explore the Data in Hi-Res Analytics

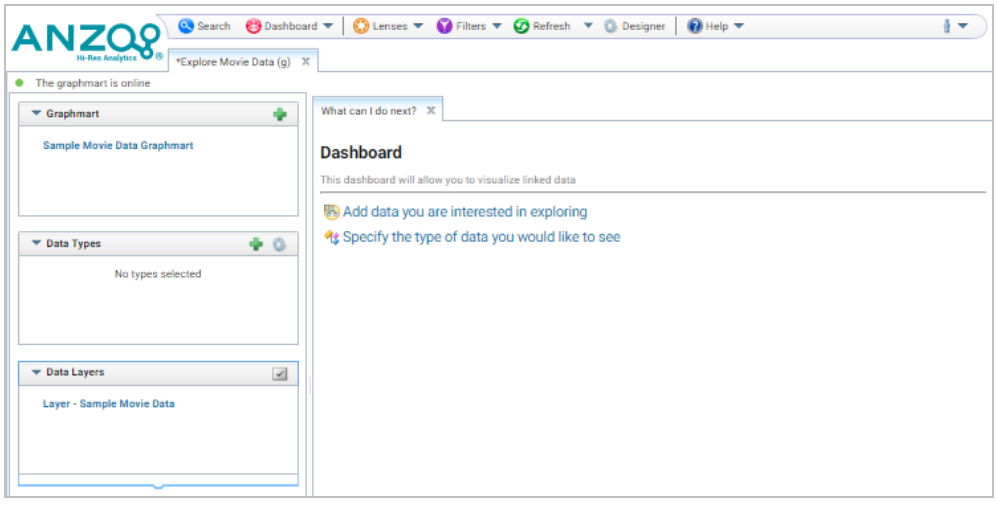
1. On the Graphmart screen for the sample movie data graphmart, click the **Create Dashboard** button. Anzo opens the Hi-Res Analytics application and displays the New Dashboard dialog box:



2. On the New Dashboard dialog box, type a **Title** for the dashboard and an optional **Description**. Leave the default **Graphmart dashboard** value in the **Type** field. For example:

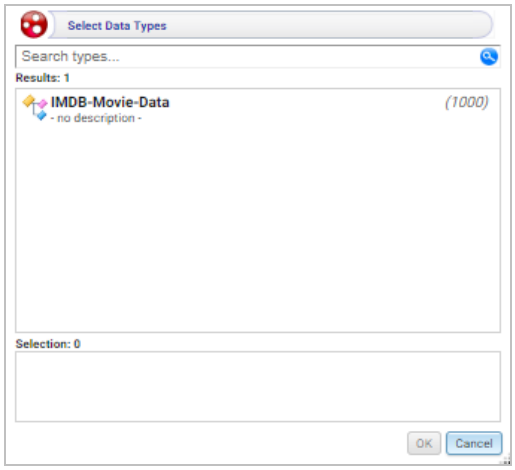


3. Click **OK** to create the dashboard. The new dashboard appears as a new tab on the screen and contains a sub-tab titled **What can I do next?**. This tab acts as a wizard to guide you through the initial dashboard creation.



Anzo populates the Graphmart and Data Layers panels with the sample movie data graphmart and default data layer for the graphmart.

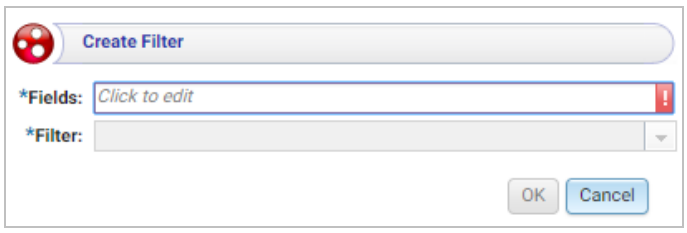
4. To define the type of data from the graphmart to display on the dashboard, click the **Specify the type of data you would like to see** link on the What can I do next tab. The Select Data Types dialog box appears and displays the available data types. Since the sample data set has one type or class of data, only IMDB-Movie-Data is listed. The value in parentheses shows the total number of instances of that type.



5. Click **IMDB-Movie-Data** to select it, and then click **OK** to add the data type to the dashboard.
6. In the main Hi-Res Analytics toolbar, click the **Dashboard** button and select **Save** to save the dashboard.

   To start to explore the data and see what values exist for the properties in the IMDB-Movie-Data class, it can help to add a filter to the dashboard. Filters reveal the values associated with properties. Learning more about the values enables you to start making decisions about what properties to group on, for example, what properties have relationships, and what results you want to visualize.

7. To create a filter, click the **Add facets to filter the data** link on the What can I do next tab. Anzo displays the Create Filter dialog box:

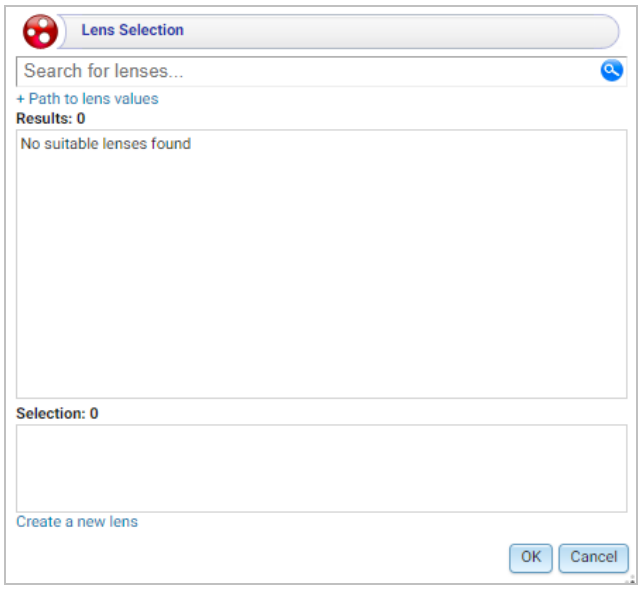


8. In the Create Filter dialog box, click the **Fields** field and browse the available properties. For this tutorial, select **Year**, and then click **Close** to close the Fields drop-down list.
9. Click the **Filter** drop-down list and browse the available filter types. For this tutorial, select **Single Select List**. Anzo adds the filter type to the Create Filter dialog box.
10. At the bottom of the screen under **Format**, click the **Type** drop-down list and select **No format**. Removing formatting from the filter ensures that commas are not displayed for the Year values.

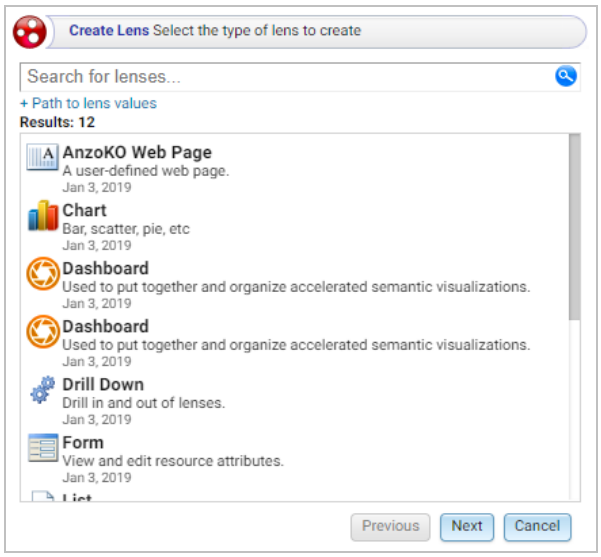
11. Click **OK** to close the dialog box and add the new filter to the dashboard. Anzo adds the Year filter to the left panel. The new filter will enable you to narrow the scope of the data when it is visualized.

Once you have a good understanding of the values and relationships that exist in the data set, you can experiment with lenses and decide on the most appropriate way to display the data. Creating a Table lens is a quick way to view the data that you filtered.

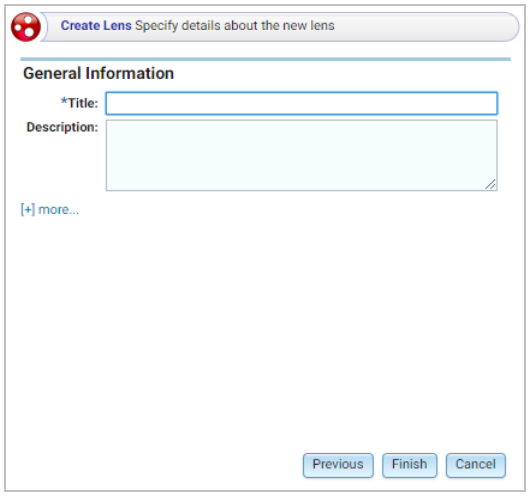
12. To create a lens, click the **Select or create visualizations of your data** link on the What can I do next tab. Anzo displays the Lens Selection dialog box. Since lenses have not been created previously, the list of lenses is empty:



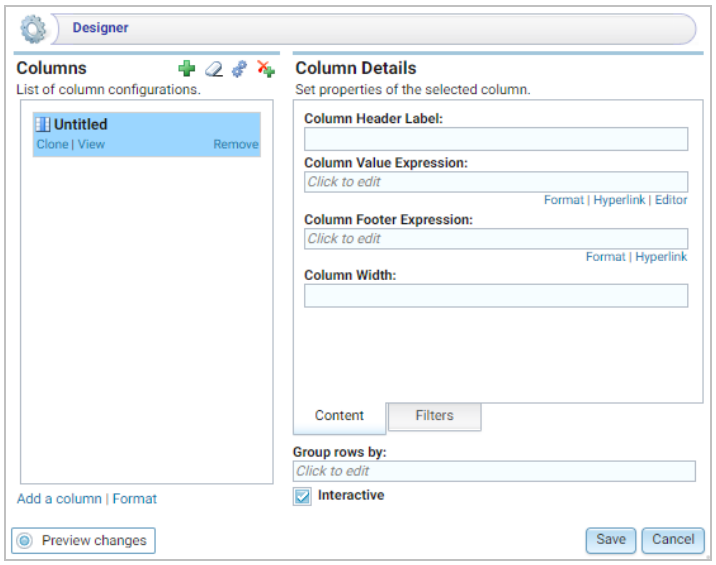
13. In the Create Lens dialog box, click the **Create a new lens** link at the bottom of the screen. Anzo displays the Create Lens dialog box:



14. In the list of lens types, select the **Table** lens and then click **Next**. Anzo displays the General Information dialog box:



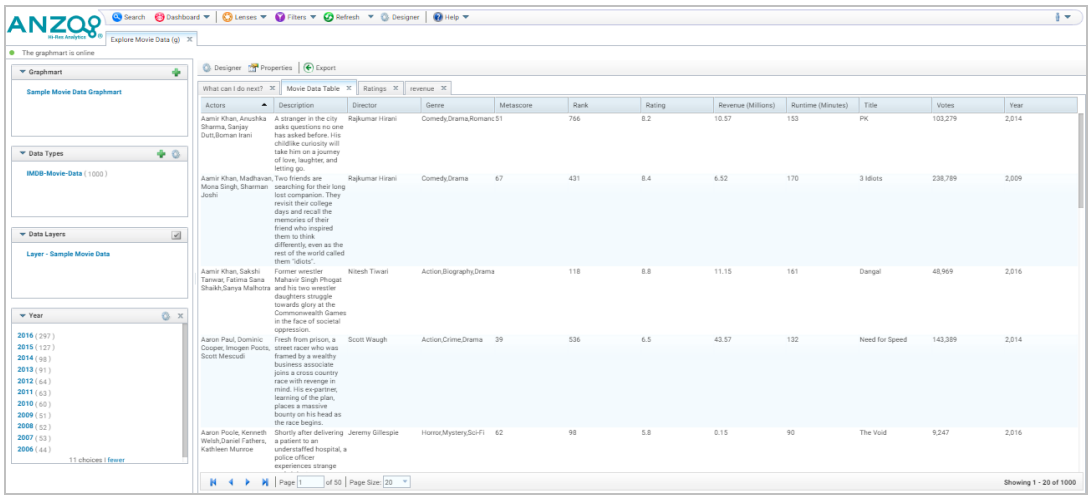
15. Type a **Title** for the lens, and then click **Finish**. Anzo adds the lens as a new sub-tab on the dashboard and opens the Table Designer:



16. In the Designer, click the **Auto-generate columns** icon ( ) to add all available columns to the table.

    Note: Since the sample data set includes only 12 columns, it is not overwhelming to view all columns at once. For larger data sets, you might want to be more selective when adding columns to table lenses.

17. Click **Save**. The table lens displays on the dashboard and populates with data. For example:

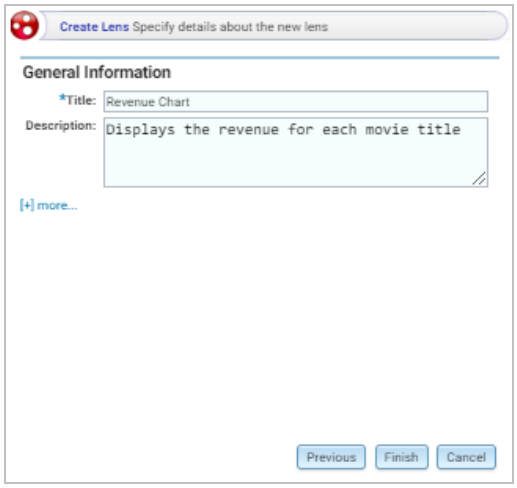


    You can reduce the number of results that are displayed by clicking a year value in the Year filter.

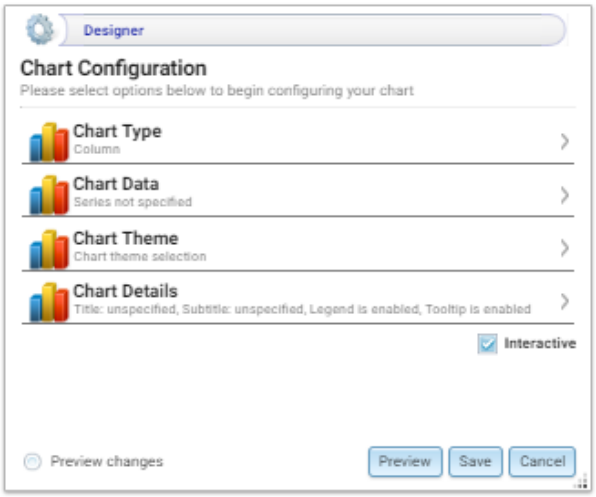
18. In the main Hi-Res Analytics toolbar, click the **Dashboard** button and select **Save** to save the changes. Remember to save dashboards periodically.

    Now that you can view a summary of the data in a table, it can help you determine how to further narrow or expand the results by adding, changing, or removing filters. In addition, you can experiment by adding other lenses to the dashboard to find the ideal way to display the data to answer the questions that you have. For example, the table includes a Revenue column. It might be interesting to see which movies made the most revenue. The next steps guide you through creating a chart lens to display movie revenue.

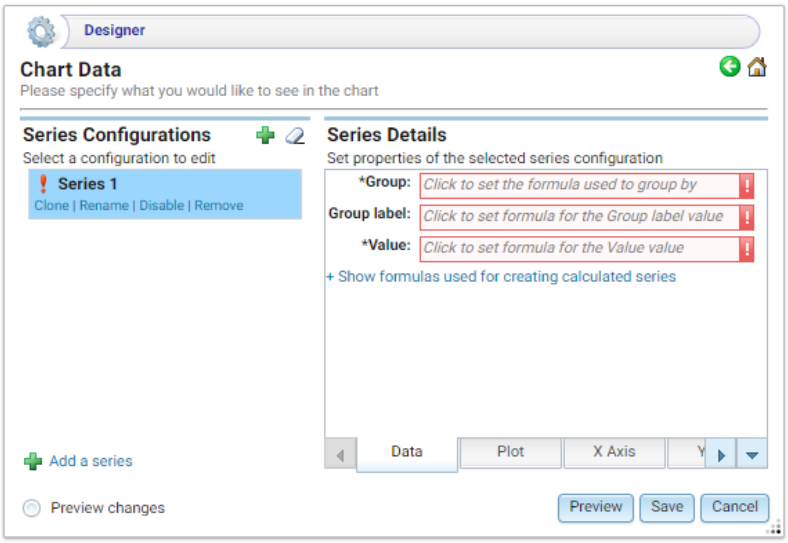
19. In the Hi-Res Analytics main toolbar, click **Lenses** and select **New**. The Create Lens dialog box opens.
20. Select **Chart** and click **Next**. Anzo displays the General Information dialog box.
21. Type a **Title** and optional **Description** for the new lens. For example:



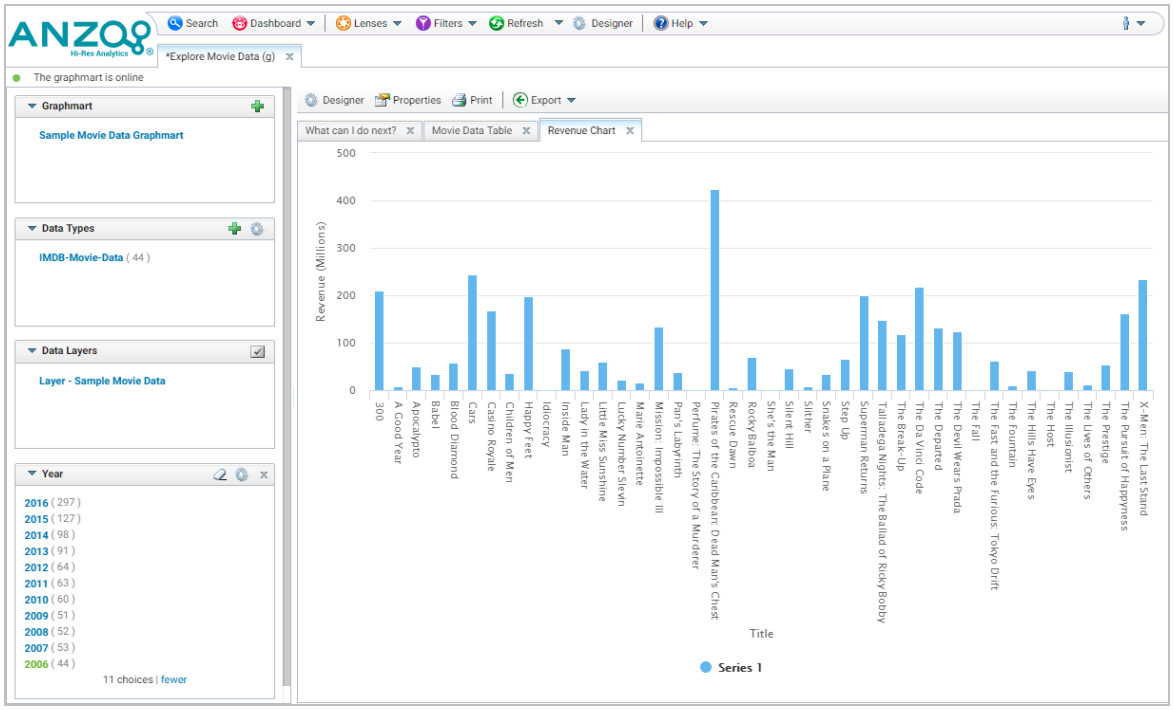
22. Click **Finish**. Anzo displays the Chart Designer, which enables you to configure the type of chart to create, such as column, pie, or line, specify the data that will populate the chart, and customize details such as the chart's theme and fonts, legend, title, and tooltip formats.



23. For this tutorial, accept the default Chart Type of "Column," and click **Chart Data** to view the Chart Data screen and configure the data to display.

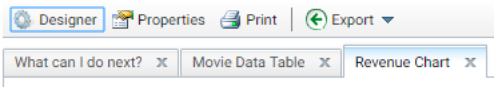


24. Click the **Group** field and select the **Title** property from the drop-down list. The Group field sets the values for the x-axis on the column chart. Anzo also populates the Group label with the Title property. Click **Close** to close the drop-down list.
25. Click the **Value** field and select the **Revenue (Millions)** property from the drop-down list. The Value field sets the y-axis values. Click **Close** to close the drop-down list.
26. Click **Save**. Anzo adds the new chart lens to the dashboard. To narrow the results that are displayed and filter on one year at a time, click a year in the Year filter. For example, by clicking **2006** in the filter, the chart lists the revenue for titles that were released in 2006 and shows that "Pirates of the Caribbean: Dead Man's Chest" had the most revenue for that year:

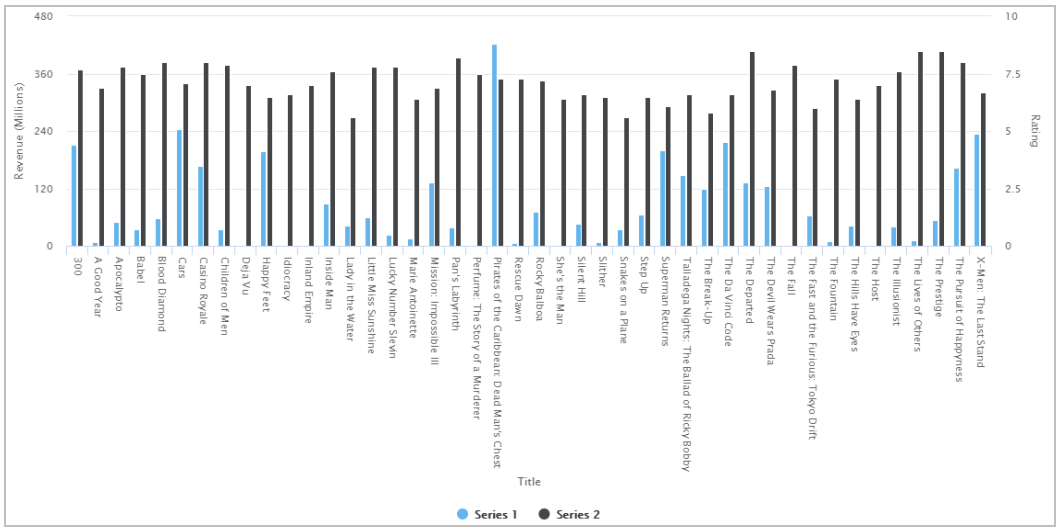


You can hover the pointer over a bar in the chart to view details.

27. If you want to add more data to the chart, such as to compare revenue versus rating for each title, follow these steps:
    a. Open the chart designer by clicking the **Designer** button above the lens tabs.



    b. In the Designer, click **Chart Data**.
    c. On the Chart Data screen next to Series Configurations, click the plus icon ( ) to add a new series.
    d. For **Series 2**, click the **Group** field and select the **Title** property from the drop-down list. Then click **Close** to close the list.
    e. Click the **Value** field and select the **Rating** property from the drop-down list. Close the list.
    f. Click **Save** to save the chart changes. The chart updates to display the Rating values in addition to the Revenue. For example:



    For more information about working with Hi-Res Analytics, see Analyzing Data with Hi-Res Analytics.

Review the New Solution's Components

This section briefly reviews the components that were created as part of the new sample solution and provides references to more details about each of the components.

During the process of automatically ingesting the sample movie data from the supplied CSV file, Anzo generated the following components:

- A data model, which describes the structure and type of data that the movie data set contains. For more information and instructions for viewing the derived model, see Modeling Data.

- A mapping, which is input to the ETL job and maps the source data to the target elements defined in the data model. For more information and instructions for viewing the generated mapping, see Working with Mappings.
- An ETL pipeline, which contains the ETL job that generates the target data set. For more information and instructions for viewing the generated pipeline, see Working with Pipelines.

Related Topics

Introduction to the User Interface

Connecting to a File Store

Creating an Anzo Data Store

Making a Basic Connection to AnzoGraph

User Guide

The User Guide provides usage information for all of the Anzo components. For an introduction to the user interface, basic setup steps, and instructions for building a sample solution from scratch, see the Getting Started Guide.

- Onboarding Structured Data
- Onboarding Unstructured Data
- Modeling Data
- Blending Data
- Accessing and Analyzing Data
- Exploring Data Provenance
- Versioning and Migrating Entities and Metadata
- Graph Data Storage Architecture

Onboarding Structured Data

Structured data sources such as relational databases or flat files are onboarded to Anzo using Anzo's built-in pipelines. These pipelines natively support CSV, JSON, XML, SAS, and Parquet files, along with all common database connections, including SQL, Oracle, MySQL, HIVE, and others.

The topics in this section provide instructions for connecting to and importing data from structured data sources, ingesting the data, and working with schemas, mappings, and pipelines.

For information about working with unstructured data, see Onboarding Unstructured Data.

- Creating a Database Data Source
- Importing Data from CSV Files
- Importing Data from JSON Files
- Importing Data from XML Files
- Importing Data from SAS Files
- Importing Data from RDF Files
- Generating Source Data Metrics
- Using Data Dictionaries
- Ingesting Data into Anzo
- Working with Schemas
- Working with Mappings
- Working with Pipelines

Creating a Database Data Source

The topics in this section provide instructions for connecting to a structured data source, such as a Microsoft, Oracle, Hadoop, Teradata, PostgreSQL, or Google database, and defining the schema to use for onboarding the data.

- Performance Considerations for Database Pipelines
- Connecting to a Database
- Defining a Database Schema
- Partitioning a Database Table for Parallel Ingestion

Performance Considerations for Database Pipelines

This topic highlights performance-related information that is helpful to consider when setting up an onboarding pipeline for a database data source.

Take Advantage of the Source Database

Onboarding data from a database involves two systems, the source database and the Spark infrastructure. The way that you configure the pipeline's schema and mappings controls which system performs some of the time-consuming

operations such as joining and filtering the data. In short, schema operations are processed by the source database, and mapping transformations are processed by Spark. Maximizing the use of the source database to join and filter data can have a significant impact on the overall performance of the ETL pipeline.

Use Schema Queries to Join and Filter Data

When defining the schema for a database source, you have the option to write SQL queries to create the schema tables. If join and/or filter operations are required, consider writing schema queries that perform those operations (see Creating a Schema from an SQL Query for more information). Since the source database runs the schema queries and then sends the filtered result set to Spark, Spark has fewer operations to perform when publishing the ETL pipeline.

Alternatively, if the schema selects all of the source data and joins or filters are configured at the mapping level, the source database sends the entire result set to Spark and Spark performs the join and filter operations when publishing the pipeline.

In general, databases perform join and filter operations much faster than Spark. And Cambridge Semantics recommends that you incorporate joins and filters in schema queries when possible, rather than transforming the data downstream in the mappings that Spark processes.

Related Topics

Connecting to a Database

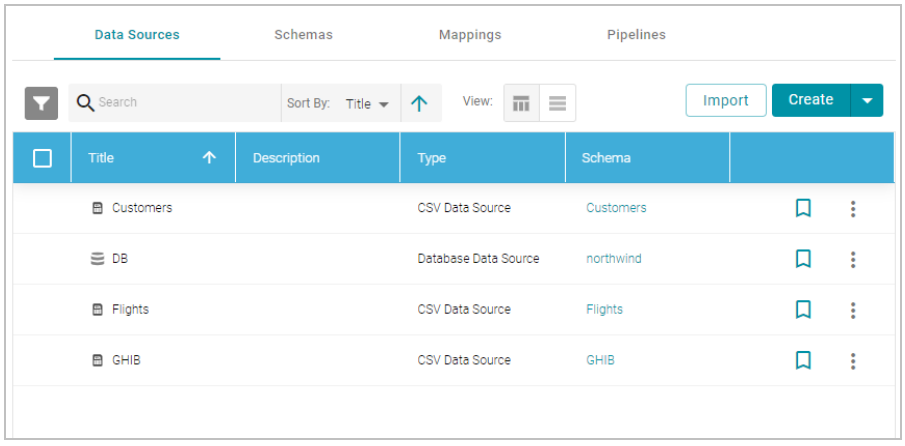
Defining a Database Schema

Partitioning a Database Table for Parallel Ingestion

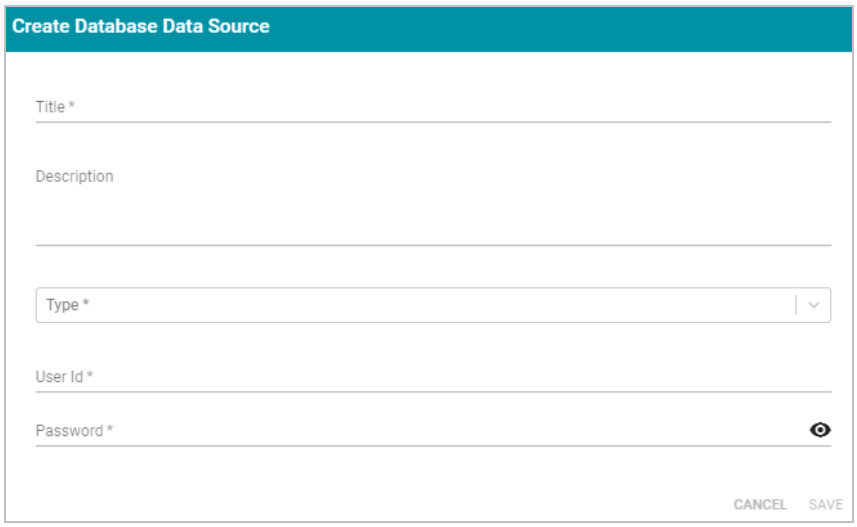
Connecting to a Database

This topic provides instructions for connecting to a structured data source, such as a Microsoft, Oracle, Hadoop, Teradata, PostgreSQL, or Google database.

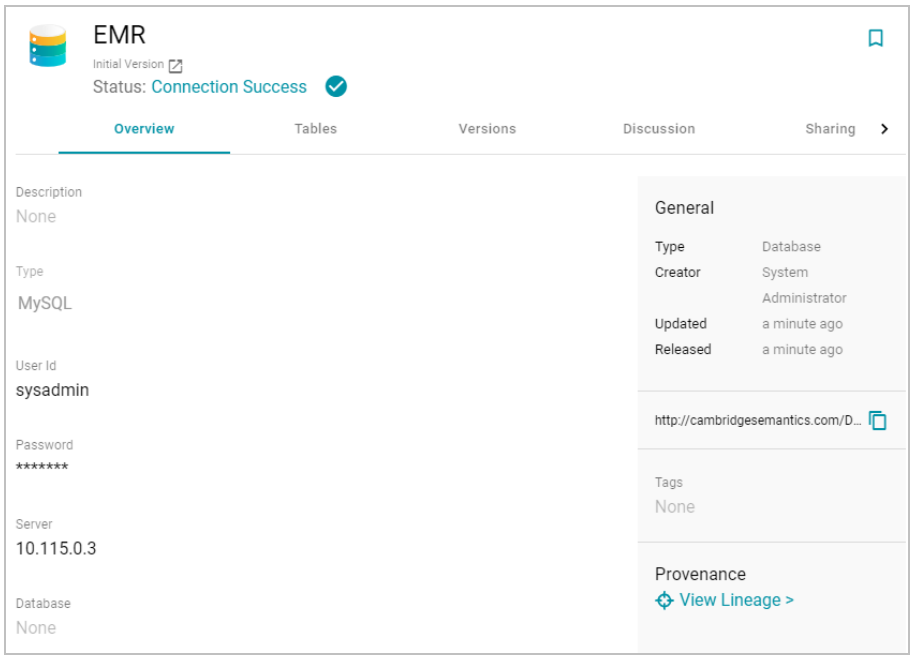
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



2. Click the **Create** button and select **Database Data Source**. Anzo opens the Create Database Data Source screen.



3. At the top of the screen, type a **Title** for the source.
4. Type an optional **Description** for the source.
5. Click the **Type** field and select the database type from the drop-down list. Depending on the type you choose, Anzo displays additional fields to complete.
6. Enter any additional details and the credentials that are required for making the source connection. The options that appear depend on the type of database that you chose:
    - **User**: Type the user name used to log in to the database.
    - **Password** and **Password Repeat**: Type the password for the user.
    - **Server**: Type the server name or IP address for the source. Include the port if necessary.
    - **Database**: If necessary, type the partition that contains the data.
    - **Extended Properties**: For Hadoop Hive or Impala databases, enter the extended attributes that you use.
7. Click **Save** to save the data source connection. Anzo tests the connectivity and displays the Overview screen. If the connection fails, adjust the data source details as needed.



After connecting to the data source, the next step is to define the schema that Anzo will use to determine the data's structure and import the data. See Defining a Database Schema for instructions.

Related Topics

Performance Considerations for Database Pipelines

Defining a Database Schema

Partitioning a Database Table for Parallel Ingestion

Defining a Database Schema

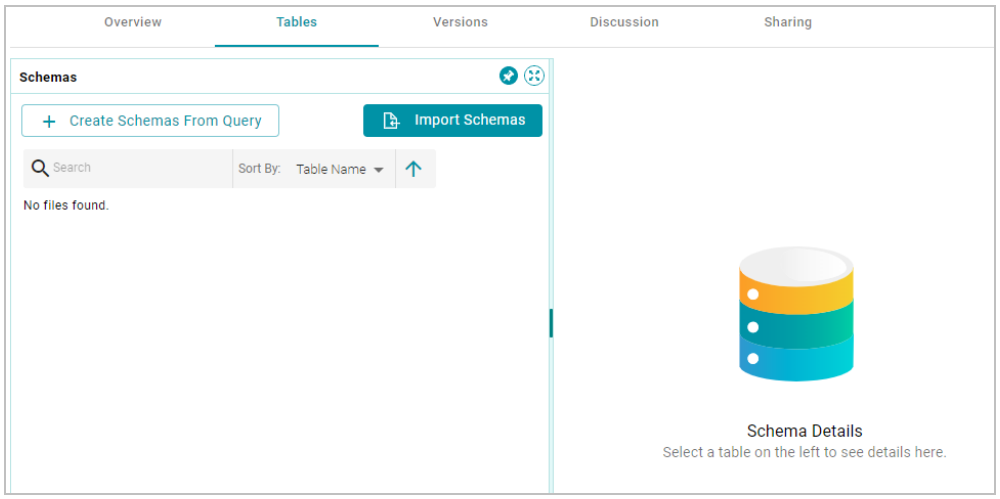
This topic provides information about creating the schema to use when importing data from a database. The schema defines the source data to onboard. Anzo supports multiple options for defining the schema. You can import a schema from the database, you can write a static SQL query that defines the data, or, if you want to import data incrementally, you can write an incremental SQL query that includes parameters that automatically increment when the ETL pipeline is run. Select an option from the list below for instructions on creating that type of schema:

- Importing a Predefined Schema
- Creating a Schema from an SQL Query
- Creating an Incremental Schema

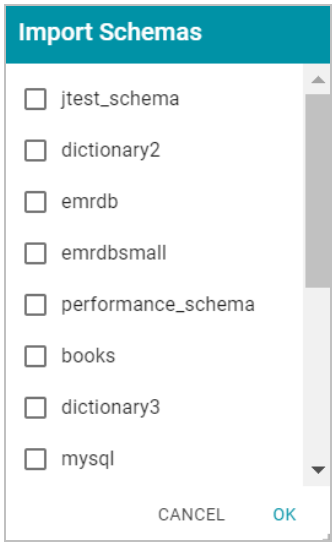
Importing a Predefined Schema

Follow the steps below to import a predefined schema from the database to Anzo. For instructions on writing a schema query, see Creating a Schema from an SQL Query below.

1. From the Overview screen (as shown in the last step of the procedure above), click the **Tables** tab. Anzo displays the Tables screen. For example:

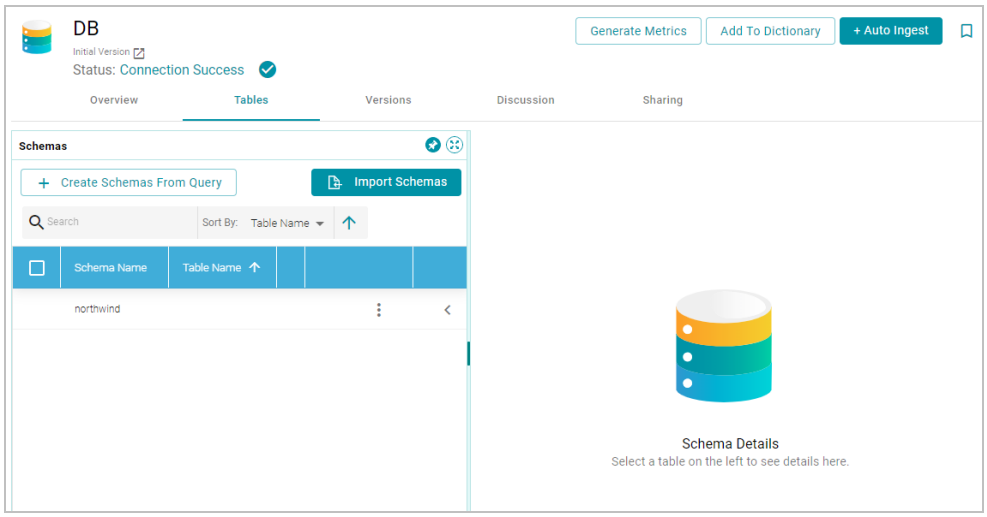


2. Click the **Import Schemas** button. Anzo displays the Import Schemas dialog box, which lists any predefined schemas in the database. For example:



   If you do not see a schema that you expect to see, make sure that you have the necessary access to the data source.

3. Select the checkbox next to each schema that you want to import, and then click **OK**. Anzo imports the selected schema or schemas and lists the imported schemas on the Tables screen. For example:



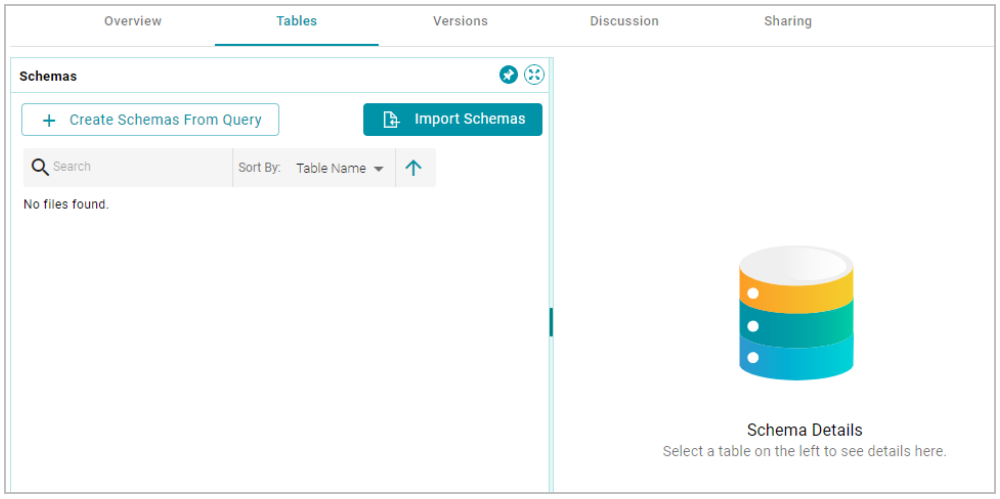
Once the schema or schemas are imported, the source data can be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo.

Creating a Schema from an SQL Query

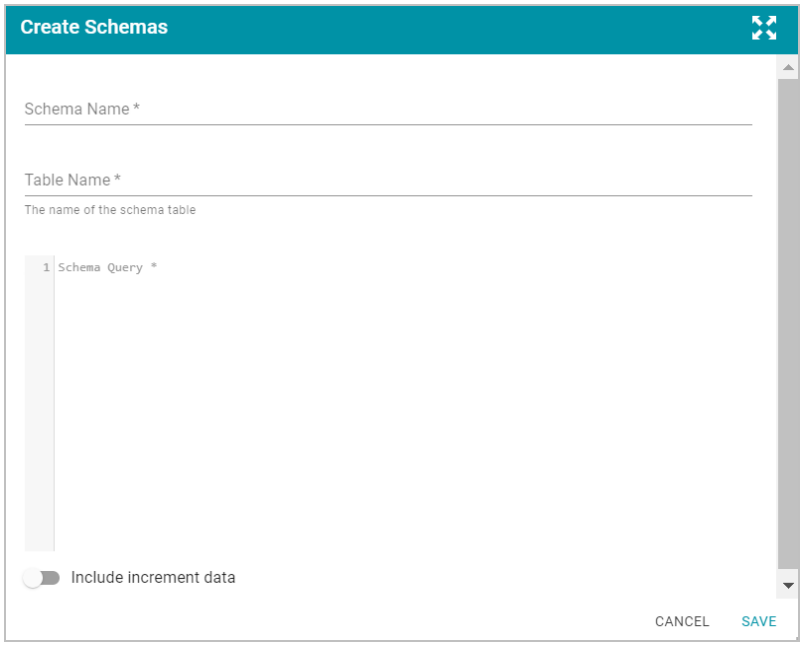
Follow the instructions below to create a schema by writing an SQL query that defines the data to onboard.

Tip For better ETL pipeline performance, it is beneficial to include joins and/or filters in schema queries rather than configuring those operations at the mapping level. For more information, see Performance Considerations for Database Pipelines.

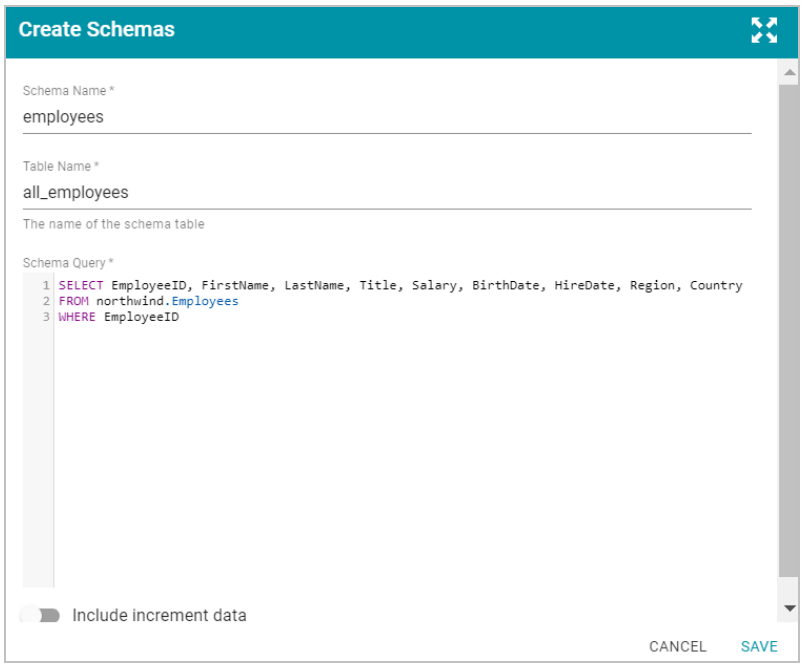
1. From the Overview screen (as shown in the last step of Connecting to a Database), click the **Tables** tab. Anzo displays the Tables screen. For example:



2. Click the **Create Schemas From Query** button. Anzo displays the Create Schemas dialog box:

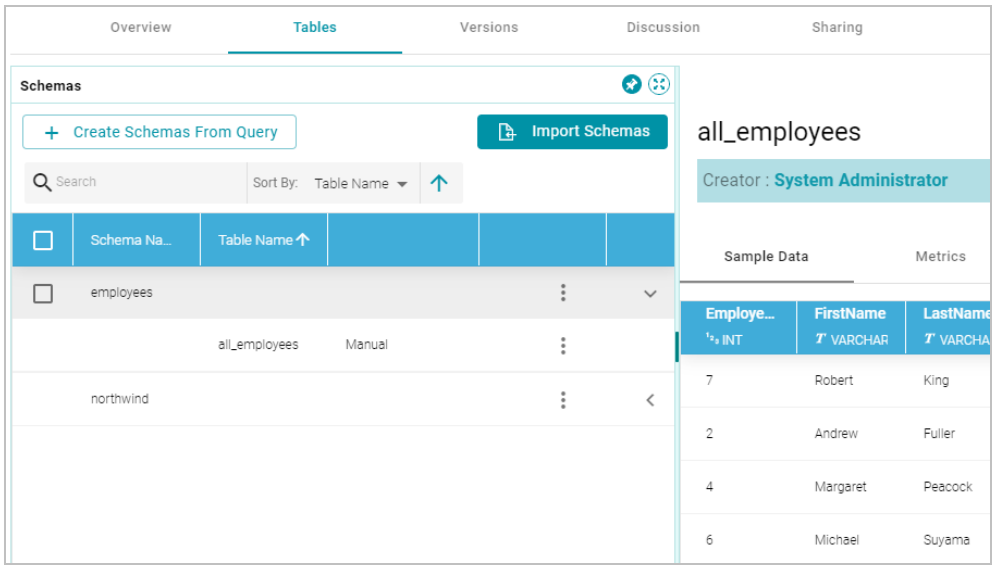


3. In the Create Schemas dialog box, specify a name for the schema in the **Schema Name** field.
4. In the **Table Name** field, specify a name for the schema table that the query will create.
5. Type the SQL statement in the text box. The statement can include any functionality that the source database supports. Anzo does not validate the SQL. For example:



For information about writing a schema query that onboards data from a database incrementally, see Creating an Incremental Schema below.

6. Click **Save** to save the query. Anzo creates the new schema and adds it to the list of schemas on the Tables screen. For example:



Once the schema or schemas are created, the source data can be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo.

Creating an Incremental Schema

Anzo provides the option to create a schema that is configured to import only the data that has been added to the data source since the last time the data was onboarded. This section provides instructions for writing a schema query to import data incrementally and gives an overview of the workflow to onboard the data.

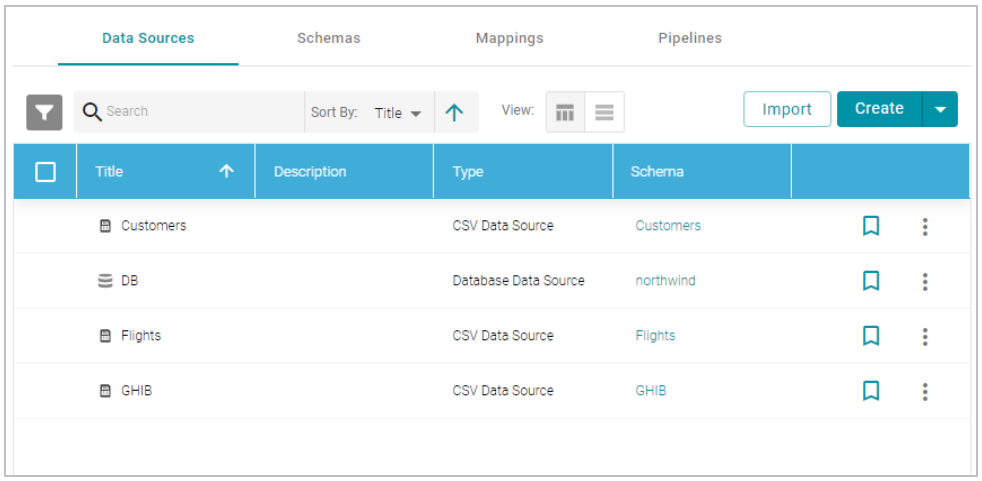
Important Anzo onboards new source data only; it does not process data that was updated or deleted in the source database. Running the ETL job for an incremental schema query replaces the existing data with the newly onboarded data. What happens to the existing data in an FLDS when I run an incremental ETL job?

- Writing an Incremental Schema Query
- Running an Incremental Pipeline

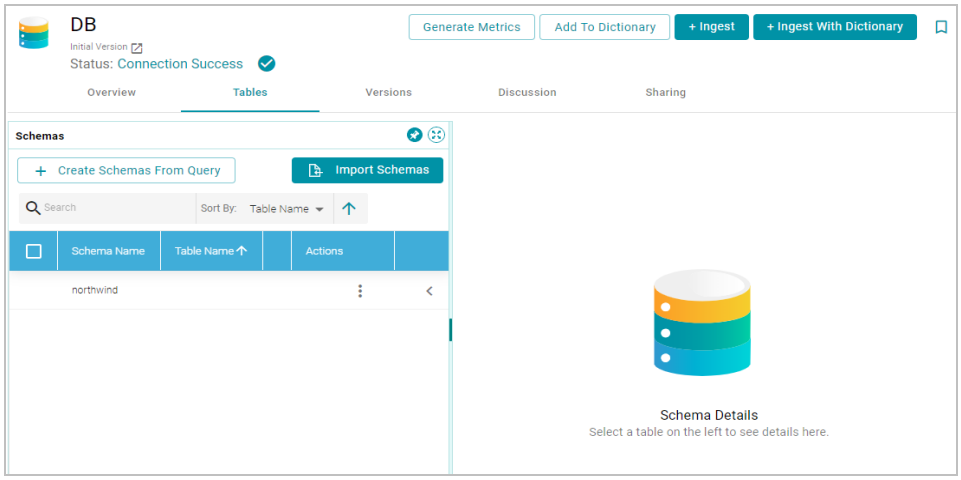
Writing an Incremental Schema Query

Tip For better ETL pipeline performance, it is beneficial to include joins and/or filters in schema queries rather than configuring those operations at the mapping level. For more information, see Performance Considerations for Database Pipelines.

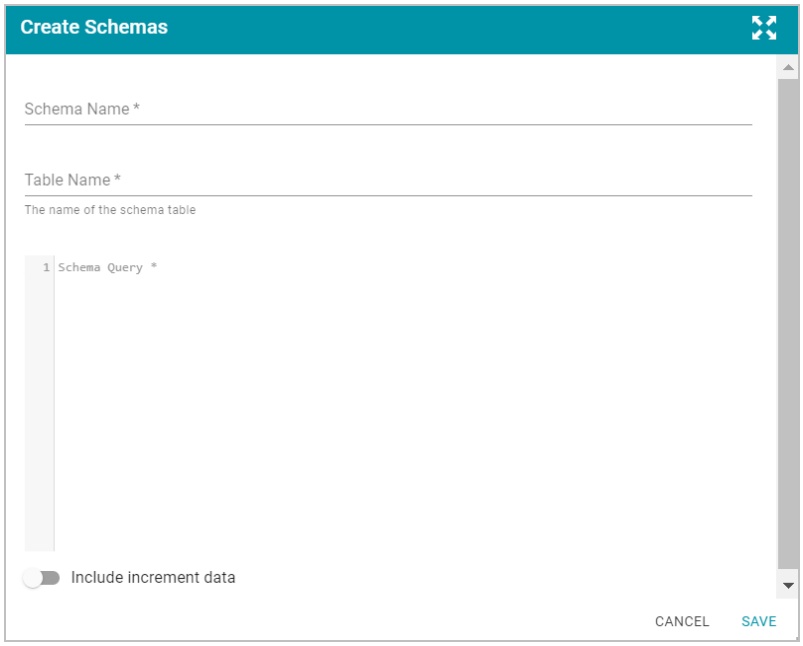
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



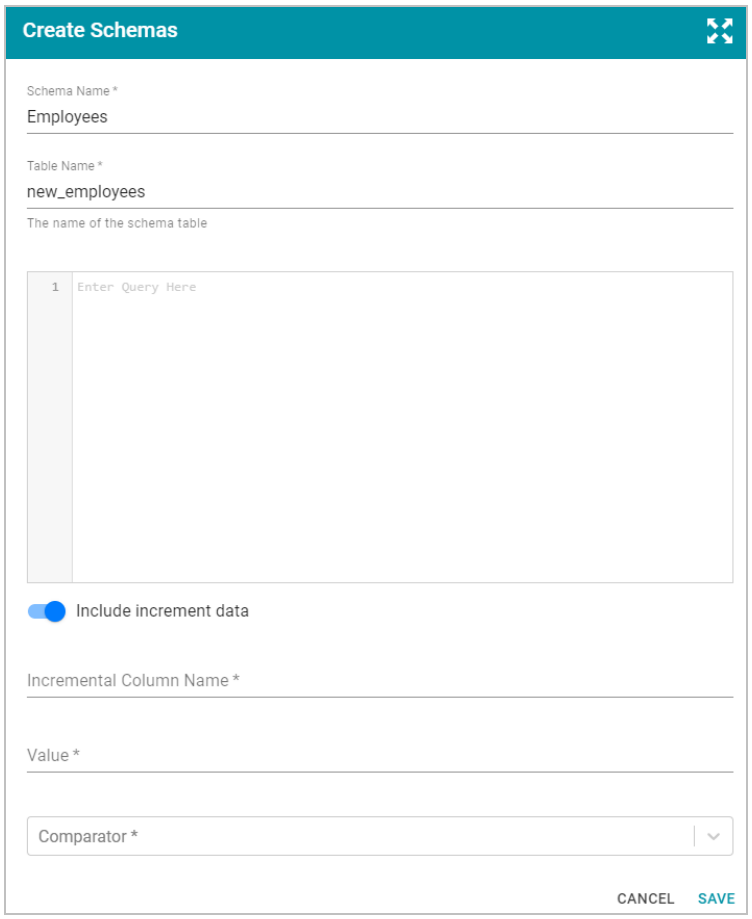
2. On the Data Sources screen, click the database data source for which you want to create an incremental schema. Anzo displays the Tables screen for that source. For example:



3. Click the **Create Schemas From Query** button. Anzo displays the Create Schemas dialog box:



4. In the Create Schemas dialog box, specify a name for the schema in the **Schema Name** field.
5. In the **Table Name** field, specify a name for the schema table that the query will create.
6. At the bottom of the screen, enable the **Include increment data** option by sliding the slider to the right. Anzo displays additional settings. For example:



7. Populate the following fields so that you can use the values as a guide for writing the schema query:
   - **Incremental Column Name**: The source column whose value will be used to increment the data.
   - **Value**: The value in the column to use as the stopping point for the current import process and the starting point for the next import.

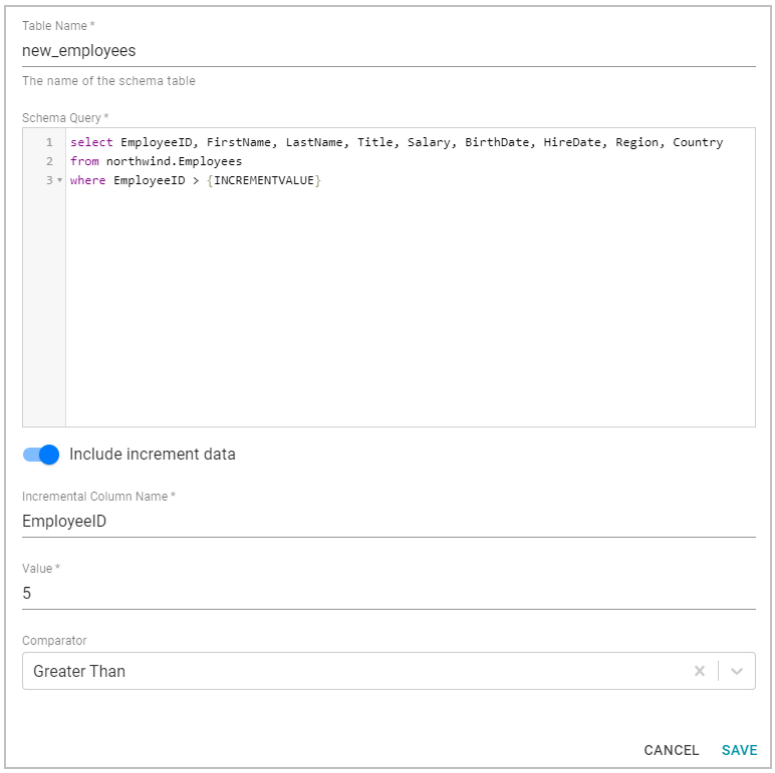
     Note Do not include quote characters in the Value field. If the SQL query requires quotes around values, such as '2010-01-01' or 'TestValue', include the quotes around the {INCREMENTVALUE} parameter in the query and not in the Value field. For example, if the value to increment on is '2010-01-01', specify **2010-01-01** in the Value field and add the quotes to the query like the following example:

     ```
     SELECT * FROM Orders WHERE OrderData > '{INCREMENTVALUE}'
     ```

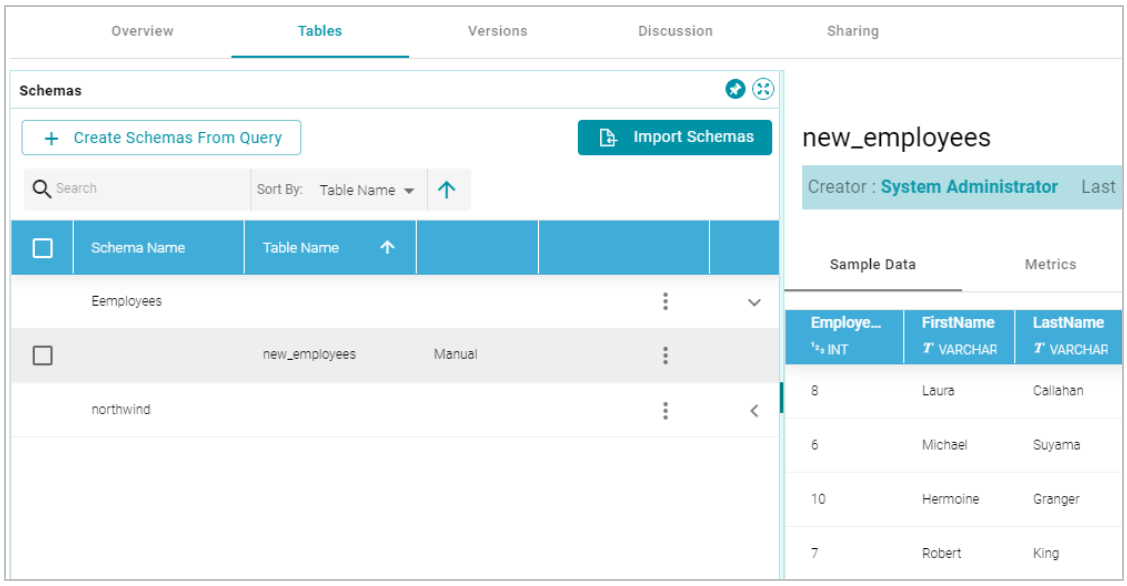
   - **Comparator**: The operator to use for comparing source values against the value above.
8. In the query text field, type the SQL statement that will target the appropriate source data. The WHERE clause must include the incremental column name, the comparison operator, and an {INCREMENTVALUE} parameter that is substituted with the **Value** at runtime. For example:

```
SELECT EmployeeID, FirstName, LastName, Title, Salary, BirthDate, HireDate,
Region, Country
FROM northwind.Employees
WHERE EmployeeID > {INCREMENTVALUE}
```

Make sure that the query includes the INCREMENTVALUE parameter and uses the same Incremental Column Name and Comparator values as the settings. For example:



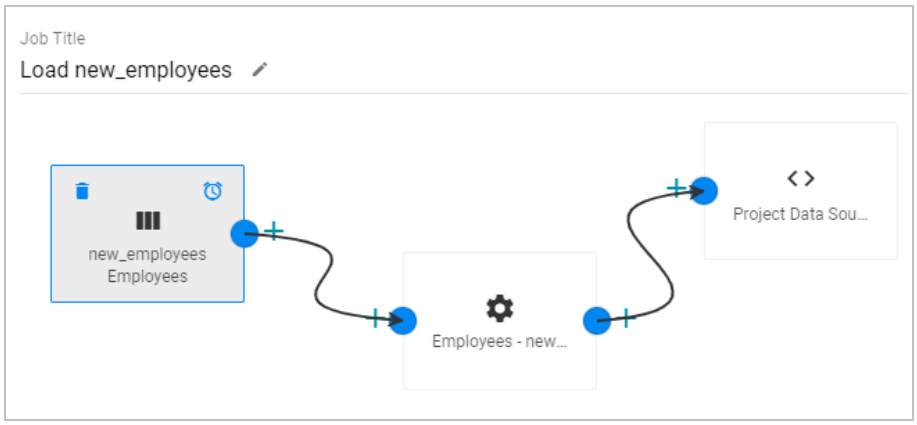
9. Click **Save** to save the query. Anzo creates the new schema and adds it to the list of schemas on the screen. For example:



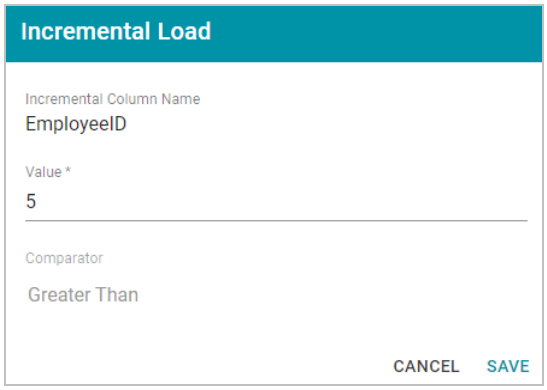
Once the schema or schemas are created, the source data can be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo. See Running an Incremental Pipeline below for important information about running a pipeline that includes an incremental schema.

Running an Incremental Pipeline

When an incremental schema is added to an ETL job, a clock icon ( ) is displayed when hovering over the component in the job. For example:

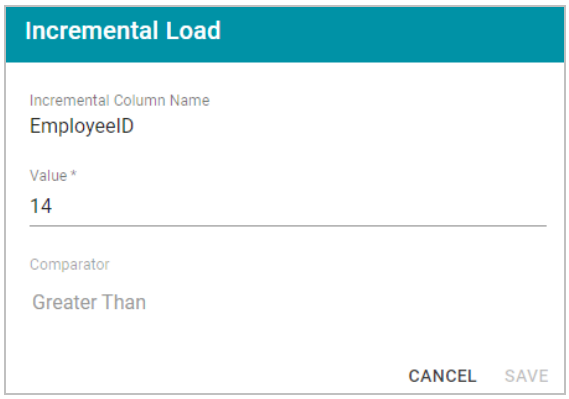


Clicking the clock icon opens the Incremental Load dialog box, which lists the Incremental Column Name, Value, and Comparator from the schema query. For example:



Publishing the job for this example will onboard only the records for which the EmployeeID is greater than 5. When the job is finished, Anzo adjusts the incremental load value to list the last value that was onboarded for the incremental column. Every time the pipeline is published, Anzo changes the incremental load value parameter to the highest or lowest value for the column, depending on the Comparator.

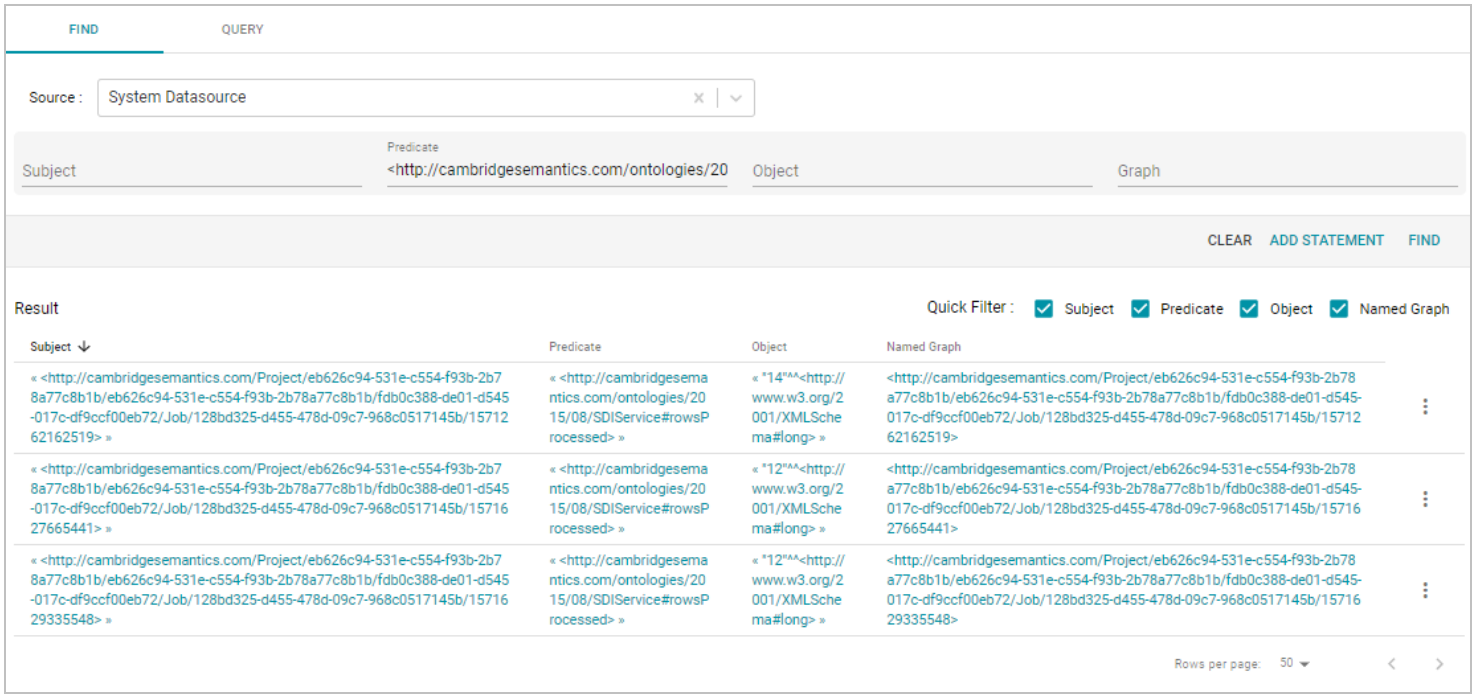
For example, viewing the Incremental Load dialog box after running the job above shows that the last EmployeeID value that was onboarded was 14:



The next time this job is run, Anzo will onboard only the records where EmployeeID is greater than 14. To view the number of rows processed after running a job, you can search the System Datasource for the following predicate on the Find tab in the Query Builder:

```
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#rowsProcessed>
```

The Object column shows the number of rows processed each time the pipeline was run. For example:



Related Topics

Performance Considerations for Database Pipelines

Connecting to a Database

Partitioning a Database Table for Parallel Ingestion

Ingesting Data into Anzo

Partitioning a Database Table for Parallel Ingestion

When you ingest data from a database, Anzo creates one ETL job for each table in the schema. When there are multiple jobs in a pipeline, Spark processes the jobs in parallel, one job per executor. If the source has a very large table, however, and one job ingests all of the data for that table, overall pipeline performance can slow down because one Spark executor processes all of the data from that table. To take advantage of parallel ingestion if a data source has one or more large tables, you can use Anzo Semantic Service calls to partition the tables. The resulting ETL job for a partitioned table has smaller sections that can be ingested in parallel by multiple executors.

This topic provides instructions on using the Anzo command line interface to compute a partition and assign the partition to a table so that Anzo can leverage the information during ingestion.

Computing and Assigning Partitions to a Table

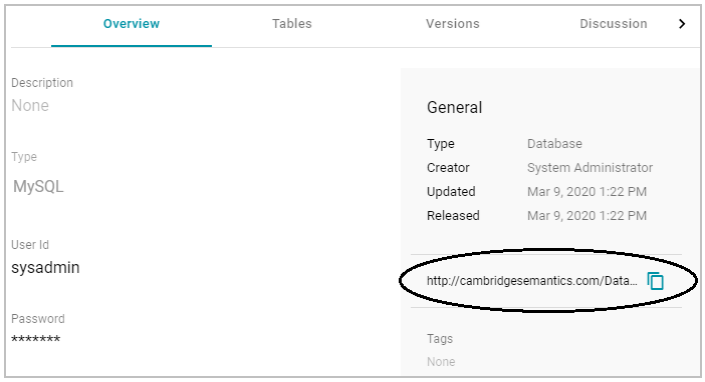
When creating a partition for a table, choose a column with an integer data type to partition on. You add metadata to that column to define the size and number of partitions, and then you call an Anzo service that computes the predicates for the partition. Once the predicates are computed, you call another service to assign the partitions to the table so that Anzo can apply the partitions when generating the ETL job. The steps below guide you through computing and assigning partitions.

Note When you supply the metadata for computing partitions, you will need to know the row count for the table that will be partitioned. Calculating the row count in Anzo requires generating statistics on the schema. You might want to generate statistics in advance before starting the steps below. For instructions, see Generating Source Data Metrics.

1. First, view the metadata for the data source so that you can retrieve the URI for the schema that contains the table to partition. Run the following command to return the data source metadata:

   ```
   anzo get data_source_uri
   ```

   The data source URI can be found on the Overview tab for the data source.

   For example:

   ```
   anzo get
   http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4
   ```

   Anzo returns the metadata for the data source.

2. In the data source metadata results, look for the schema URI for which you want to create a partition. The schema URI is the object of a triple that follows the pattern below:

   ```
   data_source_uri <http://cambridgesemantics.com/ontologies/DataSources#dbSchema>
   schema_uri
   ```

   For example, the URI below identifies the **northwind** schema:

   ```
   data_source_uri <http://cambridgesemantics.com/ontologies/DataSources#dbSchema>
     <
   http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sch
   ema/northwind>
   ```

3. Using the schema URI from the previous step, run the following command to view the metadata for the schema:

   ```
   anzo get schema_uri
   ```

For example:

```
anzo get
http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sch
ema/northwind
```

Anzo returns the metadata for the schema.

4. In the schema metadata results, find the URI for the table that you want to partition. The table URI is the object of a triple that follows the pattern below:

```
schema_uri <http://cambridgesemantics.com/ontologies/DataSources#schemaTable> table_
uri
```

For example, the URI below identifies the **ORDERS** table in the northwind schema:

```
schema_uri <http://cambridgesemantics.com/ontologies/DataSources#schemaTable>
  <
http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sch
ema/northwind/ORDERS>
```

5. Next, identify the URI for the column that you want to use for computing the partitions. The column that you choose should have an integer data type. You can view the column URIs as well as metadata for the columns in the output of the previous step, or you can run the following command to narrow the results to the list of columns for the table. This command finds all of the results for which the table URI is the subject:

```
anzo find -sub table_uri
```

For example:

```
anzo find -sub
http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sch
ema/northwind/ORDERS
```

The column URIs are the object of a triple that follows the pattern below:

```
table_uri <http://cambridgesemantics.com/ontologies/DataSources#tableColumn> column_
uri
```

For example, the URI below identifies the ORDERID column in the ORDERS table:

```
table_uri <http://cambridgesemantics.com/ontologies/DataSources#tableColumn>
  <
http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sch
ema/northwind/ORDERS/ORDERID>
```

6. Once you retrieve the column URI, create a .trig file that includes the metadata for the column. You will add new partition properties to the file. Run the following command to output a .trig file that contains the column metadata:

```
anzo find -sub column_uri --output-file /path/filename.trig
```

For example, the following command retrieves all of the results for which ORDERID is the subject. It outputs the results to a file called ComputePartitions.trig in the current directory:

```
anzo find -sub
http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sch
ema/northwind/ORDERS/ORDERID
  --output-file ComputePartitions.trig
```

The output below shows the contents of the resulting ComputePartitions.trig file.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .


<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sc
hema/northwind> {

<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sc
hema/northwind/ORDERS/ORDERID>
               a <http://cambridgesemantics.com/ontologies/DataSources#Column> ,
       <http://cambridgesemantics.com/ontologies/DataSources#DataField> ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnAutoIncrement>
"false"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnCaseSensitive>
"false"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnDerivedOwlProperty>
               <http://cambridgesemantics.com/ont/autogen/Fu/DB/northwind#Orders_OrderID> ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnIndex> "1"^^xsd:int ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnJdbcType> "integer" ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnName> "OrderID" ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnNullable>
"false"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnPrimaryKey>
"true"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnRemarks> "" ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnSize> "10"^^xsd:int ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnType> xsd:int ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnTypeName> "INT" .
}
```

7. Modify the .trig file from the previous step to specify the partitioning metadata. The metadata to add includes the number of partitions to create as well as the total number of rows in the data source table. To provide the required metadata, edit the file as follows:
   a. At the top of the file, replace the schema URI with the following service URI:

      ```
      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredicatesRequest>
      ```

      In the example above, `<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Schema/northwind>` is replaced by `<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredicatesRequest>`

   b. Towards the bottom of the file, at the end of the column metadata and inside the ending brace (}), add the following contents:

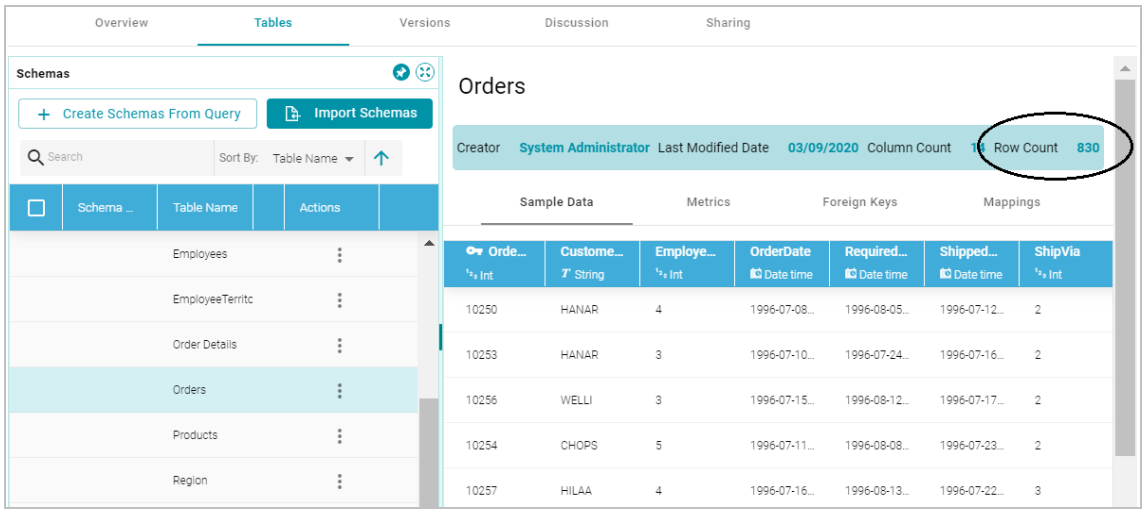
      ```
      # PARTITIONING METADATA

      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredicatesRequest>

      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#numberOfPartitions> "number_of_partitions"^^<http://www.w3.org/2001/XMLSchema#int> ;
      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#numberOfRows>
      "number_of_rows"^^<http://www.w3.org/2001/XMLSchema#long> ;
      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#tableColumn>
      column_uri ;
      a
      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#BaseComputePartitioningPredicatesRequest> ,

      <http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputeColumnBasedPartitioningRequest> .
      ```

   c. In the new triples, replace the placeholders with the appropriate values for your environment:
      - **number_of_partitions**: Specify the number of partitions to create for the table. Choose the value based on the number of Spark nodes or executors that are available. If you do not know the number, **12** is recommended.
      - **number_of_rows**: Specify the total number of rows for the table. After generating source data metrics, you can view the row count by viewing the Tables tab for the schema and clicking the table to show the metrics for that table. For example:



- **column_uri**: The URI for the partition column from step 5. You can copy the URI from the top of the file.

The example below shows the complete ComputePartitions.trig file after completing steps a, b, and c.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .


<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitionin
gPredicatesRequest> {

<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefa
ab4/Schema/northwind/ORDERS/ORDERID>
               a <http://cambridgesemantics.com/ontologies/DataSources#Column> ,
       <http://cambridgesemantics.com/ontologies/DataSources#DataField> ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnAutoIncrement>
"false"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnCaseSensitive>
"false"^^xsd:boolean ;

<http://cambridgesemantics.com/ontologies/DataSources#columnDerivedOwlProperty>
               <http://cambridgesemantics.com/ont/autogen/Fu/DB/northwind#Orders_OrderID> ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnIndex>
"1"^^xsd:int ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnJdbcType>
"integer" ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnName> "OrderID" ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnNullable>
"false"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnPrimaryKey>
```

```
"true"^^xsd:boolean ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnRemarks> "" ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnSize>
"10"^^xsd:int ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnType> xsd:int ;
       <http://cambridgesemantics.com/ontologies/DataSources#columnTypeName> "INT" .
# PARTITIONING METADATA

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitionin
gPredicatesRequest>

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#numberOfPartition
s> "12"^^<http://www.w3.org/2001/XMLSchema#int> ;
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#numberOfRows>
"830"^^<http://www.w3.org/2001/XMLSchema#long> ;
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#tableColumn>

<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefa
ab4/Schema/northwind/ORDERS/ORDERID> ;
a
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#BaseComputePartiti
oningPredicatesRequest> ,

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputeColumnBased
PartitioningRequest> .}
```

8. When the .trig file is complete, save and close the file. It becomes input to the Anzo Compute Column Based Table Partitioning Predicates service. The service returns the response to use to assign the partitions that can be used during ingestion. Run the following command to call the partitioning service:

```
anzo call
http://cambridgesemantics.com/semanticServices/SDIService#computeColumnBasedTablePart
itioningPredicates /path/filename.trig > /path/output_file.trig
```

   Where **filename.trig** is the file from step 7 and **output_file.trig** is the new file to create. For example, the following command calls the partitioning service and saves the response in a file called AssignPartitions.trig in the current directory.

```
anzo call
http://cambridgesemantics.com/semanticServices/SDIService#computeColumnBasedTablePart
itioningPredicates ComputePartitions.trig > AssignPartitions.trig
```

The service returns the list of partition predicates. The number of predicates depends on the number of partitions that were specified in the compute file. For example, a portion of the resulting AssignPartitions.trig file is shown below. You can see the complete file by clicking here.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .


<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredi
catesResponse> {
    _:u0d4a3eab-713f-4dbe-b5ad-da5676d6b721 a
<http://cambridgesemantics.com/ontologies/DataSources#PartitionPredicate> ;
      <http://cambridgesemantics.com/ontologies/DataSources#value>
"<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/S
chema/northwind/ORDERS/ORDERID> >= 69
        &&
<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sc
hema/northwind/ORDERS/ORDERID> < 138" .

    _:u255f046c-73a1-44ef-b446-c5a6126c9cc1 a
<http://cambridgesemantics.com/ontologies/DataSources#PartitionPredicate> ;
      <http://cambridgesemantics.com/ontologies/DataSources#value>
"<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/S
chema/northwind/ORDERS/ORDERID> >= 414
        &&
<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefaab4/Sc
hema/northwind/ORDERS/ORDERID> < 483" .
    ...
}
```

9. Modify the .trig file from the previous step to specify the metadata that the Anzo Assign Table Partitioning Predicates service will use to assign the partitions. To provide the required metadata, edit the file as follows:
    a. At the top of the file, replace the ComputePartitioningPredicatesResponse URI with the following Assign service URI:

    ```
    <http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePartiti
    oningPredicatesRequest>
    ```

    In the example above, `<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartiti oningPredicatesResponse>` is replaced by `<http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePar titioningPredicatesRequest>`

b. At the bottom of the file inside the ending brace (}), locate the following triple pattern:

```
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredicatesResponse>

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#partitioningPredicates>
     list_of_predicate_uris .
```

Where **list_of_predicate_uris** is a comma-separated list of all of the predicate URIs from the file. For example, this is the relevant statement from the AssignPartitions.trig file shown above:

```
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredicatesResponse>

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#partitioningPredicates> _:u0d4a3eab-713f-4dbe-b5ad-da5676d6b721 ,
    _:u255f046c-73a1-44ef-b446-c5a6126c9cc1 , _:u2aa40e92-e60d-4b11-9d75-24e84c91ec06 , _:u3472d783-ce53-4aa3-acc2-a9e57d3f318b ,
    _:u3bdca1ac-be4e-4374-b8b9-ce22c18c179a , _:u3eea38d2-8f40-4cb4-b297-8378d68d90e6 , _:u5752b0a1-4262-403a-a029-8a2f54a18f2f ,
    _:u60d4dea2-f06f-45d5-87bf-242e295494ff , _:u702c4eed-2531-4578-a7e7-5ea4120e86ce , _:ub35b741b-6020-4f7d-9a71-7d6c17adf9c9 ,
    _:ub4a0d80b-7e17-4d88-8648-e5c05cea2069 , _:ub8b7ef03-3bbc-41c4-ba43-360bc69620a0 .
```

c. Like the substep a above, replace the ComputePartitioningPredicatesResponse URI with the Assign service URI.

In the example above, `<http://cambridgesemantics.com/ontologies/2015/08/SDIService#ComputePartitioningPredicatesResponse>` is replaced by `<http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePartitioningPredicatesRequest>`

d. At the end of the list of predicate URIs, change the period (.) to a semicolon (;), and then add the following new statements after the semicolon :

```
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#tableURI> table_uri ;
a
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePartitioningPredicatesRequest> .
```

Where **table_uri** is the table URI from step 4. For example:

```
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#tableURI>

<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefa
ab4/Schema/northwind/ORDERS> ;
  a
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePartiti
oningPredicatesRequest> .
```

For example, the end of the AssignPartitions.trig file now looks like this:

```
...

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePartiti
oningPredicatesRequest>

<http://cambridgesemantics.com/ontologies/2015/08/SDIService#partitioningPredic
ates> _:u0d4a3eab-713f-4dbe-b5ad-da5676d6b721 ,
    _:u255f046c-73a1-44ef-b446-c5a6126c9cc1 , _:u2aa40e92-e60d-4b11-9d75-
24e84c91ec06 , _:u3472d783-ce53-4aa3-acc2-a9e57d3f318b ,
    _:u3bdca1ac-be4e-4374-b8b9-ce22c18c179a , _:u3eea38d2-8f40-4cb4-b297-
8378d68d90e6 , _:u5752b0a1-4262-403a-a029-8a2f54a18f2f ,
    _:u60d4dea2-f06f-45d5-87bf-242e295494ff , _:u702c4eed-2531-4578-a7e7-
5ea4120e86ce , _:ub35b741b-6020-4f7d-9a71-7d6c17adf9c9 ,
    _:ub4a0d80b-7e17-4d88-8648-e5c05cea2069 , _:ub8b7ef03-3bbc-41c4-ba43-
360bc69620a0 ;
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#tableURI>

<http://cambridgesemantics.com/DatabaseDataSource/aff6a2f7a1354140871b763dffefa
ab4/Schema/northwind/ORDERS> ;
  a
<http://cambridgesemantics.com/ontologies/2015/08/SDIService#AssignTablePartiti
oningPredicatesRequest> .
}
```

If you would like to view the complete sample file, click here.

10. When the .trig file is complete, save and close the file. It becomes input to the Assign Table Partitioning Predicates service. The service assigns the partitions to the data source to inform the ingestion process. Run the following command to call the assigning service:

```
anzo call
http://cambridgesemantics.com/semanticServices/SDIService#assignTablePartitioningPred
icates filename.trig
```

Where **filename.trig** is the file you edited in the previous step. For example:

```
anzo call
http://cambridgesemantics.com/semanticServices/SDIService#assignTablePartitioningPred
icates AssignPartitions.trig
```

When the prompt returns, the process is complete. If you view the metadata for the table that was partitioned (e.g., run `anzo find -sub table_uri`), the metadata contains a new <http://cambridgesemantics.com/ontologies/DataSources#tablePredicates> URI that lists the partition predicates.

Once the partitioning is complete, the source data can be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mappings, model, and ETL pipeline, see Ingesting Data into Anzo.

Related Topics

Connecting to a Database

Defining a Database Schema

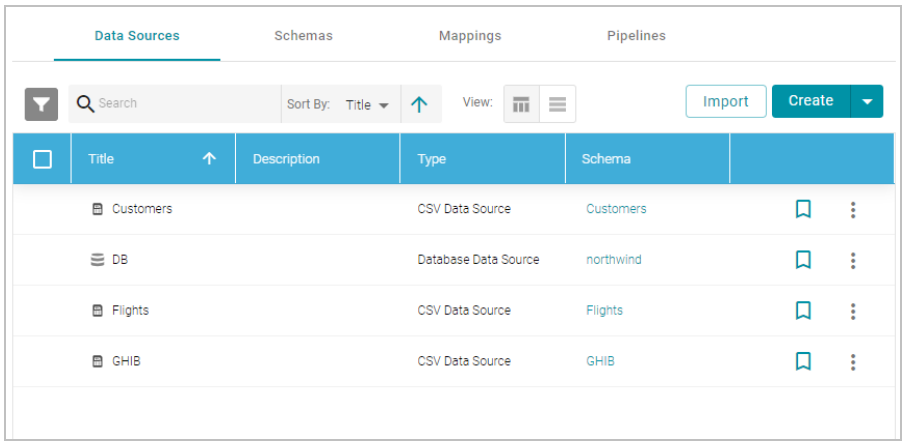
Ingesting Data into Anzo

Generating Source Data Metrics

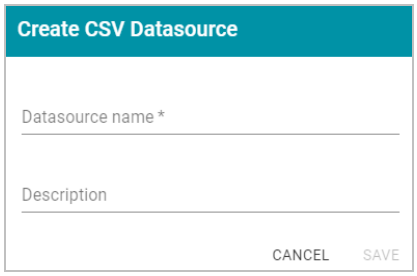
Importing Data from CSV Files

This topic provides instructions for creating a CSV data source and importing data from the files. How do I update the data in Anzo if a file in my CSV data source changes?

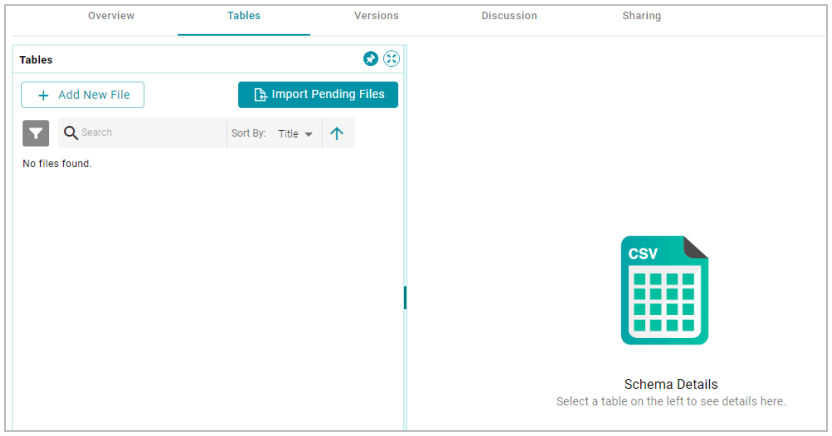
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



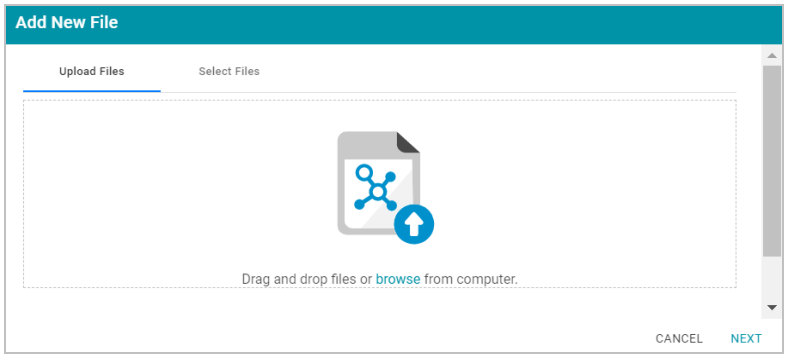
2. Click the **Create** button and select **CSV Data Source**. Anzo opens the Create CSV Datasource screen.



3. Type a name for the data source in the **Datasource name** field, and type an optional description in the **Description** field. Then click **Save**. Anzo saves the source and displays the Tables tab.

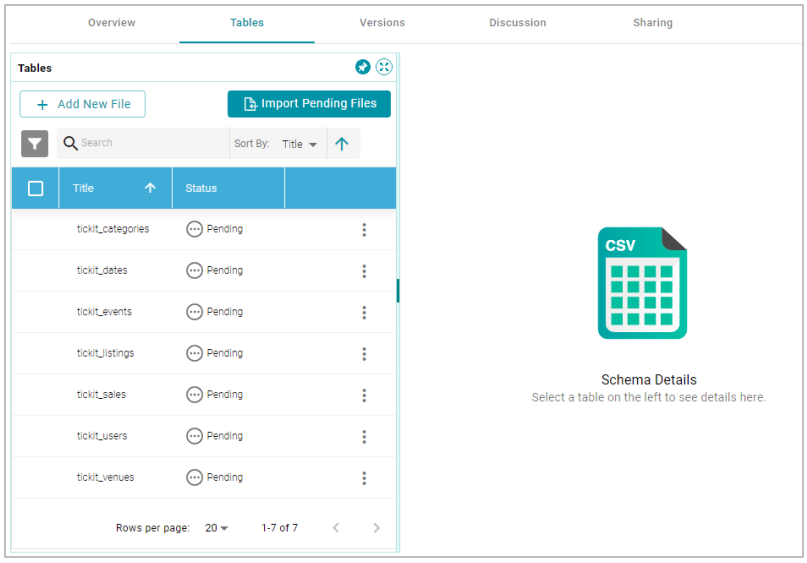


4. On the left side of the screen, click the **Add New File** button. Anzo displays the Add New File dialog box, and the **Upload Files** tab is selected.

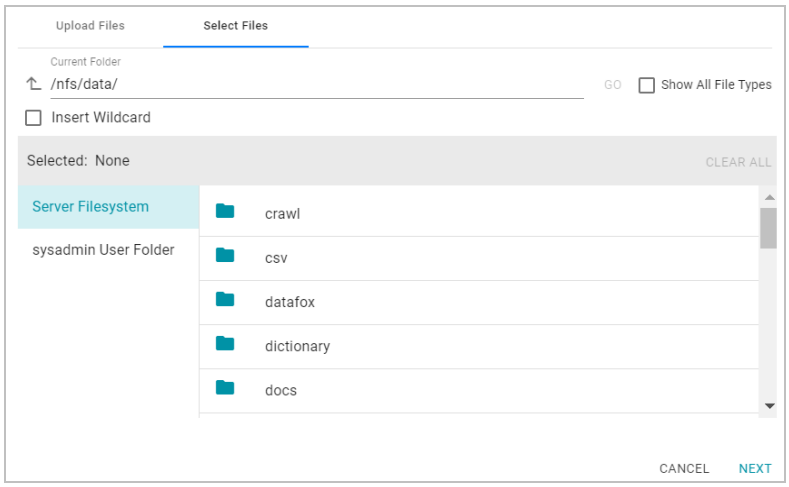


5. Follow the appropriate steps below depending on whether the CSV files are on your computer or a file connection:
   - **If the files are on your computer**:
     a. Drag and drop the files onto the Upload Files tab or click **browse** to navigate to the files and select them. Anzo attaches the files and the Next button becomes active.
     b. Click **Next**. Anzo lists the uploaded files on the left side of the screen with a status of Pending.

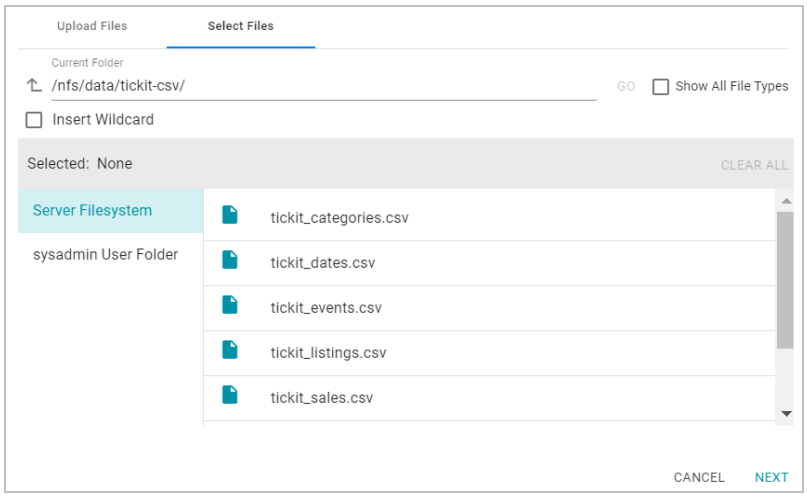
For example:



- **If the files are on a file connection**:
  a. Click the **Select Files** tab. Anzo displays the file selection dialog box.

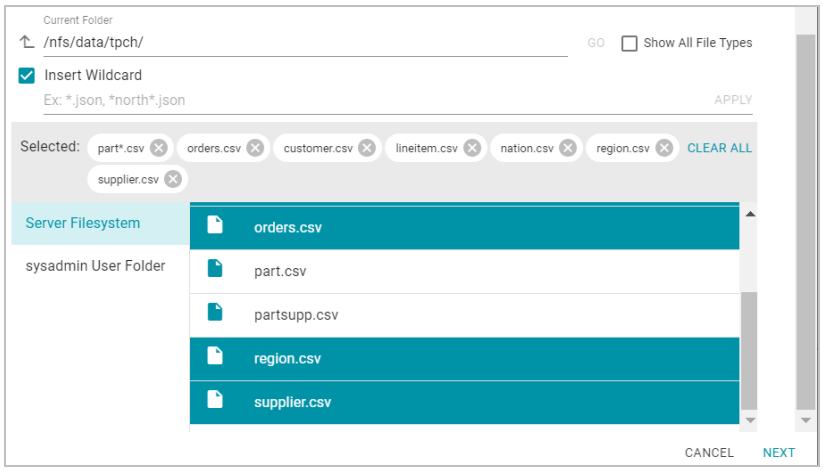


  b. On the left side of the screen, select the file connection for the CSV files. On the right side of the screen, navigate to the directory that contains the files to import. The screen displays the list of files in the directory. For example:

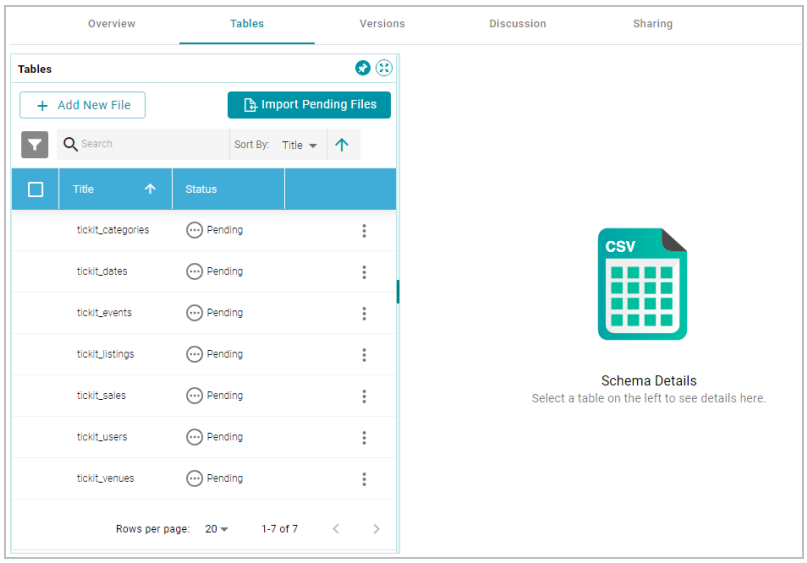


c. Select each file that you want to import. If you have multiple files with the same schema— the files contain the same columns listed in the same order—you can select the **Insert Wildcard** option. Then type a string using asterisks as wildcard characters to find the files with similar names. Files that match the specified string will be imported as **one file** and will result in one job being created in the pipeline to ingest all of the files that are selected by the specified string. After typing a string, click **Apply** to include that string in the Selected list.

Example: The image below shows a directory with several CSV files. For this example, **part.csv** and **partsupp.csv** have the same schema and can be imported as one file. The **Insert Wildcard** option is selected, and **part*.csv** is specified to identify the two files.

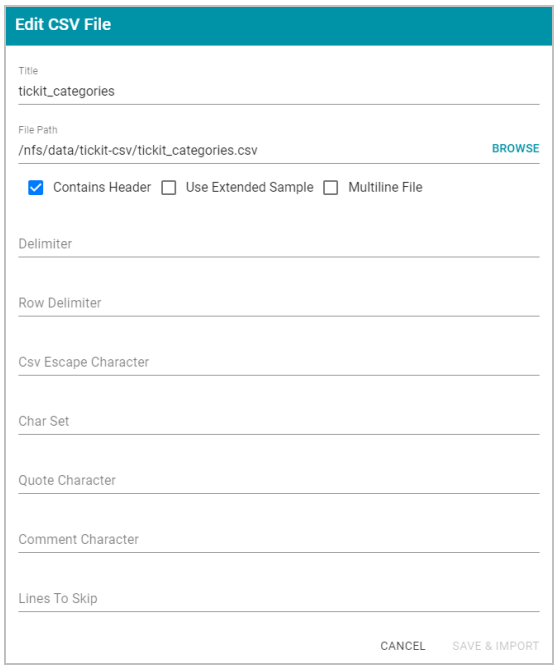


d. When you finish selecting files, click **Next** to close the dialog box. Anzo lists the uploaded files on the left side of the screen with a status of Pending. For example:



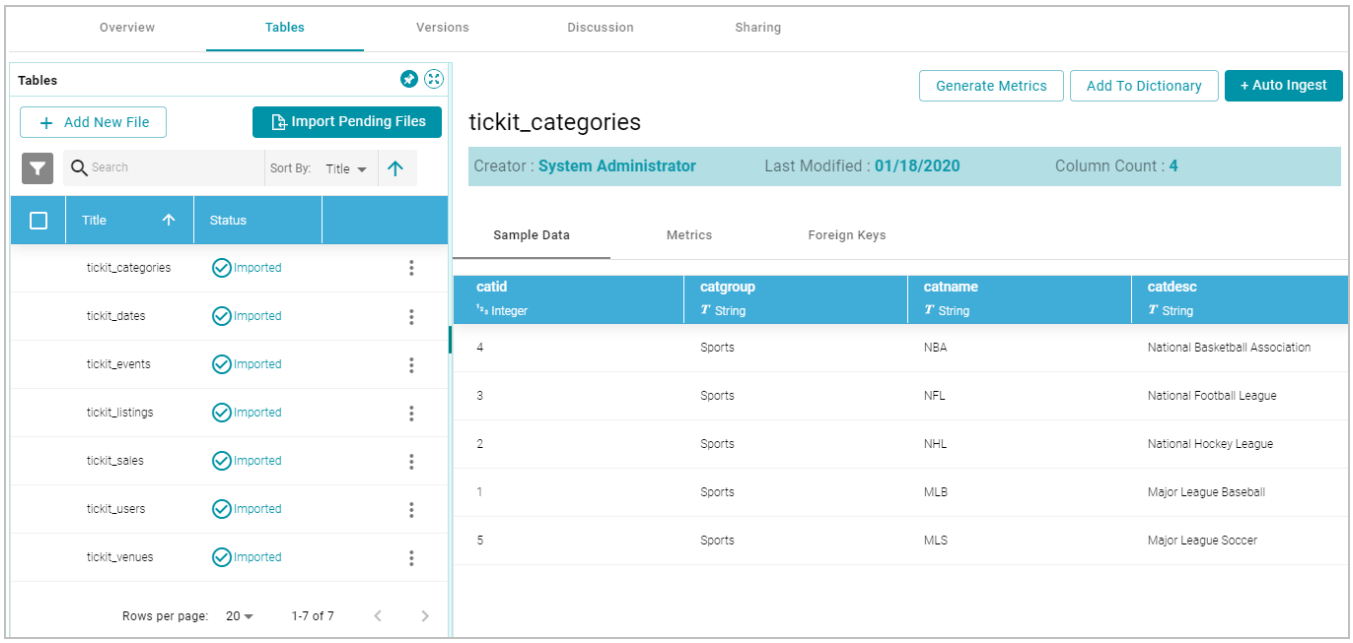
6. If you do not need to change CSV file options, click the **Import Pending Files** button to import all of the pending files. Anzo imports the data and updates the status to Imported.

   If you do need to change CSV file options, click the menu icon (⋮) for that file and select **Edit**. To change the options for multiple files, select the checkbox next to each of the files, and then click the **Edit** button at the top of the table. Anzo displays the Edit CSV File screen. For example, the image below shows the Edit screen for a single file:



   Change the options as needed and then click **Save & Import** to import the CSV file or files. Anzo imports the data and updates the status to Imported.

7. Once the files are imported, click a table row on the left side of the screen to display the schema on the right side of the screen as well as to enable the **Generate Metrics**, **Add To Dictionary**, and **Auto Ingest** options. For example:



The source data can now be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo. For information about generating metrics, see Generating Source Data Metrics. For information about adding a schema to a metadata dictionary, see Using Data Dictionaries.

Related Topics

Ingesting Data into Anzo

Generating Source Data Metrics

Using Data Dictionaries

Working with Schemas

Working with Mappings

Working with Pipelines

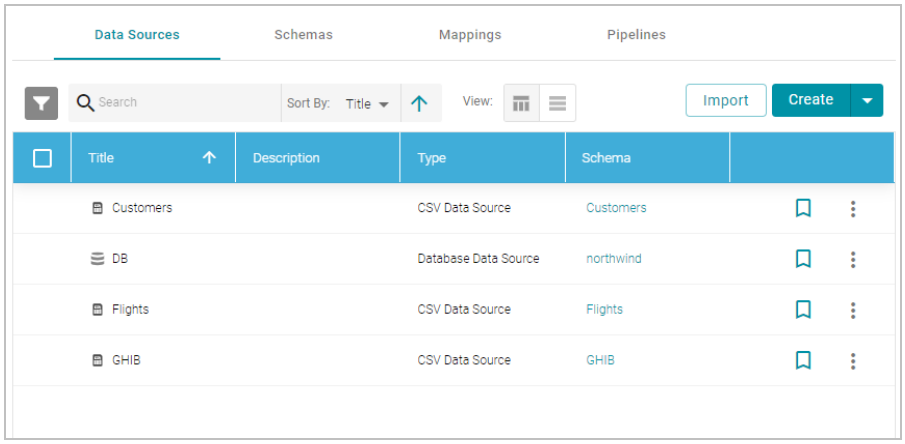
Modeling Data

Importing Data from JSON Files

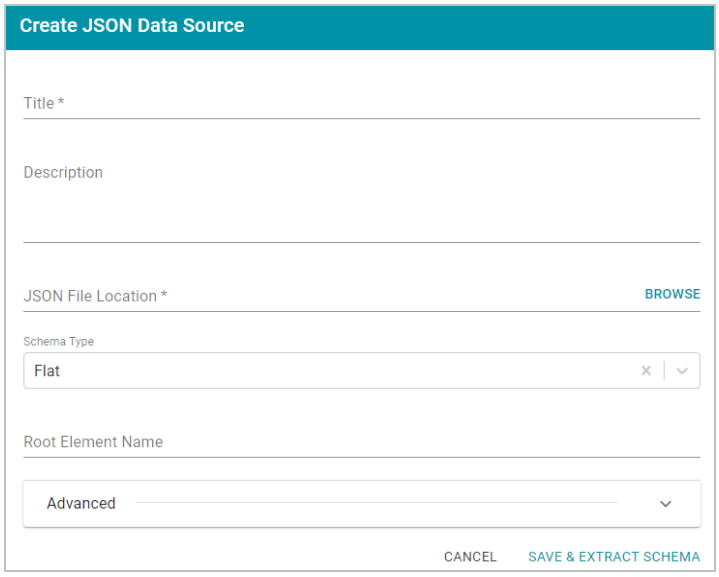
This topic provides instructions for creating a JSON data source, scanning a file, and generating the schema.

Note When a large amount of data is ingested from a single JSON file, the resulting ETL pipeline can take an extremely long time to complete because a single job is created. Since it is a single job, a single ETL engine node processes the data while other resources remain idle. The best approach to loading a large data set in JSON format is to divide the data into several smaller files and then import the batch of files. The resulting pipeline has several smaller jobs that can be processed in parallel.

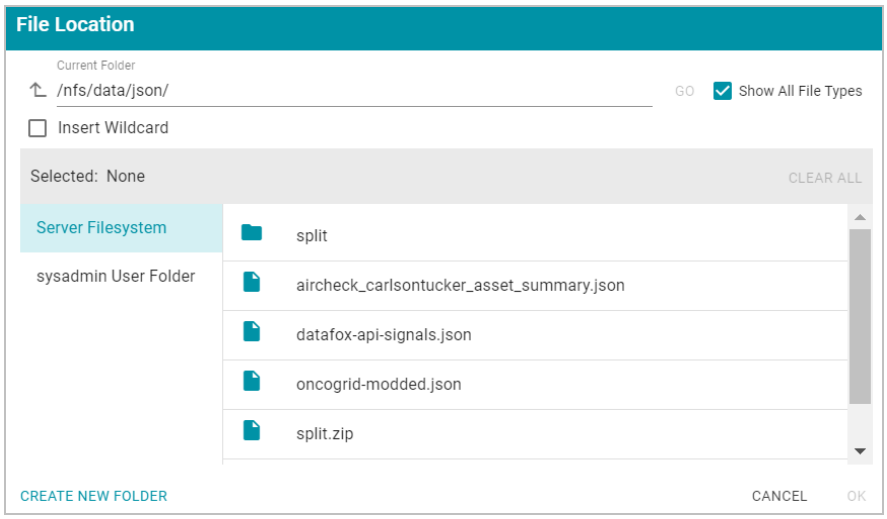
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



2. Click the **Create** button and select **JSON Data Source**. Anzo opens the Create JSON Data Source screen.



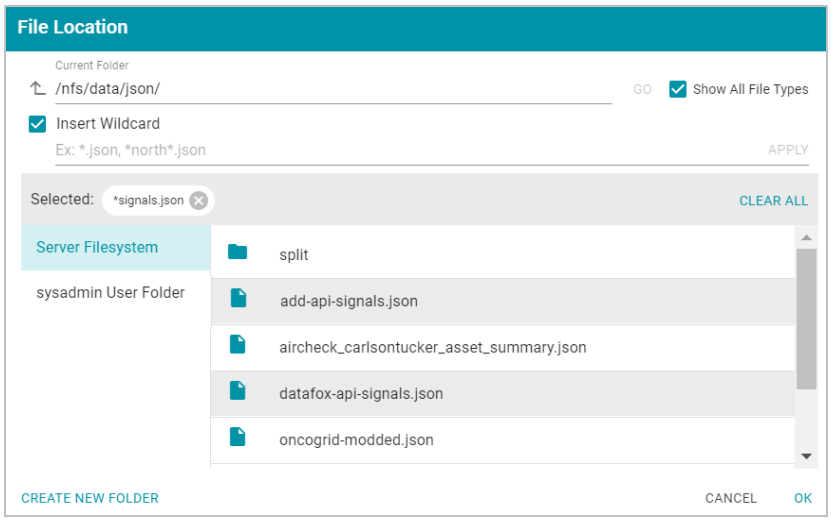
3. Type a name for the data source in the **Title** field, and type an optional description in the **Description** field.
4. Click the **JSON File Location** field to open the File Location dialog box.
5. In the File Location dialog box, on the left side of the screen, select the file connection for the JSON files. On the right side of the screen, navigate to the directory that contains the file to import. The screen displays the list of files in the directory. For example:



6. Select the file that you want to import and then click **OK** to close the dialog box. If you have multiple files with the same schema— the files contain the same arrays in the same order—you can select the **Insert Wildcard** option. Then type a string using asterisks as wildcard characters to find the files with similar names. Files that match the specified string will be imported as **one file** and will result in one job being created in the pipeline to ingest all of the files that are selected by the specified string. You can specify up to 16,000 files using a wild-card. After typing a string, click **Apply** to include that string in the Selected list.

   Important If you have a batch of files that were generated to split a large data set, do NOT specify the batch of files using the wildcard syntax. Select one file from the batch. You will select the rest of the files in a later step. Selecting all files with a wildcard essentially merges the data into one large file, resulting in one ETL job that would be processed by limited ETL engine resources rather than multiple jobs that could be processed in parallel.

   Example: The image below shows a directory with multiple JSON files. For this example, **add-api-signals.json** and **datafox-api-signals.json** have the same schema and can be imported as one file. The **Insert Wildcard** option is selected, and ***signals.json** is specified to identify the two files.



7. Specify the type of schema that Anzo should create. Click the **Schema Type** field and select one of the following types from the drop-down list:
   - **Flat**: By default, the Schema Type is set to **Flat**. A flat schema type results in a single schema table with a single mapping file and ETL job. Generating a flat schema is ideal for files with many different objects with nested relationships where there are many one-to-one relationships. If the file contains a large number of arrays or a number of arrays that are large in size, however, generating a flat schema is not recommended. The import can require extensive server resources and take a long time to process.

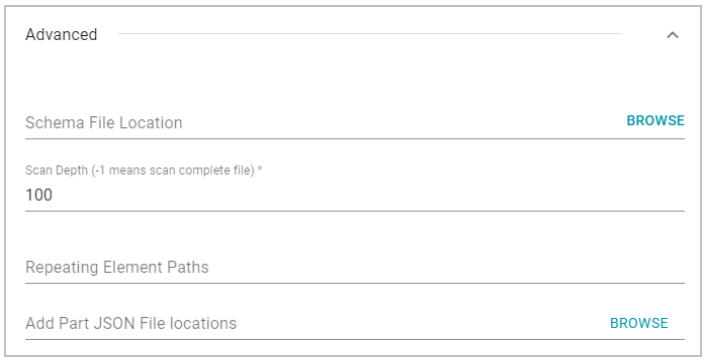
     Note In Flat mode, Anzo creates relationships that go from the parent node to the child node. For example: Person → Address.

   - **Relational**: A relational schema type results in multiple schema tables, mappings, and jobs. Generating a relational schema is ideal for files that include many arrays or a number of very large arrays. Creating a relational schema from a file that contains many different objects with one-to-one relationships can result in poor import performance and a very large number of small tables, mappings, and ETL jobs.

     Note In Relational mode, Anzo creates relationships that go from the child node to the parent node. For example: Address → Person.

   Anzo performs pre-processing before creating the schema. If the specified Schema Type would result in poor performance or require extensive resources, Anzo displays a warning and prompts you to change the schema type before proceeding with the schema creation.

8. If you are importing a batch of files or want to configure other advanced options, expand the **Advanced** section of the screen and proceed to the next steps.



9. When data is onboarded, Anzo sets the root object name to "json." If you want to specify an alternate name for this source, type the new name in the **Root Element Name** field.
10. The **Schema File Location** field defines where Anzo saves the generated schema. Cambridge Semantics recommends that you leave the field blank. If you want to designate a custom location, click **Browse** and choose a file location.
11. The value in the **Scan Depth** field indicates the number of entities in the file that Anzo should scan to find all of the unique objects to include as classes and properties in the generated model. The scan process follows nested objects, counting one object array as one row. Edit the value as needed. A value of **-1** instructs Anzo to scan the entire file.
12. If the JSON file contains lists of objects that are not defined in arrays, the file scan cannot determine if any of the objects are the same type, and Anzo treats each object as a new type. To ensure that repeating object paths are treated as the same type if the file does not include arrays, use standard JSON path syntax to define repeating element types in the **Repeating Element Paths** field. Separate paths with semicolons (;). If the file includes arrays, leave this field blank.

    For example, when Anzo scans the following sample JSON markup, people, vehicles, and maintenance would become object types without a defined relationship:

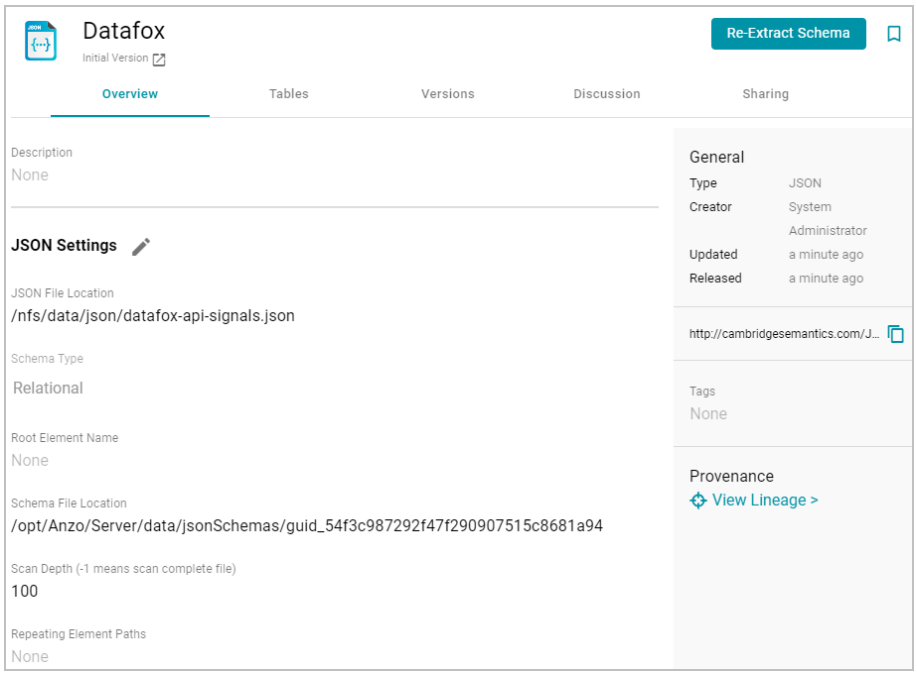
```
{
  "people":
    {
       "personal": {
         "age": 20,
         "gender": "M",
         "name": {
           "first": "John",
           "last": "Doe"
         }
       },
       "vehicles": {
         "type": "car",
```

```
          "model": "Honda Civic",
          "insurance": {
            "company": "ABC Insurance",
            "policy_num": 12345
        },
        "maintenance":
           {
             "date": "07-17-2017",
             "desc": "oil change"
           },
           {
             "date": "01-03-2018",
             "desc": "new tires"
           }
       }
    }
}
```

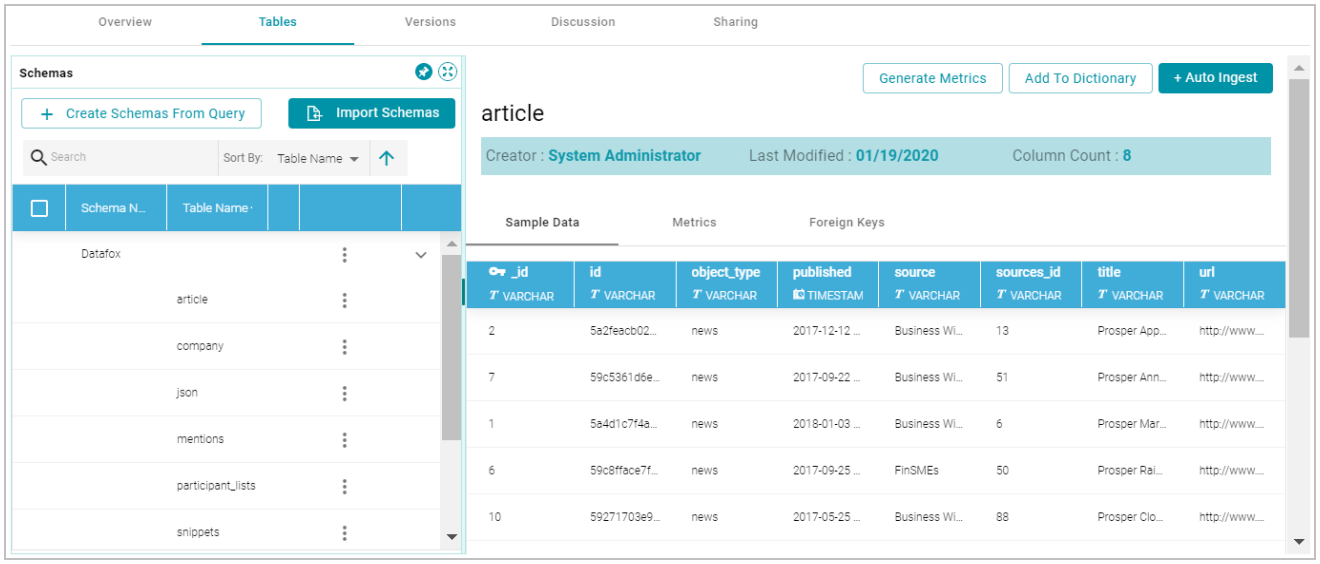
By defining the following paths in the Repeating Element Paths field, the scan knows that "people" is an object type, "vehicles" map to people, and "maintenance" is related to vehicles, which map to people.

```
$.people;$.people.vehicles;$.people.vehicles.maintenance
```

13. If you are importing a batch of files, click the **Add Part JSON File Locations** field to open the File Location dialog box. Select each of the files included in the batch and then click **OK** to close the dialog box.
14. Click **Save & Extract Schema** to scan the file and generate the schema. Anzo saves the data source, creates the schema, and displays the data source overview. For example:



To view the schema that Anzo created, you can click the Schema Name link at the bottom of the screen under **Schema Details**. Anzo opens the Tables screen for the schema, where you can access schema details. For example:



The source data can now be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo. For information about adding a schema to a metadata dictionary, see Using Data Dictionaries. If you imported data from a batch of JSON files, you must create a data dictionary to onboard the source.

Related Topics

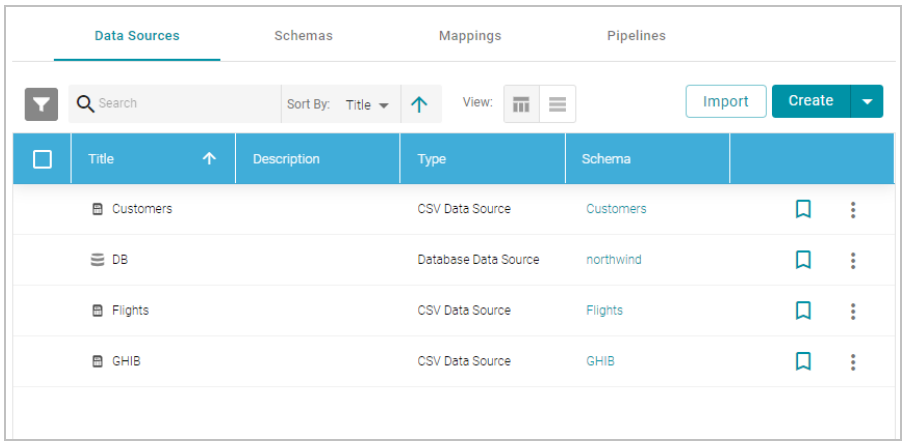
Ingesting Data into Anzo

Generating Source Data Metrics

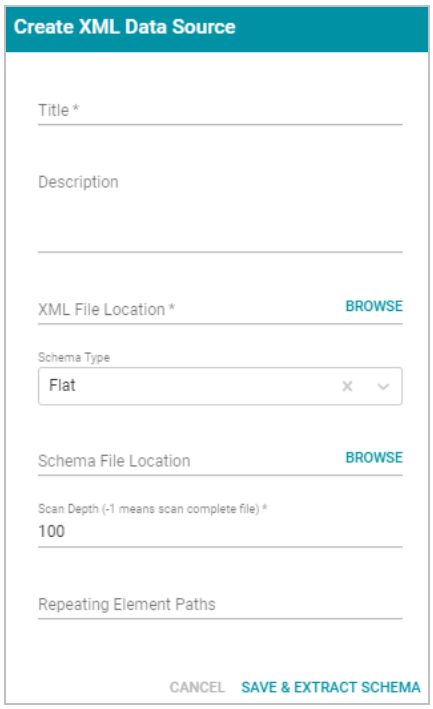
Importing Data from XML Files

This topic provides instructions for creating an XML data source, scanning a file, and generating the schema.

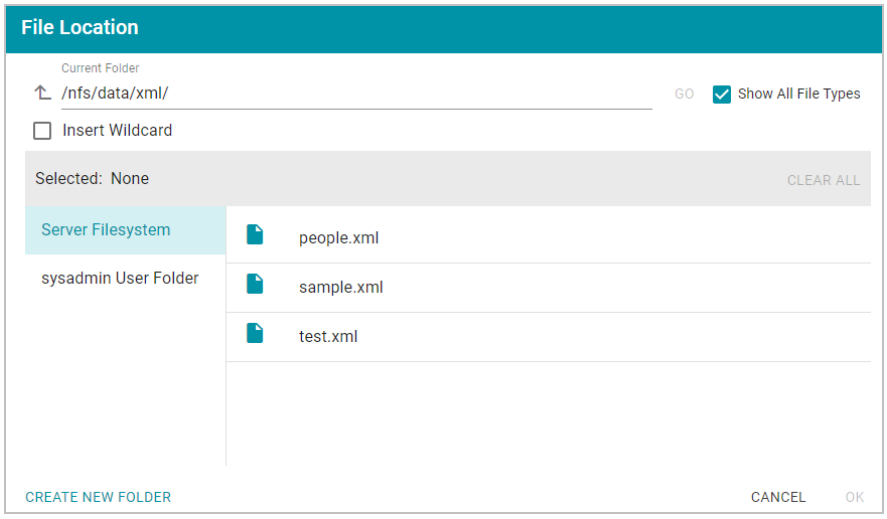
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



2. Click the **Create** button and select **XML Data Source**. Anzo opens the Create XML Data Source screen.



3. Type a name for the data source in the **Title** field, and type an optional description in the **Description** field.
4. Click the **XML File Location** field to open the File Location dialog box.
5. In the File Location dialog box, on the left side of the screen, select the file connection for the file. On the right side of the screen, navigate to the directory that contains the file to import. The screen displays the list of files in the directory. For example:



6. Select the file that you want to import and then click **OK** to close the dialog box. If you have multiple files with the same schema— the files contain the same elements in the same order—you can select the **Insert Wildcard** option. Then type a string using asterisks as wildcard characters to find the files with similar names. Files that match the specified string will be imported as **one file** and will result in one job being created in the pipeline to ingest all of the files that are selected by the specified string. After typing a string, click **Apply** to include that string in the Selected list.
7. Specify the type of schema that Anzo should create. Click the **Schema Type** field and select one of the following types from the drop-down list:
    - **Flat**: By default, the Schema Type is set to **Flat**. A flat schema type results in a single schema table with a single mapping file and ETL job. Generating a flat schema is ideal for files with many different objects with nested relationships where there are many one-to-one relationships. If the file contains a large number of arrays or a number of arrays that are large in size, however, generating a flat schema is not recommended. The import can require extensive server resources and take a long time to process.

      Note In Flat mode, Anzo creates relationships that go from the parent node to the child node. For example: Person → Address.

    - **Relational**: A relational schema type results in multiple schema tables, mappings, and jobs. Generating a relational schema is ideal for files that include many arrays or a number of very large arrays. Creating a relational schema from a file that contains many different objects with one-to-one relationships can result in poor import performance and a very large number of small tables, mappings, and ETL jobs.

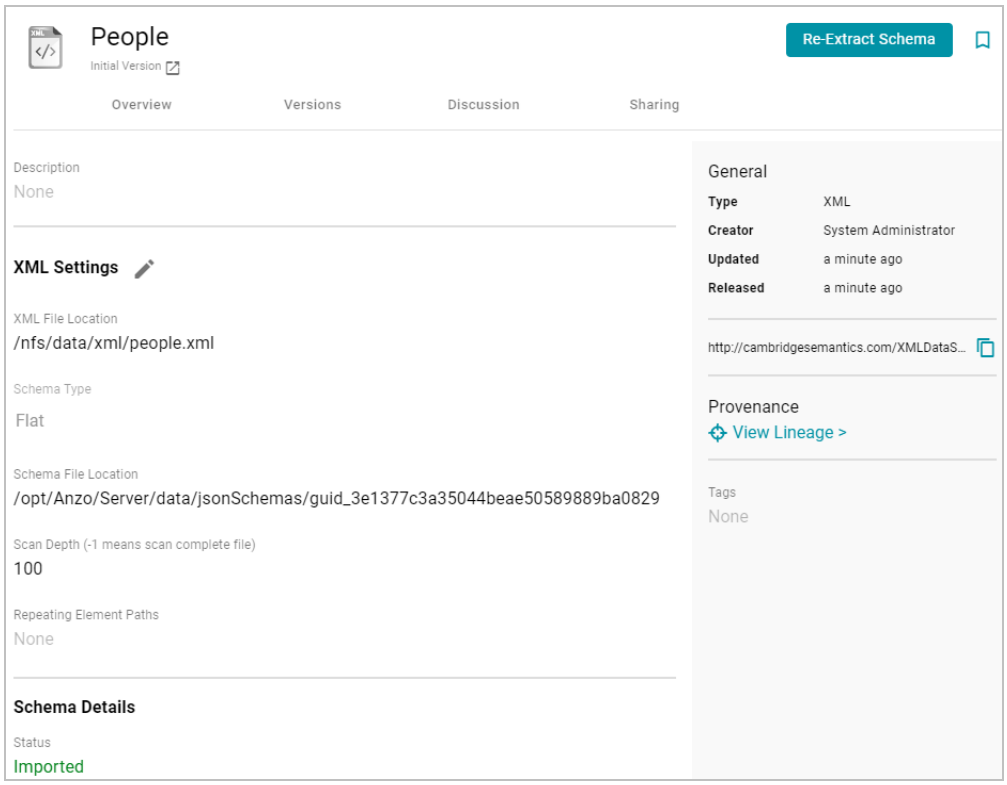
> **Note** In Relational mode, Anzo creates relationships that go from the child node to the parent node. For example: Address → Person.

Anzo performs pre-processing before creating the schema. If the specified Schema Type would result in poor performance or require extensive resources, Anzo displays a warning and prompts you to change the schema type before proceeding with the schema creation.

8. The **Schema File Location** field defines where Anzo saves the generated schema. Cambridge Semantics recommends that you leave the field blank. If you want to designate a custom location, click **Browse** and choose a file location.
9. The value in the **Scan Depth** field indicates the number of entities in the file that Anzo should scan to find all of the unique objects to include as classes and properties in the generated model. The scan process follows nested objects, counting one object array as one row. Edit the value as needed. A value of **-1** instructs Anzo to scan the entire file.
10. If the XML file contains lists of objects that are not nested, the file scan cannot determine if any of the objects are the same type, and Anzo treats each object as a new type. To ensure that repeating object paths are treated as the same type if the XML elements are all at the same level, use standard XML path (XPath) syntax to define the repeating element types in the **Repeating Element Paths** field. If the file nests elements, leave this field blank. Separate paths with semicolons (;). For example:

```
/root/people;/root/people/vehicles;/root/people/vehicles/maintenance
```

11. Click **Save & Extract Schema** to scan the file and generate the schema. Anzo saves the data source, creates the schema, and displays the data source overview. For example:



To view the schema that Anzo created, you can click the Schema Name link at the bottom of the screen under **Schema Details**. Anzo opens the Tables screen for the schema, where you can access schema details.

The source data can now be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo. For information about generating metrics, see Generating Source Data Metrics. For information about adding a schema to a metadata dictionary, see Using Data Dictionaries.

Related Topics

Ingesting Data into Anzo

Generating Source Data Metrics

Using Data Dictionaries

Working with Schemas

Working with Mappings

Working with Pipelines

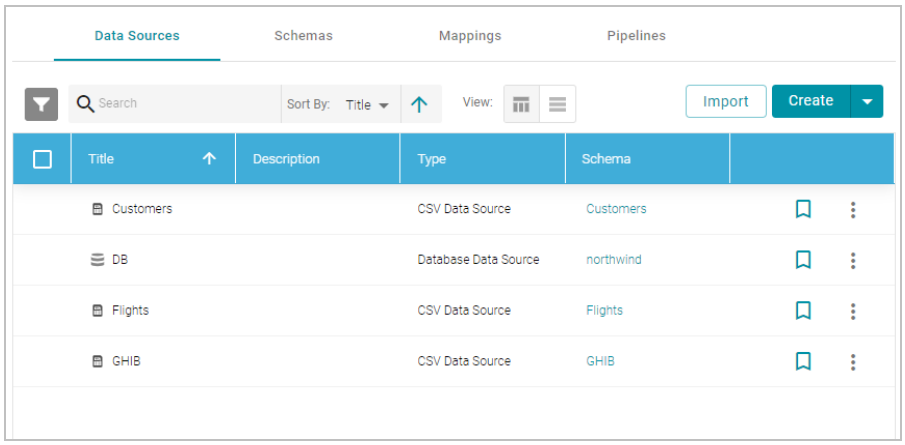
Modeling Data

Importing Data from SAS Files

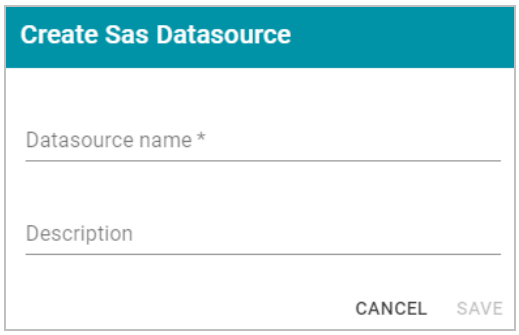
This topic provides instructions for creating a SAS data source and importing data from SAS7BDAT files.

Note When importing data from SAS files, Anzo imports any metadata that is defined in the files. The metadata only becomes visible, however, when a metadata dictionary is created for the source. For more information, see Using Data Dictionaries.

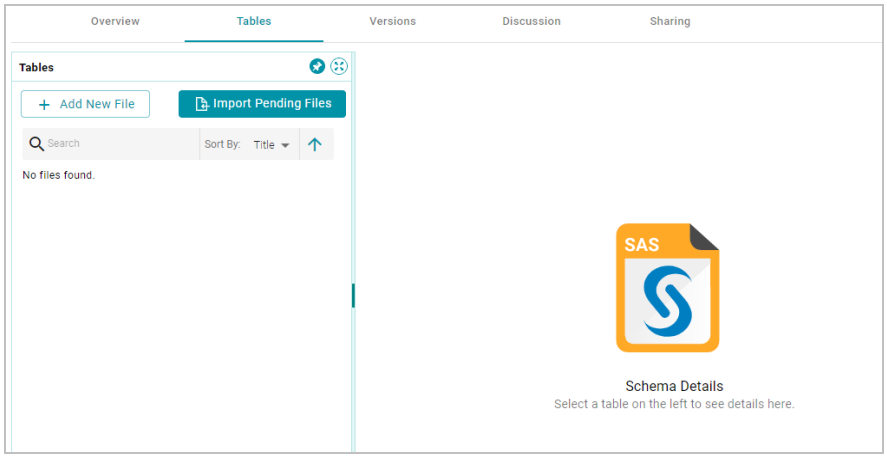
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



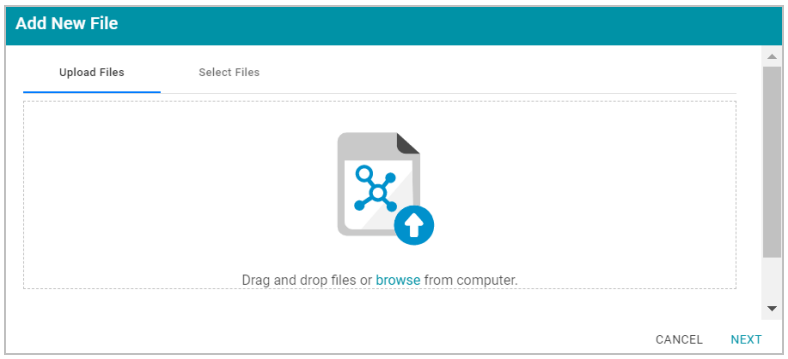
2. Click the **Create** button and select **SAS Data Source**. Anzo opens the Create Sas Datasource screen.



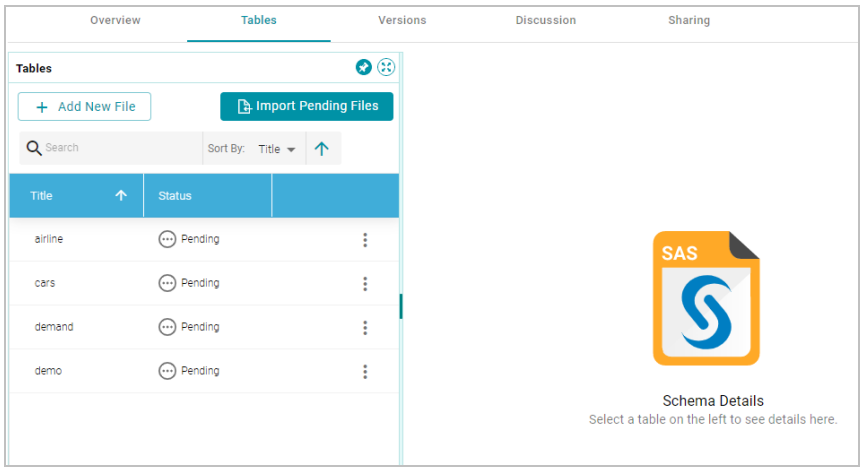
3. Type a name for the data source in the **Datasource name** field, and type an optional description in the **Description** field. Then click **Save**. Anzo saves the source and displays the Tables tab.



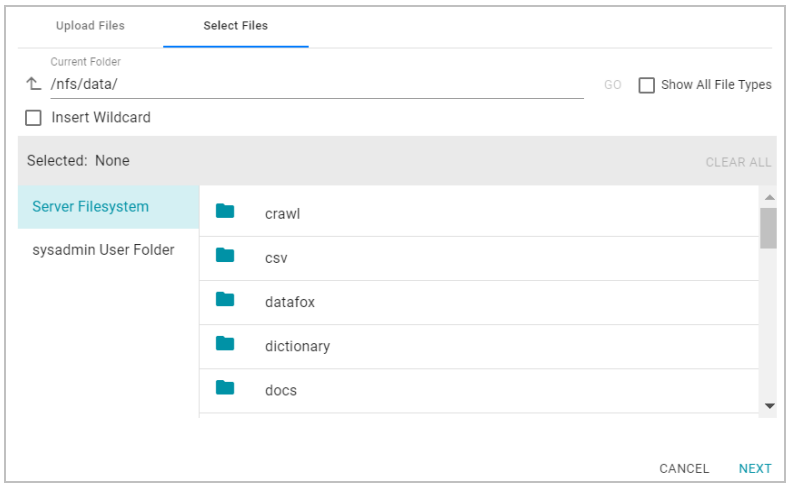
4. On the left side of the screen, click the **Add New File** button. Anzo displays the Add New File dialog box, and the **Upload Files** tab is selected.



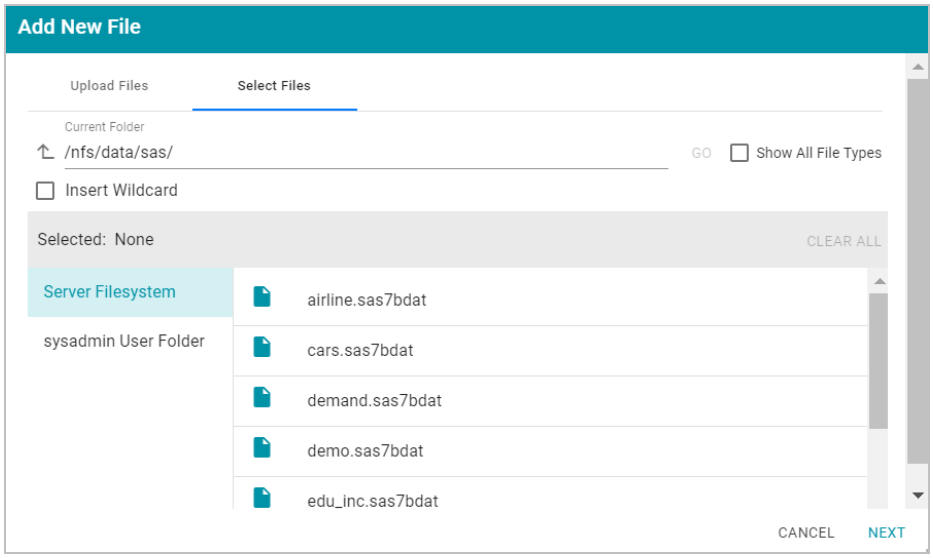
5. Follow the appropriate steps below depending on whether the SAS files are on your computer or a file connection:
    - **If the files are on your computer**:
        a. Drag and drop the files onto the Upload Files tab or click **browse** to navigate to the files and select them. Anzo attaches the files and the Next button becomes active.
        b. Click **Next**. Anzo lists the uploaded files on the left side of the screen with a status of Pending. For example:



- **If the files are on a file connection**:
  a. Click the **Select Files** tab. Anzo displays the file selection dialog box.

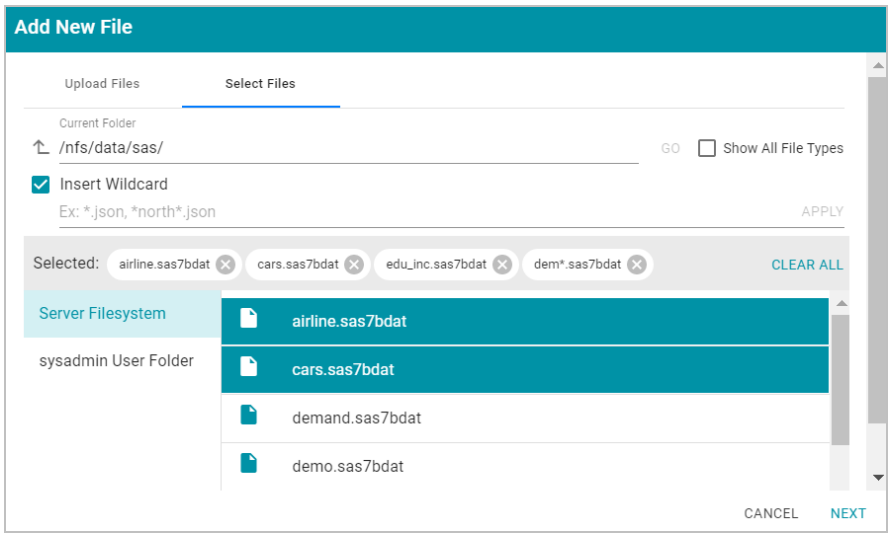


  b. On the left side of the screen, select the file connection for the SAS files. On the right side of the screen, navigate to the directory that contains the files to import. The screen displays the list of files in the directory. For example:

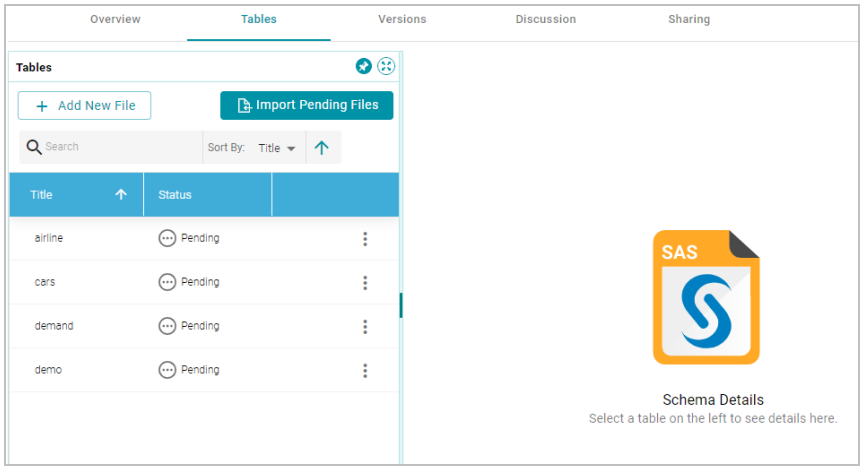


  c. Select each file that you want to import. If you have multiple files with the same schema— the files contain the same columns listed in the same order—you can select the **Insert Wildcard** option. Then type a string using asterisks as wildcard characters to indicate find the files with similar names. Files that match the specified string will be imported as one file. After typing a string, click **Apply** to include that string in the Selected list.

  Example: The image below shows a directory with several SAS files. For this example, **demand.sas7bat** and **demo.sas7bat** have the same schema and can be imported as one file. The **Insert Wildcard** option is selected, and **dem*.sas7bat** is specified to identify the two files.

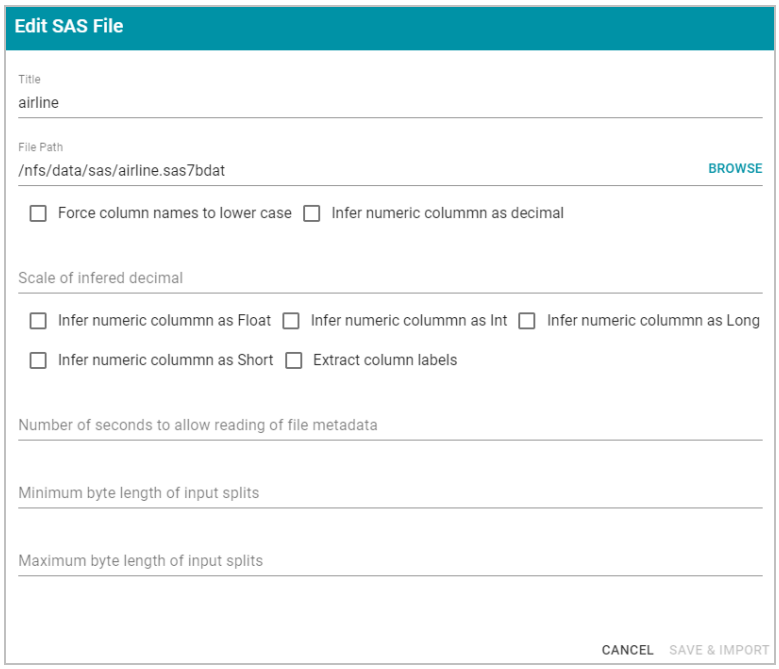


d. When you finish selecting files, click **Next** to close the dialog box. Anzo lists the uploaded files on the left side of the screen with a status of Pending. For example:



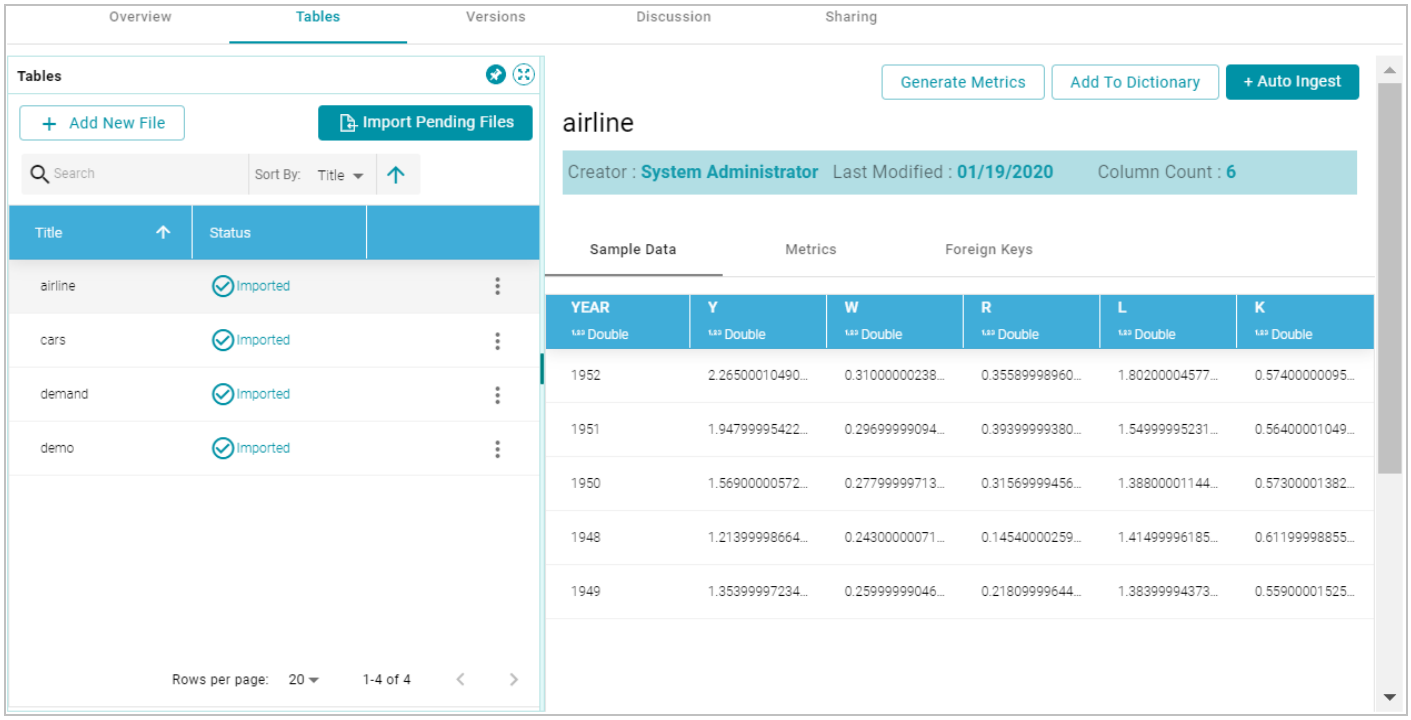
6. If you do not need to change SAS file options, click the **Import Pending Files** button to import all of the pending files. Anzo imports the data and updates the status to Imported.

   If you do need to change SAS file options, click the menu icon (⋮) for that file and select **Edit**. To change the options for multiple files, select the checkbox next to each of the files, and then click the **Edit** button at the top of the table. Anzo displays the Edit SAS File screen. For example, the image below shows the Edit screen for a single file:



Change the options as needed and then click **Save & Import** to import the SAS file or files. Anzo imports the data and updates the status to Imported.

7. Once the files are imported, click a table row on the left side of the screen to display the schema on the right side of the screen as well as to enable the **Generate Metrics**, **Add To Dictionary**, and **Auto Ingest** options. For example:



The source data can now be onboarded to Anzo. For instructions on onboarding the data by letting Anzo automatically generate the mapping, model, and ETL pipeline, see Ingesting Data into Anzo. For information about generating metrics, see Generating Source Data Metrics. For information about adding a schema to a metadata dictionary, see Using Data Dictionaries.

Related Topics

Ingesting Data into Anzo

Generating Source Data Metrics

Using Data Dictionaries

Working with Schemas

Working with Mappings

Working with Pipelines

Modeling Data

Importing Data from RDF Files

Source data that is not in RDF format is onboarded through structured or unstructured pipelines, where the data is imported to Anzo and converted to RDF format before becoming available in the Dataset catalog. Certain RDF file types, however, can be added to the catalog directly, making the data available for loading and analyzing in AnzoGraph.

Users can add to the Dataset catalog any pre-existing file-based linked data set (FLDS), such as when migrating an FLDS from one Anzo server to another. Or they can point Anzo to a directory of Turtle, N-Triple, N-Quad, or TriG files and Anzo will create the FLDS and add the data set to the catalog.

Note To import data from CSV, JSON, XML, or SAS files, use the import processes described in Importing Data from CSV Files, Importing Data from JSON Files, Importing Data from XML Files, or Importing Data from SAS Files.

This topic provides instructions for making RDF files available in the catalog.

- File Requirements
- Importing RDF Files
- Importing an FLDS

File Requirements

To add a data set to the Anzo catalog, the location of the files, the file format, and the directory structure must meet the following requirements.

- **Supported File Locations**: Files must be staged on a configured file store.
- **Supported File Formats**: Files must be in one of the following formats.
  - Turtle (.ttl file type)
  - N-Triple (.n3 and .nt file types)
  - N-Quad (.nq and .quads file types)
  - TriG (.trig file type)

  Any of the file types listed above can be compressed in GZIP format and imported as *filename.filetype*.gz files.
- **Supported Directory Structure**: The directory structure that is required depends on whether you are importing a File-Based Linked Data Set (FLDS)—a data set that was previously created by onboarding data to Anzo—or files that are not yet part of an FLDS:
  - **FLDS Imports**: FLDS directories should contain an **flds.trig** file, an **onts** directory that includes the model .trig file, and an **rdf.ttl** or **rdf.ttl.gz** directory that contains the data files. For example:

```
LoadEmployees_f7b1f
├── flds.trig
├── onts
│   └── Employees.trig
└── rdf.ttl.gz
    └── Loadnew_employees_8be23.ttl.gz
        └── 20191021034225.ttl.gz
            └── part-00000.ttl.gz
            └── part-00001.ttl.gz
             └── part-00003.ttl.gz
```

    Note Models must be in TriG format, regardless of the file type of the data files.

  - **RDF File Imports**: When importing RDF files that are not part of an FLDS, the files must be placed in a directory named **rdf.ttl** or **rdf.ttl.gz**. Use one of those names regardless of the file format. Stage N-Triple, N-Quad, and TriG files in a directory named rdf.ttl. Place uncompressed files in an rdf.ttl directory and gzipped files in an rdf.ttl.gz directory.

    For example:

```
External-RDF-Top-Level-Directory
└── rdf.ttl.gz
    └── external-rdf-file1.ttl.gz
    └── external-rdf-file2.ttl.gz
    └── external-rdf-file3.ttl.gz
```

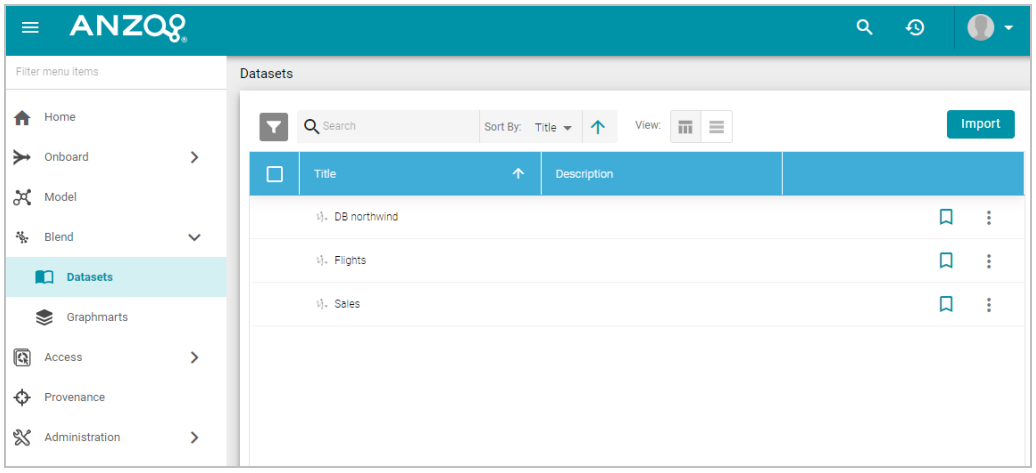
Important All files inside an rdf.ttl or rdf.ttl.gz directory must be the same format and end in the same extension. Data in mixed formats will not load successfully. If you plan to import multiple file types, organize files into separate directories by file extension type, and then import each directory separately.

Importing RDF Files

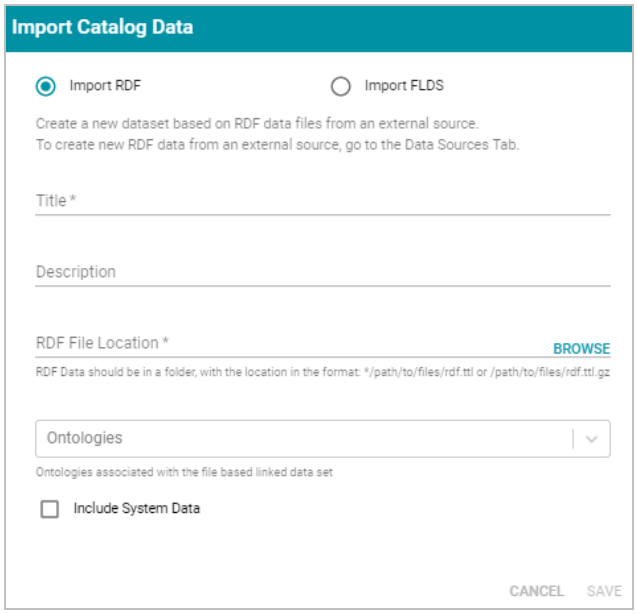
Follow the instructions below to create an FLDS catalog entry from a directory of Turtle, N-Triple, N-Quad, or TriG files. Make sure that the files and directory meet the requirements in File Requirements.

Tip Anzo provides the option to link the files to an existing data model during the import. If the model is not yet available in Anzo, consider uploading it before importing the RDF files. See Uploading a Model to Anzo for instructions. You are not required to include a model at import time; a model can be associated with a data set at any time. How do I associate a model with an existing data set?

1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Datasets screen, which lists the catalog of data sets. For example:



2. On the Datasets screen, click **Import**. Anzo opens the Import Catalog Data dialog box.



3. The **Import RDF** radio button is selected by default. Type a name for the data set in the **Title** field and an optional description in the **Description** field.
4. Click the **RDF File Location** field to open the File Location dialog box. Find and select the **rdf.ttl** or **rdf.ttl.gz** directory that you want to import, and then click **OK** to close the dialog box.
5. If you want to associate a model with this data set, click the **Ontologies** drop-down list and select the model. To include a system model, select the **Include System Data** checkbox. If you do not want to associate a model with the data at this time, leave the **Ontologies** field blank.

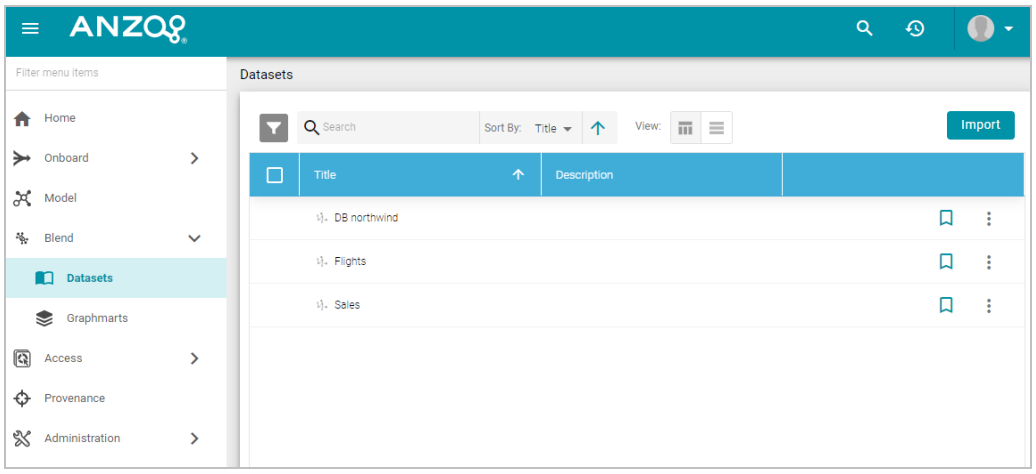
   Note: Data sets without a model cannot be viewed in Hi-Res Analytics dashboards, but the imported data can still be queried. A model can be associated with the data set at a later time. How do I associate a model with an existing data set?
6. Click **Save** to create the FLDS, add it to the catalog, and return to the Datasets screen. You can now select the FLDS from the catalog and create a graphmart. See Creating a Graphmart for instructions.

   Note: Anzo generates an flds.trig file at the same level as the rdf.ttl or rdf.ttl.gz directory. The file contains metadata about the load files.

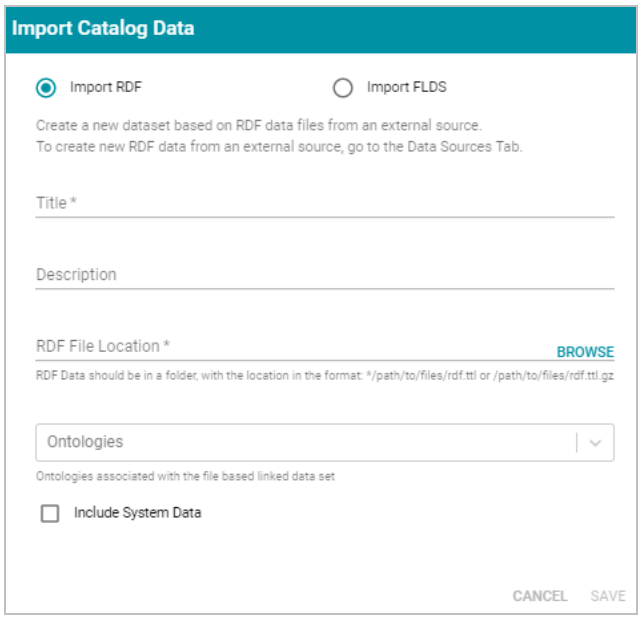
Importing an FLDS

Follow the instructions below to add an FLDS to the catalog. Make sure that the FLDS meets the requirements in File Requirements.

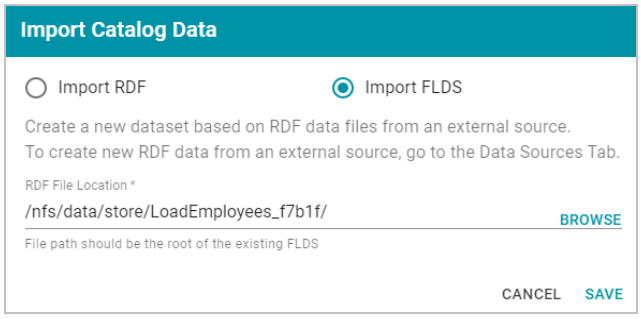
1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Datasets screen, which lists the catalog of data sets. For example:



2. On the Datasets screen, click **Import**. Anzo opens the Import Catalog Data dialog box.



3. Select the **Import FLDS** radio button.
4. Click the **RDF File Location** field to open the File Location dialog box. Select the root directory for the FLDS, the directory that contains the flds.trig file, the onts directory, and the rdf.ttl directory. For example:



5. Click **Save** to import the FLDS and return to the Datasets screen. You can now select the FLDS in the catalog and create a graphmart. See Creating a Graphmart for instructions.

Related Topics

Importing Data from CSV Files

Importing Data from JSON Files

Importing Data from XML Files

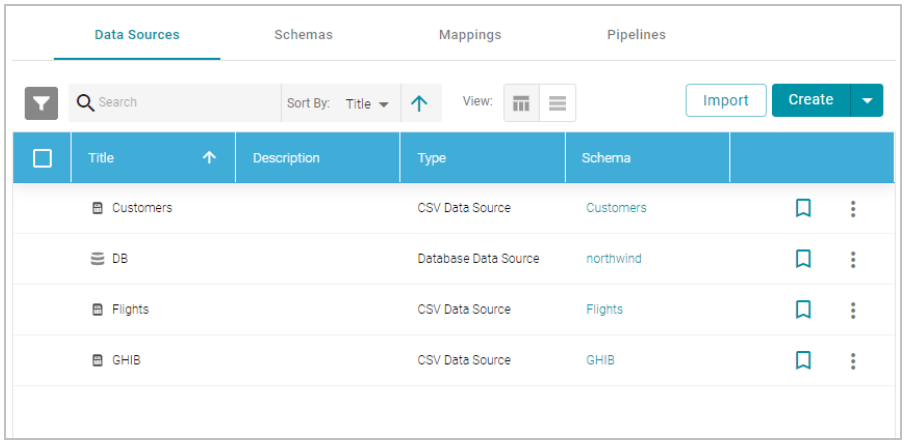
Importing Data from SAS Files

Generating Source Data Metrics

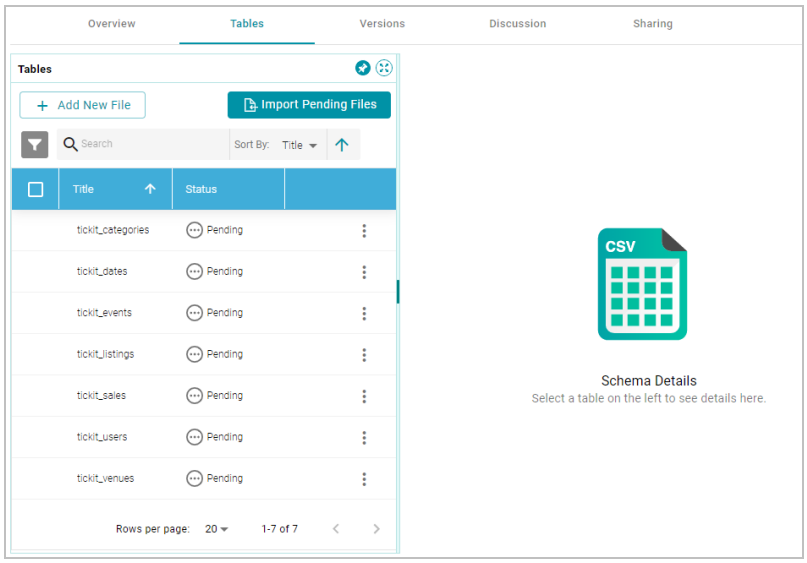
To help users assess the quality of the data coming from a data source, Anzo provides the ability to calculate metrics for each source. When metrics are generated, Anzo profiles the entire source data set and reports statistics for each table in the schema, such as the number of populated, null, or empty rows for each column in a table and the number of rows for each column grouped by value. It also reports column-level metrics such as the smallest and largest values in a column, the number of unique values, and the value that appears most often. For schemas with multiple tables, Anzo also generates a list of foreign key suggestions between tables that include the same column.

Follow the instructions below to generate and review data quality metrics for a data source.

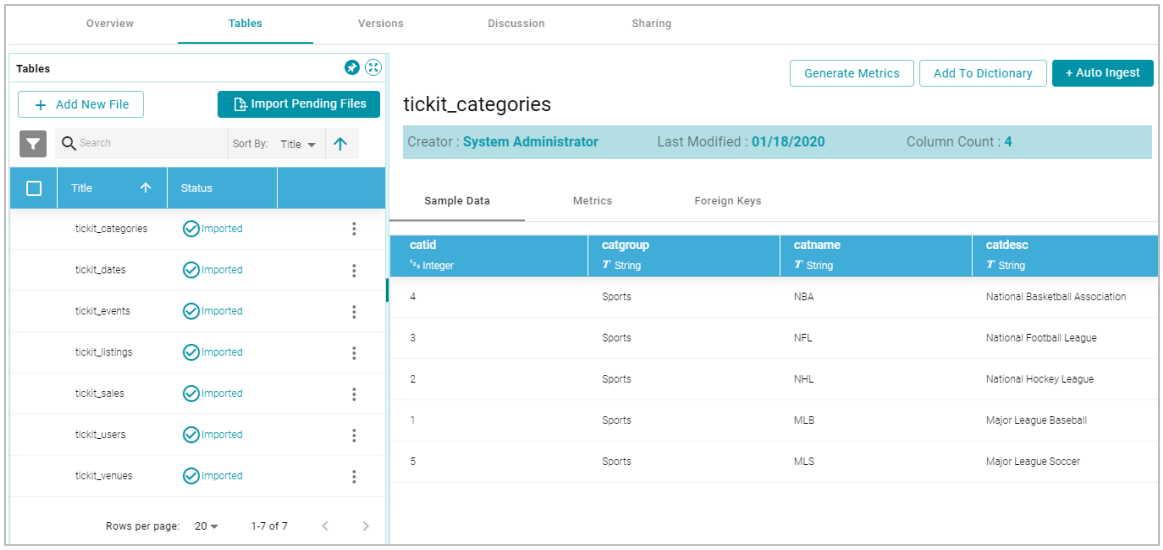
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



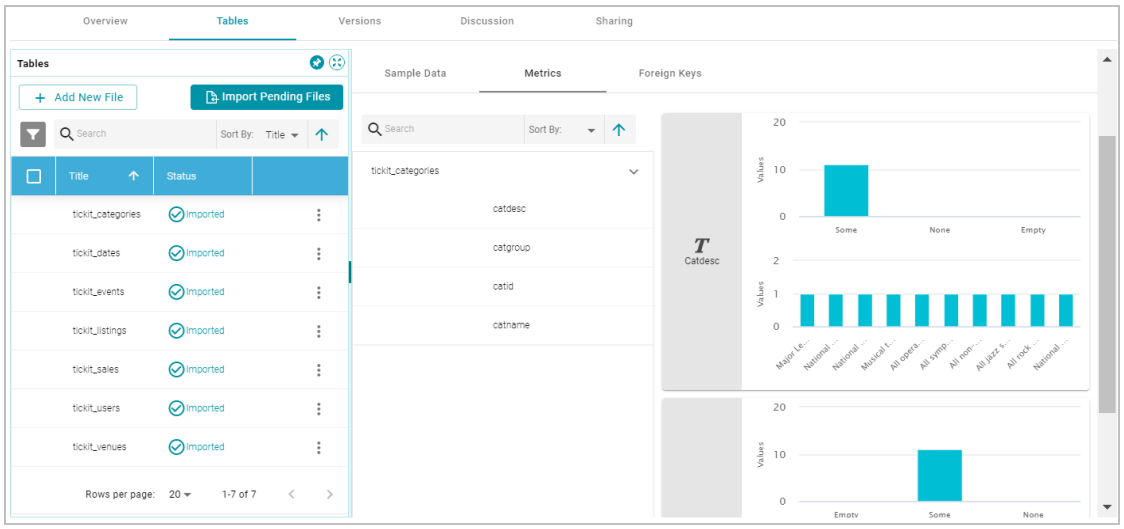
2. Click the data source for which you want to generate metrics. Anzo displays the Tables tab for the source, which lists the schema and table details. For example:



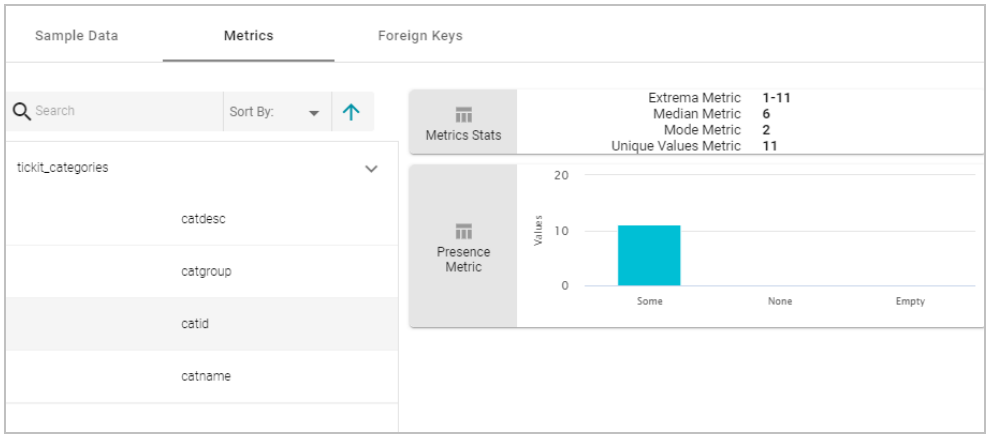
3. On the left side of the screen, select a row for a table in the schema. A view of the sample data and additional buttons are displayed on the right side of the screen. For example:



4. Click the **Generate Metrics** button at the top of the screen. Anzo displays a warning that the process may consume a lot of server resources. Click **Yes** to continue. Anzo calculates metrics for the schema. The process may take several minutes. You can check the status of the process in the Activity Log.
5. When the process is complete, click the **Metrics** tab next to the Sample Data tab in the middle of the screen. Anzo displays the metrics for the selected table. For example:



On the far right of the screen, Anzo shows the metrics for each column in the selected table. To select a different table, click another row on the left side of the screen. To view additional metrics for a column in the table, click a column name in the middle of the screen. For example:



Depending on the data type of the column, one or more of the following metrics are shown:

- Extrema Metric: Shows the smallest and largest values.
- Median Metric: Shows the middle value.
- Mode Metric: Shows the value that appears most often.
- Unique Values Metric: Shows the number of unique values.

For additional metrics based on the type of data quality checks needed, contact Cambridge Semantics. For information about generating metrics for an onboarded data set, see Generating Graph Data Metrics.

Related Topics

Importing or Creating a Schema

Using Data Dictionaries

Metadata dictionaries are similar to data models in that they define the desired business meaning and structure of the data after it is onboarded to Anzo and converted to the graph model. Unlike data models, though, metadata dictionaries offer maximum flexibility for normalizing the data that comes from various sources and structures. A single dictionary can be used to link conceptually identical elements (columns) from many different data source schemas, independent of any models and mappings. The metadata dictionary structure becomes the basis for creating and reusing models and mappings. As models and mappings are generated, deleted, and recreated over time, the growing body of information about business meaning and the concepts that link source schema elements to properties in the model remain available in the data dictionaries.

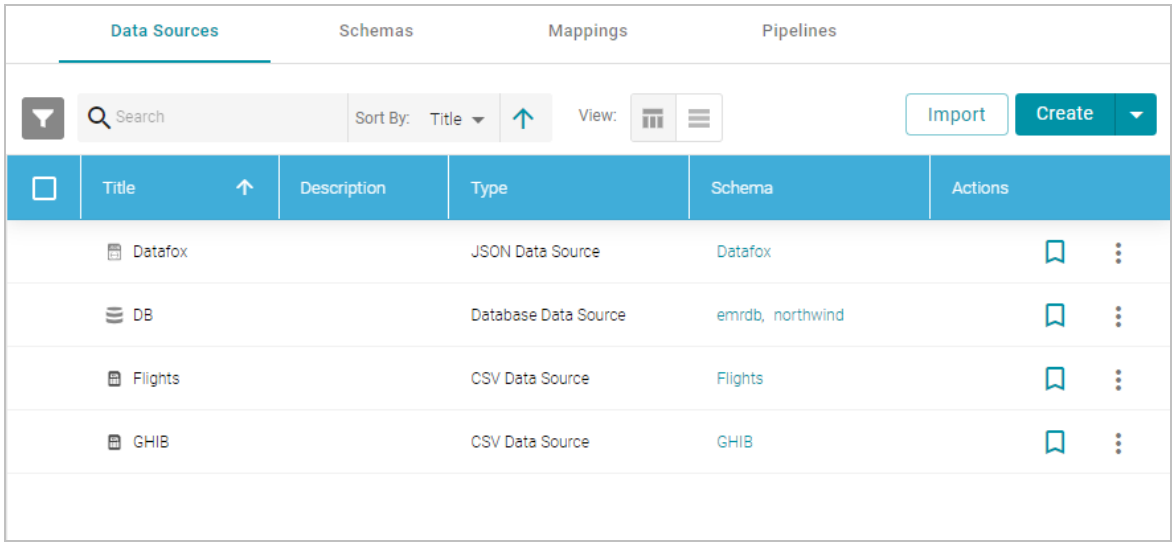
This topic provides instructions for creating and managing data dictionaries.

Creating a Metadata Dictionary from a Schema

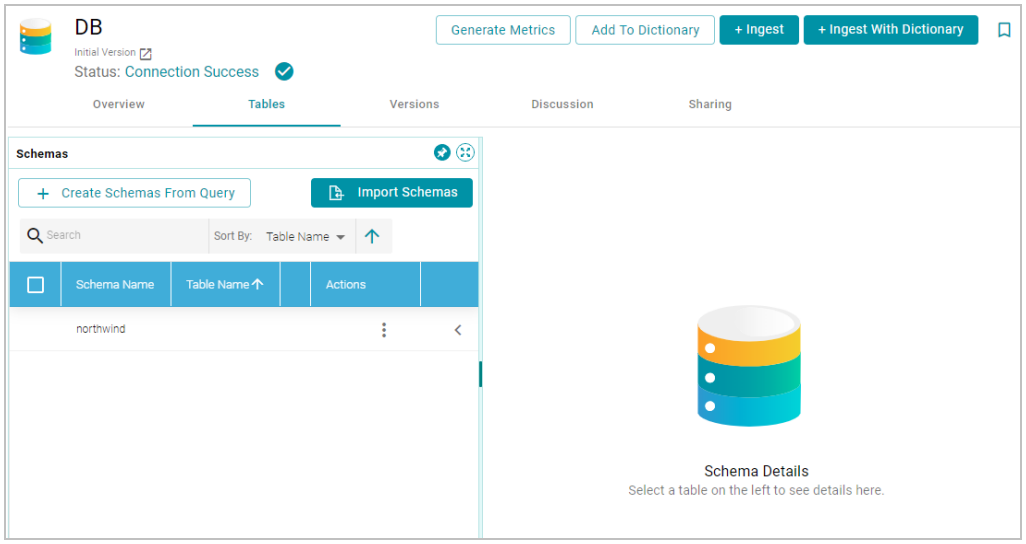
Follow the instructions below to create a new metadata dictionary from a schema.

Note The steps below start with viewing a schema and then adding that schema to a new dictionary. That method allows for flexibility in choosing which schema tables are added to the dictionary. However, you can also create a data dictionary first and then add an entire schema to it. To do so, select **Metadata Hub** from the **Onboard** menu. On the Dictionaries screen, click the **Create** button and select **From Schema**. In the Create Metadata Dictionary dialog box, select the schema to add to the new dictionary.

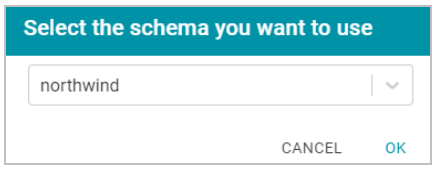
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists the available data sources. For example:



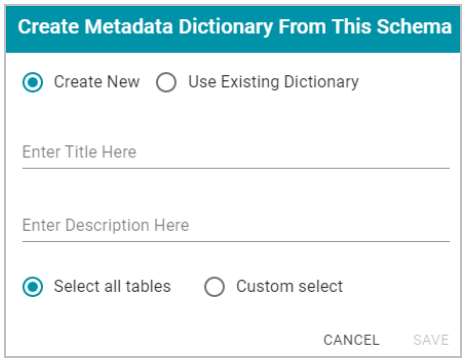
2. On the Data Sources screen, click the name of the data source for which you want to create a data dictionary. Anzo displays the Tables screen for the source. For example:



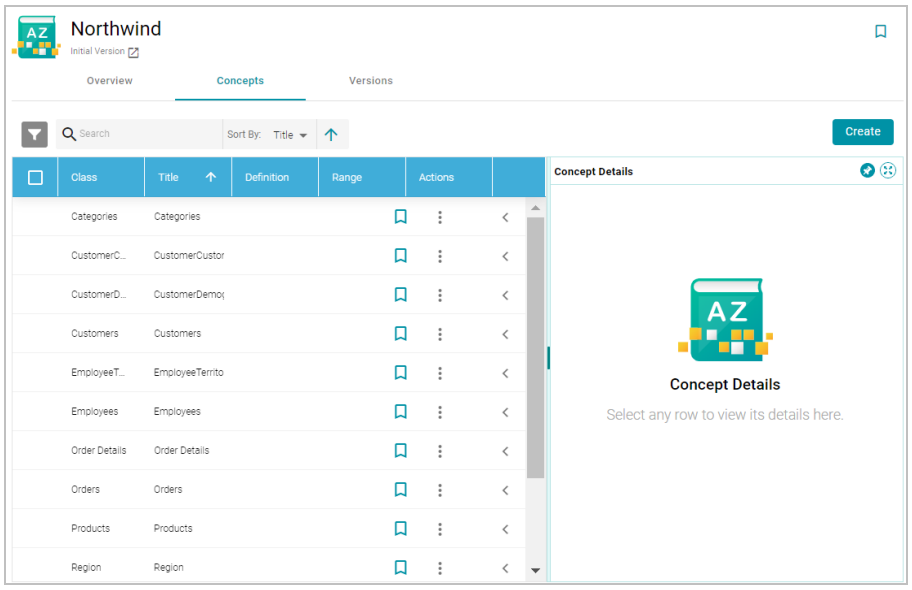
3. Click the **Add To Dictionary** button. If the source has more than one schema, Anzo displays the select schema dialog box. In the drop-down list, select the schema to add to the dictionary, and then click **OK**. For example:



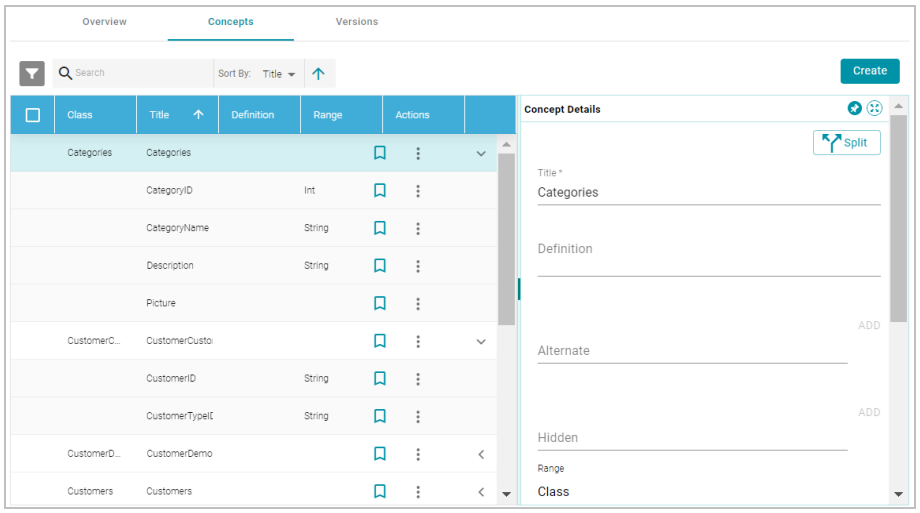
Anzo opens the Create Metadata Dictionary From This Schema dialog box.



4. In the dialog box, leave the **Create New** radio button selected.
5. Enter a name for the dictionary in the **Title** field and specify an optional description in the **Description** field.
6. If you want to include all of the schema tables, select the **Select all tables** radio button. If you want to include a subset of tables, click the **Custom select** radio button and then select each of the tables to add.
7. Click **Save**. Anzo creates the new dictionary and displays the Concepts tab, which lists the class and properties that are derived from the schema. For example:



8. Click a row in the list of concepts on the left to view the concept details on the right side of the screen. Click the **<** character in the table to expand a class concept and view its property concepts. For example:

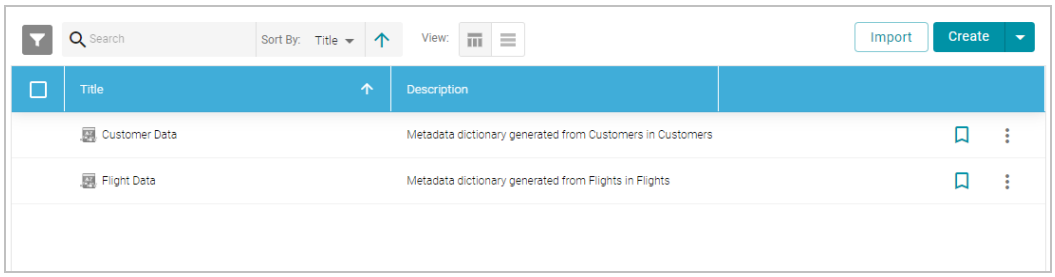


Create and edit concepts as needed. See Defining Concepts in a Metadata Dictionary below for information about working with concepts.

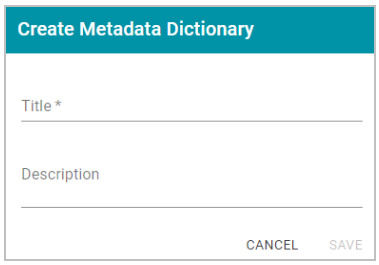
Creating a Metadata Dictionary from Scratch

Follow the instructions below to create a new metadata dictionary from scratch.

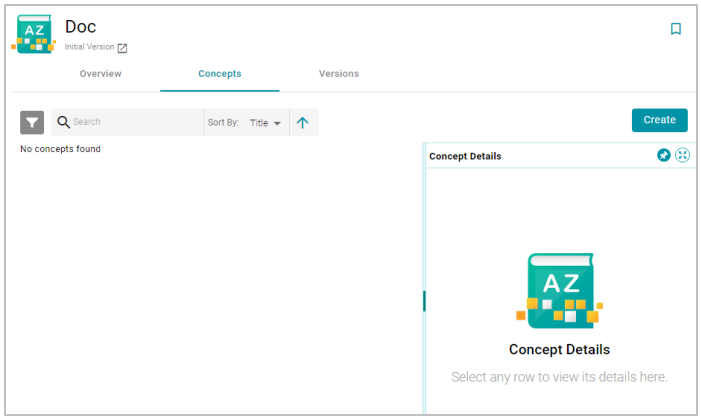
1. In the Anzo console, expand the **Onboard** menu and click **Data Dictionaries**. Anzo displays the Dictionaries screen. For example:



2. Click the **Create** button at the top of the screen, and select **Manual**. Anzo displays the Create Metadata Dictionary dialog box.



3. Type a name for the dictionary in the **Title** field and supply an optional description in the **Description** field.
4. Click **Save** to create the new dictionary. Anzo saves the dictionary and displays the empty Concepts tab. For example:



Create and edit concepts as needed. See Defining Concepts in a Metadata Dictionary below for information about working with concepts.

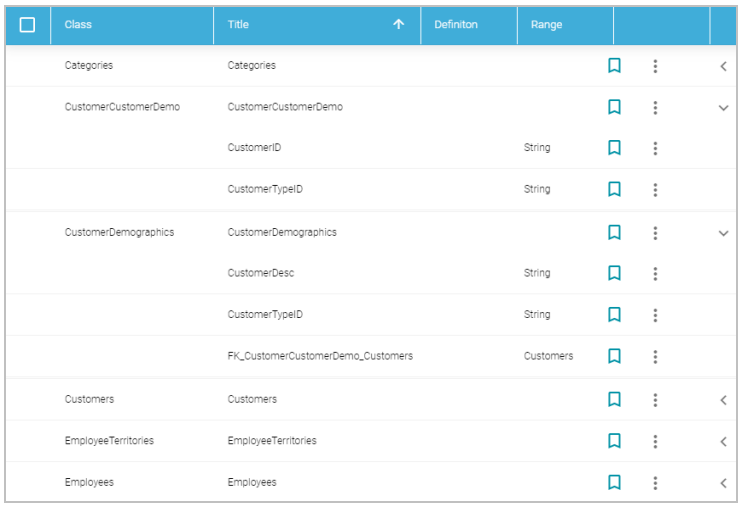
Defining Concepts in a Metadata Dictionary

This section provides examples and instructions for defining the concepts in a data dictionary.

- Merging Concepts
- Creating a New Concept
- Splitting a Concept

Merging Concepts

It is common to encounter schemas where the same concept is conveyed using different names and whose properties are shared across tables in the data source. For example, the concept list below has a "Customers" class, a "CustomerDemographics" class, and a class called "CustomerCustomerDemo."

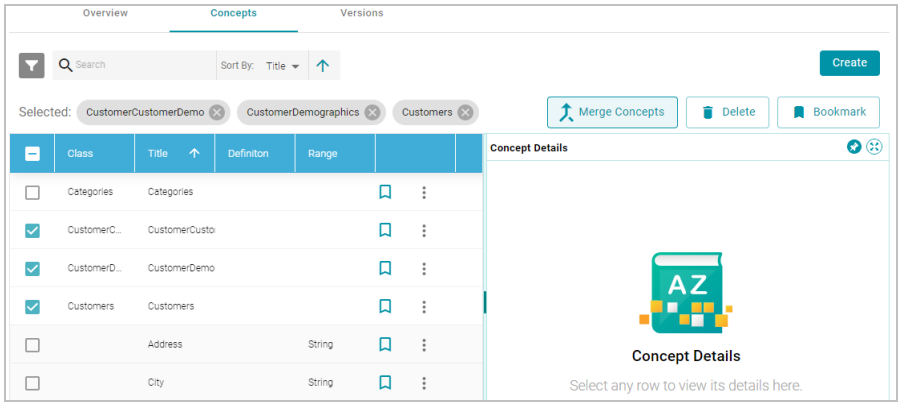


The three customer concepts share properties such as CustomerID and CustomerTypeID, which are foreign key relationships across the tables/classes. The three classes that share the same concept, customer-related data, can be merged into a single concept, creating one class in the model that contains all of the customer-related properties.

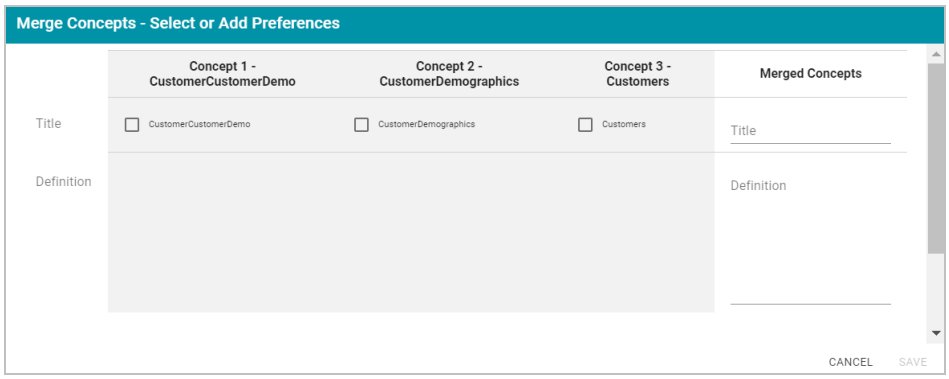
Note: Modifications that you make to a data dictionary do not change the source schema.

To merge concepts

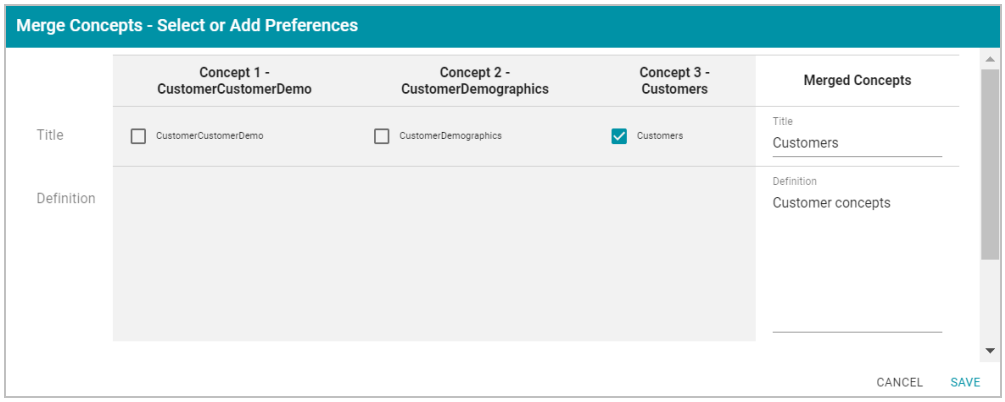
1. Select the checkbox next to each concept that you want to merge, and then click the **Merge Concepts** button on the right side of the screen. For example:



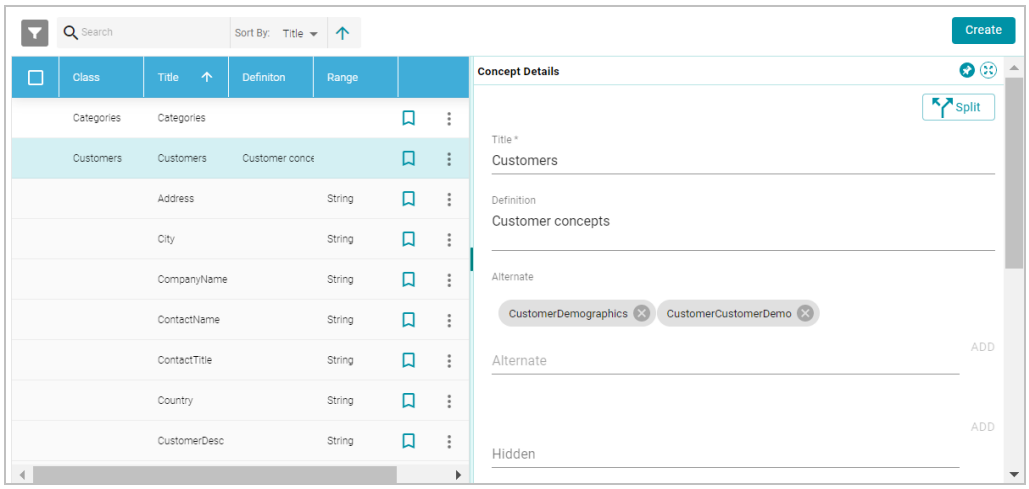
Anzo displays the Merge Concepts dialog box, which lists the classes to merge and enables you to specify the title and description of the new, merged class. For example:



2. On the Merge Concepts screen, if you want to name the merged class with one of the existing class names, select the checkbox next to that class. The **Title** field on the right is populated with that name and you have the option to edit it. If you do not want to use any existing titles, type a new title in the **Title** field.
3. In the **Definition** field, type an optional description for the class. For example:



4. Click **Save** to merge the concepts. Anzo displays a confirmation dialog box that lists the concepts that will be merged and asks if you want to proceed. Click **OK** to complete the merge.
5. When the merge is complete, the concept list is displayed with the changes. You can select the merged class to view and modify concept details on the right side of the screen. For example, the image below shows the details for the merged "Customers" concept. The names of the concepts that were merged to "Customers" are listed in the **Alternate** field. Sources that include those labels, "CustomerDemographics" and "CustomerCustomerDemo," will be mapped to "Customers" in the model. You can edit the Alternate field to add other labels that might come from future source schemas.

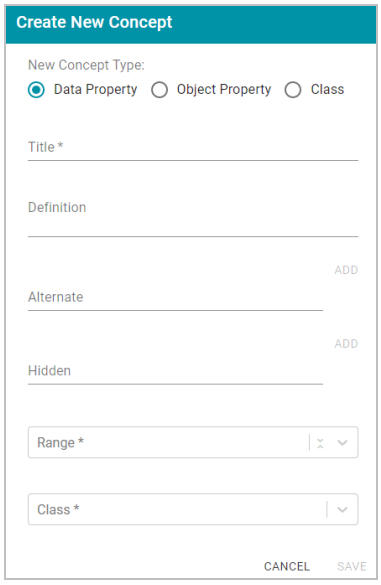


6. From the concept list you can select, edit, and remove classes or properties. For example, since foreign keys were present in the classes that were merged in the sample above, the foreign keys can be removed.

Creating a New Concept

Follow the instructions below to create a new class or property concept in a data dictionary.

1. To add a new concept, click the **Create** button on the right side of the screen. Anzo displays the Create New Concept screen.

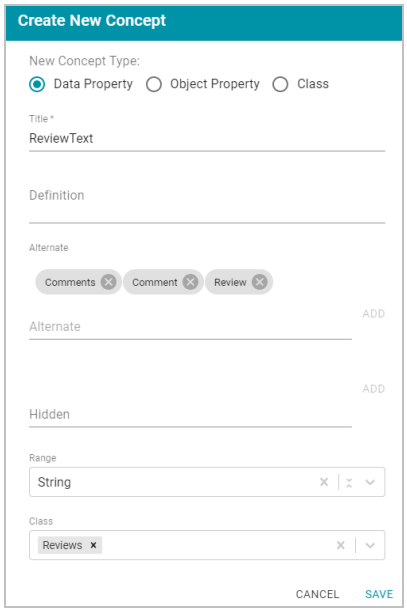


2. Under **New Concept Type**, select the radio button for the type of concept to create:
   - **Data Property**: A data property has an object that is a literal value. For example, a property like FirstName is a data property. Its object has a value such as "Jane."

- **Object Property**: An object property has an object that relates a class to another class. These types of relationships are usually foreign keys in the source. For example, a property like CustomerID might relate the Customers class to the Orders class.
- **Class**: A class concept contains a group of related properties, such as a table name from a source schema.

3. Depending on the type of concept you are creating, specify the appropriate required and optional details:
   - **Title**: The name for this class or property concept.
   - **Definition**: An optional description for the new concept.
   - **Alternate**: An optional list of labels that should map to this new class or property concept.
   - **Hidden**: An optional list of labels that should be hidden in the data model that is generated from this dictionary.
   - **Range**: For property concepts, this required field specifies the data type for the property.
   - **Class**: For property concepts, this required field lists the class or classes the property belongs to.

   For example, the image below creates a data property for reviews of orders. The new property is named ReviewText and "Comment," "Comments," and "Review" are included as Alternate labels so that those properties in source schemas are mapped to ReviewText in the model when the data is onboarded.

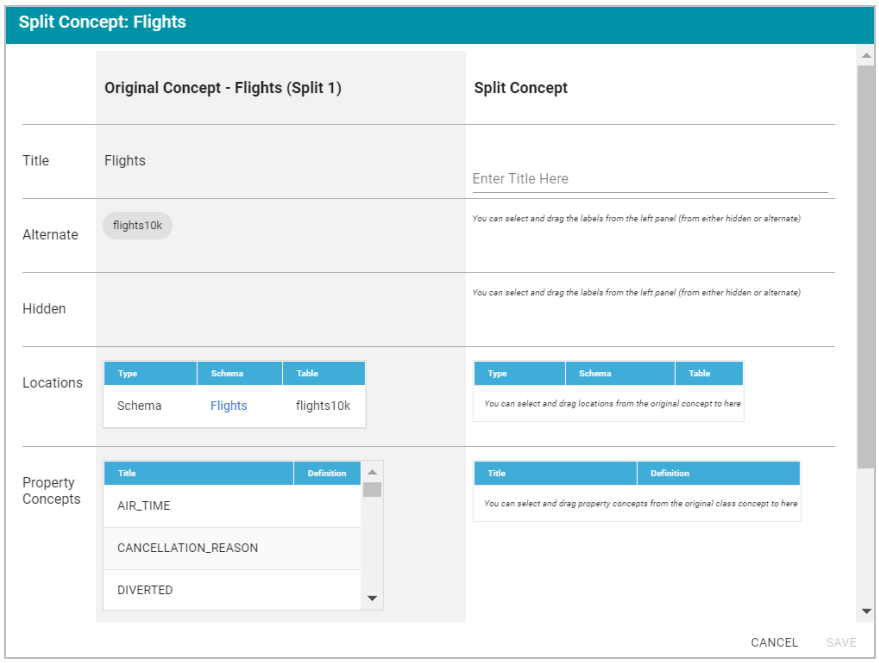


4. Click **Save** to add the new concept to the dictionary.

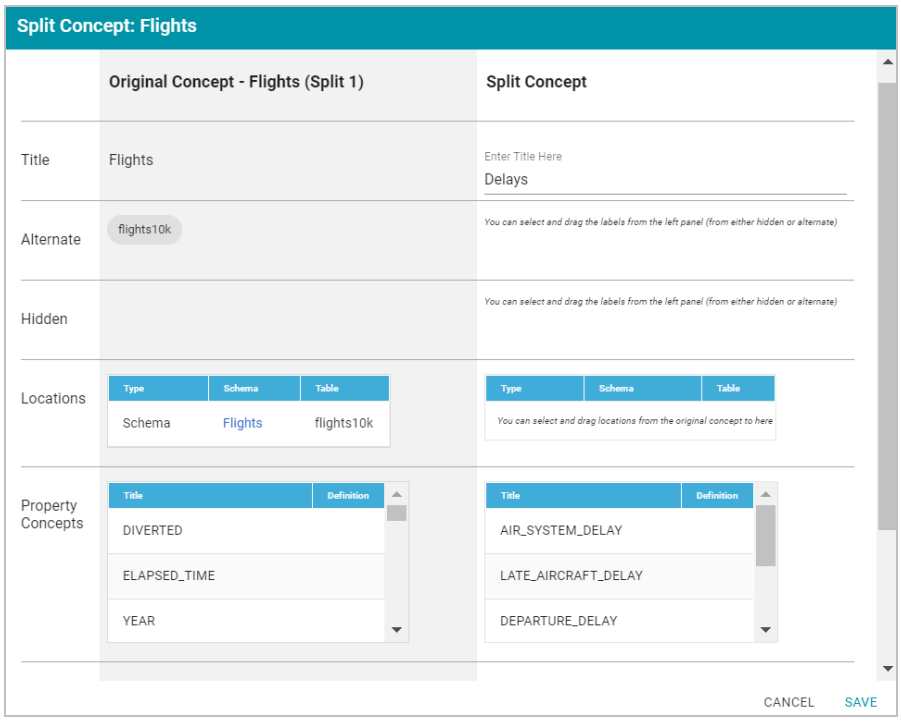
Splitting a Concept

If you determine that one concept should be separated into multiple concepts, you can quickly split the concept and create an additional one by moving any of the original concept's elements to a new concept. Follow the instructions below to split a concept.

1. In the list of concepts, select the row for concept that you want to split and then click the **Split** button in the Concept Details. Anzo displays the Split Concept screen, which lists the original concept on the left and the new concept on the right. For example:



2. Under **Split Concept**, type a name for the new concept in the **Title** field.
3. For the rest of the fields, you can drag elements from the Original Concept to the Split Concept. For example, the image below creates a new Delays class concept and moves the delay-related properties from the original concept to the new concept.



4. When you are finished configuring the new concept, click **Save**. Anzo displays a confirmation dialog box that lists the concepts that will be split and asks if you want to proceed. Click **OK** to complete the split and return to the Concepts screen.

For instructions on onboarding data using a data dictionary, see Ingesting Data with a Data Dictionary.

Related Topics

Ingesting Data with a Data Dictionary

Ingesting Data into Anzo

This topic provides instructions for onboarding data from structured data sources using the ingest process. You can configure the process to automatically generate new mappings, a model, and an ETL pipeline. Or you can configure the process to update existing artifacts by ingesting the new data according to a data dictionary or previously created data model.

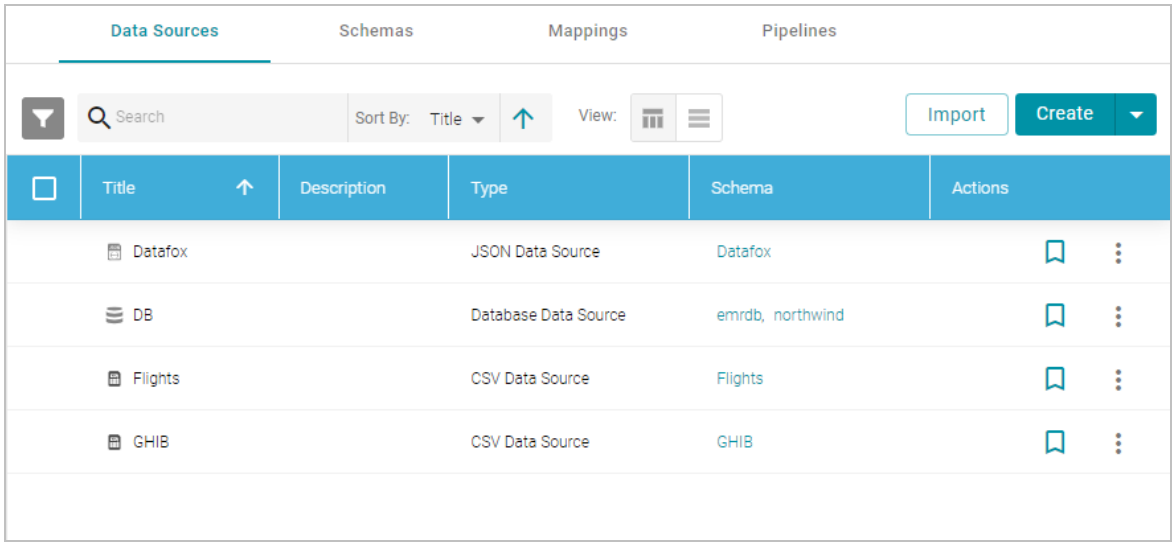
The instructions that you follow depend on whether or not the source you are onboarding has an associated metadata dictionary. Select the appropriate instructions below:

- Ingesting Data without a Data Dictionary
- Ingesting Data with a Data Dictionary

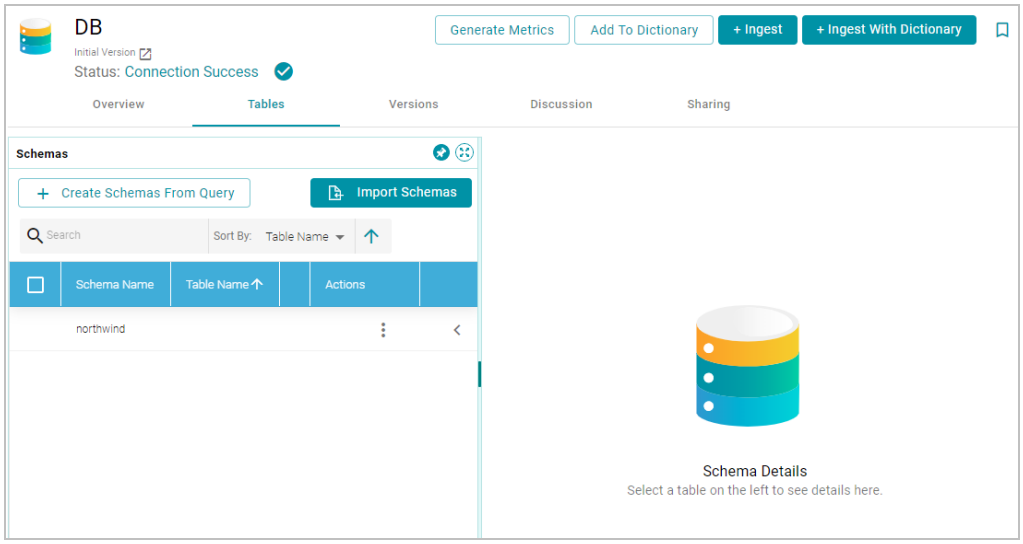
Ingesting Data without a Data Dictionary

Follow the instructions below to ingest data for a schema that does not have an associated data dictionary.

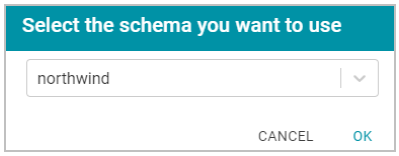
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists the available data sources. For example:



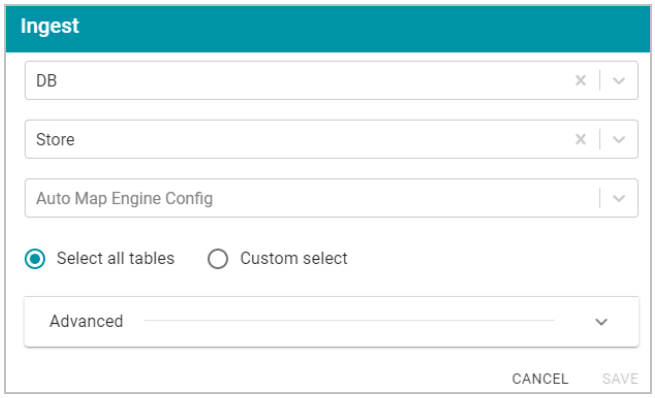
2. On the Data Sources screen, click the name of the data source for which you want to ingest data. Anzo displays the Tables screen for the source. For example:



3. Click the **Ingest** button. If the source has more than one schema, Anzo displays the select schema dialog box. In the drop-down list, select the schema to use, and then click **OK**. For example:



Anzo opens the Ingest dialog box and automatically populates the data source as well as the Anzo data store if only one store is configured. For example:

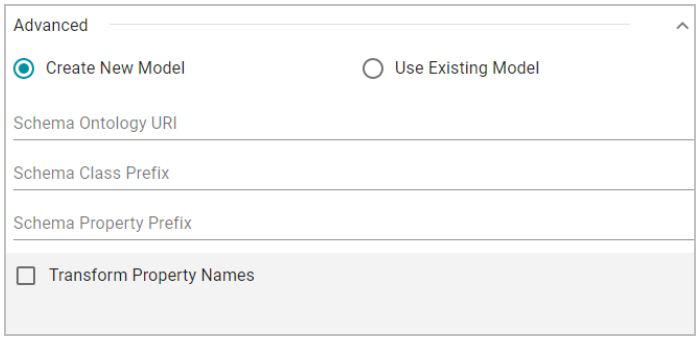


4. If necessary, click the **Anzo Data Store** field and select the data store for this pipeline. For information about creating an Anzo data store, see Creating an Anzo Data Store.
5. Click the **Auto Map Engine Config** field and select the ETL engine to use for this new pipeline.
6. By default, Anzo enables the **Select all tables** radio button to ingest the data for all tables in the schema. If you do not want to add all tables, click the **Custom select** radio button and then select each of the tables to add.
7. Click **Advanced** to view additional configuration options. If this source has not been ingested previously, the workflow defaults to **Create New Model**. If the source has been ingested, the workflow defaults to **Use Existing Model**. Click a description below to view instructions for configuring the workflow to create a new model or use an existing one:

   Create a new model

   Follow the instructions below if you want to create a new model.

   a. Select the **Create New Model** radio button. Additional options become available:



   b. If you want to define the model, class, or property URIs, type the desired prefixes in the Schema Ontology URI, Schema Class Prefix, and Schema Property Prefix fields:

- **Schema Ontology URI**: The URI for the data model. When the Schema Ontology URI field is blank, Anzo generates the model URI for this schema with the following format:

  ```
  http://cambridgesemantics.com/ont/autogen/xx/schema_name
  ```

  Where *xx* is a hash snippet based on the model's globally unique identifier (GUID). If you want Anzo to use a specific URI format when creating the model for this schema, you can type that URI into the Schema Ontology URI field. For example, you can specify a URI such as `http://mycompany.com.ontology/movies`. This format results in an ontology URI of `http://mycompany.com.ontology/movies`.

  Important: Make sure that you type a unique Schema Ontology URI for each schema. Schemas with the same model URI overwrite each other when the projects run.

- **Schema Class Prefix**: The URI prefix format to use for classes in the data model. When the Schema Class Prefix field is blank, Anzo generates the ontology's class URIs using the following format:

  ```
  http://cambridgesemantics.com/ont/autogen/xx/schema_name#class_
  name
  ```

  Where *xx* is a hash snippet based on the model's GUID. If you want Anzo to use a specific URI format when creating classes in the model, you can type that URI into the Schema Class Prefix field. For example, you can specify a URI such as `http://mycompany.com.ontology/class`. This format results in class URIs like `http://mycompany.com.ontology/class#className`.

  Tip: To generate models from multiple sources and use the same class URIs across models, you can set the Schema Class Prefix to the same value across schemas.

- **Schema Property Prefix**: The URI prefix format to use for properties in the data model. When the Schema Property Prefix field is blank, Anzo generates the ontology's property URIs using the following format:

  ```
  http://cambridgesemantics.com/ont/autogen/xx/schema_name#class_
  name_property_name
  ```

  Where *xx* is a hash snippet based on the model's GUID. If you want Anzo to use a specific URI format when creating properties in the model, you can type that URI into the Schema Property Prefix field. For example, you can specify a URI such as

`http://mycompany.com.ontology/property`. This format results in property URIs like `http://mycompany.com.ontology/property#`*`className_propertyName`*.

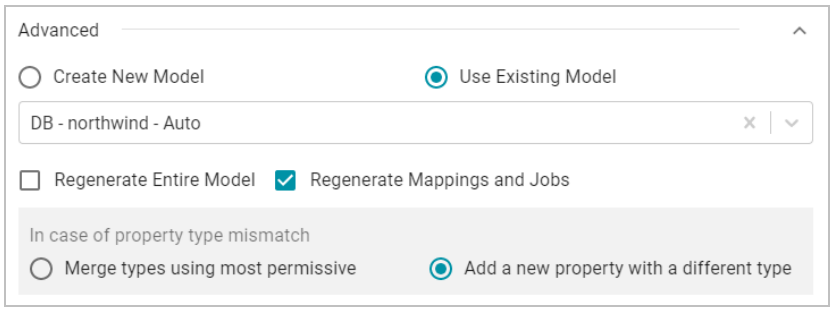
Tip: To generate models from multiple sources and use the same property URIs across models, you can set the Schema Property Prefix to the same value across schemas.

c. If you want to transform property names to upper or lower case letters, select the **Transform Property Names** checkbox. Then select the **To lowercase** radio button if you want to convert property names to lowercase or select the **To UPPERCASE** radio button if you want to convert property names to upper-case.

Use an existing model for this workflow

Follow the instructions below if you want to associate this source with an existing model.

a. Select the **Use Existing Model** radio button. Additional options become available:



b. Select the model to use from the **Model** drop-down list.

c. Specify whether to regenerate artifacts and which artifacts to regenerate. The list below describes each of the options:

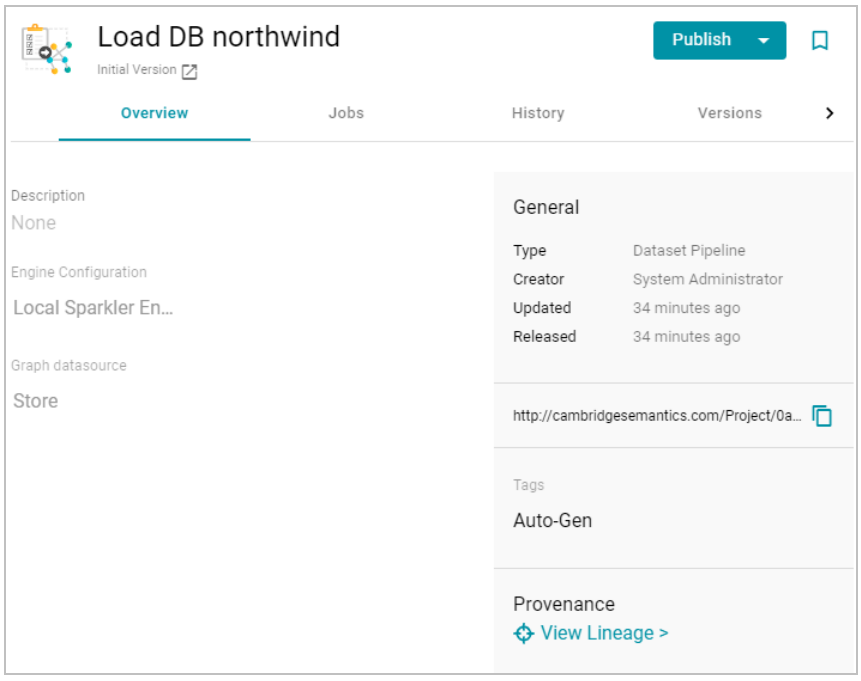
- **Regenerate Entire Model**: Selecting this option means that Anzo deletes all entities from the existing model and recreates them. The model that results from the current ingestion process will contain only the data from the current process. For example, if a previous run generated a model that contains classes A, B, and C, and the current data contains Classes C, D, and E, selecting **Regenerate Entire Model** results in a model that contains only classes C, D, and E. If **Regenerate Entire Model** is NOT selected, the resulting model will contain classes A, B, C, D, and E.
- **Regenerate Mappings and Jobs**: Selecting this option means that Anzo deletes all entities from the existing mappings and jobs and recreates them. The artifacts that result from the current ingestion process will contain only the data from the current process. For example, if a previous run generated mappings and jobs that contain tables A and B and the current run is ingesting tables C and D, selecting **Regenerate Mappings and Jobs** results in artifacts that contain only

tables C and D. If **Regenerate Mappings and Jobs** is NOT selected, the resulting artifacts contain tables A, B, C, and D.

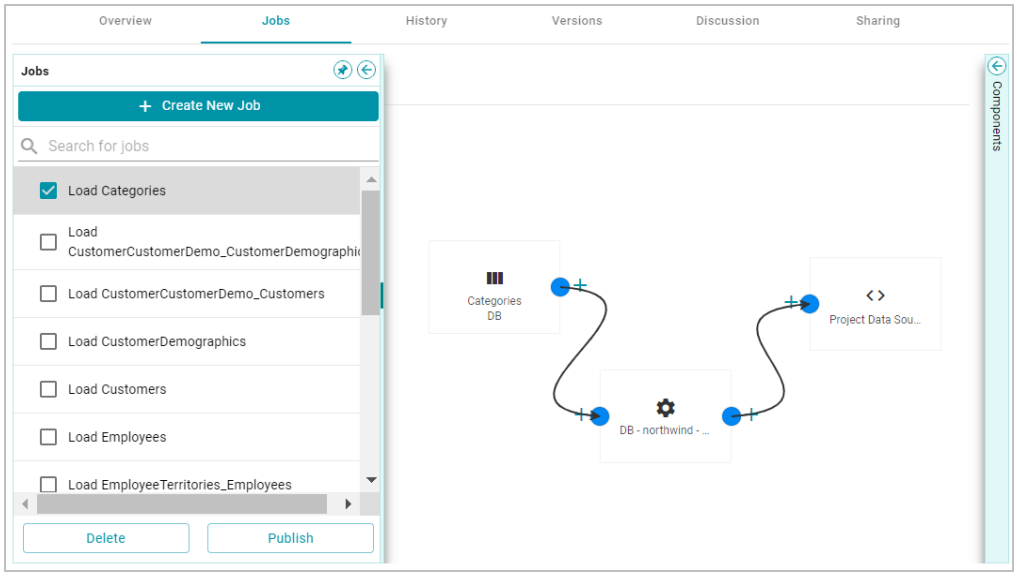
d. Specify what Anzo should do if there is a property type mismatch between the existing model and the new schema:
   - **Merge types using most permissive**: Anzo looks at the inferred types in both schemas and chooses the type that covers all inputs. In most cases Anzo sets the type to **String**.
   - **Add a new property with a different type**: If Anzo encounters a type mismatch, it adds a new property with the new type to the existing model.

Note: When associating column names in the new schema with the existing model, the match is case-insensitive. Anzo matches the names based on the spelling. For example, "myInt" matches "MYint."

8. Click **Save**. Anzo creates a pipeline (or updates the existing one) and generates or updates the model and mappings according to the options you specified.
9. In the main navigation menu under **Onboard**, click **Structured Data**. Then click the **Pipelines** tab.
10. Click the name of the pipeline to run. Anzo displays the pipeline overview screen. For example:



11. If you would like to see the jobs that Anzo created for this data source, click the **Jobs** tab. The jobs are listed on the left side of the screen. A job exists for each of the tables that were imported. For example:



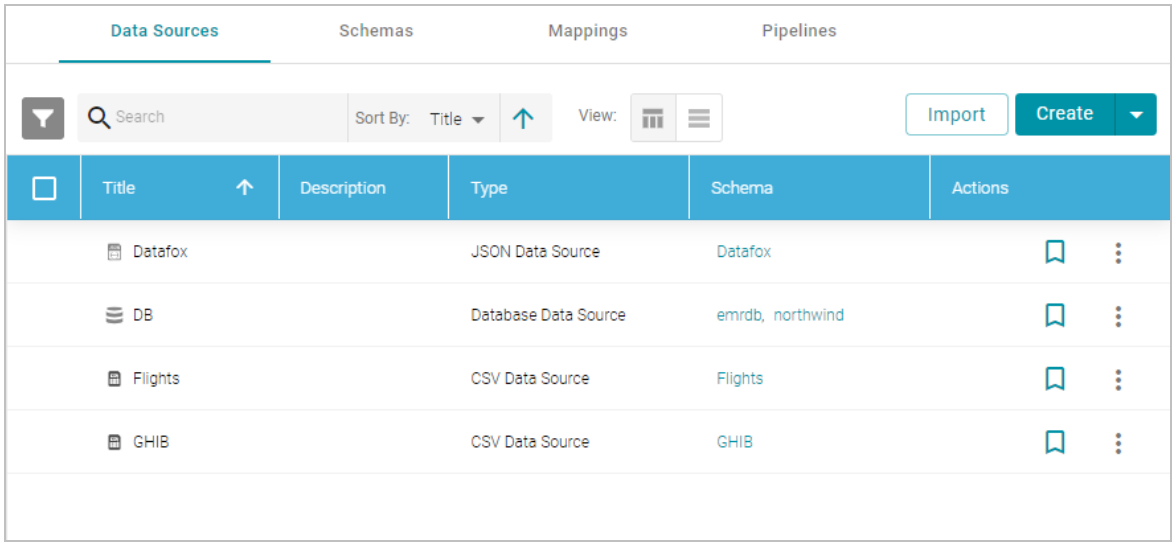
12. To run all of the jobs, click the **Publish** button at the top of the screen. To publish a subset of the jobs, select the checkbox next to each job that you want to run and then click the **Publish** button below the list of jobs. Anzo runs the pipeline and generates the resulting RDF files in a new subdirectory under the specified base directory for the data store.

When the pipeline finishes, you can generate metrics on the graph data to start exploring the data. See Generating Graph Data Metrics for more information. You can also add the new data set to a graphmart and load it to AnzoGraph so that you can explore and analyze the data using Hi-Res Analytics. See Creating a Graphmart for instructions.

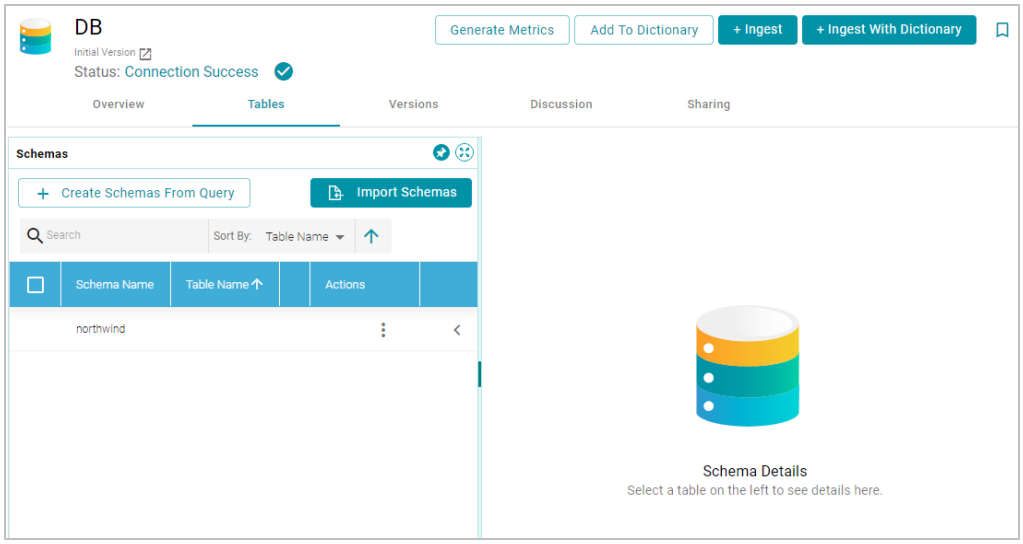
Ingesting Data with a Data Dictionary

Follow the instructions below to ingest data for a schema that has an associated data dictionary.

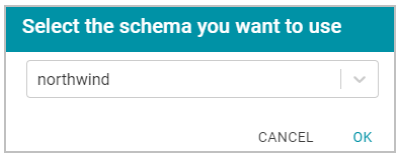
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists the available data sources. For example:



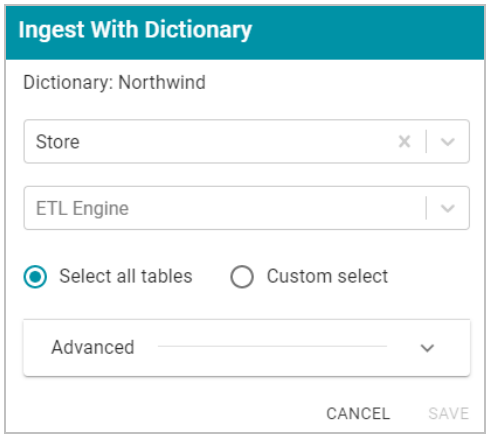
2. On the Data Sources screen, click the name of the data source for which you want to ingest data. Anzo displays the Tables screen for the source. For example:



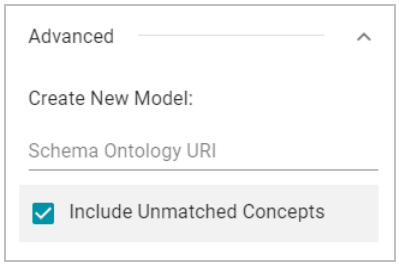
3. Click the **Ingest With Dictionary** button to build the model and mappings from the data dictionary. If the source has more than one schema, Anzo displays the select schema dialog box. In the drop-down list, select the schema to use, and then click **OK**. For example:



Anzo opens the Ingest With Dictionary dialog box, which lists the dictionary to use and automatically populates the Anzo data store if only one store is configured. For example:



4. If necessary, click the **Anzo Data Store** field and select the Anzo data store for this pipeline. For information about creating an Anzo data store, see Creating an Anzo Data Store.
5. Click the **ETL Engine** field and select the ETL engine to use for this new pipeline.
6. By default, Anzo enables the **Select all tables** radio button to ingest the data for all tables in the schema. If you do not want to add all tables, click the **Custom select** radio button and then select each of the tables to add.
7. To view model and dictionary options for this pipeline, expand the **Advanced** section of the dialog box.
    - **If this source has not been previously ingested and no model exists**, Anzo displays the following advanced options:



        - **Create New Model**: Anzo will create a new model and you have the option to specify a custom Schema Ontology URI.
        - **Schema Ontology URI**: The URI for the data model. When the Schema Ontology URI field is blank, Anzo generates the model URI for this schema with the following format:

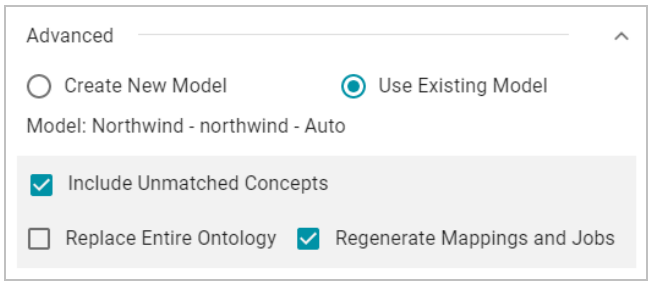
          ```
          http://cambridgesemantics.com/ont/autogen/xx/schema_name
          ```

          Where *xx* is a hash snippet based on the model's globally unique identifier (GUID). If you want Anzo to use a specific URI format when creating the model for this schema, you can specify that URI in the Schema Ontology URI field. For example, you can specify a URI such as `http://mycompany.com.ontology/movies`, which results in a model URI of `http://mycompany.com.ontology/movies`.

Important: Make sure that you type a unique Schema Ontology URI for each schema. Schemas with the same model URI overwrite each other when the pipelines are run.

- **Include Unmatched Concepts**: This option specifies whether to ingest new data that does not map to the concepts that are defined in the dictionary. For example, the dictionary includes a class concept with 10 properties that map to columns in the schema. The new data contains 15 columns, 5 of which are not mapped to properties in the dictionary. If **Include Unmatched Concepts** is enabled, Anzo will ingest the data for all 15 columns and update the model to include the 5 unmatched properties. The 5 new properties are added to a new subclass of the class that is defined in the dictionary. If **Include Unmatched Concepts** is disabled, Anzo will ingest only the 10 columns that match the concepts in the dictionary and will not add new properties to the model.

- **If this source has been previously ingested and a model exists**, Anzo displays the following advanced options:



  - **Create New Model**: Select this option if you want Anzo to create a new model. Selecting Create New Model presents the option to specify a custom Schema Ontology URI.
  - **Schema Ontology URI**: The URI for the data model. When the Schema Ontology URI field is blank, Anzo generates the model URI for this schema with the following format:

    ```
    http://cambridgesemantics.com/ont/autogen/xx/schema_name
    ```

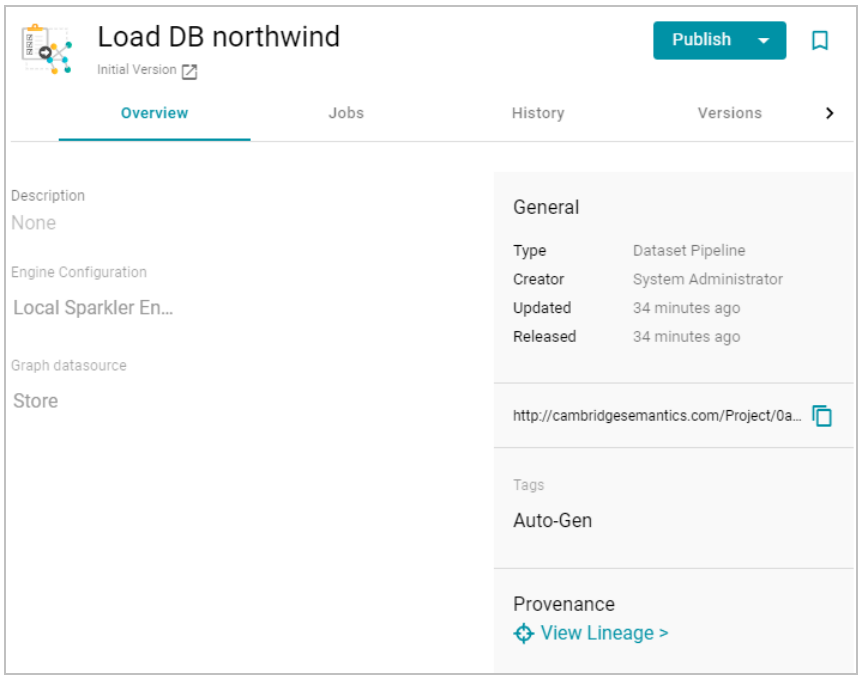
    Where *xx* is a hash snippet based on the model's globally unique identifier (GUID). If you want Anzo to use a specific URI format when creating the model for this schema, you can specify that URI in the Schema Ontology URI field. For example, you can specify a URI such as `http://mycompany.com.ontology/movies`, which results in a model URI of `http://mycompany.com.ontology/movies`.

    Important: Make sure that you type a unique Schema Ontology URI for each schema. Schemas with the same model URI overwrite each other when the pipelines are run.

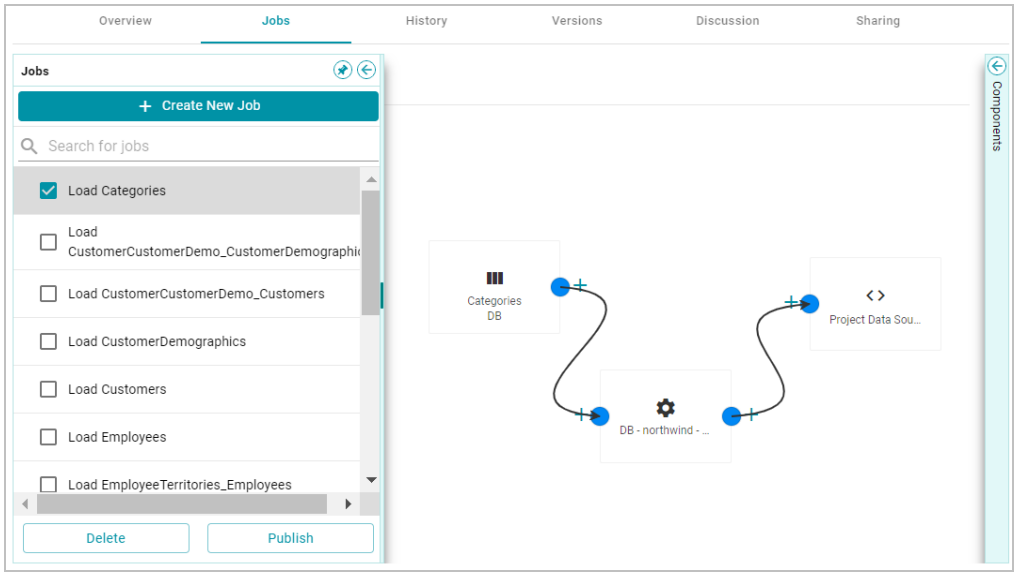
- **Use Existing Model**: Select this option if you want to reuse the existing model. Anzo displays the name of the model that will be used and presents additional model-related options at the bottom of the screen.
- **Include Unmatched Concepts**: This option specifies whether to ingest new data that does not map to the concepts that are defined in the dictionary. For example, the dictionary includes a class concept with 10 properties that map to columns in the schema. The new data contains 15 columns, 5 of which are not mapped to properties in the dictionary. If **Include Unmatched Concepts** is enabled, Anzo will ingest the data for all 15 columns and update the model to include the 5 unmatched properties. The 5 new properties are added to a new subclass of the class that is defined in the dictionary. If **Include Unmatched Concepts** is disabled, Anzo will ingest only the 10 columns that match the concepts in the dictionary and will not add new properties to the model.
- **Replace Entire Ontology**: Selecting this option means that Anzo deletes all entities from the existing model and recreates them. The model that results from the current ingestion process will contain only the data from the current process. For example, if a previous run generated a model that contains classes A, B, and C, and the current data contains Classes C, D, and E, selecting **Replace Entire Ontology** results in a model that contains only classes C, D, and E. If **Replace Entire Ontology** is NOT selected, the resulting model will contain classes A, B, C, D, and E.
- **Regenerate Mappings and Jobs**: Selecting this option means that Anzo deletes all entities from the existing mappings and jobs and recreates them. The artifacts that result from the current ingestion process will contain only the data from the current process. For example, if a previous run generated mappings and jobs that contain tables A and B and the current run is ingesting tables C and D, selecting **Regenerate Mappings and Jobs** results in artifacts that contain only tables C and D. If **Regenerate Mappings and Jobs** is NOT selected, the resulting artifacts contain tables A, B, C, and D.

8. Click **Save**. Anzo creates a pipeline (or updates the existing one) and generates or updates the model and mappings according to the options you specified.
9. In the main navigation menu under **Onboard**, click **Structured Data**. Then click the **Pipelines** tab.

10. Click the name of the pipeline to run. Anzo displays the pipeline overview screen. For example:



11. If you would like to see the jobs that Anzo created for this data source, click the **Jobs** tab. The jobs are listed on the left side of the screen. A job exists for each of the tables that were imported. For example:



12. To run all of the jobs, click the **Publish** button at the top of the screen. To publish a subset of the jobs, select the checkbox next to each job that you want to run and then click the **Publish** button below the list of jobs. Anzo runs the pipeline and generates the resulting RDF files in a new subdirectory under the specified base directory for the data store.

When the pipeline finishes, you can generate metrics on the graph data to start exploring the data. See Generating Graph Data Metrics for more information. You can also add the new data set to a graphmart and load it to AnzoGraph so that you can explore and analyze the data using Hi-Res Analytics. See Creating a Graphmart for instructions.

Related Topics

Generating Graph Data Metrics

Creating a Graphmart

Working with Pipelines

Using Data Dictionaries

Working with Schemas

The schemas manager enables users to work with the schemas in Anzo. You can add or edit tables, change the foreign and primary keys, create custom URIs for models, and you can control user access to the schema. The topics in this section provide information about modifying schemas and importing or creating new schemas for existing data sources.

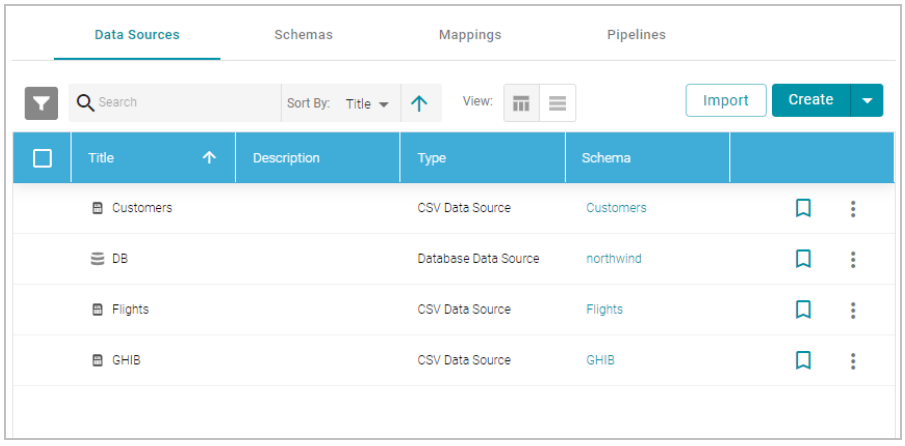
For information about modifying the mappings associated with a schema, see Working with Mappings.

- Importing or Creating a Schema

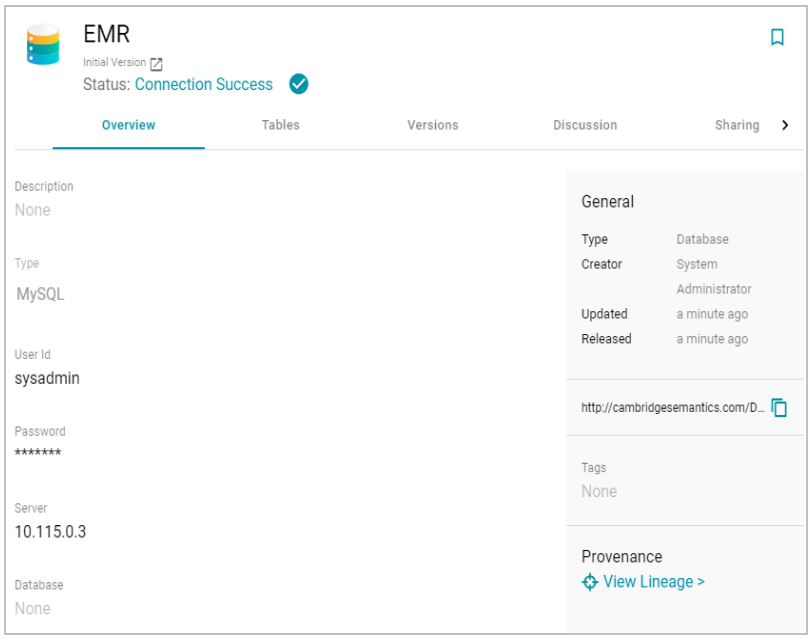
Importing or Creating a Schema

Follow the instructions below to import a predefined schema or create a custom schema for an existing structured data source.

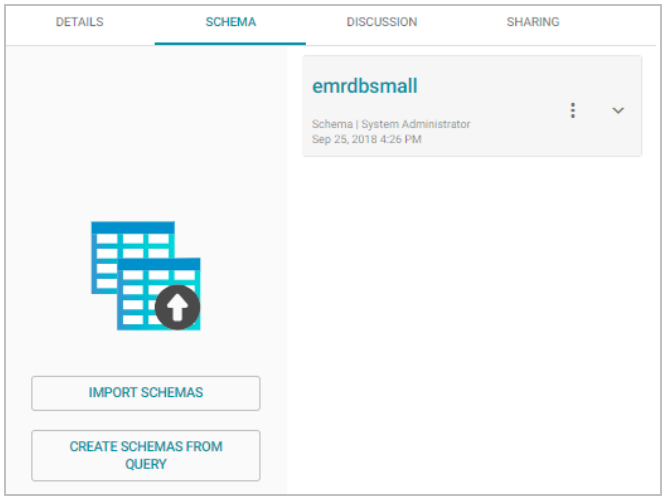
1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Anzo displays the Data Sources screen, which lists any existing data sources. For example:



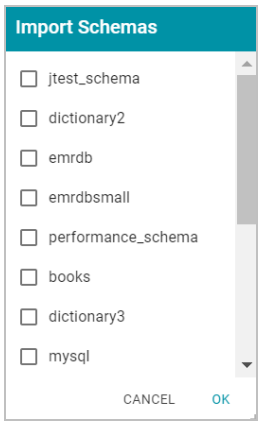
2. On the Data Sources screen, click the name of the source for which you want to import or create a schema. Anzo displays the details for the source. For example:



3. Click the **Schemas** tab. Anzo lists any schemas associated with the source. For example:



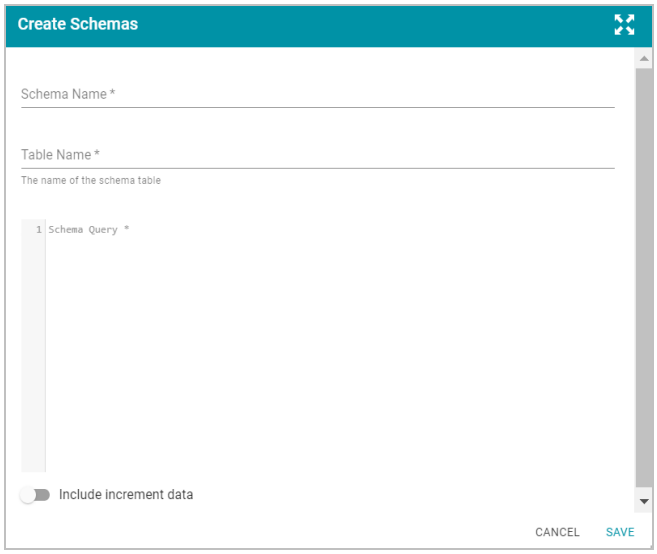
4. If you want to import one or more predefined schemas, follow these steps:
    a. Click the **Import Schemas** button. Anzo displays the Import Schemas dialog box. For example:



    If you do not see a schema that you expect to see, make sure that you have the necessary access to the data source.

    b. Select the checkbox next to each schema that you want to import, and then click **OK**. Anzo imports the selected schemas.

5. If you want to create a schema by writing an SQL statement that specifies a customized set of data to retrieve, follow these steps:
    a. Click the **Create Schemas From Query** button. Anzo displays the Create Schemas dialog box:



    b. In the Create Schemas dialog box, specify a name for the schema in the **Schema Name** field.
    c. In the **Table Name** field, specify a name for the schema table.
    d. Type the SQL statement in the text box. The statement can include any functionality that the database supports. Anzo does not validate the SQL. For information about writing a schema query that onboards

data from a database incrementally, see Creating an Incremental Schema.

e. Click **Save** to save the query.

Related Topics

Generating Source Data Metrics

Working with Mappings

Anzo enables you to map and transform your data with the Anzo for Office plugin for Microsoft Excel. For instructions on installing Anzo for Office, see Installing the Anzo for Excel and Office Plugins. The topics in this section provide information about creating and editing the mappings that describe the relationships between your schemas and models:

- Creating a New Mapping
- Configuring Mappings to Ingest a Subset of the Source Data
- Transforming Data in Mappings
- Supported Mapping Functions

Creating a New Mapping

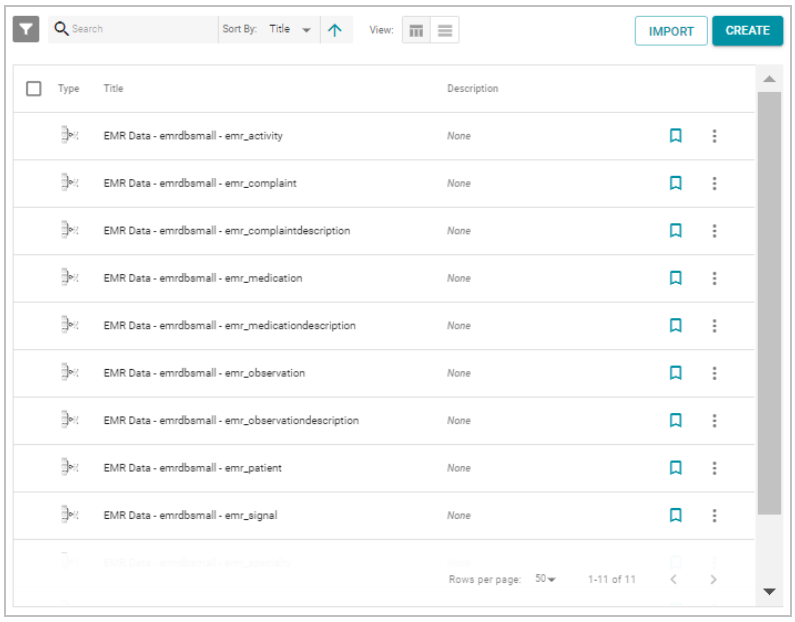
This topic provides instructions for using the Anzo console and the Anzo for Office Excel plugin to create a new basic mapping. Typically users create one mapping for each target and source pair. For example, if you have a project that ingests data from 10 tables in a source or 10 CSV files, the project will likely include 10 mappings. You can create mappings where multiple sources map to one target, but one mapping cannot include multiple targets.

Follow these steps to create a new mapping:

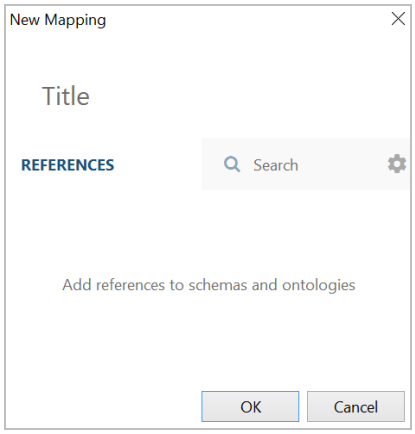
1. Create a Mapping and Select References
2. Define the Source for the Mapping
3. Define the Target for the Mapping
4. Map the Source Elements to Target Elements

Create a Mapping and Select References

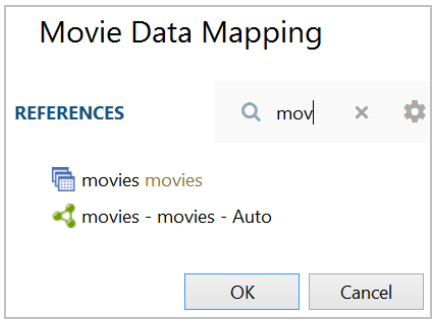
1. In the Anzo application, expand the **Onboard** menu item and click **Structured Data**. Then click the **Mappings** tab. Anzo displays the Mappings screen, which lists any existing mappings. For example:



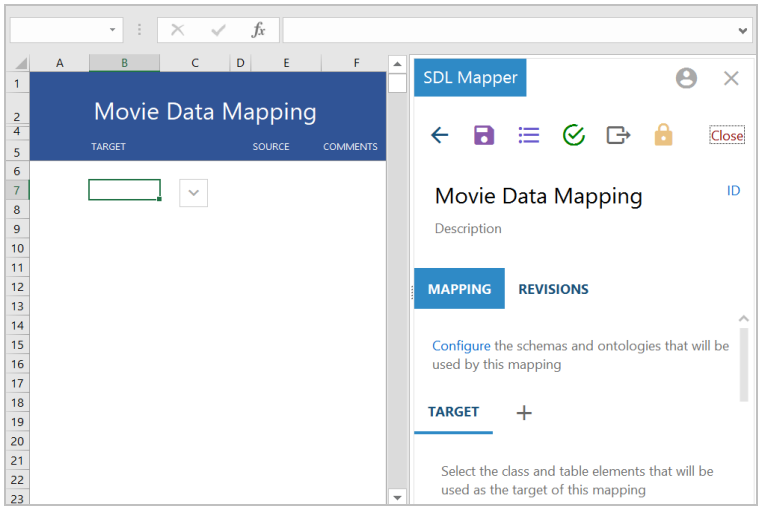
2. Click the **Create** button at the top of the screen. Anzo opens the Anzo for Office plugin in Microsoft Excel and prompts you to enter the Anzo server connection information.
3. Provide your server connection and login information and then click the arrow icon ( ) to connect to the server and open the mapping tool. Anzo displays the New Mapping dialog box.



4. In the **Title** field, type a name for the mapping.
5. In the **References** field, select the schemas and models that the new mapping should have access to. Type a value to search for available sources. If you want to include system items in the search, you can click the cog icon ( ) to select it. Anzo lists the sources that match the search text.



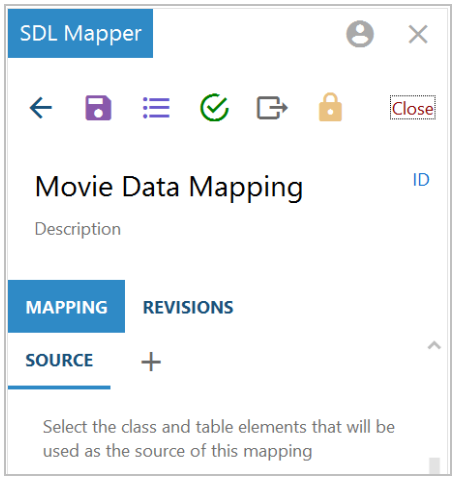
6. Hover over a source in the list of references to display the plus icon (+). Click the icon for each source that you want to add as a reference, and then click **OK**. The new mapping opens. The mapping workbook is in the center pane and the mapping menu and configuration details are displayed in the right pane. For example:



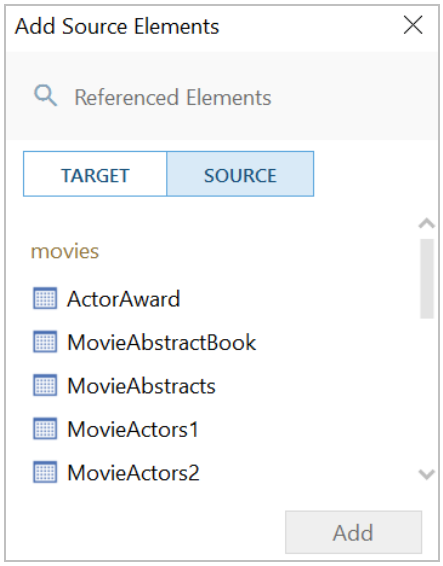
Define the Source for the Mapping

Complete these steps to define the source to use for the mapping:

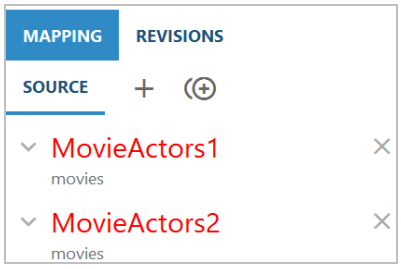
1. In the right pane, scroll to the **Source** section in the **Mapping** tab.



2. Click the plus icon (+) next to **Source**. Anzo opens the Add Source Elements dialog box.



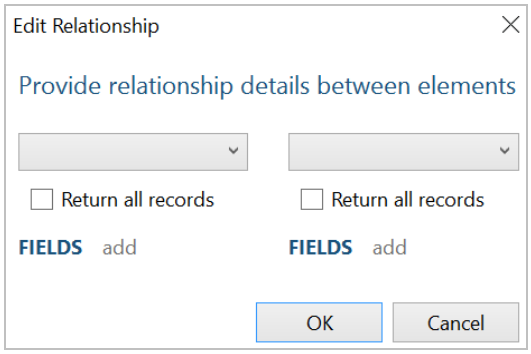
3. Select a source to add to the mapping and click **Add**. If you want to add additional sources, select another source and click **Add** again. Anzo adds the source to the list under Source. You can click the X icon to the right of a source name to delete that source from the list.



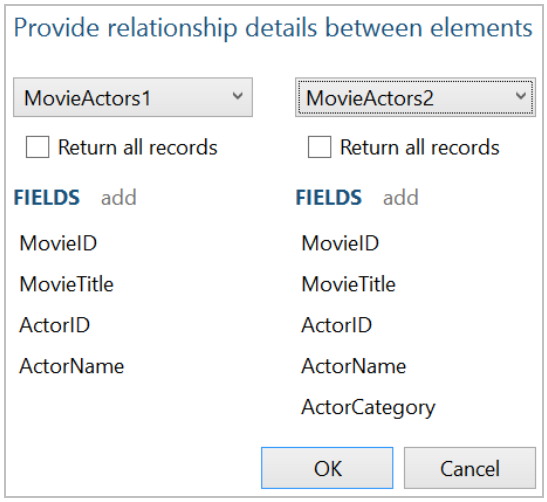
4. When you are finished adding sources, close the Add Source dialog box.
5. If you added one source, proceed to the next step. If you added multiple sources, Anzo displays an Add Relationship icon ( ) next to the plus icon so that you can create a relationship between the sources by specifying the join criteria.

   Follow these steps to specify relationships:

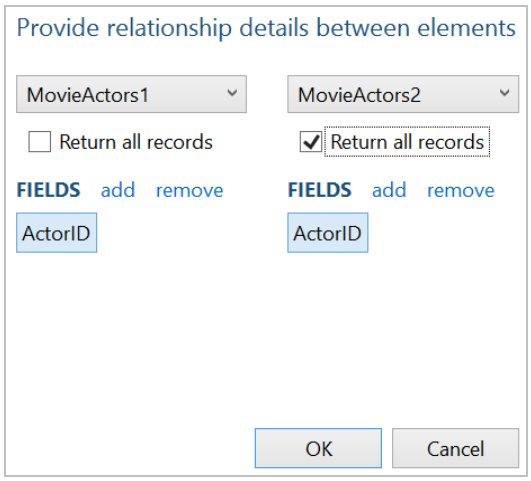
a. Click the Add Relationship icon ( ). Anzo displays the Edit Relationship dialog box.



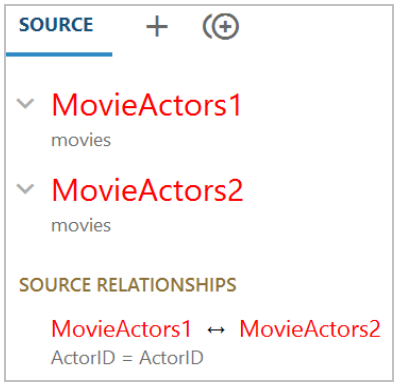
b. In the drop-down lists at the top of the dialog box, select one source on the left and the other source on the right. Anzo displays the fields under each source.



c. Determine which field to join on from each source, and then select those fields. Double-click a field on the left to select it, then double-click the join field on the right. You can use the **Return all records** check boxes above the field lists to specify whether to return all records from either the right or the left side if no matching field is found.



d. Click **OK** to create the relationship and close the Edit Relationship dialog box. Anzo lists the relationship definition at the bottom of the right pane.



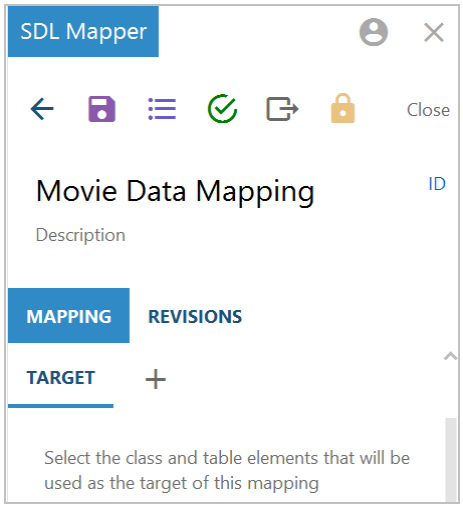
If you have additional sources to join, click the Add Relationship icon again and repeat these steps to relate each source.

6. Click the save icon ( ) at the top of the mapping configuration pane to save the mapping changes.

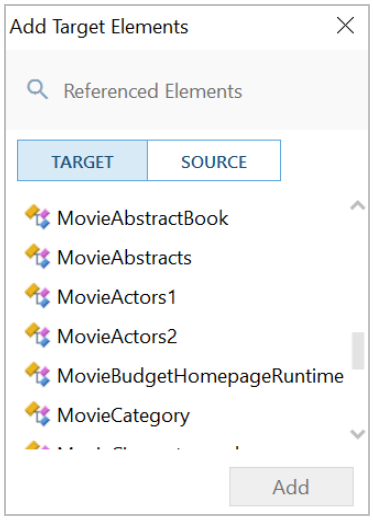
Define the Target for the Mapping

Complete these steps to define the class and table elements that the mapping should target:

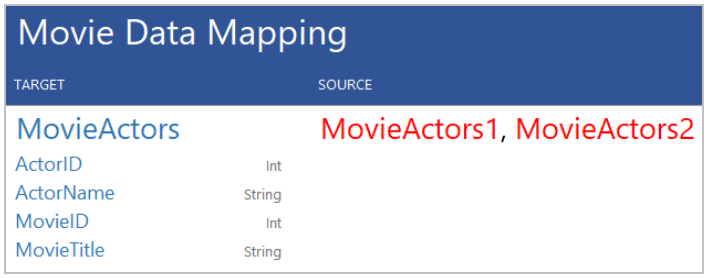
1. In the right pane, scroll to the **Target** section in the **Mapping** tab.



2. Click the plus icon (+) next to **Target**. Anzo opens the Add Target Elements dialog box.



3. Select the model or table that you want to map to and click **Add**. Anzo adds the target to the Target list in the right pane and populates the workbook with the target elements. For example:



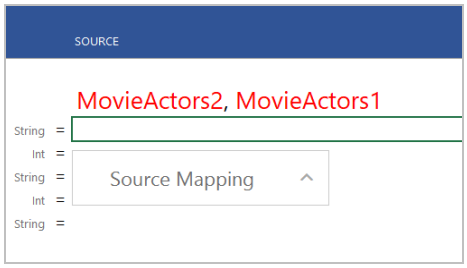
4. Close the Add Target dialog box and click the save icon ( ) at the top of the mapping configuration pane to save the mapping changes.

Note For information about setting up parameters to ingest a subset of the source data, see Configuring Mappings to Ingest a Subset of the Source Data.

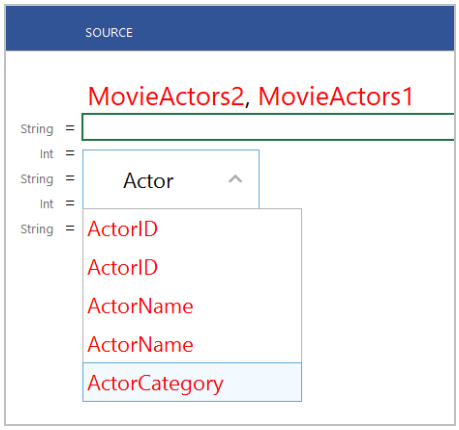
Map the Source Elements to Target Elements

In the workbook, follow these steps to complete the mapping by specifying which source element maps to each target element:

1. Click in a source cell next to a target field. Anzo displays the **Source Mapping** text box below the cell.

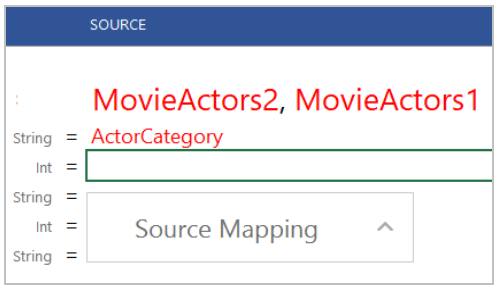


2. In the Source Mapping text box, start typing the source field name. Anzo displays a list of fields that match the text.



Important Type all values in the Source Mapping text box. Do not type in any cells. Any text typed in a cell directly is invalid and can cause issues with the mapping.

3. In the list of results, click the source field that maps to the target. Anzo adds the field name to the cell and opens the Source Mapping text box for the next source.



4. If you chose a model or ontology as the target, some of the classes in the target likely include additional properties or related classes. In the workbook, the properties for related classes in are indented the target list. For example:

TARGET
emr_medication_ibfk_5
SPECIALTYID
emr_medication_ibfk_6
STUDYID
MEDICATIONACTIVEFLAG
MEDICATIONACTIVITYID
MEDICATIONAGE
MEDICATIONCREATEDDATE

In the example, SPECIALTYID and STUDYID are object properties in the related emr_medication_ibfk_5 and emr_medication_ibfk_6 classes.

To map the source for these properties, first map the source for the class and then the source for the object properties becomes available in the mapping. For example:

TARGET		SOURCE
emr_medication_ibfk_5	emr_specialty →	emr_specialty
SPECIALTYID	Int =	MEDICATIONSPECIALTYID
emr_medication_ibfk_6	emr_study →	emr_study
STUDYID	Int =	MEDICATIONSTUDYID
MEDICATIONACTIVEFLAG	String =	MEDICATIONACTIVEFLAG
MEDICATIONACTIVITYID	Int =	MEDICATIONACTIVITYID
MEDICATIONAGE	Int =	MEDICATIONAGE
MEDICATIONCREATEDDATE	Date =	MEDICATIONCREATEDDATE

In the example, once emr_medication_ibfk_5 is mapped to the emr_specialty source class or table and emr_medication_ibfk_6 is mapped to emr_study, the source for the SPECIALTYID and STUDYID object properties are available to map.

5. Complete the mapping by entering the appropriate source for each target that you want to map. You do not have to enter source for all targets. For information about using functions to transform the source data, see Transforming Data in Mappings.

   Note To enter a literal value in the source mapping, type the value in the Source Mapping text box and then press **Enter**. The literal value is added to the cell with green text to distinguish it from fields.

6. If you want to add a new target and source pair to the mapping, click the target field that is above the cell where you want to add the new target. Then click the lines icon ( ) that appears to the left of the target name. Click the Add icon (+) that becomes available and choose the target and source elements by following the same process that you used when you mapped the source elements.

7. When you are finished mapping fields, click the save icon ( ) at the top of the mapping configuration pane to save the mapping changes. For instructions on performing other common editing tasks, see Editing Mappings below.

Editing Mappings

The table below provides instructions for working with mapping components. When changing mappings, click Save ( ) periodically to save your changes.

What do you want to do?	Instructions
Add a target and source pair	1. Click the target element above the row where you want to add a new pair. 2. Click the lines icon ( ) that appears to the left of the target name. 3. Click the Add icon ( ) that becomes available under the lines icon. Anzo selects the target cell and opens the **Target Mapping** text box. 4. In the Target Mapping text box, start typing the target field name. Anzo displays a list of fields that match the text. Select the target element in the results list. 5. In the Source column, click the cell that corresponds to the target you added. Anzo opens the Source Mapping text box. 6. In the Source Mapping text box, start typing the source field name. Anzo displays a list of fields that match the text. Select the source element in the results list.
Delete a target and source pair	Click the target element in the row that you want to delete. Then click the lines icon ( ) that appears to the left of the target name. Click the Trashcan icon ( ) that becomes available.
Modify the mapping data references	To change selected data that the mapping can access, click the Configuration icon ( ) in the right pane. In the References dialog box, add or remove elements as needed.
Validate changes to a mapping	Click the Validate icon ( ) to validate the mapping. Anzo displays any errors in the Validation Results screen.

For more advanced information about working with mappings, see Transforming Data in Mappings. For information about using a mapping in a pipeline, see Working with Pipelines.

Related Topics

Configuring Mappings to Ingest a Subset of the Source Data

Transforming Data in Mappings

Supported Mapping Functions

Configuring Mappings to Ingest a Subset of the Source Data

Anzo mappings include an option to set up parameters or criteria for ingesting source data so that you can create a graph data set that contains a subset of the data rather than all values. For example, if you want to import data that has decades worth of historical information but you are only interested in ingesting data from certain years, you can set criteria to filter out data that does not fall between those years.

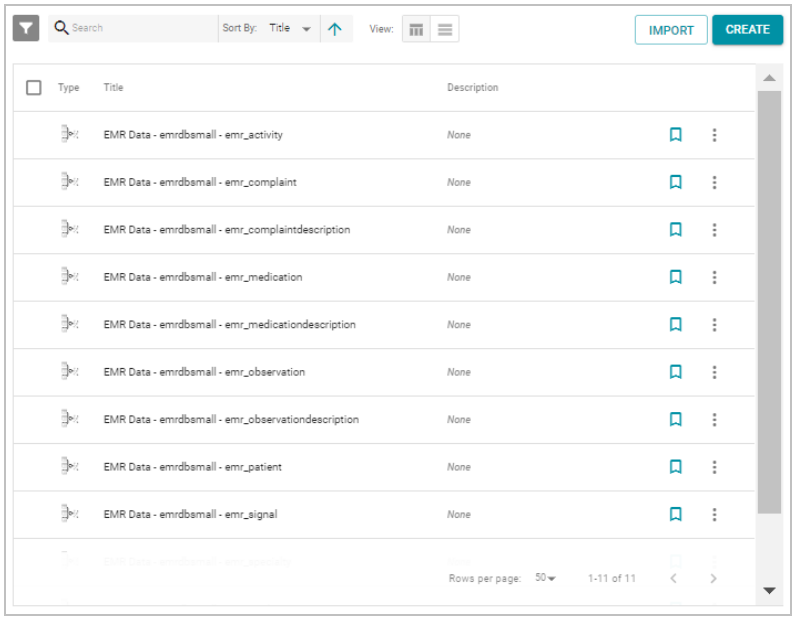
Note If the data source is a database, you can typically achieve better overall ETL pipeline performance by using schema queries to join and/or filter data rather than configuring mappings to perform those types of operations. For more information, see Performance Considerations for Database Pipelines.

Follow the instructions below to set up the parameters to use as criteria and add the criteria to filters:

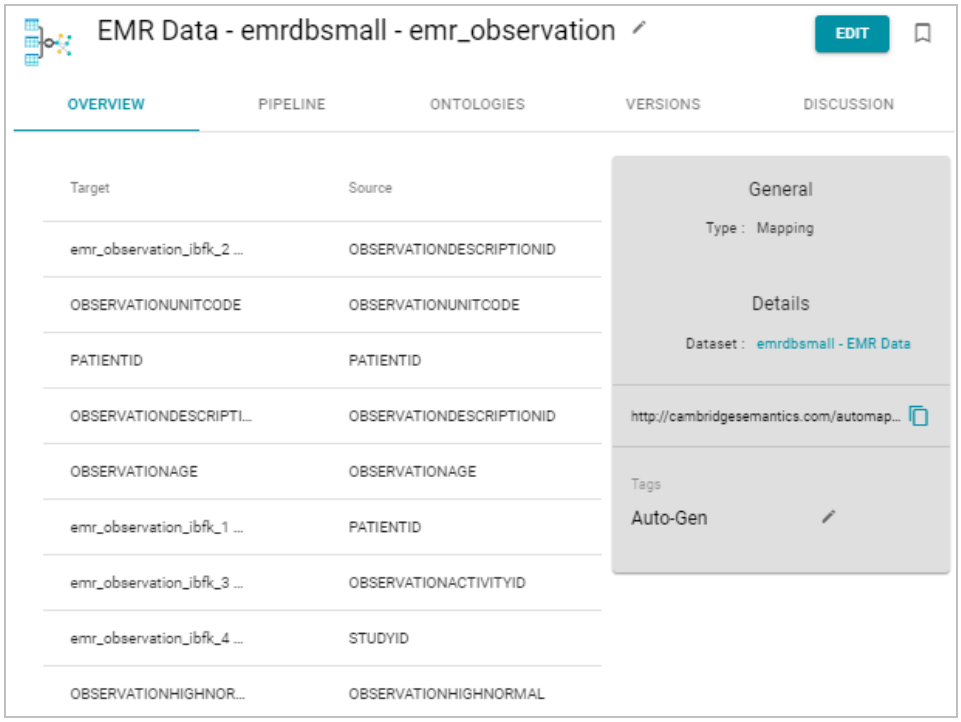
1. Open the Mapping
2. Create Parameters to Use as Filter Criteria
3. Add Filters to the Mapping to Apply the Criteria

Open the Mapping

1. In the Anzo application, expand the **Onboard** menu item and click **Structured Data**. Then click the **Mappings** tab. Anzo displays the Mappings screen, which lists any existing mappings. For example:



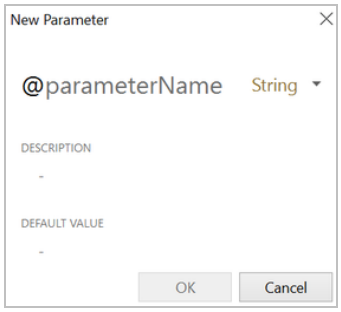
2. Click the name of the mapping that you want to edit. Anzo displays the mapping details. For example:



3. Click **Edit** at the top of the screen. Provide your server connection and login information and then click the arrow icon ( ) to connect to the server and open the mapping.

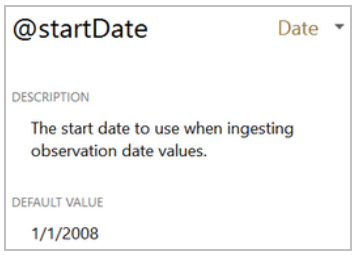
Create Parameters to Use as Filter Criteria

1. In the mapping configuration pane, click the plus icon (+) next to **Parameters**. Anzo opens the New Parameter dialog box.



2. Type a name for the new parameter in the **parameterName** field.
3. Next to the parameter name, click the data type drop-down list and select the data type of the source field whose value you want to use as filter criteria.
4. Type an optional description for the parameter in the **Description** field.
5. In the **Default Value** field, type the literal value to use as criteria for the source data. The value that you type must match the format for the data type that you chose. Do not include functions or formulas that transform the

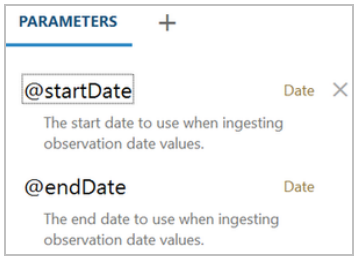
value in the Default Value field. You can transform the values when you create the filter that applies this parameter. For example:



6. Click **OK** to save the new parameter and add it to the Parameters list. Repeat the steps in this section to create any additional parameters, for example, if you are filtering on dates and need to set the beginning and end dates to filter on. For example:

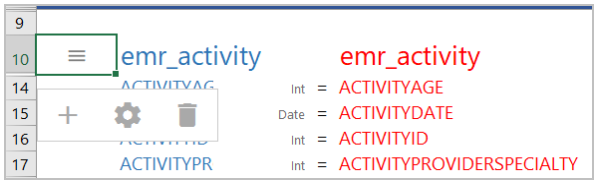
7. Click the save icon ( ) at the top of the mapping configuration pane to save the mapping changes.

To edit parameters in the Parameters list in the mapping configuration pane, click a parameter name to open the Edit Parameter dialog box. To delete parameters, hover the pointer over the parameter name and click the X that appears to the right of the parameter. For example:

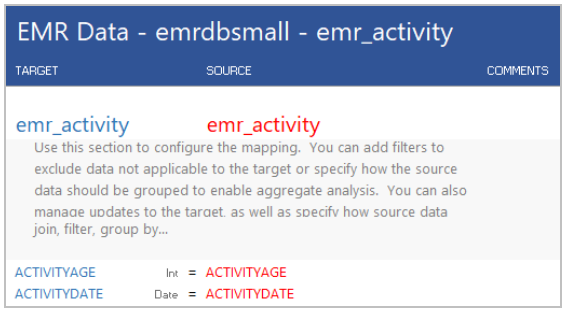


Add Filters to the Mapping to Apply the Criteria

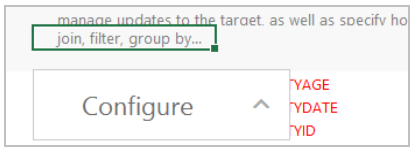
1. Click the cell to the left of the target table name to open the menu. For example:



2. Click the Cog icon ( ) in the menu to open the configuration section of the mapping.



3. In the configuration section, click the cell that contains the **join, filter, group by...** text in the Target column. If necessary, click the drop-down arrow next to the cell to open the Configure text box.

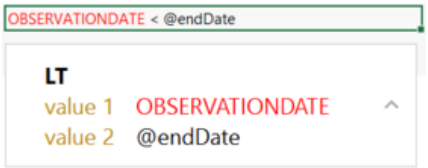


4. In the Configure text box, start typing **filter**. When "filter" appears below the text box, click it to add the filter keyword to the cell.
5. Click the cell in the Source column that corresponds to the filter you entered in the Target column. Anzo displays the **Source Mapping** text box below the cell. If necessary, click the drop-down arrow ( ) next to the cell to open the Source Mapping text box.
6. In the Source Mapping text box enter the expression to use for the filter. Type **@** to display the list of parameters to select from. For instructions on using functions in the filter, see Transforming Data in Mappings.

   For example, the following expression filters on values whose OBSERVATIONDATE is greater than the date in the @startDate parameter:

7. Create any additional filters by repeating steps 2 – 5.

   For example, the following expression filters on values whose OBSERVATIONDATE is less than the date in the @endDate parameter:



By adding filters for the @startDate and @endDate parameters, the example mapping is configured to ingest only the source records with observation dates that fall between the start and end dates.

8. Click the save icon ( ) at the top of the mapping configuration pane to save the mapping changes.

Related Topics

Creating a New Mapping

Transforming Data in Mappings

Supported Mapping Functions

Transforming Data in Mappings

Using the Anzo for Office plugin for Microsoft Excel, you can transform data to further define relationships between data elements, perform lightweight data preparation, or create sophisticated transformations. This topic provides information about creating advanced mappings and using Excel-like functions to transform data during the ETL process. This topic also describes the most commonly used mapping functions.

Note If the data source is a database, you can typically achieve better overall ETL pipeline performance by using schema queries to join and/or filter data rather than configuring mappings to perform those types of operations. For more information, see Performance Considerations for Database Pipelines.

For instructions on creating a new mapping, see Creating a New Mapping. For instructions on setting up parameters to ingest a subset of the source data, see Configuring Mappings to Ingest a Subset of the Source Data.

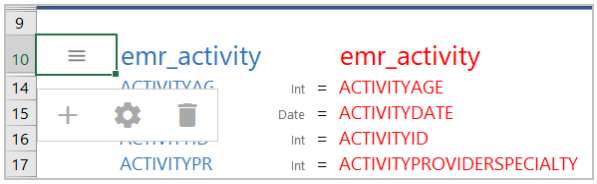
- Configuring Groups, Filters, Joins, Updates, and Merges
- Using Functions to Transform Source Data
- Commonly Used Functions

Configuring Groups, Filters, Joins, Updates, and Merges

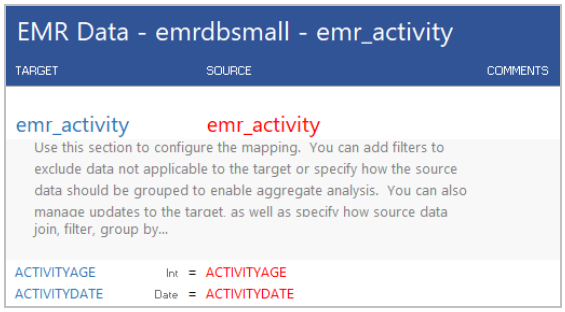
This section provides information about configuring groups, joins, filters, references, and merges at the mapping level so that they can be used by any functions that you use to transform the source data. The table below the steps describes each of the mapping level configuration options.

To implement a mapping level configuration:

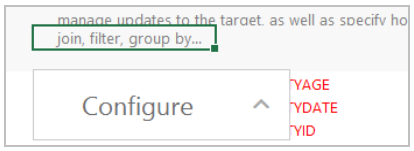
1. Click the cell to the left of the target table name to open the menu. For example:



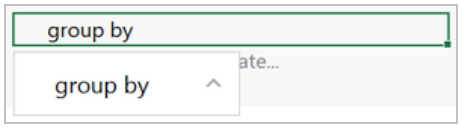
2. Click the Cog icon ( ) in the menu to open the configuration section of the mapping.



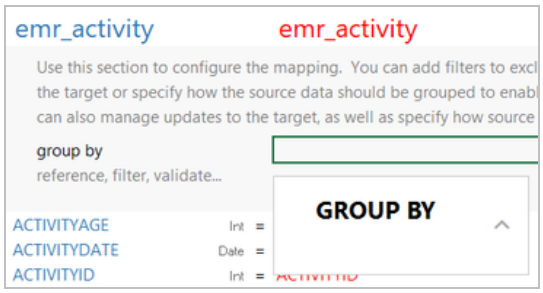
3. In the configuration section, click the cell that contains the **join, filter, group by...** text in the Target column. If necessary, click the drop-down arrow next to the cell to open the Configure text box.



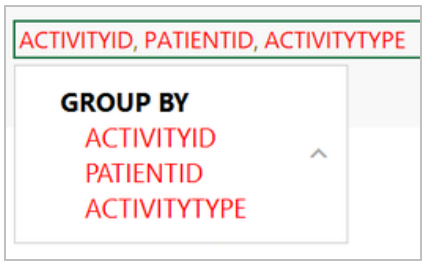
4. In the Configure text box, start typing the option that you want to configure. Anzo displays the options that match the text. Select an option to add it to the cell. For example:



5. Click the cell in the Source column that corresponds to the option you entered in the Target column. Anzo populates the Source text box with the appropriate keywords and arguments. For example:

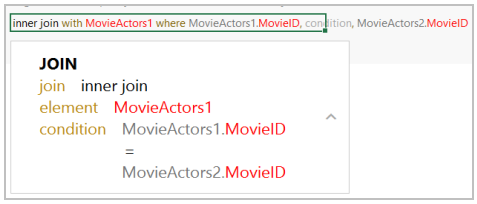
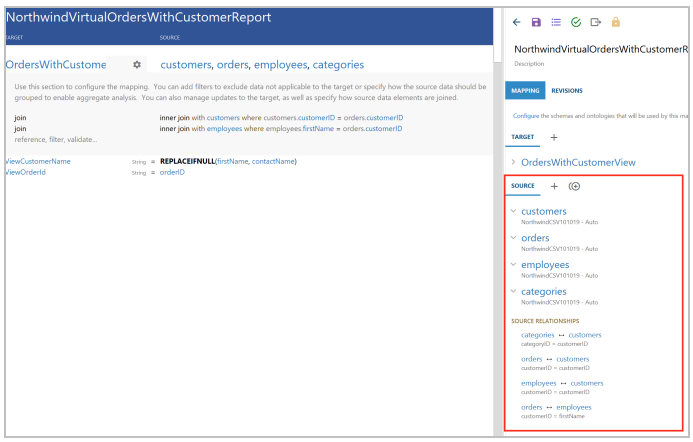


6. In the Source text box, click next to an argument or under a keyword and start typing column names for the columns that you want to add. Press **Ctrl +** to enter multiple columns. Then click the up arrow to enter the columns in the cell. For example:



The table below describes the mapping configuration options:

Option & Arguments	Description
FILTER	The FILTER keyword restricts the results that the mapping functions return. FILTER supports a single expression, and the expression must return a boolean value.
GROUP BY	The GROUP BY clause designates data groups and is required for aggregate functions. When an aggregate function is used, the solution is first divided into the groups defined by the GROUP BY clause, and then the aggregate value is calculated for each group.

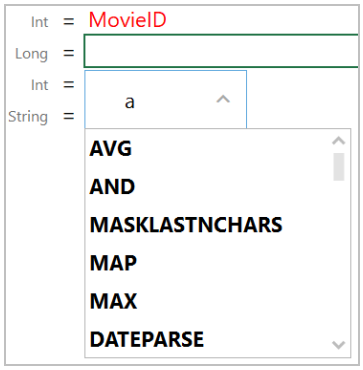
Option & Arguments	Description
JOIN join element condition	A JOIN combines rows from two tables based on related columns. You can specify joins when you map two sources to one target. • **join**: The kind of join to use. Type one of the following options: ▪ **inner join**: Returns only the records that have matching values in both tables. ▪ **outer join**: Returns all records from both tables when there is a match in either the left or right table. ▪ **left join**: Returns all records from the left table and joins only the records from the right table that match the condition. ▪ **right join**: Returns all records from the right table and joins only the records from the left table that match the condition. • **element**: One of the tables to join. The table that you specify depends on the type of join you are creating. For right joins, choose the left table. For left joins, choose the right table. • **condition**: The condition to use to join the two tables. For example, the following join uses an inner join to join all of the records from the MovieActors1 and MovieActors2 tables when the MovieID is the same in both tables.  **Important**: When including joins in mappings, do not create joins that result in multiple primary tables. Mappings with two or more primary tables are invalid. For example, the following example mapping is invalid because it has two primary tables, orders and categories. 

Option & Arguments	Description
MERGE BY	The MERGE BY clause enables you to merge multiple source rows into a single target row.
REFERENCE element condition	The REFERENCE keyword enables you to create a referential join between two tables.
UPDATE action key	

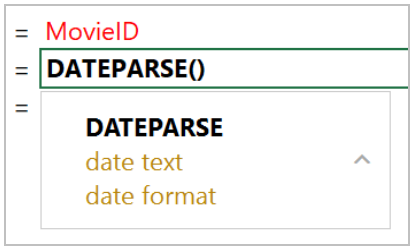
Using Functions to Transform Source Data

This section provides information about how to add functions to perform operations on source data.

The Source Mapping text box that you use to select source fields also includes a list of functions that you can apply to transform the source data. When you type in the Source Mapping box, in addition to available source fields, the mapping tool also displays a list of the functions that match the text you type. For example:

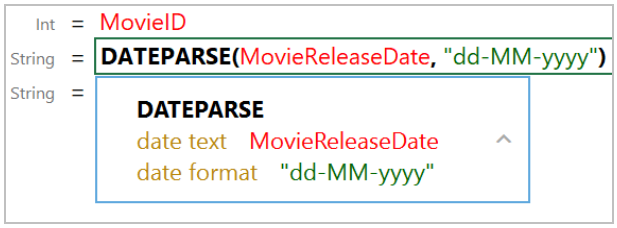


To add a function, select it from the list. The mapping tool adds the function to the cell and the Source Mapping box shows the arguments for the function that you chose. For example:



In the Source Mapping text box, enter the source field name, text, or additional functions that you want to include for the arguments. Enter arguments by typing to the right of the argument name. When entering literal values, press **Enter** to submit the value.

For example, the DATEPARSE function below converts MovieReleaseDate values from string types to dates in dd-MM-yyyy format:



Note The format that you specify for dates is flexible. For example, typing the format "dd-MMM-yy" displays values such as "01-JAN-19."

Commonly Used Functions

The table below describes the functions that are commonly used in mappings. For a complete list of the supported functions, see Supported Mapping Functions.

Note Any time you type a literal value into a function argument, press **Enter** to submit the value.

Function & Arguments	Description
IF test value if true value if false value if error	This function evaluates the condition in the **test** argument and assigns the value in **value if true** or **value if false** based on the results. • **test**: Use boolean columns or functions that return boolean: LE, LT, GE, GT, EQUALS, NOT_EQUAL, ISNULL, NOT, IN. • **value if true**: The value to output if **test** returns true. • **value if false**: The value to output if **test** returns false. • **value if error**: Cambridge Semantics recommends that you leave this argument blank.

Function & Arguments	Description
DATEPARSE date text date format	This function converts a string that contains a date value (**date text**) to the specified **date format**. • **date text**: The property that contains the date value in string format. • **date format**: The format that you want the date to follow. Specify days as "d," months as "M," and years as "y." For example, "yyyy-MM-dd." For example, the source mapping below converts the MovieReleaseDate values from strings to dates in the format "dd-MM-yyyy": **DATEPARSE** date text MovieReleaseDate date format "dd-MM-yyyy" The format that you specify for dates is flexible. For example, typing the format "dd-MMM-yy" displays values such as "01-JAN-19."
DATETIMEPARSE date text date format	This function converts a string that contains a datetime value (**date text**) to the specified **date format**. • **date text**: The property that contains the datetime value in string format. • **date format**: The format that you want the datetime to follow. For the date, specify days as "d," months as "M," and years as "y." For the time, specify "H" for hours, "m" for minutes, and "s" for seconds. For example, "yyyy-MM-dd HH:mm:ss." For example, the source mapping below converts the PATIENTLASTPMODATE from a string value to a datatime value in the format "MM-dd-yyyy HH:mm:ss": **DATETIMEPARSE** date text PATIENTLASTPMODATE date format "MM-dd-yyyy HH:mm:ss"
UPPER value	This function converts a string **value** to upper case letters.
LOWER value	This function converts a string **value** to lower case letters.

Function & Arguments	Description
REPLACEIFNULL expression if null expression	This function evaluates the **expression**. If the result is null, Anzo replaces the null with the value in **if null expression**. • **expression**: The source column or expression to evaluate. • **if null expression**: The expression to replace null values with. The resulting value must be the same data type as the target. For example, if mapping to a target with a double data type, "10.01" is valid but the string "missing" is not. For example, the source mapping below replaces any null values in the PATIENTID integer column with the integer 999: **REPLACEIFNULL** expression PATIENTID if null expression 999
REPLACEIFNULLOREMPTY string expression if null or empty expression	This function evaluates the **string expression**. If the result is null or empty (""), Anzo replaces the empty or null with the value in **if null or empty expression**. • **string expression**: The source column or expression that evaluates to string. • **if null or empty expression**: The expression to replace null or empty values with. The resulting value must be a string. For example, the source mapping below replaces any null or empty values in the GENDER column with "Not Specified": **REPLACEIFNULLOREMPTY** string expression GENDER if null or empty string expression "Not Specified"
ISNULL expression	This function evaluates the source column values in **expression** and returns "true" if the value is null and "false" if it is not null. You must choose a column in the **expression** argument; do not type a literal value or a function.

Function & Arguments	Description
SPLIT string delimiter	This function splits a **string** value into multiple values based on the specified **delimiter**. • **string**: The source column or function that evaluates to a string. • **delimiter**: The character to use to delimit the **string**.
SPLITARRAY string delimiter index	This function splits a **string** value into an array based on the **delimiter**. From the array, the function retrieves only the portion of the value that you specify in the **index**. • **string**: The source column or function that evaluates to a string. • **delimiter**: The character to use to delimit the string. • **index**: An integer that specifies the portion of the array to retrieve. Indexes start at zero. The first portion of the array is 0, the second is 1, and so on. Choose an index that you know exists or the mapping becomes invalid. For example, the following source mapping retrieves only the last four digits of social security numbers: **SPLITARRAY** string SSN delimeter "-" index 2

Function & Arguments	Description
IN value set to check	This function checks whether a given **value** exists in a set of values (**set to check**). If the value exists in the set, IN returns "true." If the value does not exist in the set, IN returns "false." IN does not do comparisons on string values. • **value**: The value to look for in the set. • **set to check**: The set of values to compare the value against. After typing a character, press **Enter** to submit the value, then press **Ctrl +** to add the next value. All items in the set must be the same data type. For example, the following source mapping checks to see if PATIENTID falls in the set of 1, 100, 1000: **IN** value PATIENTID set to check 1 100 1000
MAKELIST expression	This function maps multiple source columns to a single target property. The function does not create a list; it creates new rows, one for each column that is mapped to the target. • **expression**: The list of columns that you want to map to the target. After adding a source column press **Ctrl +** to select the next column.

Function & Arguments	Description
REGEX input regex replace	This function finds all patterns in the **input** string that match the specified regular expression (**regex**). It replaces the input patterns with the value in **replace** and returns the resulting string. • **input**: The source column or expression that evaluates to a string. • **regex**: The regular expression to use to find matches in the input string. For information about REGEX syntax, see the W3C Regular Expression Syntax specification. • **replace**: The string that should replace the input patterns that match regex. For example, the source mapping below uses the REGEX function to search for the pattern "PS" in the COMPLAINTSTRING values and replaces each PS with a hyphen (-): **REGEX** input COMPLAINTSTRING regex "PS" replace "-"
CONCATENATE text	This function concatenates multiple string values (**text**) and returns a single string. • **text**: The string values to concatenate, including any delimiters that you want to use. Press **Ctrl +** to enter multiple values. For example, the source mapping below concatenates PATIENTHOMESTATE and PATIENTHOMEZIP: **CONCATENATE** text PATIENTHOMESTATE PATIENTHOMEZIP
EQUAL value1 value2	This function compares numeric values and returns "true" if **value1** is equal to **value2** and "false" if the values are not equal (value1 = value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.

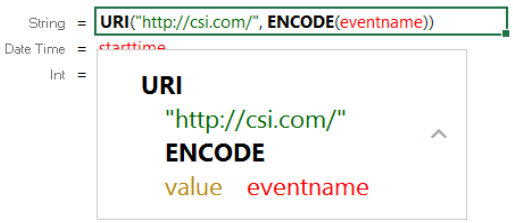
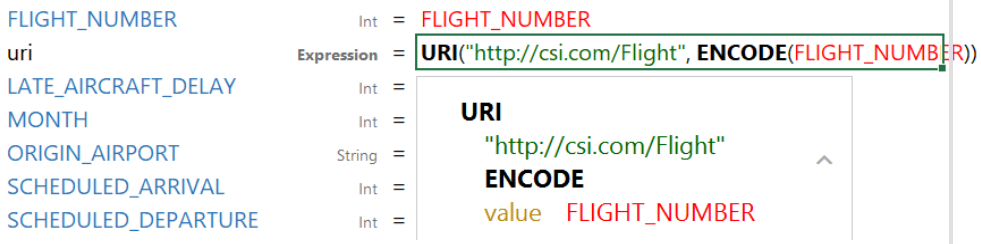
Function & Arguments	Description
NOT_EQUAL value1 value2	This function compares numeric values and returns "true" if **value1** does not equal **value2** and "false" if the values are equal (value1 != value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
GE value1 value2	This function compares numeric values and returns "true" if **value1** is greater than or equal to **value2** and "false" if value1 is less than value2 (value1 >= value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
GT value1 value2	This function compares numeric values and returns "true" if **value1** is greater than **value2** and "false" if value1 is less than or equal to value2 (value1 > value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
LE value1 value2	This function compares numeric values and returns "true" if **value1** is less than or equal to **value2** and "false" if value1 is greater than value2 (value1 <= value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
LT value1 value2	This function compares numeric values and returns "true" if **value1** is less than **value2** and "false" if value1 is greater than or equal to value2 (value1 < value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.

Function & Arguments	Description
AND logical1 logical2	This logical function evaluates two or more logical statements (**logical1**, **logical2**) and returns "true" if all conditions are met or "false" if any condition is not met. All logical statements must evaluate to the same data type. • **logical1**: The first logical condition to evaluate. This argument needs to include a logical function that returns a boolean value, such as AND, OR, GT, GE, LE, LT, EQUAL, NOT_EQUAL, ISNULL, NOT, IN. • **logical2**: The second logical condition to evaluate. This argument also needs to include a logical function that returns a boolean value.
OR logical1 logical2	This logical function evaluates two or more logical statements (**logical1**, **logical2**) and returns "true" if any of the conditions are met or "false" if none of the conditions are met. All logical statements must evaluate to the same data type. • **logical1**: The first logical condition to evaluate. This argument needs to include a logical function that returns a boolean value, such as AND, OR, GT, GE, LE, LT, EQUAL, NOT_EQUAL, ISNULL, NOT, IN. • **logical2**: The second logical condition to evaluate. This argument also needs to include a logical function that returns a boolean value.
NOT logical	This logical function evaluates whether data does not meet the condition (**logical**) that you specify. • **logical**: The logical condition to evaluate. This argument needs to include a logical function that returns a boolean value, such as AND, OR, GT, GE, LE, LT, EQUAL, NOT_EQUAL, ISNULL, NOT, IN.
NUMERIC_ADD v1 v2	This function adds the values of the numeric expressions that you specify (**v1** + **v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.

Function & Arguments	Description
DIVIDE v1 v2	This function divides the values of the numeric expressions that you specify (**v1**/**v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
MULTIPLY v1 v2	This function multiplies the values of the numeric expressions that you specify (**v1** x **v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
NUMERIC_SUBTRACT v1 v2	This function subtracts the values of the numeric expressions that you specify (**v1 - v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
LOOKUP from get fields values	This function enables you to look up values in a supplemental table. LOOKUP joins the **from** lookup table to a source table on the columns specified in the **fields** and **values** arguments. The function returns the value from **get** in the lookup table that corresponds to each row's value in the **values** argument. • **from**: The lookup table to perform the join against. • **get**: The field or property to retrieve from the lookup table in the **from** argument. • **fields**: The field or fields from the lookup table to compare with the values from the primary table. • **values**: The values from the primary table to compare with the **fields** from the lookup table.

Function & Arguments	Description
MAP value map	This function retrieves values from a map that you define. The map is a collection of key/value pairs. The function uses the specified **value** as a key in the **map** and returns the value associated with the key. • **value**: The key or keys to use to look up the value from the map. • **map**: The map to use to look up the values. You can click the **map** argument name to open the Edit Map dialog box and define or change a map.

Function & Arguments	Description

URI()	When specified in the Source column in mappings, this function transforms the property values to URI format by concatenating each of the components that you specify in the function. To ensure that values with spaces and other characters are encoded as valid URIs, the URI function is often used with the ENCODE function.
	
	For example, the following mapping for a "tickit_events" table transforms an "eventname" string to URI format by prepending "http://csi.com/" to the encoded event names:
	
	 
	
	This example mapping results in triples such as:
	
	`<tickit_events> <eventname> "http://csi.com/Rolling+Stones"`
	
	You can also enter URI() in the Target column to specify that the Expression in the Source column should be the URI of the entity that is being created. For example, the mapping below generates an entity URI by prepending "http://csi.com/Flight" to the flight number value:
	
	 
	
	The example URI specification results in triples such as:
	
	`<http://csi.com/Flight1234> <FLIGHT_NUMBER> 1234`
	`<http://csi.com/Flight1234> <ORIGIN_AIRPORT> "BOS"`

Related Topics

Creating a New Mapping

Configuring Mappings to Ingest a Subset of the Source Data

Supported Mapping Functions

Supported Mapping Functions

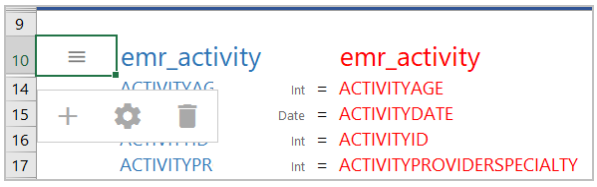
This topic describes the mapping functions that Anzo supports. For information about adding functions to mappings, see Transforming Data in Mappings.

- Aggregate Functions
- Boolean Operators
- Conditional Expressions
- Data Type Conversion Functions
- Lookup and Mapping Functions
- Numeric Functions
- String Functions

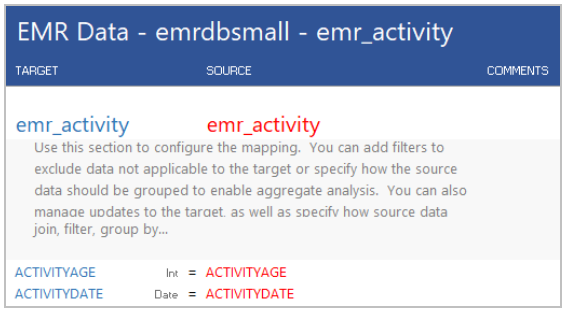
Aggregate Functions

Aggregate functions rely on the groups that you define by configuring a GROUP BY statement for the mapping. All aggregate functions use the GROUP BY that you specify. Follow these instructions to configure a GROUP BY statement:

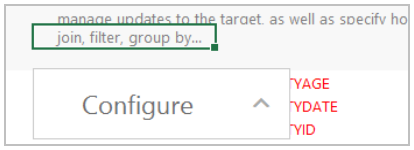
1. Click the cell to the left of the target table name to open the menu. For example:



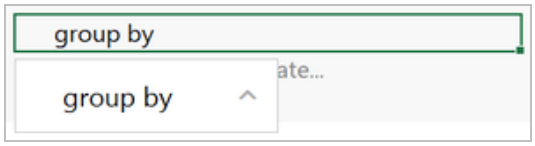
2. Click the Cog icon ( ) in the menu to open the configuration section of the mapping.



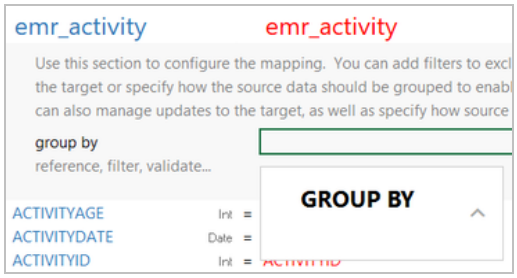
3. In the configuration section, click the cell that contains the **join, filter, group by...** text in the Target column. If necessary, click the drop-down arrow next to the cell to open the Configure text box.



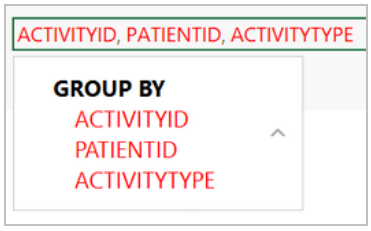
4. In the Configure text box, start typing "group by." Anzo completes the text and displays **group by** in the box. Click the up arrow to enter group by in the cell.



5. Click the cell in the Source column that corresponds to the group by you entered in the Target column. Anzo enters GROUP BY in the Source text box.



6. In the Source text box, click under GROUP BY and start typing column names for the columns that you want to group on. Press **Ctrl +** to enter multiple columns. Then click the up arrow to enter the columns in the cell. For example:



When you finish configuring the GROUP BY, save the mapping. When you use aggregate functions in the mapping, the functions group data according to the configured GROUP BY.

The table below describes the supported aggregate functions.

Function & Arguments	Description
AVG number	This function calculates the arithmetic mean for the group of numeric values that you specify in the **number** argument. • **number**: The column or expression that evaluates to a numeric value. The average is computed for the group or groups in the mapping's GROUP BY statement. For example, the following source mapping calculates the average NUMBER_OF_BYTES for each event. The GROUP BY statement for the mapping includes EVENTID. **AVG** number NUMBER_OF_BYTES
COUNT value	This function counts the number of instances for a grouped **value**. This function does not perform COUNT DISTINCT.
MAX value	This function calculates the maximum value for the group of numeric values that you specify in the **value** argument.
MIN value	This function calculates the minimum value for the group of numeric values that you specify in the **value** argument.
SUM number	This function calculates the sum of the group of numeric values that you specify in the **number** argument.

Boolean Operators

This section describes the boolean operators that you can use to target specific data and expand or reduce the number of records that are returned.

Function & Arguments	Description
EQUAL value1 value2	This function compares numeric values and returns "true" if **value1** is equal to **value2** and "false" if the values are not equal (value1 = value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.

Function & Arguments	Description
GE value1 value2	This function compares numeric values and returns "true" if **value1** is greater than or equal to **value2** and "false" if value1 is less than value2 (value1 >= value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
GT value1 value2	This function compares numeric values and returns "true" if **value1** is greater than **value2** and "false" if value1 is less than or equal to value2 (value1 > value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
IN value set to check	This function checks whether a given **value** exists in a set of values (**set to check**). If the value exists in the set, IN returns "true." If the value does not exist in the set, IN returns "false." IN does not do comparisons on string values. • **value**: The value to look for in the set. • **set to check**: The set of values to compare the value against. After typing a character, press **Enter** to submit the value, then press **Ctrl +** to add the next value. All items in the set must be the same data type. For example, the following source mapping checks to see if PATIENTID falls in the set of 1, 100, 1000: **IN** value PATIENTID set to check 1 100 1000
ISNULL expression	This function evaluates the source column values in **expression** and returns "true" if the value is null and "false" if it is not null. You must choose a column in the **expression** argument; do not type a literal value or a function.

Function & Arguments	Description
LE value1 value2	This function compares numeric values and returns "true" if **value1** is less than or equal to **value2** and "false" if value1 is greater than value2 (value1 <= value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
LT value1 value2	This function compares numeric values and returns "true" if **value1** is less than **value2** and "false" if value1 is greater than or equal to value2 (value1 < value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.
NOT_EQUAL value1 value2	This function compares numeric values and returns "true" if **value1** does not equal **value2** and "false" if the values are equal (value1 != value2). • **value1**: The numeric value to compare to value2. • **value2**: The numeric value to compare to value1.

Conditional Expressions

This section describes the functions that you can use to perform different computations based on whether a conditional expression evaluates to true or false.

Function & Arguments	Description
AND logical1 logical2	This logical function evaluates two or more logical statements (**logical1**, **logical2**) and returns "true" if all conditions are met or "false" if any condition is not met. All logical statements must evaluate to the same data type. • **logical1**: The first logical condition to evaluate. This argument needs to include a logical function that returns a boolean value, such as AND, OR, GT, GE, LE, LT, EQUAL, NOT_EQUAL, ISNULL, NOT, IN. • **logical2**: The second logical condition to evaluate. This argument also needs to include a logical function that returns a boolean value.

Function & Arguments	Description
IF test value if true value if false value if error	This function evaluates the condition in the **test** argument and assigns the value in **value if true** or **value if false** based on the results. • **test**: Use boolean columns or functions that return boolean: LE, LT, GE, GT, EQUALS, NOT_EQUAL, ISNULL, NOT, IN. • **value if true**: The value to output if **test** returns true. • **value if false**: The value to output if **test** returns false. • **value if error**: Cambridge Semantics recommends that you leave this argument blank.
OR logical1 logical2	This logical function evaluates two or more logical statements (**logical1**, **logical2**) and returns "true" if any of the conditions are met or "false" if none of the conditions are met. All logical statements must evaluate to the same data type. • **logical1**: The first logical condition to evaluate. This argument needs to include a logical function that returns a boolean value, such as AND, OR, GT, GE, LE, LT, EQUAL, NOT_EQUAL, ISNULL, NOT, IN. • **logical2**: The second logical condition to evaluate. This argument also needs to include a logical function that returns a boolean value.
NOT logical	This logical function evaluates whether data does not meet the condition (**logical**) that you specify. • **logical**: The logical condition to evaluate. This argument needs to include a logical function that returns a boolean value, such as AND, OR, GT, GE, LE, LT, EQUAL, NOT_EQUAL, ISNULL, NOT, IN.

Function & Arguments	Description
REPLACEIFNULL expression if null expression	This function evaluates the **expression**. If the result is null, Anzo replaces the null with the value in **if null expression**. • **expression**: The source column or expression to evaluate. • **if null expression**: The expression to replace null values with. The resulting value must be the same data type as the target. For example, if mapping to a target with a double data type, "10.01" is valid but the string "missing" is not. For example, the source mapping below replaces any null values in the PATIENTID integer column with the integer 999: **REPLACEIFNULL** expression PATIENTID if null expression 999
REPLACEIFNULLOREMPTY string expression if null or empty expression	This function evaluates the **string expression**. If the result is null or empty (""), Anzo replaces the empty or null with the value in **if null or empty expression**. • **string expression**: The source column or expression that evaluates to string. • **if null or empty expression**: The expression to replace null or empty values with. The resulting value must be a string. For example, the source mapping below replaces any null or empty values in the GENDER column with "Not Specified": **REPLACEIFNULLOREMPTY** string expression GENDER if null or empty string expression "Not Specified"

Data Type Conversion Functions

This section describes functions that you can use to convert values from one data type to another.

Function & Arguments	Description
BOOLEANPARSE value	This function converts a string (**value**) that contains "true" and "false" values to boolean format. **Note**: Specifying a source column for which some instances do not contain "true" or "false" values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEBOOLEAN unless you are certain that all instances of **value** contain the words "true" or "false."
DATEPARSE date text date format	This function converts a string that contains a date value (**date text**) to the specified **date format**. **Note**: Specifying a source column for which some instances do not contain date values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEDATE unless you are certain that all instances of **date text** contain a date. • **date text**: The property that contains the date value in string format. • **date format**: The format that you want the date to follow. Specify days as "d," months as "M," and years as "y." For example, "yyyy-MM-dd." For example, the source mapping below converts the MovieReleaseDate values from strings to dates in the format "dd-MM-yyyy": **DATEPARSE** date text MovieReleaseDate date format "dd-MM-yyyy" The format that you specify for dates is flexible. For example, typing the format "dd-MMM-yy" displays values such as "01-JAN-19."

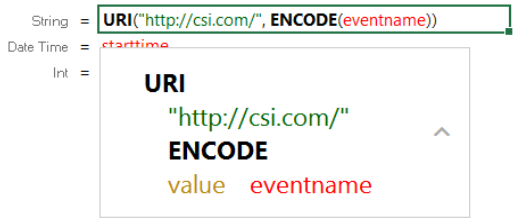
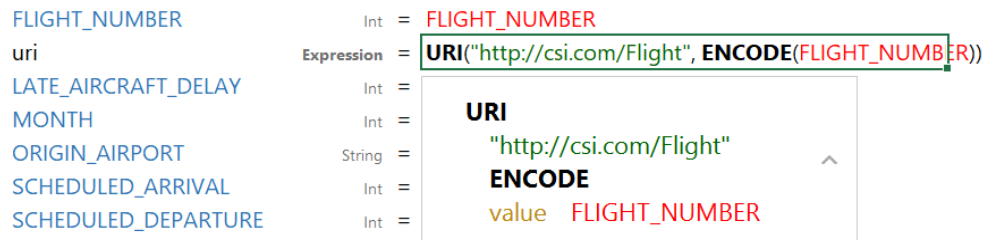
Function & Arguments	Description
DATETIMEPARSE date text date format	This function converts a string that contains a datetime value (**date text**) to the specified **date format**. **Note**: Specifying a source column for which some instances do not contain datetime values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEDATETIME unless you are certain that all instances of **date text** contain a datetime. • **date text**: The property that contains the datetime value in string format. • **date format**: The format that you want the datetime to follow. For the date, specify days as "d," months as "M," and years as "y." For the time, specify "H" for hours, "m" for minutes, and "s" for seconds. For example, "yyyy-MM-dd HH:mm:ss." For example, the source mapping below converts the PATIENTLASTPMODATE from a string value to a datatime value in the format "MM-dd-yyyy HH:mm:ss": **DATETIMEPARSE** date text PATIENTLASTPMODATE date format "MM-dd-yyyy HH:mm:ss"
DECIMALPARSE value	This function converts a string (**value**) that contains a decimal value to decimal format. **Note**: Specifying a source column for which some instances do not contain decimal values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEDECIMAL unless you are certain that all instances of **value** contain a decimal.
DOUBLEPARSE value	This function converts a string (**value**) that contains a double value to double format. **Note**: Specifying a source column for which some instances do not contain double values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEDOUBLE unless you are certain that all instances of **value** contain a double.
FLOATPARSE value	This function converts a string (**value**) that contains float values to float format. **Note**: Specifying a source column for which some instances do not contain float values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEFLOAT unless you are certain that all instances of **value** contain floats.

Function & Arguments	Description
INTPARSE value	This function converts a string (**value**) that contains integer values to integer format. **Note**: Specifying a source column for which some instances do not contain integer values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSEINT unless you are certain that all instances of **value** contain integers.
LONGPARSE value	This function converts a string (**value**) that contains a long integer value (from -2,147,483,648 to 2,147,483,647) to long format. **Note**: Specifying a source column for which some instances do not contain long values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSELONG unless you are certain that all instances of **value** contain long data.
SHORTPARSE value	This function converts a string (**value**) that contains a short integer value (from -32,678 to 32,767) to short format. **Note**: Specifying a source column for which some instances do not contain short values can cause the ETL job to fail. Cambridge Semantics recommends using TRYPARSESHORT unless you are certain that all instances of **value** contain short data.
TIMEPARSE time text time format	This function converts a string that contains **time text** to a time value in the **time format** that you specify. **Note**: Specifying a source column for which some instances do not contain time values can cause the ETL job to fail. Use this function only when all instances of **time text** contain a time value. • **time text**: The property that contains the time value in string format. • **time format**: The format that you want the time value to follow. Specify "H" for hours, "m" for minutes, and "s" for seconds. For example, "HH:mm:ss."
TOSTRING value format	This function converts a **value** that is a double data type to string **format**. • **value**: The double type values that you want to convert to string format. • **format**: The format code for the new string value. For example, "%.0f".

Function & Arguments	Description
TRYPARSEBOOLEAN value if error	This function attempts to convert a string **value** to a boolean data type. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains "true" or "false" values. • **if error**: The boolean value to replace the string with if an error occurs with the conversion.
TRYPARSEDATE value date format if error	This function attempts to convert a string **value** to a date data type in the **date format** that you specify. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains date data. • **date format**: The format that you want the date to follow. Specify days as "d," months as "M," and years as "y." For example, "yyyy-MM-dd." Or if your data has values such as 09APR2020, specify the date format "ddMMMyyyy." • **if error**: The date value to replace the string with if an error occurs with the conversion.
TRYPARSEDATETIME value date format if error	This function attempts to convert a string **value** to a datetime data type in the **date format** that you specify. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains datetime data. • **date format**: The format that you want the datetime to follow. For the date, specify days as "d," months as "M," and years as "y." For the time, specify "H" for hours, "m" for minutes, and "s" for seconds. For example, "yyyy-MM-dd HH:mm:ss." • **if error**: The datetime value to replace the string with if an error occurs with the conversion.
TRYPARSEDECIMAL value if error	This function attempts to convert a string **value** to a decimal data type. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains decimal data. • **if error**: The decimal value to replace the string with if an error occurs with the conversion.

Function & Arguments	Description
TRYPARSEDOUBLE value if error	This function attempts to convert a string **value** to a double data type. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains double data. • **if error**: The double value to replace the string with if an error occurs with the conversion.
TRYPARSEFLOAT value if error	This function attempts to convert a string **value** to a float data type. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains float values. • **if error**: The float value to replace the string with if an error occurs with the conversion.
TRYPARSELONG value if error	This function attempts to convert a string **value** to a long data type. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains long data (-2,147,483,648 to 2,147,483,647). • **if error**: The long value to replace the string with if an error occurs with the conversion.
TRYPARSESHORT value if error	This function attempts to convert a string **value** to a short data type. If an instance cannot be converted, Anzo replaces the string with the value in **if error**. • **value**: The string value that contains short data (-32,678 to 32,767). • **if error**: The short value to replace the string with if an error occurs with the conversion.

Function & Arguments	Description

URI()	When specified in the Source column in mappings, this function transforms the property values to URI format by concatenating each of the components that you specify in the function. To ensure that values with spaces and other characters are encoded as valid URIs, the URI function is often used with the ENCODE function.
	For example, the following mapping for a "tickit_events" table transforms an "eventname" string to URI format by prepending "http://csi.com/" to the encoded event names:
	 
	This example mapping results in triples such as:
	`<tickit_events> <eventname> "http://csi.com/Rolling+Stones"`
	You can also enter URI() in the Target column to specify that the Expression in the Source column should be the URI of the entity that is being created. For example, the mapping below generates an entity URI by prepending "http://csi.com/Flight" to the flight number value:
	 
	The example URI specification results in triples such as:
	`<http://csi.com/Flight1234> <FLIGHT_NUMBER> 1234`
	`<http://csi.com/Flight1234> <ORIGIN_AIRPORT> "BOS"`

Lookup and Mapping Functions

This section describes the lookup and map functions that Anzo supports.

Function & Arguments	Description
LOOKUP from get fields values	This function enables you to look up values in a supplemental table. LOOKUP joins the **from** lookup table to a source table on the columns specified in the **fields** and **values** arguments. The function returns the value from **get** in the lookup table that corresponds to each row's value in the **values** argument. • **from**: The lookup table to perform the join against. • **get**: The field or property to retrieve from the lookup table in the **from** argument. • **fields**: The field or fields from the lookup table to compare with the values from the primary table. • **values**: The values from the primary table to compare with the **fields** from the lookup table.
MAKELIST expression	This function maps multiple source columns to a single target property. The function does not create a list; it creates new rows, one for each column that is mapped to the target. • **expression**: The list of columns that you want to map to the target. After adding a source column press **Ctrl +** to select the next column.
MAP value map	This function retrieves values from a map that you define. The map is a collection of key/value pairs. The function uses the specified **value** as a key in the **map** and returns the value associated with the key. • **value**: The key or keys to use to look up the value from the map. • **map**: The map to use to look up the values. You can click the **map** argument name to open the Edit Map dialog box and define or change a map.

Numeric Functions

This section describes functions that operate on values with numeric data types.

Function & Arguments	Description
CEILING value	This function rounds the **value** up to the next whole number if the value has a fractional part. • **value**: The source values that you want to round up to the next whole number.

Function & Arguments	Description
DIVIDE v1 v2	This function divides the values of the numeric expressions that you specify (**v1**/**v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
FLOOR value	This function rounds the **value** down to a whole number if the value has a fractional part. • **value**: The source values that you want to round down to a whole number.
MULTIPLY v1 v2	This function multiplies the values of the numeric expressions that you specify (**v1** x **v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
NUMERIC_ ADD v1 v2	This function adds the values of the numeric expressions that you specify (**v1** + **v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
NUMERIC_ SUBTRACT v1 v2	This function subtracts the values of the numeric expressions that you specify (**v1** - **v2**). • **v1**: The numeric value that you want to add with v2. • **v2**: The numeric value that you want to add with v1.
RANDOM value min range max range	This function replaces **value** with a random integer from within the **min range** and **max range** that you specify. • **value**: The source values that you want to replace with a random integer. • **min range**: The integer that indicates the lowest number in the range that the function can choose from. • **max range**: The integer that indicates the highest number in the range that the function can choose from.
ROUND value	This function rounds the **value** up or down to the closest whole number. • **value**: The source values that you want to round up or down.

String Functions

This section describes functions that operate on values with string data types.

Function & Arguments	Description
CONCATENATE text	This function concatenates multiple string values (**text**) and returns a single string. • **text**: The string values to concatenate, including any delimiters that you want to use. Press **Ctrl +** to enter multiple values. For example, the source mapping below concatenates PATIENTHOMESTATE and PATIENTHOMEZIP: **CONCATENATE** text PATIENTHOMESTATE PATIENTHOMEZIP
DATEPARSE date text date format	This function converts a string that contains a date value (**date text**) to the specified **date format**. • **date text**: The property that contains the date value in string format. • **date format**: The format that you want the date to follow. Specify days as "d," months as "M," and years as "y." For example, "yyyy-MM-dd." For example, the source mapping below converts the MovieReleaseDate values from strings to dates in the format "dd-MM-yyyy": **DATEPARSE** date text MovieReleaseDate date format "dd-MM-yyyy" The format that you specify for dates is flexible. For example, typing the format "dd-MMM-yy" displays values such as "01-JAN-19."

Function & Arguments	Description
DATETIMEPARSE date text date format	This function converts a string that contains a datetime value (**date text**) to the specified **date format**. • **date text**: The property that contains the datetime value in string format. • **date format**: The format that you want the datetime to follow. For the date, specify days as "d," months as "M," and years as "y." For the time, specify "H" for hours, "m" for minutes, and "s" for seconds. For example, "yyyy-MM-dd HH:mm:ss." For example, the source mapping below converts the PATIENTLASTPMODATE from a string value to a datatime value in the format "MM-dd-yyyy HH:mm:ss": **DATETIMEPARSE** date text PATIENTLASTPMODATE date format "MM-dd-yyyy HH:mm:ss"
LEFT text num chars	This function starts on the left side of a **text** string, keeps the number of characters in **num chars**, and returns the truncated string.
LOWER value	This function converts a string **value** to lower case letters.

Function & Arguments	Description
REGEX input regex replace	This function finds all patterns in the **input** string that match the specified regular expression (**regex**). It replaces the input patterns with the value in **replace** and returns the resulting string. • **input**: The source column or expression that evaluates to a string. • **regex**: The regular expression to use to find matches in the input string. For information about REGEX syntax, see the W3C Regular Expression Syntax specification. • **replace**: The string that should replace the input patterns that match regex. For example, the source mapping below uses the REGEX function to search for the pattern "PS" in the COMPLAINTSTRING values and replaces each PS with a hyphen (-): **REGEX** input COMPLAINTSTRING regex "PS" replace "-"
RIGHT text num chars	This function starts on the right side of a **text** string, keeps the number of characters in **num chars**, and returns the truncated string.
SPLIT string delimiter	This function splits a **string** value into multiple values based on the specified **delimiter**. • **string**: The source column or function that evaluates to a string. • **delimiter**: The character to use to delimit the **string**.

Function & Arguments	Description
SPLITARRAY string delimiter index	This function splits a **string** value into an array based on the **delimiter**. From the array, the function retrieves only the portion of the value that you specify in the **index**. • **string**: The source column or function that evaluates to a string. • **delimiter**: The character to use to delimit the string. • **index**: An integer that specifies the portion of the array to retrieve. Indexes start at zero. The first portion of the array is 0, the second is 1, and so on. Choose an index that you know exists or the mapping becomes invalid. For example, the following source mapping retrieves only the last four digits of social security numbers: **SPLITARRAY** string SSN delimeter "-" index 2
STRLEN term	This function returns the number of characters in the specified text string (**term**).
UPPER value	This function converts a string **value** to upper case letters.

Related Topics

Creating a New Mapping

Configuring Mappings to Ingest a Subset of the Source Data

Transforming Data in Mappings

Working with Pipelines

The topics in this section provide information about creating and editing pipelines and ETL jobs.

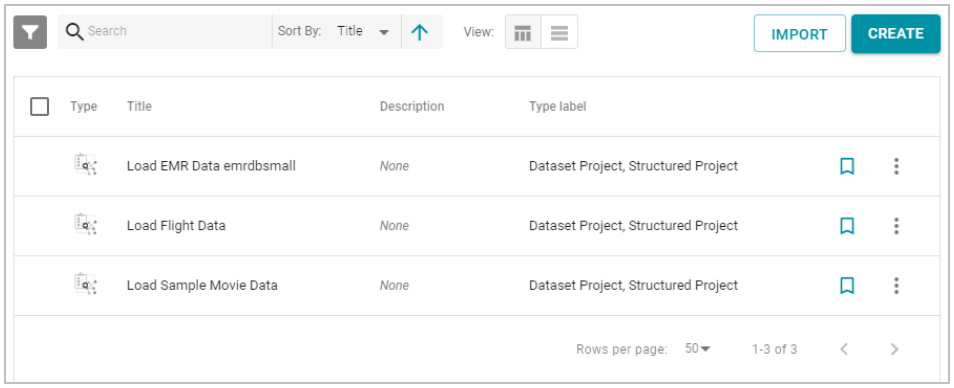
- Creating a New Dataset Pipeline to Ingest Data into Anzo
- Creating a New ETL Pipeline to Move Data to a File or Database
- Migrating a Pipeline from One Server to Another

Creating a New Dataset Pipeline to Ingest Data into Anzo

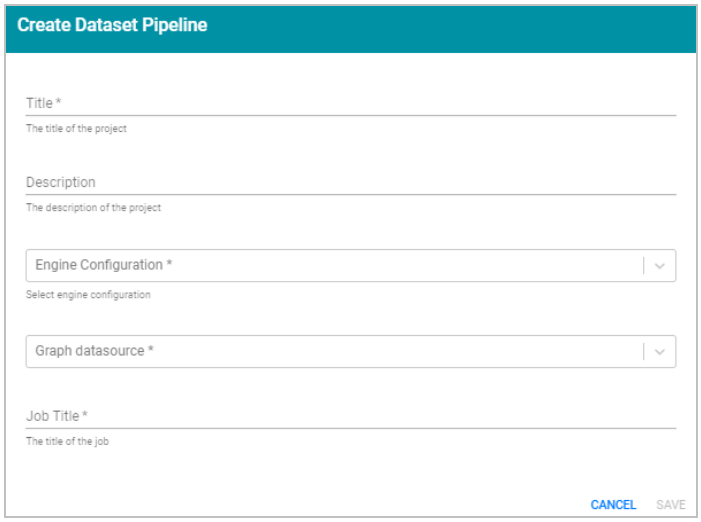
This topic provides instructions for creating a new **Dataset Pipeline** to ingest data into Anzo. Dataset pipelines produce a new dataset in the Dataset catalog and generate RDF load files for AnzoGraph. All auto-ingested projects are dataset pipelines.

For instructions on creating a pipeline that moves data to a file or database, see Creating a New ETL Pipeline to Move Data to a File or Database. For information about creating unstructured pipelines, see Creating an Unstructured Pipeline.

1. In the Anzo console, expand the **Onboard** menu and click **Structured Data**. Then click the **Pipelines** tab. Anzo displays the Pipelines screen. For example:

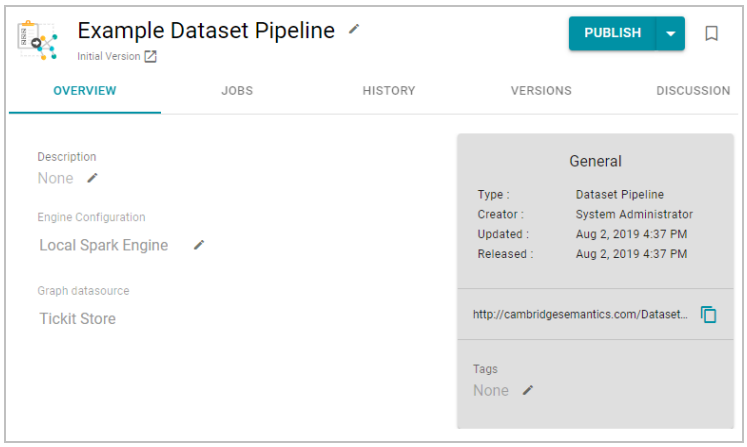


2. Click the **Create** button at the top of the screen and select **Dataset Pipeline**. Anzo displays the Create Dataset Pipeline screen.

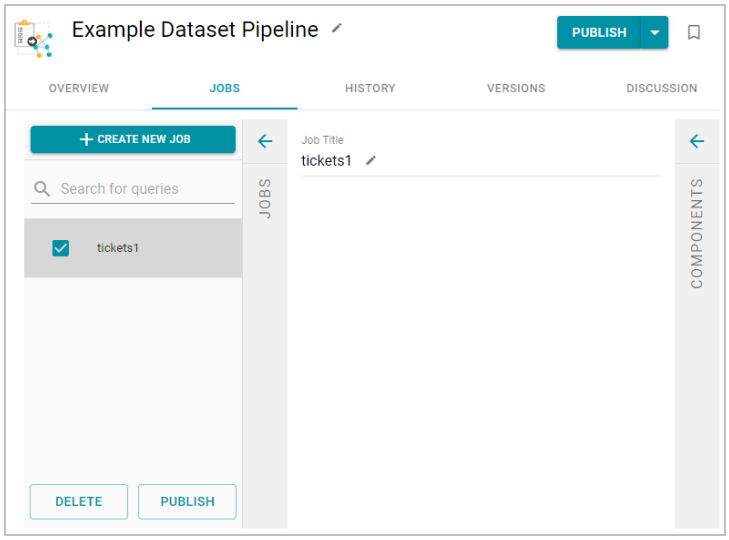


3. Type a name for the pipeline in the **Title** field and enter an optional **Description**.
4. Click the **Engine Configuration** drop-down list and select the ETL engine for this pipeline.
5. Click the **Graph datasource** drop-down list and select the graph source where you want Anzo to save the AnzoGraph load files that are generated when jobs in this pipeline are published.

6. In the **Job Title** field, type a name for the first job in the pipeline.
7. Click **Save** to create the pipeline. Anzo displays the pipeline overview screen. For example:

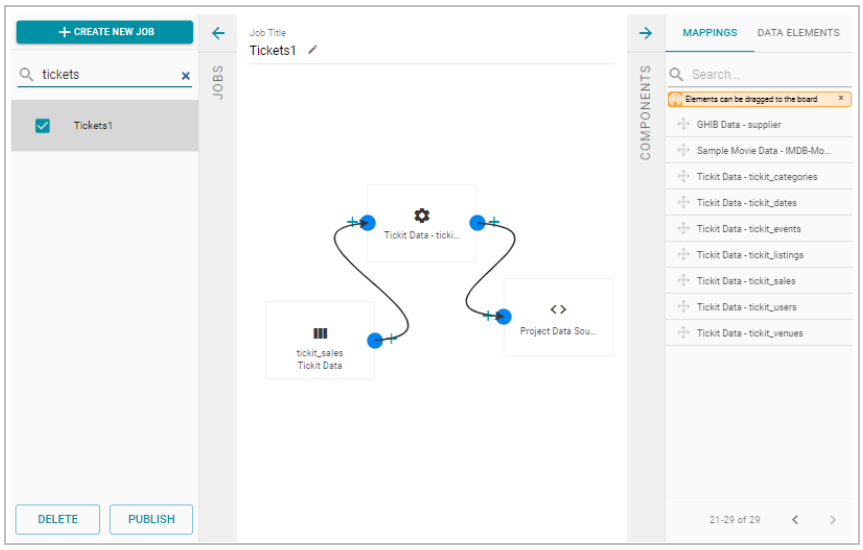


8. To view the job that was created in the pipeline, click the **Jobs** tab. Anzo displays the Jobs screen. For example:



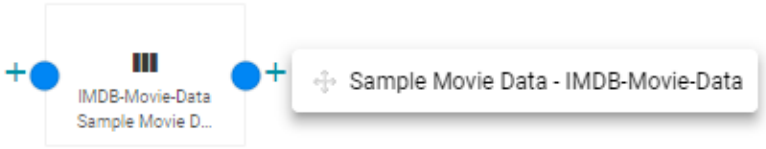
   The Jobs pane on the left side of the screen lists the jobs in the pipeline. The Components pane on the right side of the screen lists the data elements and mappings that are available for building the job. The white space in main part of the screen is the working area where you construct the job.

9. To configure the job, drag and drop a data source element from the Mappings or Data Elements tab onto the main part of the screen. Expand an element to drill down to a selectable source. For example, in the image below the Tickit Data - tickit_sales mapping is added to the job:

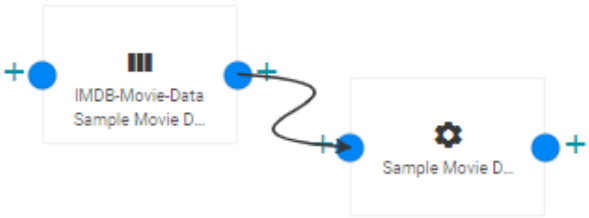


If you drag a mapping onto the job canvas and there is only one source and one target for the mapping, Anzo automatically adds that source and target to the job.

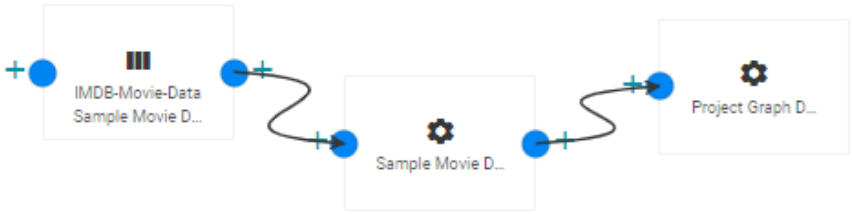
10. If necessary, specify the mapping to use to map the source data to the target for this pipeline. To do so, click the plus icon to the right of the source element. Anzo suggests a mapping to use based on the data source. For example:



11. To add the mapping to the job, select the item in the tooltip and drag it onto the screen. For example:



12. If necessary, complete the job by specifying the target. Click the plus icon to the right of the mapping element. Anzo suggests a target to use based on the data source. Select **Project Graph Data Source** as the target to use the graph data source that you specified when you created the pipeline. Drag the target onto the working space. For example:



13. If you want to create additional jobs for this pipeline, click the **Create New Job** button on the left side of the screen. Repeat the steps above to complete the job.
14. To run the pipeline and all of the jobs, click the **Publish** button at the top of the screen. If you want to run a single job in the pipeline, click the checkbox next to the job, then click the **Publish** button under the list of jobs.

When the pipeline completes, the new (or updated) data set becomes available in the Dataset catalog. From the catalog, data sets can be blended, explored, and analyzed. For guidance on next steps, see Blending Data.

Creating a New ETL Pipeline to Move Data to a File or Database

This topic is in progress and will provide instructions for creating a new **ETL Pipeline** that ingests data into a file or database. Create an ETL pipeline when you do not want to generate a new data set entry in the Anzo Dataset catalog or produce RDF load files for AnzoGraph. ETL pipelines require a mapping that defines a file schema or database as the target.

For instructions on creating a dataset pipeline to ingest data into Anzo, see Creating a New Dataset Pipeline to Ingest Data into Anzo. For information about creating unstructured pipelines, see Creating an Unstructured Pipeline.

Migrating a Pipeline from One Server to Another

For instructions on migrating a pipeline from one server to another by exporting the pipeline and then importing it on the destination server, see Exporting Versions.

Onboarding Unstructured Data

Anzo processes unstructured data through configurable text analytics and natural language processing (NLP) pipelines that find and extract data and convert it to the graph data model. Anzo can process all common file types such as Office documents, PDFs, web pages, and email messages, and can analyze text within Excel, databases, and knowledgebases, or XML columns, properties, and fields. Anzo finds, analyzes, extracts, and ingests concepts, entities, sentiment, topics, classifications, events, facts, and thousands of types of relationships.

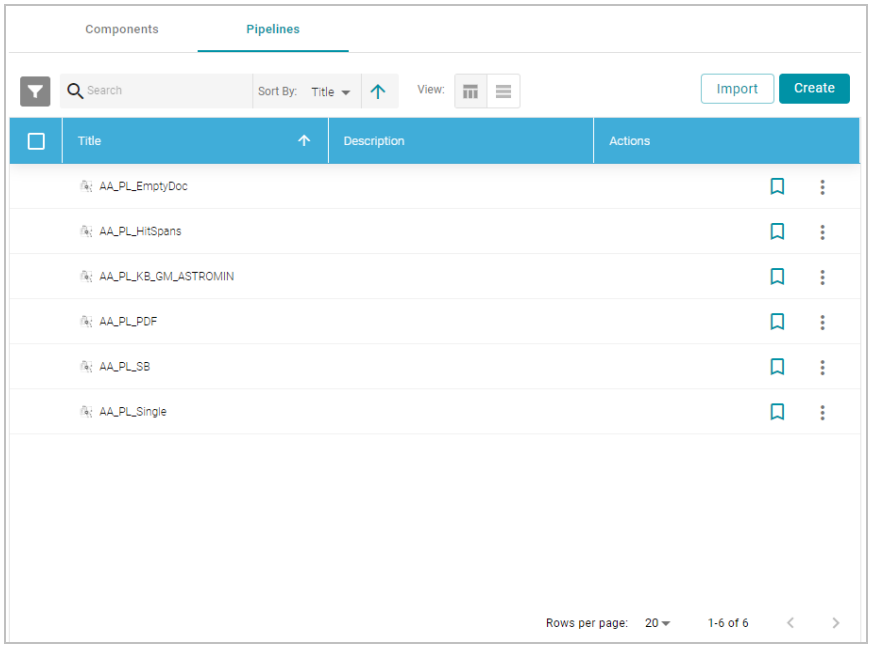
The topics in this section provide instructions for creating pipelines and onboarding unstructured data. For information about setting up the Anzo Unstructured environment, see Deploying the Anzo Unstructured Infrastructure.

- Creating an Unstructured Pipeline
- Running an Unstructured Pipeline

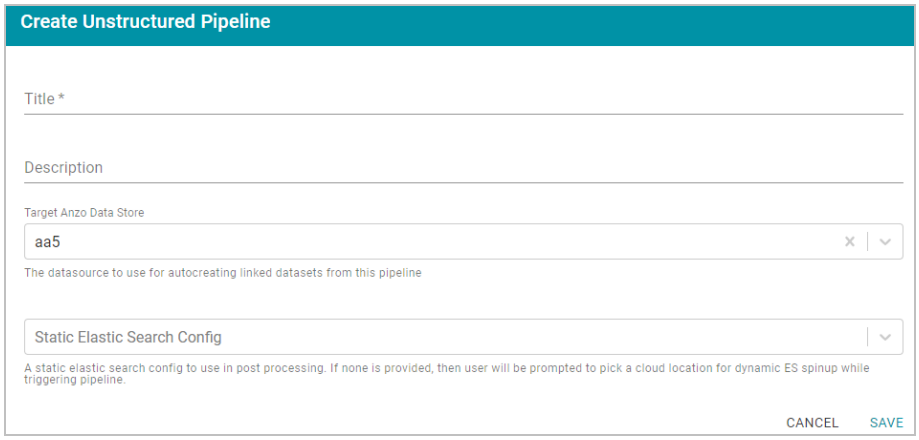
Creating an Unstructured Pipeline

This topic provides instructions for creating a new pipeline to ingest unstructured data.

1. In the Anzo console, expand the **Onboard** menu and click **Unstructured Data**. Anzo displays the Pipeline screen, which lists any existing unstructured pipelines. For example:



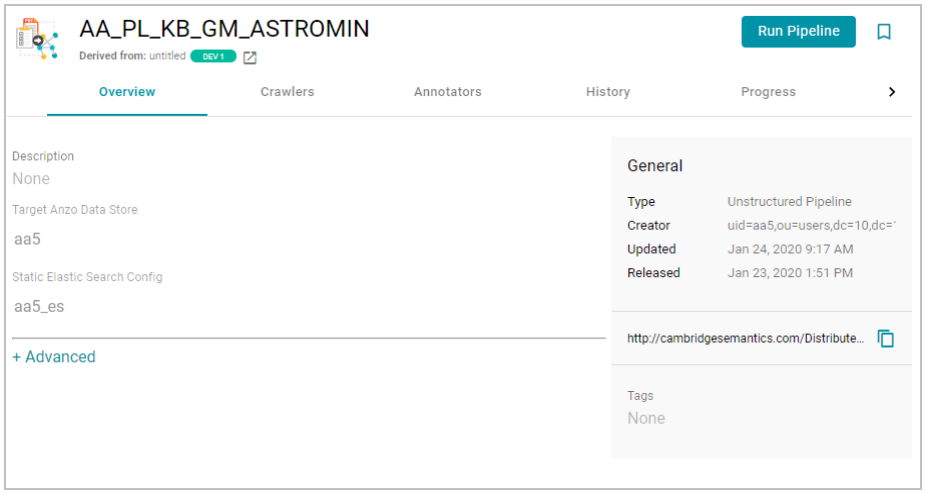
2. Click the **Create** button. Anzo opens the Create Unstructured Pipeline dialog box. For example:



3. In the **Title** field, type a name for the pipeline.

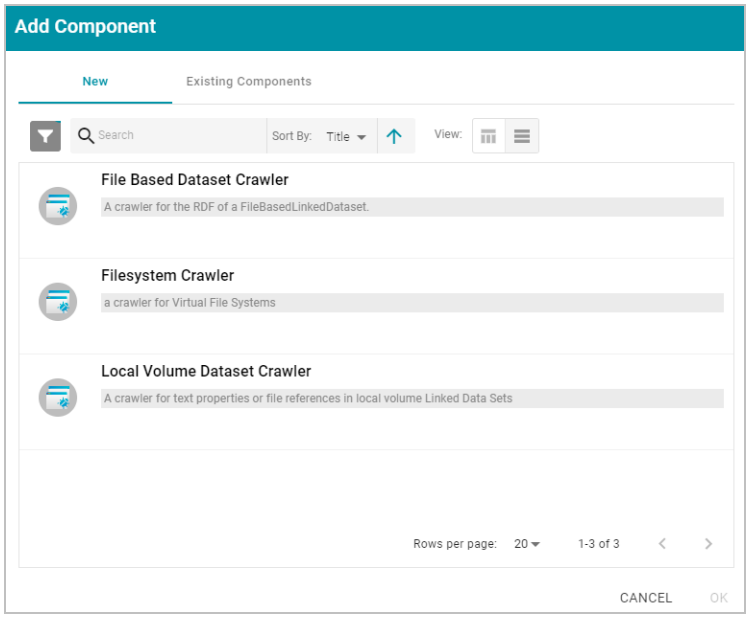
   Note: This title serves as a key to identify this pipeline and its corpus in multiple contexts. Specify a title that is unique and stable. The pipeline's corpus data set name is derived from this title.

4. Type an optional description for the pipeline in the **Description** field.
5. If necessary, click the **Target Anzo Data Store** field and select the data store for this pipeline. For information about creating an Anzo data store, see Creating an Anzo Data Store.
6. If necessary, click the **Static Elasticsearch Config** field and select the Elasticsearch connection to use for this pipeline. If you use dynamic deployments to deploy Elasticsearch instances on-demand, leave this field blank. Anzo will prompt the user to choose a cloud location when the pipeline is run. For information about creating a static Elasticsearch connection, see Connecting to Elasticsearch.
7. Click **Save** to create the pipeline. Anzo displays the pipeline Overview screen. For example:



   Note: A pipeline configuration saves automatically and constantly undergoes validation to make sure that the pipeline is valid based on the current configuration. Anzo displays validation issues in red on the top of the screen. The warnings will disappear as you add components to the pipeline.

8. If necessary, click **Advanced** to view and configure the advanced pipeline settings. Descriptions of the advanced settings are in progress.
9. Click the **Crawlers** tab and follow the substeps below to add a crawler to the pipeline:
    a. Click **Add Input** to select a crawler. Anzo opens the Add Component dialog box.

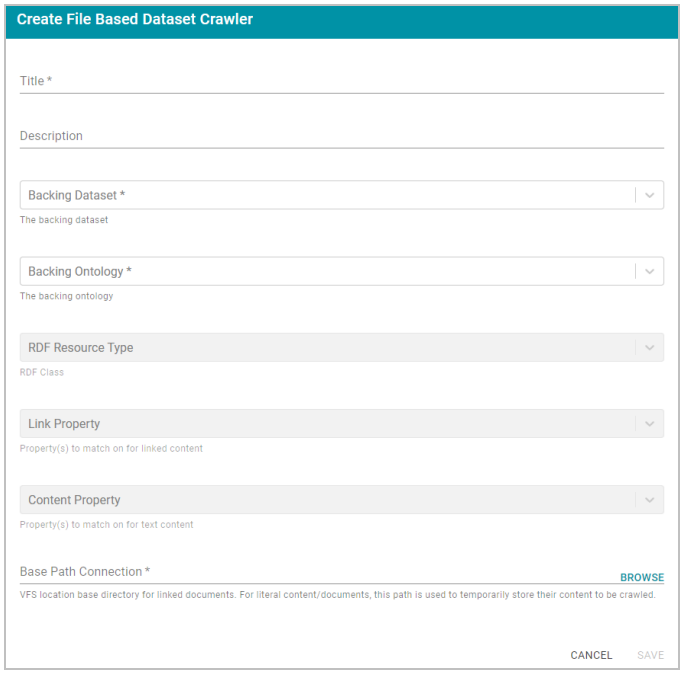


    In the Add Component dialog box, the **New** tab lists the default crawlers and the **Existing Components** tab lists crawlers that have been previously configured for other pipelines.

    b. To add a new crawler to the pipeline, click the crawler name to select it. To add an existing crawler to the pipeline, click the **Existing Components** tab, and then select a crawler. The list below describes each of the default crawlers:
        - **File Based Dataset Crawler**: Include this crawler to process data from a file-based linked data set (FLDS) in Anzo.
        - **Filesystem Crawler**: Include this crawler to process documents, such as email messages, PDF, XML, PowerPoint, Excel, OneNote, or Word files, and images, that are available on a file store.
        - **Local Volume Dataset Crawler**: Include this crawler to process RDF data that is stored as a linked data set (LDS) in an Anzo journal.
    c. After selecting a crawler, click **OK**. Anzo opens the Create dialog box for the component. Complete the fields to configure the crawler. The list below provides details about the settings for each crawler. Click a crawler name to view the details for that component:

File Based Dataset Crawler

This section describes the settings that are available on the Create File Based Dataset Crawler screen:



- **Title**: Required field that specifies the unique name for this crawler.
- **Description**: Optional field that provides a description of this crawler.
- **Backing Dataset**: Required field that specifies the Anzo data set to crawl. Click the field and select a data set from the drop-down list.
- **Backing Ontology**: Required field that specifies the model for the backing data set. Click the field and select a model from the drop-down list.
- **RDF Resource Type**: Required field that specifies the resource type or class of data to target with this crawler. Click the field and select a resource type from the drop-down list.
- **Link Property**: Optional field that specifies whether there is a link property to crawl. A link property is a property whose value identifies a linked document. For example, in the triples below, **fileLocation** is a link property:

```
<urn://someUnstructuredDocument> <urn://someProperty> "metadata
about the file" ;
<urn://fileLocation> "/path/to/file.pdf" .
```

- **Content Property**: Optional field that specifies whether there is a content property to crawl. A content property is a property whose value is a string literal, and you want Anzo to crawl and annotate the string. For example, in the triples below, **longDescription** is a content property:

```
<urn://someUnstructuredDocument> <urn://someProperty> "metadata
about the file" ;
<urn://longDescription> "this is some interesting, likely long,
```

```
unstructured text
with a lot of information, and I want to annotate it" .
```

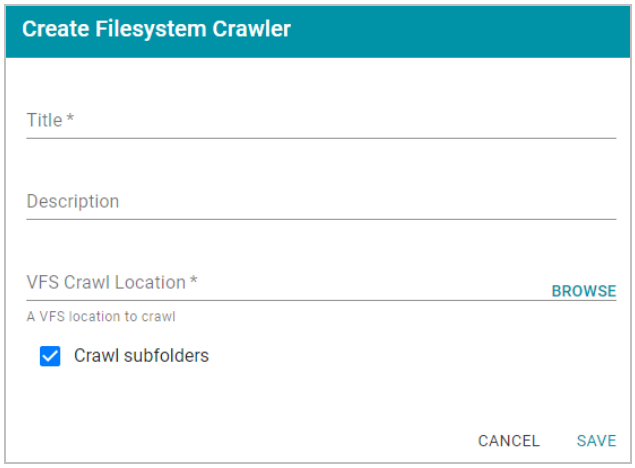
- **Base Path Connection**: Required field whose value depends on whether a link property was specified or a content property was specified:
    - If a **Link Property** was specified, the Base Path Connection is the base path to use for resolving relative file paths in the link property values. For example, in sample triples above, <urn://fileLocation> has a value of "/path/to/file.pdf." That value could be the relative path to s3://location/bucket/path/to/file.pdf or /opt/anzoshare/data/path/to/file.pdf.

      To specify the base path, click the **Base Path Connection** field. Then type or select the base path to the linked files in the File Location dialog box.
    - If a **Content Property** was specified, the Base Path Connection is a directory on the file store where Anzo can save a copy of the content property string values for the Anzo Unstructured worker instances. Saving the content to a shared file location avoids the overhead of sending the strings to the workers over the network.

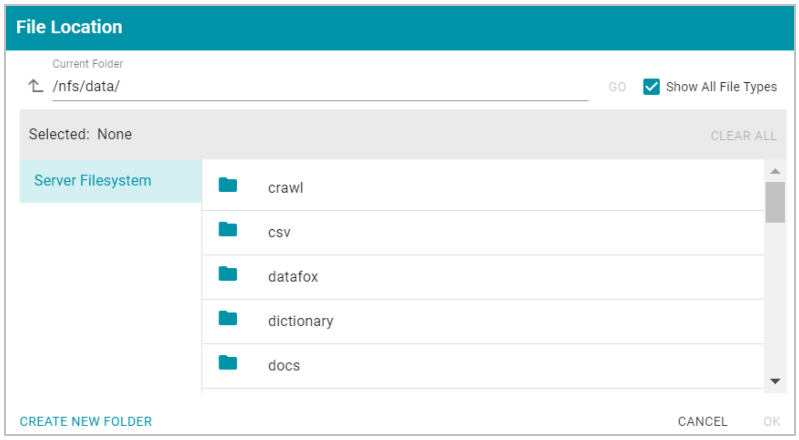
      To specify the path connection, click the **Base Path Connection** field. In the File Location dialog box, select the directory where Anzo should save the content property values.

Filesystem Crawler

This section describes the settings that are available on the Create Filesystem Crawler screen:



- **Title**: Required field that specifies the unique name for this crawler.
- **Description**: Optional field that provides a description of this crawler.
- **VFS Crawl Location**: Required field that specifies the virtual file system crawl location. Click the field to open the File Location dialog box:

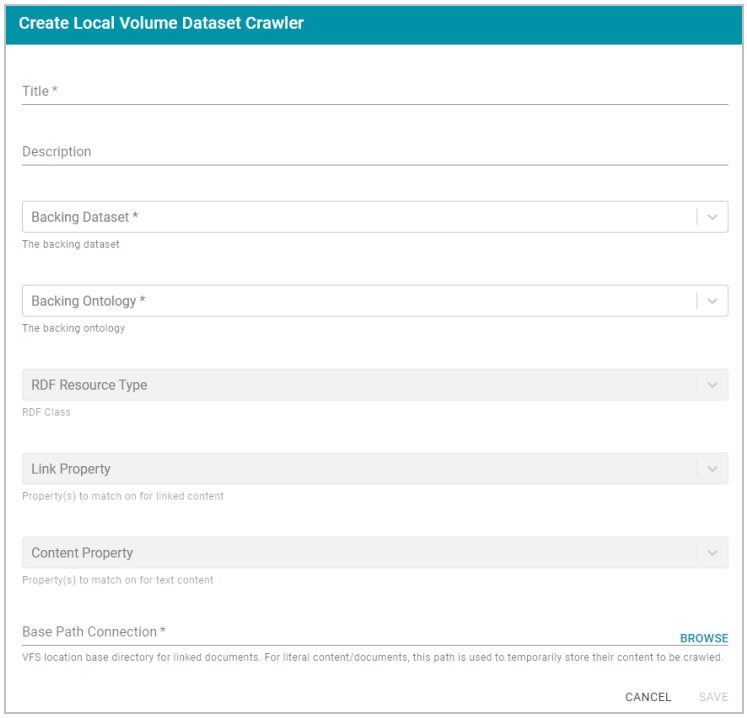


On the left side of the screen, select the storage location for the files to crawl. On the right side of the screen, navigate to the directory that contains the files. Select a directory, and then click **OK**.

- **Crawl subfolders**: Optional field that specifies whether to crawl the subdirectories under the VFS Crawl Location. To crawl the subdirectories, select the **Crawl subfolders** checkbox. To ignore subdirectories, clear the **Crawl subfolders** checkbox.

Local Volume Dataset Crawler

This section describes the settings that are available on the Create Local Volume Dataset Crawler screen:



- **Title**: Required field that specifies the unique name for this crawler.
- **Description**: Optional field that provides a description of this crawler.
- **Backing Dataset**: Required field that specifies the Anzo data set to crawl. Click the field and select a data set from the drop-down list.
- **Backing Ontology**: Required field that specifies the model for the backing data set. Click the field and select a model from the drop-down list.
- **RDF Resource Type**: Required field that specifies the resource type or class of data to target with this crawler. Click the field and select a resource type from the drop-down list.
- **Link Property**: Optional field that specifies whether there is a link property to crawl. A link property is a property whose value identifies a linked document. For example, in the triples below, **fileLocation** is a link property:

```
<urn://someUnstructuredDocument> <urn://someProperty> "metadata
about the file" ;
<urn://fileLocation> "/path/to/file.pdf" .
```

- **Content Property**: Optional field that specifies whether there is a content property to crawl. A content property is a property whose value is a string literal, and you want Anzo to crawl and annotate the string. For example, in the triples below, **longDescription** is a content property:

```
<urn://someUnstructuredDocument> <urn://someProperty> "metadata
about the file" ;
<urn://longDescription> "this is some interesting, likely long,
unstructured text
with a lot of information, and I want to annotate it" .
```

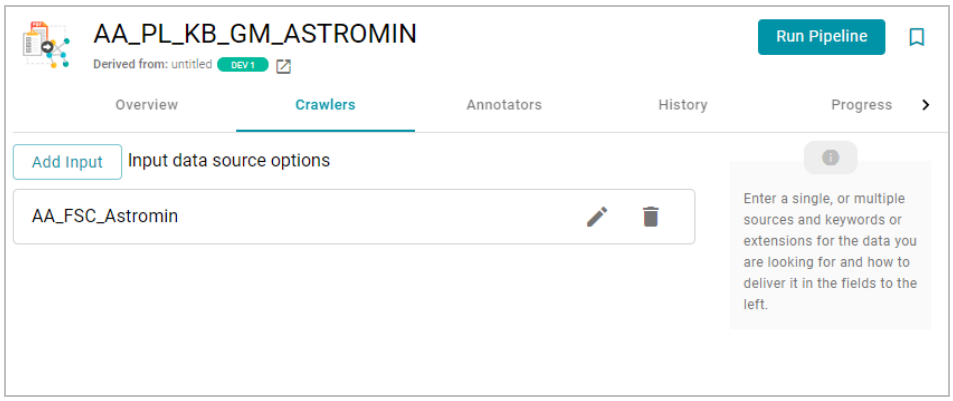
- **Base Path Connection**: Required field whose value depends on whether a link property was specified or a content property was specified:
  - If a **Link Property** was specified, the Base Path Connection is the base path to use for resolving relative file paths in the link property values. For example, in sample triples above, <urn://fileLocation> has a value of "/path/to/file.pdf." That value could be the relative path to s3://location/bucket/path/to/file.pdf or /opt/anzoshare/data/path/to/file.pdf.

    To specify the base path, click the **Base Path Connection** field. Then type or select the base path to the linked files in the File Location dialog box.
  - If a **Content Property** was specified, the Base Path Connection is a directory on the file store where Anzo can save a copy of the content property string values for the Anzo Unstructured worker instances. Saving the content to a shared file location avoids the

overhead of sending the strings to the workers over the network.

To specify the path connection, click the **Base Path Connection** field. In the File Location dialog box, select the directory where Anzo should save the content property values.

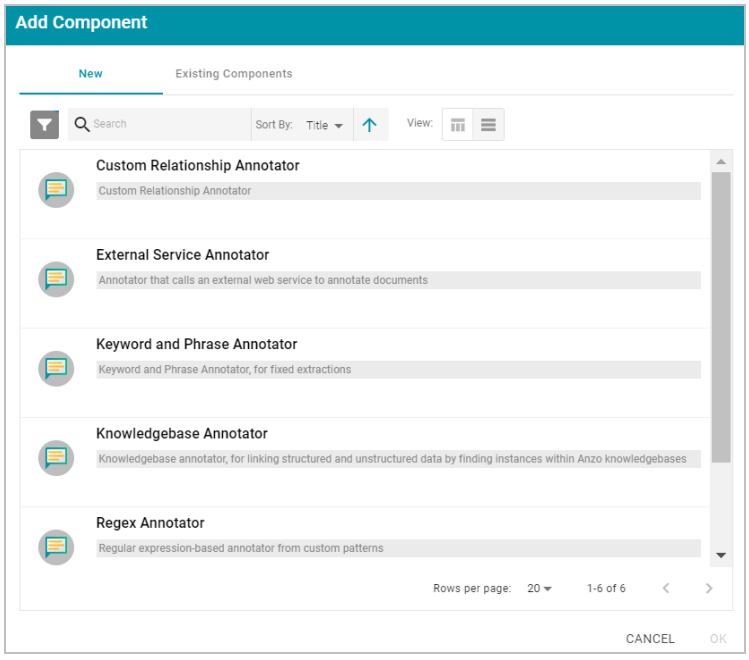
d. When you have finished configuring the crawler, click **Save**. Anzo adds the crawler to the pipeline and returns to the Crawlers screen. For example:



e. If you want to change the crawler configuration, click the Edit icon (✎) for the crawler and modify the settings as needed. If you want to add another crawler to the pipeline, repeat substeps a – d.

10. Click the **Annotators** tab and follow the substeps below to add an annotator to the pipeline:

a. Click **Add Output** to select an annotator. Anzo opens the Add Component dialog box.

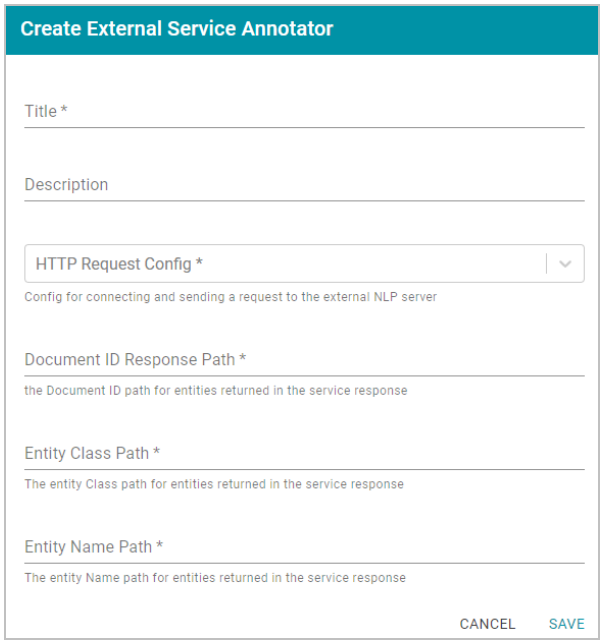


In the Add Component dialog box, the **New** tab lists the default annotators and the **Existing Components** tab lists annotators that have been previously configured for other pipelines.

b. To add a new annotator to the pipeline, click the annotator name to select it. To add an existing annotator to the pipeline, click the **Existing Components** tab, and then select an annotator. The list below describes each of the default annotators:
    - **Custom Relationship Annotator**: Include this annotator to map relationships between annotations based on the number of characters between the annotations.
    - **External Service Annotator**: Include this annotator to hit an HTTP endpoint that provides annotations.
    - **Keyword and Phrase Annotator**: Include this annotator to create annotations based on the phrases that you specify.
    - **Knowledgebase Annotator**: Include this annotator to link structured and unstructured data by finding instances in data layers, graphmarts, or Anzo linked datasets. Based on the names and aliases of entities present or patterns that are indicative of the entities, this annotator marks up the documents with the structured entities linked.
    - **Regex Annotator**: Include this annotator to use regular expression rules to identify entities such as email addresses, URLs, phone numbers, or any other entity that can be matched using a regular expression.
    - **Semantria Annotator**: Include this annotator to use the Semantria web service to find entities, sentiment, and topics in documents. It requires an Semantria API access key from Lexalytics.
    - **Significant Phrases Annotator**: Include this annotator to annotate statistically significant words and phrases.
c. After selecting an annotator, click **OK**. Anzo opens the Create dialog box for the component. Complete the fields to configure the annotator. The list below provides details about the settings for the annotators that are typically used in pipelines. Click an annotator name to view the details for that component:

    #### External Service Annotator

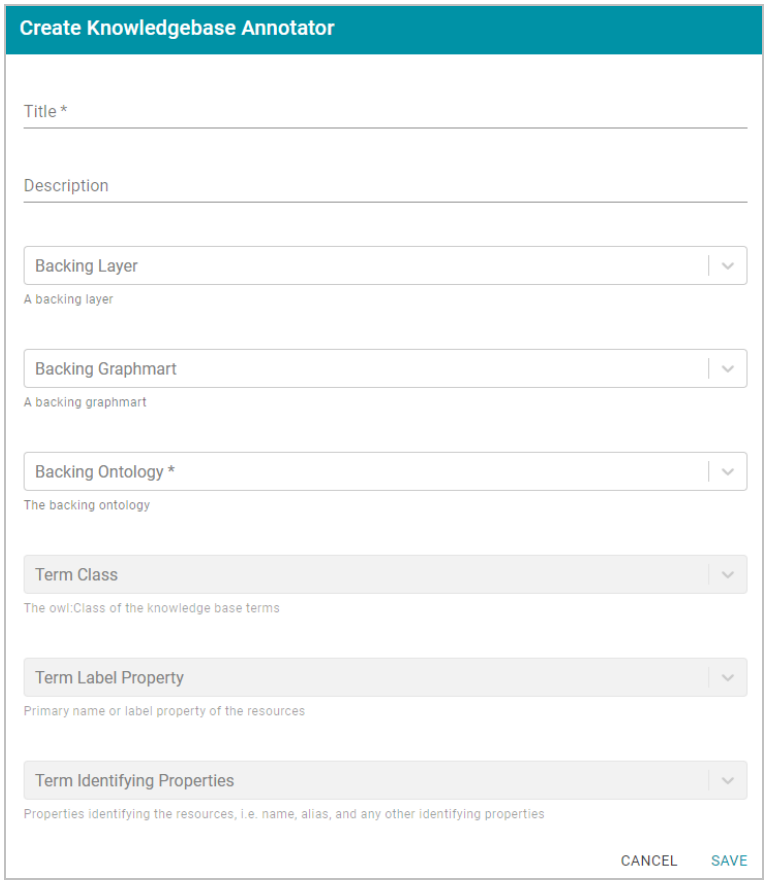
    This section describes the settings that are available on the Create External Service Annotator screen:



- **Title**: Required field that specifies the unique name for this annotator.
- **Description**: Optional field that provides a description of this annotator.
- **HTTP Request Config**: Required field that specifies the HTTP source object that contains the URL and method to use when sending data for annotations.
- **Document ID Response Path**: Required field that specifies where to find the document ID in the response.
- **Entity Class Path**: Required field that specifies the class URI for an annotation.
- **Entity Name Path**: Required field that specifies the annotation object name path.

Knowledgebase Annotator

This section describes the settings that are available on the Create Knowledgebase Annotator screen:



- **Title**: Required field that specifies the unique name for this annotator.
- **Description**: Optional field that provides a description of this annotator.
- **Backing Layer**: Optional field that specifies the data layer or layers to annotate.

  Note: The Backing Layer and Backing Graphmart fields are treated independently. Layers that you select do not have to be part of the graphmart that you specify in **Backing Graphmart**. And specifying a layer does not mean that you must select a Backing Graphmart. However, any layers or graphmarts that you select must contain classes and properties from the **Backing Ontology** or the data will not be annotated.

- **Backing Graphmart**: Optional field that specifies the graphmart or graphmarts to annotate.

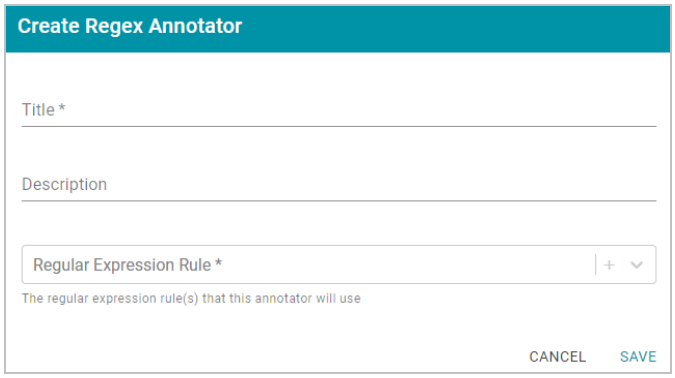
  Note: If you want the annotator to run against a linked dataset or Anzo knowledgebase instead of a data layer or graphmart, leave the Backed Layer and Backed Graphmart fields blank. After saving the pipeline, you can edit the pipeline and specify a **Backed Dataset** at that time.

- **Backing Ontology**: Required field that specifies the model for the backing data layers and/or graphmart. Click the field and select a model from the drop-down list.
- **Term Class**: Required field that specifies the class of data for the annotation.

- **Term Label Property**: Required field that lists the property for which to find entities.
- **Term Identifying Properties**: Required field that specifies the properties that contain names, aliases, or other identifiers by which you want to find entities.

Regex Annotator

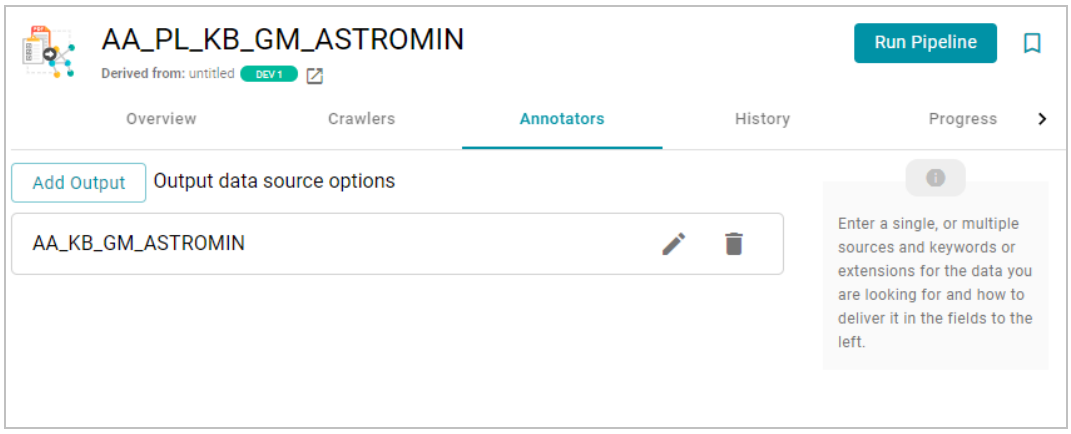
This section describes the settings that are available on the Create Regex Annotator screen:



- **Title**: Required field that specifies the unique name for this annotator.
- **Description**: Optional field that provides a description of this annotator.
- **Regular Expression Rule**: Required field the lists the regular expression rules for this annotator. To add a rule, click the plus icon (+) in the field. Anzo opens the Create Regular Expression Rule dialog box where you can define the rule:

  - **Title**: Required field that specifies the name of the rule.
  - **Description**: Optional field that describes the rule.
  - **Regular Expression**: Required field that specifies the regular expression to use for finding matching entities.
  - **Class Structure**: Required field that specifies the class structure for the entities in the format `group_number:class_name`. For example, `0:person,1:Company`.

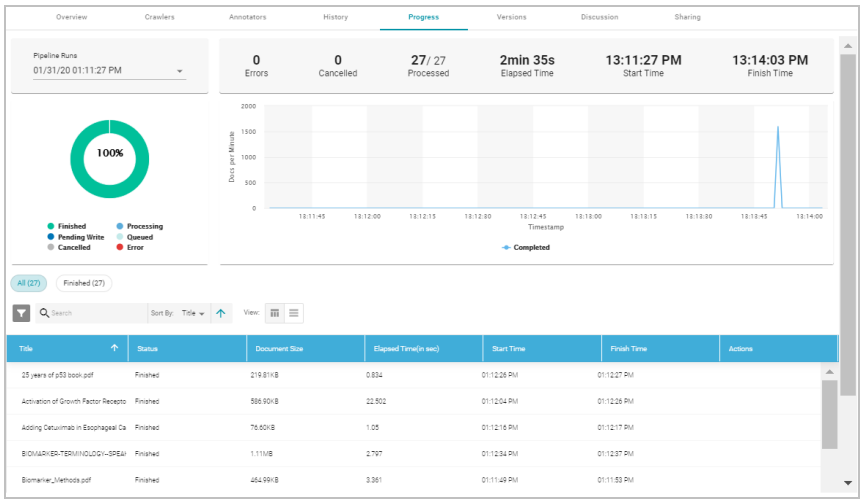
d. When you have finished configuring the annotator, click **Save**. Anzo adds the annotator to the pipeline and returns to the Annotators screen. For example:



e. If you want to change the annotator configuration, click the Edit icon ( ) for the annotator and modify the settings as needed. If you want to add another annotator to the pipeline, repeat substeps a – d.

11. When you have finished adding crawlers and annotators to the pipeline, click the **Run Pipeline** button to run the pipeline.

The process can take several minutes to complete. You can click the **Progress** tab to view details such as the pipeline status, runtime, number of documents processed, and errors. For example:



When the pipeline finishes, you can generate metrics on the graph data to start exploring the data. See Generating Graph Data Metrics for more information. You can also add the new data set to a graphmart and load it to AnzoGraph so that you can explore and analyze the data using Hi-Res Analytics. See Creating a Graphmart for instructions.

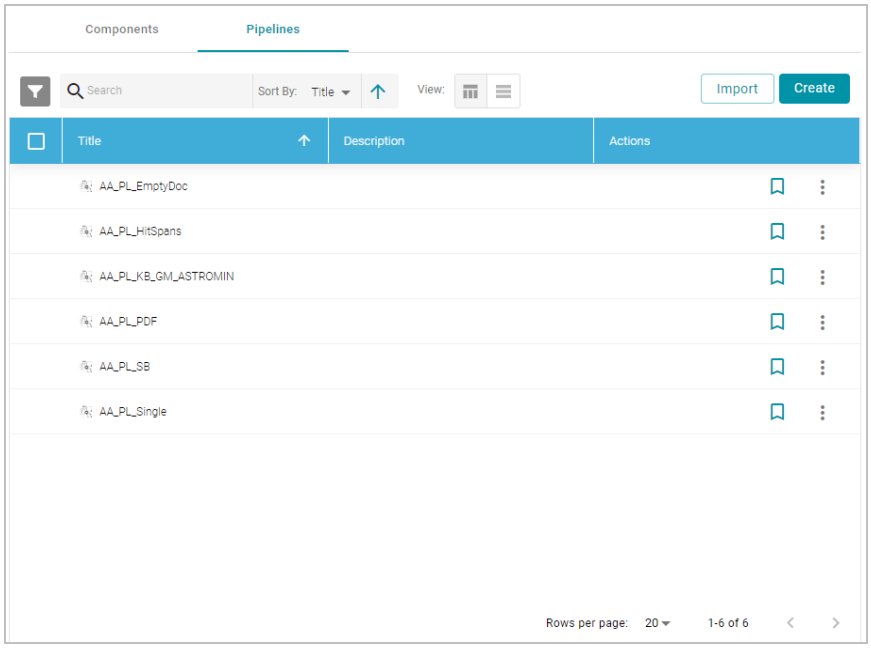
Related Topics

Running an Unstructured Pipeline

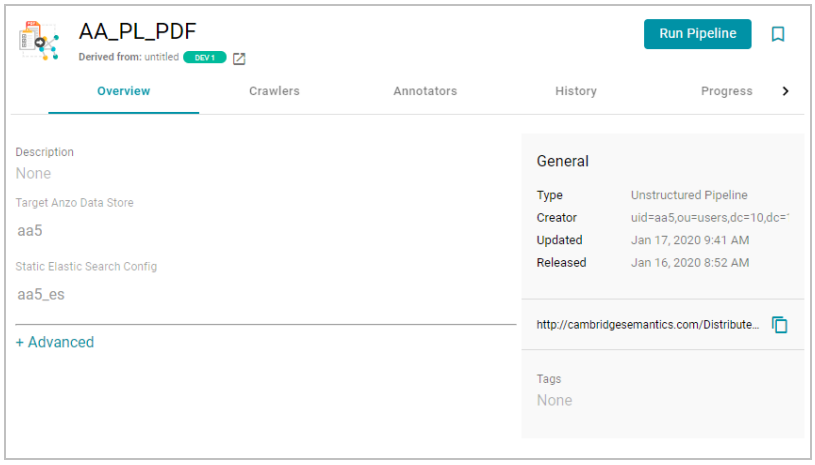
Running an Unstructured Pipeline

This page provides instructions for running an unstructured pipeline.

1. In the Anzo console, expand the **Onboard** menu and click **Unstructured Data**. Anzo displays the Pipeline screen, which lists any existing unstructured pipelines. For example:

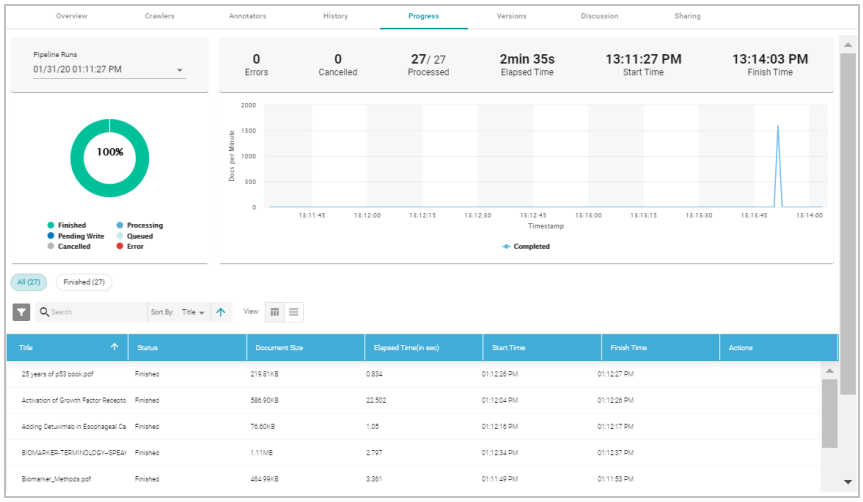


2. Click the name of the pipeline that you want to run. Anzo displays the pipeline Overview screen. For example:



3. Click **Run Pipeline** to run the pipeline.

The process can take several minutes to complete. You can click the **Progress** tab to view details such as the pipeline status, runtime, number of documents processed, and errors. For example:



When the pipeline finishes, you can generate metrics on the graph data to start exploring the data. See Generating Graph Data Metrics for more information. You can also add the new data set to a graphmart and load it to AnzoGraph so that you can explore and analyze the data using Hi-Res Analytics. See Creating a Graphmart for instructions.

Related Topics

Creating an Unstructured Pipeline

Modeling Data

Models define the business meaning of the source data. They describe the concepts, attributes, and relationships in or across data sets. Instead of reflecting the format or schema of the source data, models reflect the desired structure of the data after it is onboarded to Anzo. Anzo links data to models to provide flexibility for capturing data coming from various sources and structures and to enable users to search for and visualize data in Hi-Res Analytics dashboards or other applications.

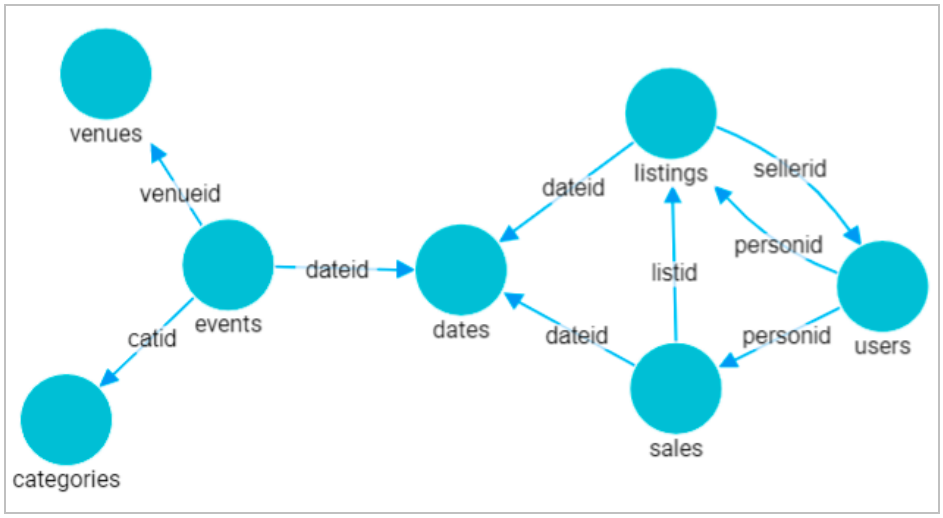
Users can import existing or external models (OWL ontologies) to Anzo, and models can be shared and reused. The topics in this section provide introductory information about data model terminology, describe model requirements and recommendations, and include instructions for creating and editing models.

- Introduction to Models
- Model Requirements and Recommendations
- Uploading a Model to Anzo
- Creating a New Model
- Editing a Model
- Setting Class Instance URI Patterns
- Downloading a Model

Introduction to Models

This topic provides a brief introduction to data models and defines the terminology that is used in Anzo.

The following image shows a portion of the model for a data set that captures sales activity for a fictional website where people buy and sell tickets for sporting events, shows, and concerts.



Class

Models are made up of classes. Classes describe a concept or a group of related objects. For example, the model above contains events, dates, categories, sales, users, and listings classes.

Property

Properties are attributes that describe the data in a class. For example the users class has properties such as firstName, lastName, and personID. The events class has properties such as eventName, dateID, and startTime.

Anzo uses two kinds of properties:

- **Data property**: Relates a class to a simple value. For example, in the users class, the firstName and lastName properties relate to simple values.
- **Object property**: Relates a class to another class. For example, the listID property relates to the sales and listing classes.

Property Type

The specific type that can be used as the value of a property. Also known as "range."

Instance

Instances are concrete occurrences of a class. For example, an event's name is an instance of the events class.

Simple value

Also known as literals. For example:

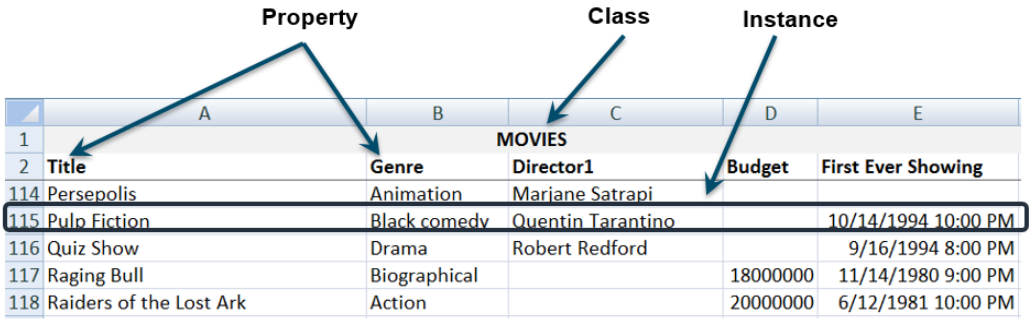
- Numbers (for example, 15, -9, 10.35)
- Text strings (for example, "Jane Doe" or "a long description")
- Dates and times (for example, "13-Dec-2008", or "April, 2017")
- Boolean (true or false)

Type

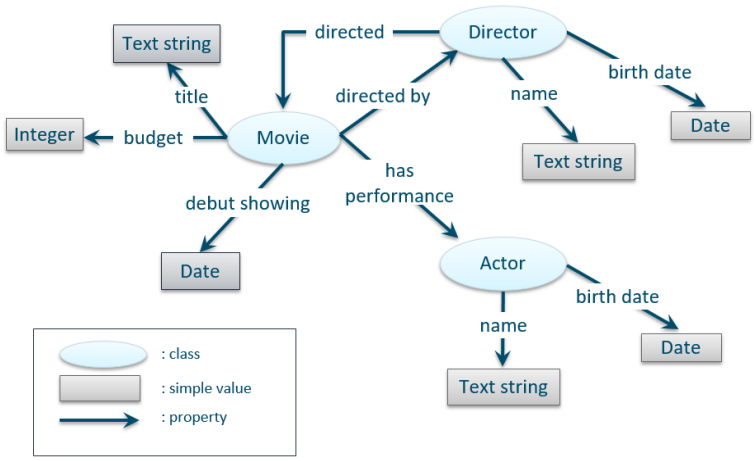
Either a class or a simple value.

Example: A Film Ontology

The example below shows classes, properties, and instances in a worksheet.

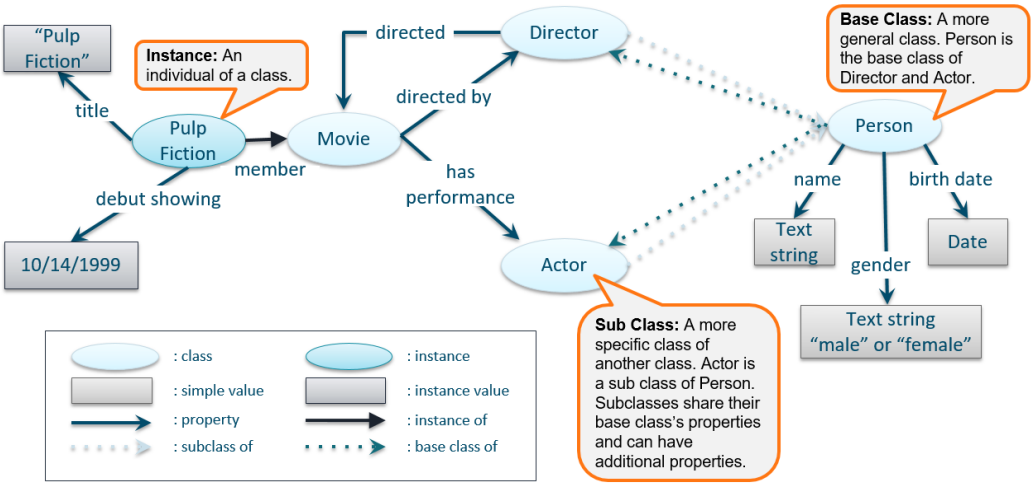


In a model, you can define relationships between the properties, instances, and classes.



Instances, Subclasses, and Base Classes

In the example below, "Pulp Fiction" is an instance of the Movie class. Person is the more general class, or base class, for Actor and Director. And Actor and Director are subclasses of Person.



Related Topics

Model Requirements and Recommendations

Uploading a Model to Anzo

Model Requirements and Recommendations

Anzo uses models to describe and manage RDF data sets. To ensure that data structures are properly defined, Anzo requires that data models include certain information and avoid unsupported information. This topic provides details about the requirements and guidelines to follow when uploading or creating models.

Requirements

This section lists the requirements or rules to follow when uploading or creating a data model. Models that are generated by Anzo during the auto-ingest process conform to these rules.

- Define each model as an owl:Ontology
- Define the model name with rdfs:label
- The named graph URI must match the ontology URI
- Define classes and concepts with owl:Class
- Define taxonomy with rdfs:subClassOf
- Define properties as owl:DatatypeProperty or owl:ObjectProperty
- Include rdfs:domain and rdfs:range for all properties
- Reference only Anzo-stored models

Define each model as an owl:Ontology

Define each data model as an **owl:Ontology**. To do so, include the following triple in the model:

```
<myOntology> a owl:Ontology
```

Where `myOntology` is the URI that names the model. The URI must be unique. To avoid unexpected results when saving a model, do not include a hash (#) character at the end of the model URI.

Define the model name with rdfs:label

Use an **rdfs:label** property to define name of the model as a string. Include the following triple:

```
<myOntology> rdfs:label "My Ontology"^^xsd:string .
```

For example, you can use the following statement as a template for inserting owl:Ontology and rdfs:label into the model:

```
<myOntology> a owl:Ontology ;
  rdfs:label "My ontology"^^xsd:string .
```

The named graph URI must match the ontology URI

Make sure that the named graph URI for the model matches the ontology URI. For example:

```
<myOntology> { <myOntology> a owl:Ontology . }
```

Like a linked data set, an ontology is a core component that is used throughout the system. The registries that store and track the graphs for core components, such as the ontology registry, expect that each graph contains a resource that matches the graph URI and specifies the type of graph. Having a mismatched graph and ontology URI can break core Anzo functionality.

Define classes and concepts with owl:Class

Use **owl:Class** for class or concept definitions. Do NOT include skos:Concept or rdfs:Class. For example, the following statement requires modification to make it valid in an Anzo model:

```
<myConcept> a skos:Concept
```

Changing the statement as follows correctly uses owl:Class instead of skos:Concept:

```
<myConcept> a owl:Class ;
  rdfs:label <businessFacingClassLabel> .
```

Define taxonomy with rdfs:subClassOf

Use **rdfs:subClassOf** for taxonomy. Do NOT use skos:broader. For example, the following statement requires modification to make it valid in an Anzo model:

```
<childSkosConcept> skos:broader <parentSkosConcept> .
```

Changing the statement as follows correctly uses rdfs:subClassOf instead of skos:broader:

```
<childOwlClass> rdfs:subClassOf <parentOwlClass> .
```

Define properties as owl:DatatypeProperty or owl:ObjectProperty

Define properties using **owl:DatatypeProperty** or **owl:ObjectProperty**. For example:

```
<myObjectProperty> a owl:ObjectProperty .
```

Or

```
<myDataTypeProperty> a owl:DatatypeProperty .
```

Include rdfs:domain and rdfs:range for all properties

Define **rdfs:domain** and **rdfs:range** for all properties. For example, the following property definition is incomplete:

```
<myObjectProperty> a owl:ObjectProperty .
```

The statement below completes the definition by adding rdfs:label, rdfs:domain, and rdfs:range:

```
<myObjectProperty> a owl:ObjectProperty ;
  rdfs:label <businessFacingPropertyLabel> ;
  rdfs:domain <myClass> ;
  rdfs:range <myOtherClass> .
```

The example below shows a valid data type definition:

```
<myDataTypeProperty> a owl:DatatypeProperty ;
  rdfs:label <businessFacingPropertyLabel> ;
  rdfs:domain <myClass> ;
  <myDataTypeProperty> rdfs:range <literal> .
```

Important: When defining the property range for integer values, use **xsd:int** instead of xsd:integer.

Reference only Anzo-stored models

Models must be self-contained or include references only to models that are stored in Anzo.

Guidelines

This section lists additional guidelines and important information to know when working with data models in Anzo.

- Property Range Guidelines
- TriG is the preferred format for models to upload
- Load RDFS and OWL vocabularies as graphs
- Axiomatically defined classes and property hierarchies not processed

Property Range Guidelines

When creating or editing properties in the model editor, Anzo offers several RDF property ranges or data types to choose from. Certain types are preferred over others, however, because they are treated consistently and predictably across systems. Cambridge Semantics recommends that you specify one of the following preferred property range values:

- **Boolean**: For true or false values.
- **Byte**: For 1-byte integers from -128 to 127.
- **Date**: For date values that follow a format such as YYYY-MM-DD.
- **Date time**: For 8-byte date and time values that follow a format such as YYYY-MM-DDThh:mm:ss.
- **Double**: For up to 8-byte double floating point values.
- **Duration**: For a duration of time expressed as a number of years, months, days, hours, minutes, and seconds in a format such as PnYnMnDTnHnMnS.

- **Float**: For up to 4-byte floating point values with potential decimal places.
- **Int**: For up to 4-byte integers from -2,147,483,648 to 2,147,483,647.
- **Long**: For up to 8-byte integers from –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.
- **Short**: For up to 2-byte integers from -32,768 to 32,767.
- **String**: For character values of varying length.
- **Time**: For time values that follow a format such as hh:mm:ss.

TriG is the preferred format for models to upload

The preferred format for models that will be uploaded to Anzo is **TriG** (.trig) format.

Load RDFS and OWL vocabularies as graphs

Anzo loads but does not process additional vocabulary data (such as rdf:subPropertyOf, owl:sameAs, and owl:intersectionOf, etc.) if they are encoded in models. Models that contain vocabularies rather than structural information should be loaded as RDF graphs instead. Anzo can load any valid RDF data. Since RDFS, SKOS, and OWL are valid RDF formats, the vocabulary information can be loaded as a graph, and the data can be interpreted with SPARQL in data layers and Hi-Res Analytics.

Axiomatically defined classes and property hierarchies not processed

When models include axiomatically defined classes or property hierarchies, Anzo loads the information but does not process the data. For example, Anzo does not infer information from axiomatically defined classes.

Related Topics

Introduction to Models

Uploading a Model to Anzo

Creating a New Model

Editing a Model

Downloading a Model

Uploading a Model to Anzo

This topic provides instructions for uploading an existing model to Anzo. Follow these instructions if you have a model that was created outside of Anzo or was downloaded from Anzo as described in Downloading a Model. Anzo accepts model files in OWL (.owl), RDF (.rdf), TriG (.trig), TTL (.ttl), and XML (.xml) format.

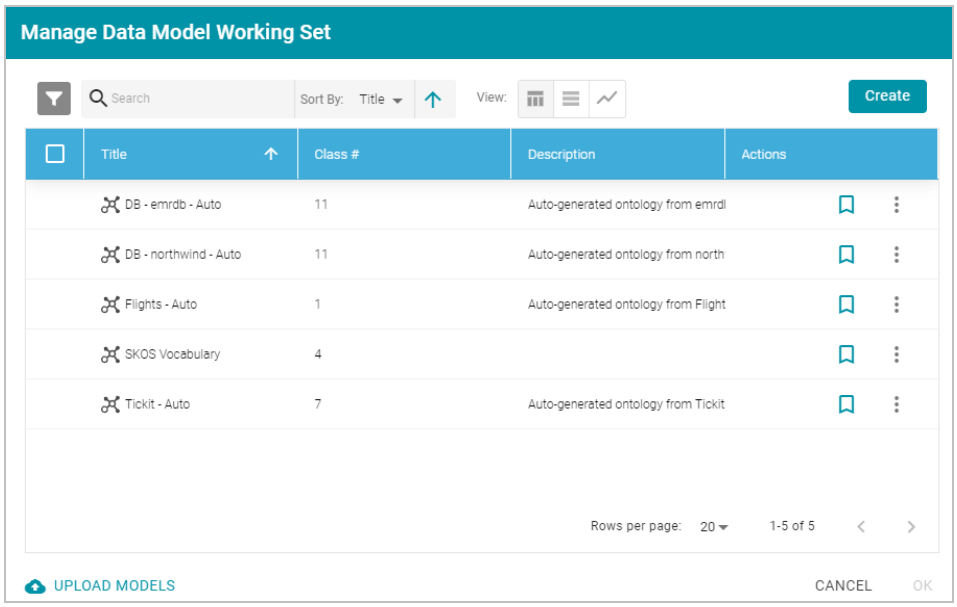
If you want to import a version of a model that was exported from Anzo as described in Exporting Versions, follow the instructions in Importing Exported Versions to import the model.

Important: Before uploading a data model to Anzo, make sure that the model meets the requirements defined in Model Requirements and Recommendations.

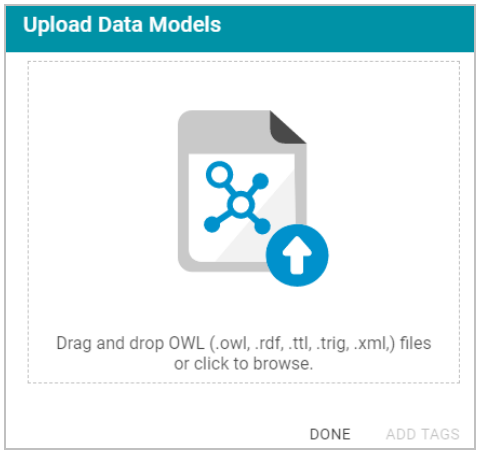
Note One of the following outcomes will occur if two users upload the same data model:

- If the second user does not have permission to modify the model that the first user uploaded, the second user receives an access denied error and cannot upload the model.
- If the second user does have permission to modify the model that the first user uploaded, Anzo overwrites the existing model with the version from user two.

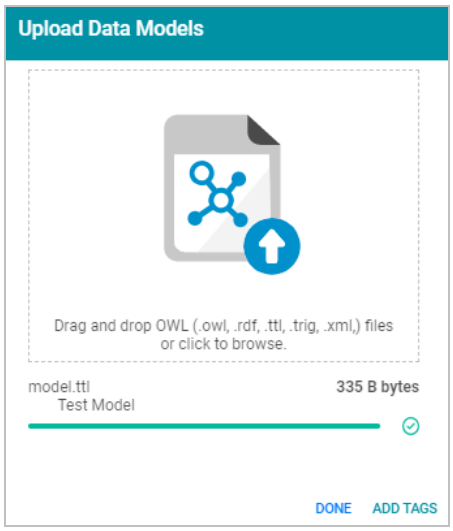
1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



2. On the bottom left corner of the screen, click **Upload Models**. The Upload Data Models dialog box opens.



3. To upload a model, drag and drop the file onto the dialog box or click the text to browse and select the file on your computer. Anzo uploads the model that you selected and displays the file name and size. For example:



If you want to upload additional models, you can repeat the process and drag and drop or select files on the Upload Data Models dialog box.

4. Click **Done** when you finish uploading models. The new models become available on the Manage Data Model Working Set screen.

For information about editing models using the Data Model editor, see Editing a Model.

Related Topics

Introduction to Models

Model Requirements and Recommendations

Creating a New Model

Editing a Model

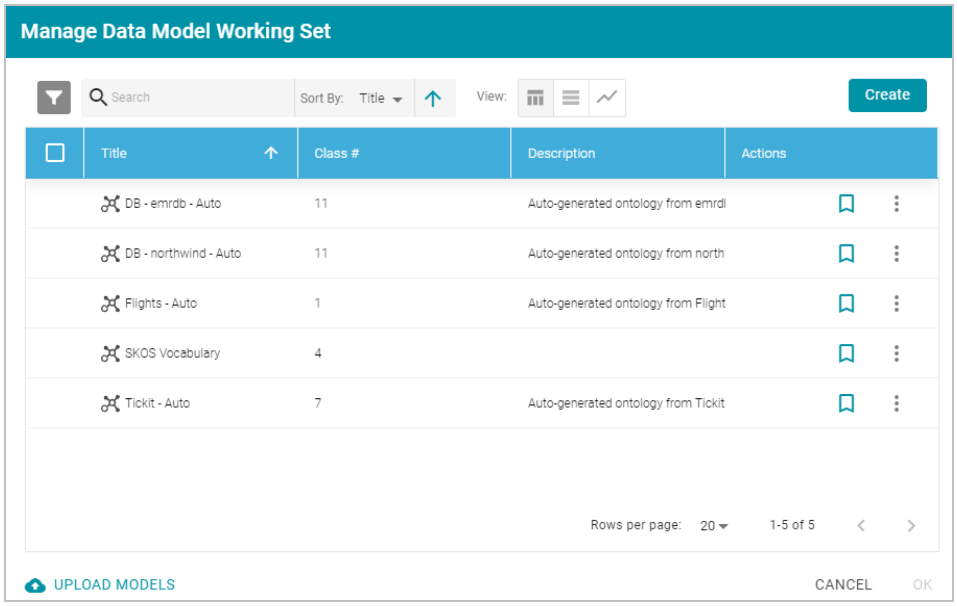
Downloading a Model

Creating a New Model

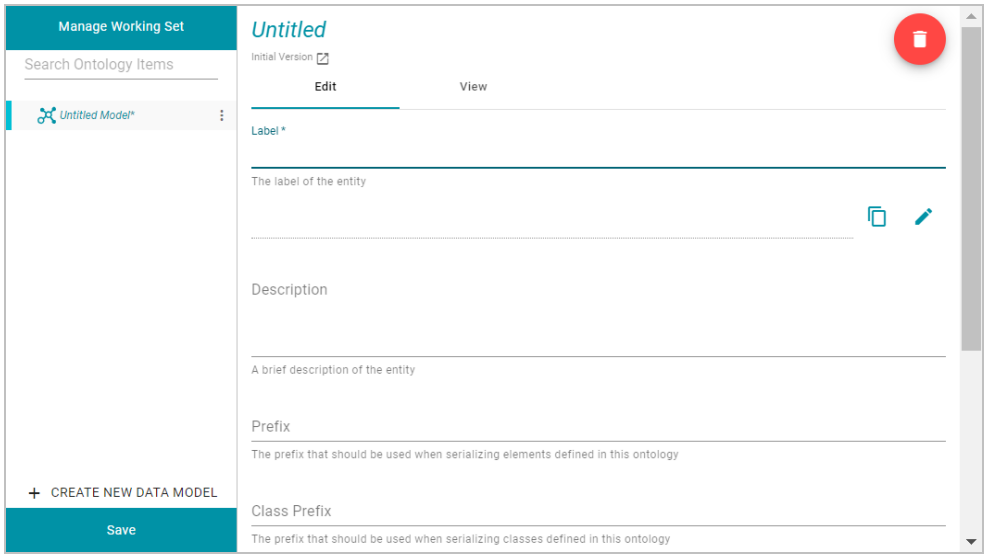
This topic provides instructions for creating a new data model in the Anzo console.

Important: When creating a new data model, follow the requirements and guidelines defined in Model Requirements and Recommendations.

1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



2. Click the **Create** button on the top right of the screen. Anzo displays the Model editor.



3. In the **Label** field, type a unique name for the model.
4. Provide the following optional information as needed:
    - **Description**: A brief description of the model.
    - **Prefix**: Enables you to include an prefix as a hint for TriG serialization. For example, the prefix for the Friend of a Friend (FOAF) model is "foaf," and the prefix for Dublin Core is "dc." Enter a prefix if you want customize the way the data looks when Anzo serializes it.
    - **Imports**: Enables you to import the definitions from another model into this model. To select models to import, click in the Imports field and select a model from the drop-down list. You can press **Ctrl** and select multiple models.

- **System Model**: Indicates that the data model is a system model only and not related to business data.
- **Hidden Model**: Hides the data model so that it is not associated with business data.

5. Click **Save** to save the model.

For information about adding classes and properties to the new model, see Editing a Model. You can change or create a mapping to associate the new model with a data set. For information, see Working with Mappings.

Related Topics

Introduction to Models

Model Requirements and Recommendations

Uploading a Model to Anzo

Editing a Model

Downloading a Model

Editing a Model

This topic provides information about using the Anzo model editor to open a data model and modify it to add, edit, or remove classes, properties, data ranges, and annotations.

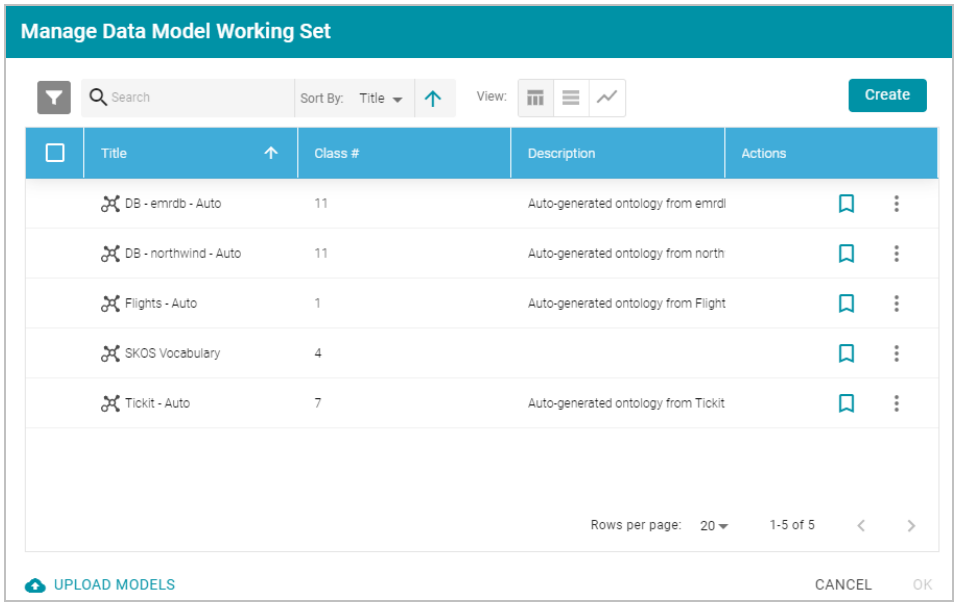
Important: When editing data models, follow the requirements and guidelines defined in Model Requirements and Recommendations.

- Opening Models in the Editor
- Adding, Deleting, or Changing Model Components

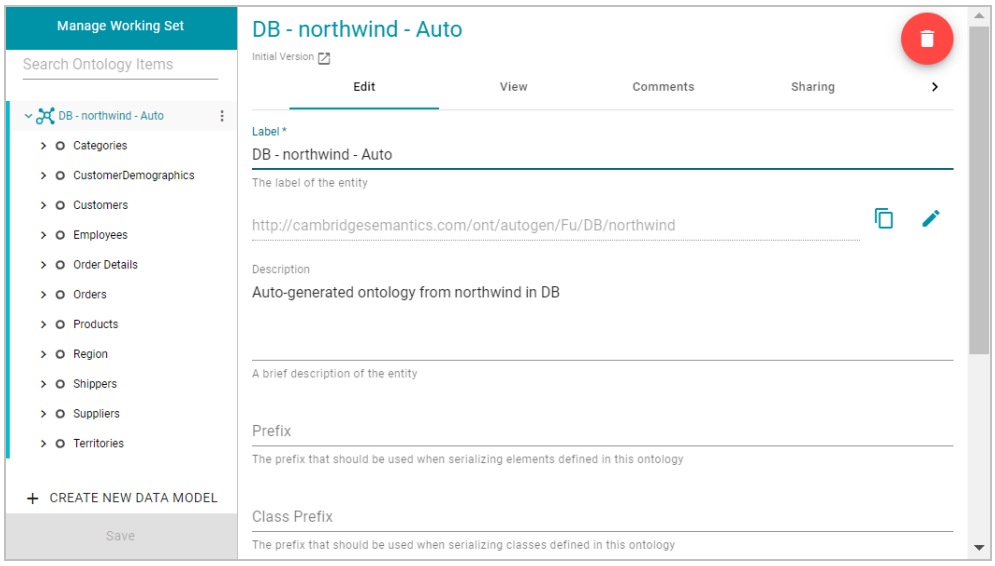
Note Before editing a data model, you have the option to create a backup of the current version. For more information, see Creating and Restoring Versions.

Opening Models in the Editor

1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



2. On the Manage Working Set screen, select the model that you want to edit, and then click **OK**. Anzo opens the selected model in the editor. For example:

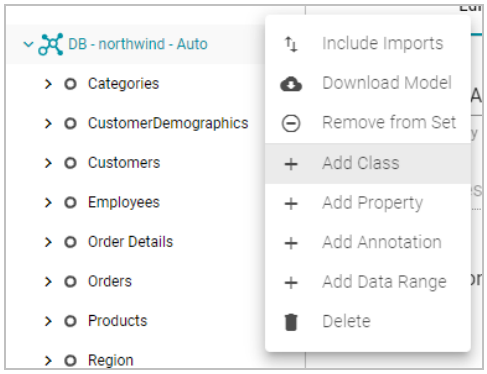


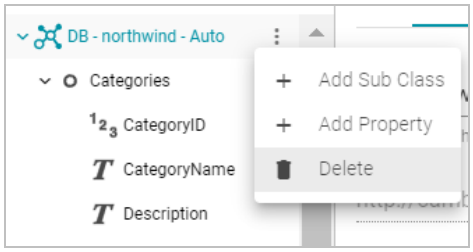
3. You can edit the following model-level settings or view the Adding, Deleting, or Changing Model Components section below for information about working with classes, properties, annotations, and data ranges.
    - **Description**: A brief description of the model.
    - **Prefix**: Enables you to include an prefix as a hint for TriG serialization. For example, the prefix for the Friend of a Friend (FOAF) model is "foaf," and the prefix for Dublin Core is "dc." Enter a prefix if you want customize the way the data looks when Anzo serializes it.

- **Imports**: Enables you to import the definitions from another model into this model. To select models to import, click in the Imports field and select a model from the drop-down list. You can press **Ctrl** and select multiple models.
- **System Model**: Indicates that the data model is a system model only and not related to business data.
- **Hidden Model**: Hides the data model so that it is not associated with business data.

Adding, Deleting, or Changing Model Components

The table below provides instructions for working with model components. When modifying models, make sure that you click **Save** periodically to save your changes.

What do you want to do?	Instructions
Create a new class	Open the model menu by clicking the menu icon (⋮) to the right of the model name. Then select **Add Class**.  Anzo opens the class editor so that you can configure the new class. See Class Editor Reference below for information about class settings.
Create a new property in a class	Open the class menu by clicking the menu icon (⋮) to the right of the class name. Then select **Add Property**. Anzo opens the property editor so you can configure the new property. See Property Editor Reference below for information about property settings.
Add an existing property to a class	To add an existing property to a class, click the class in the left pane to display the class details in the editor. In the editor, click in the **Properties** field and select the property that you want to add from the drop-down list.

What do you want to do?	Instructions
Edit a class	To change an existing class, select the class in the left pane of the working set. Anzo expands the class to show its properties and displays the details for that class in the editor. You can make changes in the editor. See Class Editor Reference below for information about class settings.
Delete a property from a class	In the left pane of the working set, select the property that you want to delete. Anzo opens that property in the editor. To remove the property, click the trashcan icon ( ) on the top right of the screen. Then click **OK** in the Delete dialog box to confirm that you want to delete the property.
Delete a class from the model	Open the class menu by clicking the menu icon (⋮) to the right of the class that you want to remove from the model.  Click **Delete** and then click **OK** in the Delete dialog box to confirm that you want to delete the class. **Note**: This action cannot be undone. Anzo removes the class and saves the model.
Add a data range	Open the model menu by clicking the menu icon (⋮) to the right of the model name. Then select **Add Data Range**. Anzo opens the data range editor so that you can configure the new range.
Add an annotation	Open the model menu by clicking the menu icon (⋮) to the right of the model name. Then select **Add Annotation**. Anzo opens the editor so that you can configure the annotation.

Class Editor Reference

This section describes each of the fields that are available for configuring classes.

Field	Description
Label	The name of the class.
Description	A brief description of the class.
Model	The name of the model or models that the class belongs to.
Parent Classes	Lists any parent classes under which this class becomes a child or subclass. Click in the field to select parent classes from the drop-down list. Or click the X to the left of a class name to remove that parent class from the list.
Properties	Lists the properties under this class. Click in the field to select properties from the drop-down list. Or click the X to the left of a property name to remove that property from the list.
Resource Template	Defines the Uniform Resource Identifier (URI) template to use for instances of the class. You can construct URI templates by typing a value and pressing **Enter** or by choosing an available property from the drop-down list. For more information, see Setting Class Instance URI Patterns.
Graph Template	Defines the graph URI template to use for instances of the class. You can construct graph URI templates by typing a value and pressing **Enter** or by choosing an available property from the drop-down list. You can concatenate the specified graph template value with values of properties in the class. For example: `http://cambridgesemantics.com/graph/` and `Title`
Preview Property	Defines a property from the class to use as the "name" or entity on default displays. For example, if there is a reference to entity X, and entity X has Name, Title, and Label properties, you could specify that you want Title to display by default instead of "X."

Property Editor Reference

This section describes each of the fields that are available for configuring properties.

Field	Description
Label	The name of the property.

Field	Description
Description	A brief description of the property.
Required	Indicates whether a value is required for this property.
Multi Value	Indicates whether more than one value can exist for this property. **Note**: Some business intelligence (BI) applications have limitations on the retrieval of multi-value properties. If you use the Anzo Data on Demand service to query data from third-party BI tools, consider whether your application supports multi-value properties before creating them.
Domain	Lists the class or classes that the property belongs to.
Has Data Range	Indicates whether the property has a single data type or a data range. Selecting this checkbox displays the Data Range field so that you can choose the data range.
Property Range	The data type for the property. See Property Range Guidelines for recommendations on choosing property ranges.
Min Cardinality	The minimum number of distinct values a property can have. When Min Cardinality is blank, the number of values is unrestricted.
Max Cardinality	The maximum number of distinct values a property can have. When Max Cardinality is blank, the number of values is unrestricted.
Value Restriction	Indicates whether to restrict the property's values to certain data types or specific values in a list.

Related Topics

Introduction to Models

Model Requirements and Recommendations

Uploading a Model to Anzo

Creating a New Model

Downloading a Model

Setting Class Instance URI Patterns

Setting Class Instance URI Patterns

When you open a data model in the Model editor, there is a **Resource Template** setting for each of the classes in the model. A Resource Template defines the Uniform Resource Identifier (URI) pattern that Anzo should follow when ingesting data and generating the URIs for the instances of each class.

When using the Ingest workflow (with Anzo-generated models, mappings, and pipelines), if a Resource Template is not defined for the classes in a model, Anzo generates class URIs by following this pattern:

```
<uri_prefix>/<class_name>/<primary_key>
```

Anzo uses the URI prefix of **http://csi.com/**, appends the name of the table (class), and adds the primary key value for each instance of the table. For example, the following URI is generated for an instance of a class called MovieActors2. The primary key for the MovieActors2 table is ActorID, so the ActorID value is appended to the URI.

```
<http://csi.com/MovieActors2/31211756>
```

Note For property URIs, the default URI prefix is **http://cambridgesemantics.com/**. The value is controlled by the URI Prefix option in system settings. See Changing URI Prefixes and SPARQL Options for more information.

Defining a Resource Template for the classes in your models helps link and relate data by using URI patterns that express the meaning of the data and combine similar concepts. Additionally, simpler and more meaningful URIs are easier to read and therefore easier to write in queries.

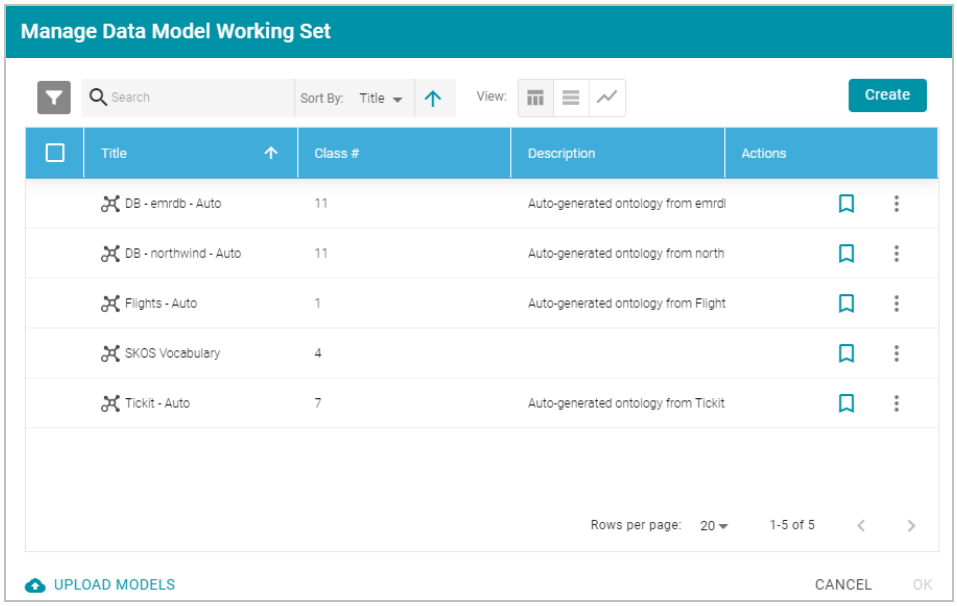
Example If you ingest movie data from multiple sources and each source assigns a movie ID as the primary key, the same movie title will likely be associated with multiple IDs. If the auto-generated URI pattern for the Movies class is http://csi.com/Movies/<movie_ID>, then all of the data for the same movie title will not be joined by the same class instance URI. In this case, defining a resource template that uses the movie title as the uniqueness condition rather than the ID would automatically join movie data from different sources.

Follow the steps below to configure resource templates for each of the classes in a data model. For automatic ingestion workflows, configure resource templates after Anzo generates the model, mappings, and pipeline and before the pipeline is published. To configure resource templates for pipelines that have been published, edit the model and then re-publish the pipeline to update the instance data.

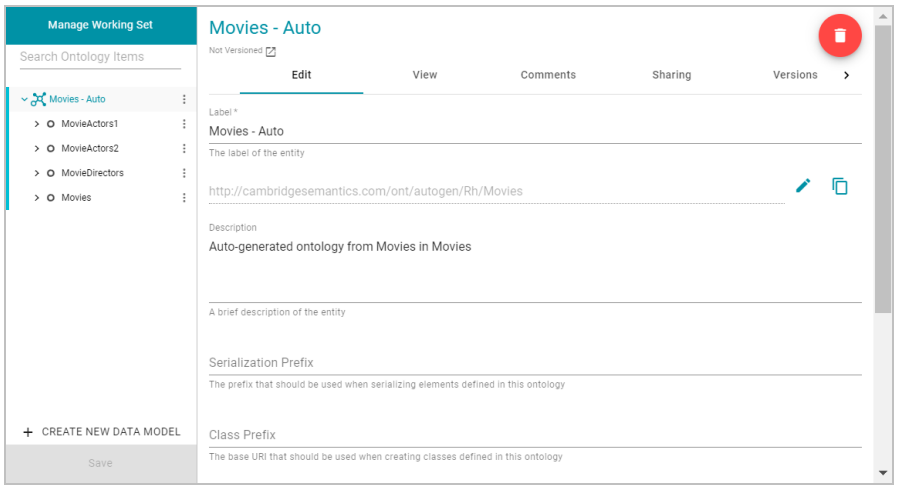
Important Keep the following points in mind when defining class instance URI patterns:

- Avoid joining data that should not be joined. For example, using a property such as YearProduced in a movies Resource Template would group all movies from a given year as a single instance.
- Resource Templates with multiple components must have all components present. If a component is missing, Anzo generates random strings for missing Resource Template components.
- Resource templates do not work across different classes. You must define resource templates on individual classes.

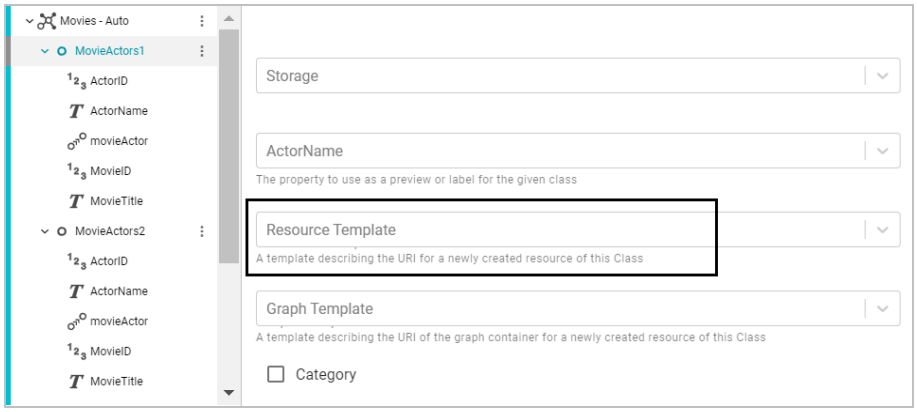
1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



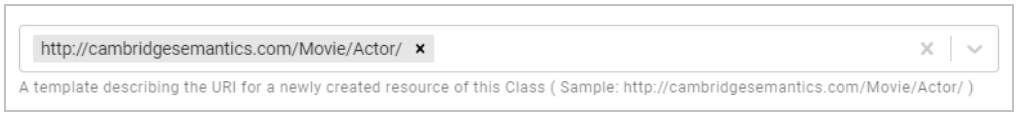
2. On the Manage Working Set screen, select the checkbox next to the model (or models) that you want to add to the working set for editing. Then click **OK**. Anzo opens the selected model in the editor. For example:



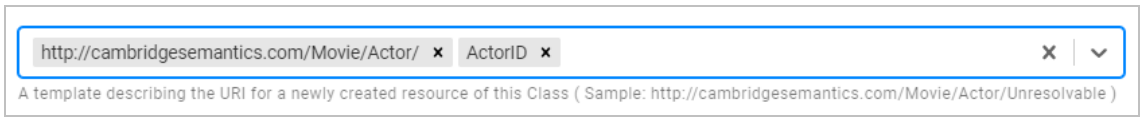
3. Select a class in the model to display the settings for that class. Then scroll down to the **Resource Template** field. For example, the image below shows the Resource Template field for the selected MovieActors1 class.



4. Click the **Resource Template** field and specify the URI pattern to use for instances of this class. First, type a base value in the field and press **Enter** to add the value to the field. For example, for MovieActors1 in the step above: http://cambridgesemantics.com/Movie/Actor.



Then click the field again and select a property in the class that defines the class, i.e., contains unique values. For example, in the MovieActors1 class, ActorID provides unique values.



5. Click **Save** to save the change, and then select another class for which to set a Resource Template. Repeat the step above for each class in the model.

Related Topics

Introduction to Models

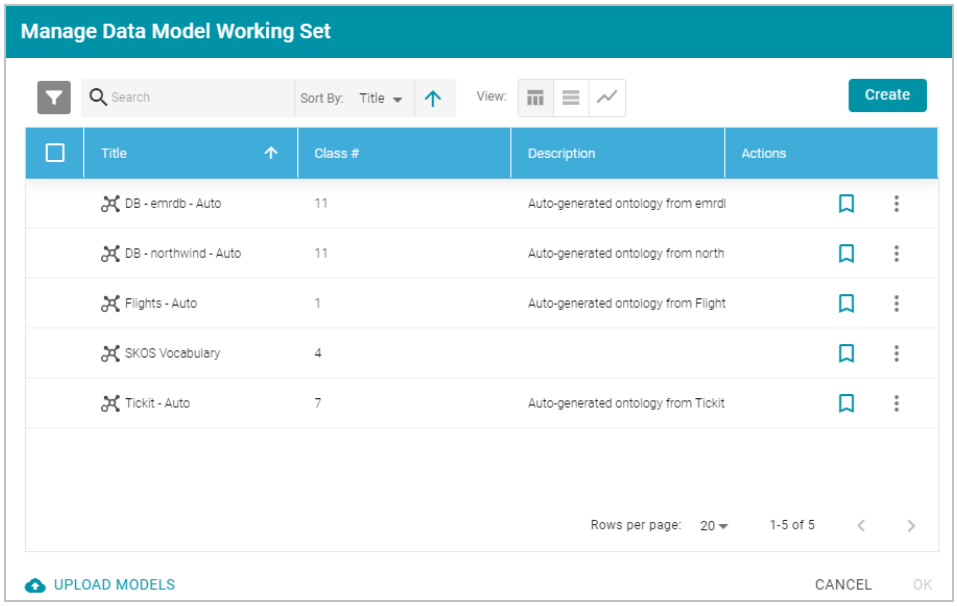
Model Requirements and Recommendations

Editing a Model

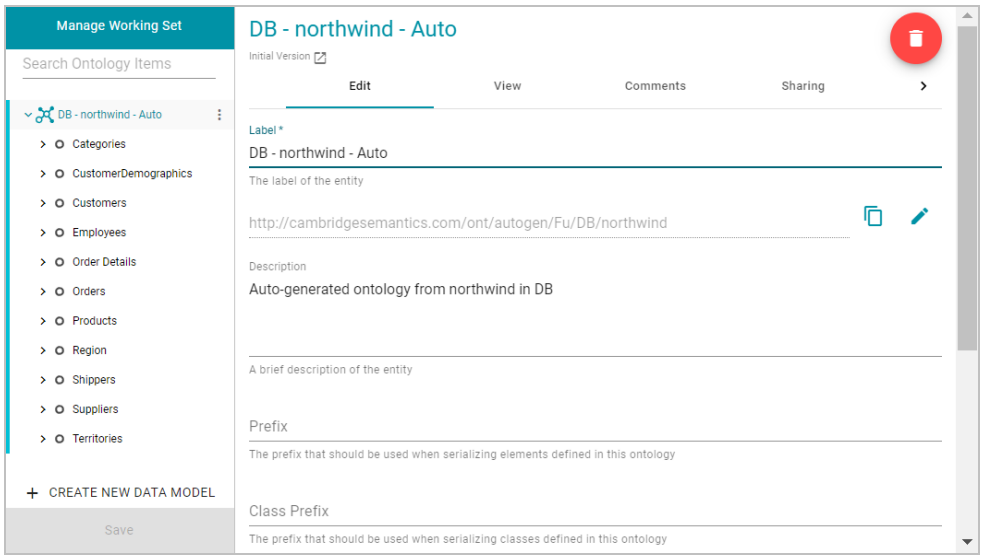
Downloading a Model

This topic provides instructions for downloading a data model from Anzo.

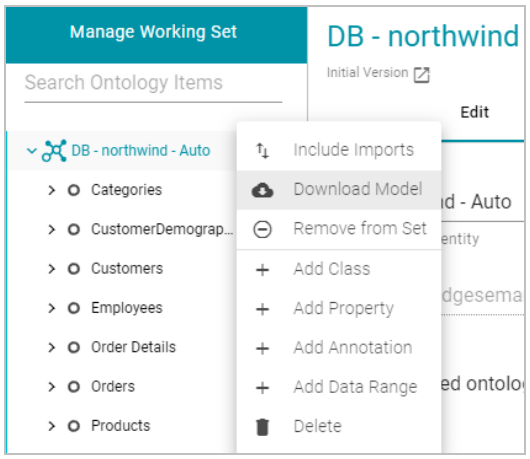
1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



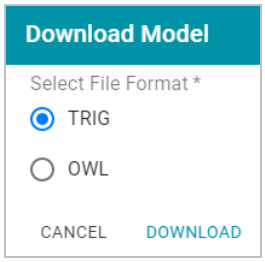
2. On the Manage Working Set screen, select the checkbox next to the model that you want to export, and then click **OK**. Anzo opens the selected model in the editor. For example:



3. Open the model menu by clicking the menu icon (⋮) to the right of the model name. Then select **Download Model**.



Anzo displays the Download Model dialog box:



4. In the Download Model dialog box, select the format to save the model in. By default Anzo saves models in **TRIG** format. If you want to save the file in OWL format, select the **OWL** radio button. Then click **Download**.

Anzo downloads the model to your computer in the selected format.

Note: When a data model is downloaded from Anzo, the resulting TriG or OWL file size can be significantly larger than the file size of the original data model file that was uploaded. The original model likely includes prefix specifications and abbreviated URIs. When a model is exported, however, Anzo replaces the prefixes with full URIs. In addition, the downloaded model includes the Anzo-generated metadata for the model. For example, the following simple example TTL content shows part of a data model that uses prefixes:

```
@prefix csi: <http://cambridgesemantics.com/2017/02/ont#> .
csi:testModel a owl:Ontology ;
  rdfs:label "Test Model"^^xsd:string .
csi:DOB a owl:Class;
  rdfs:domain csi:Demographics ;
  rdfs:label "DOB" ;
  rdfs:range xsd:string .
csi:HEIGHT a owl:Class;
  rdfs:domain csi:Demographics ;
  rdfs:label "HEIGHT" ;
  rdfs:range xsd:decimal .
```

After uploading the TTL file and then downloading the model in TriG format, the resulting file includes full URIs as well as the model's metadata:

```
<http://cambridgesemantics.com/2017/02/ont#testModel> {
  <http://cambridgesemantics.com/2017/02/ont#DOB> a <http://www.w3.org/2002/07/owl#Class>
;
    <http://www.w3.org/2000/01/rdf-schema#domain>
<http://cambridgesemantics.com/2017/02/ont#Demographics> ;
    <http://www.w3.org/2000/01/rdf-schema#label> "DOB" ;
    <http://www.w3.org/2000/01/rdf-schema#range> <http://www.w3.org/2001/XMLSchema#string>
.

  <http://cambridgesemantics.com/2017/02/ont#HEIGHT> a
<http://www.w3.org/2002/07/owl#Class> ;
    <http://www.w3.org/2000/01/rdf-schema#domain>
<http://cambridgesemantics.com/2017/02/ont#Demographics> ;
    <http://www.w3.org/2000/01/rdf-schema#label> "HEIGHT" ;
    <http://www.w3.org/2000/01/rdf-schema#range>
<http://www.w3.org/2001/XMLSchema#decimal> .
  <http://cambridgesemantics.com/2017/02/ont#testModel> a
<http://www.w3.org/2002/07/owl#Ontology> ;
    <http://www.w3.org/2000/01/rdf-schema#label> "Test Model" .
}
<http://cambridgesemantics.com/registries/Ontologies> {
  <http://cambridgesemantics.com/registries/Ontologies>
    <http://openanzo.org/ontologies/2008/07/Anzo#defaultNamedGraph>
    <http://cambridgesemantics.com/2017/02/ont#testModel> ;
    a <http://openanzo.org/ontologies/2008/07/Anzo#Dataset> .
}
<http://openanzo.org/metadataGraphs
(http%3A%2F%2Fcambridgesemantics.com%2F2017%2F02%2Font%23testModel)> {
  <http://cambridgesemantics.com/2017/02/ont#testModel>
    <http://openanzo.org/ontologies/2008/07/Anzo#canBeAddedToBy>
    <http://openanzo.org/system/internal/sysadmin> ;
  <http://openanzo.org/ontologies/2008/07/Anzo#canBeReadBy>
    <http://openanzo.org/Role/everyone> , <http://openanzo.org/system/internal/sysadmin> ;
    <http://openanzo.org/ontologies/2008/07/Anzo#canBeRemovedFromBy>
    <http://openanzo.org/system/internal/sysadmin> .
  ...
}
```

Related Topics

Introduction to Models

Model Requirements and Recommendations

Uploading a Model to Anzo

Blending Data

Data in Anzo's Dataset catalog can be blended to meet the needs of the overall business data model. Users can create links between related but previously siloed data elements, they can apply cleansing, transformation, or validation steps, and they can harmonize and align data to meet analytic needs.

The topics in this section provide information about creating and loading graphmarts as well as enhancing graphmarts with data layers.

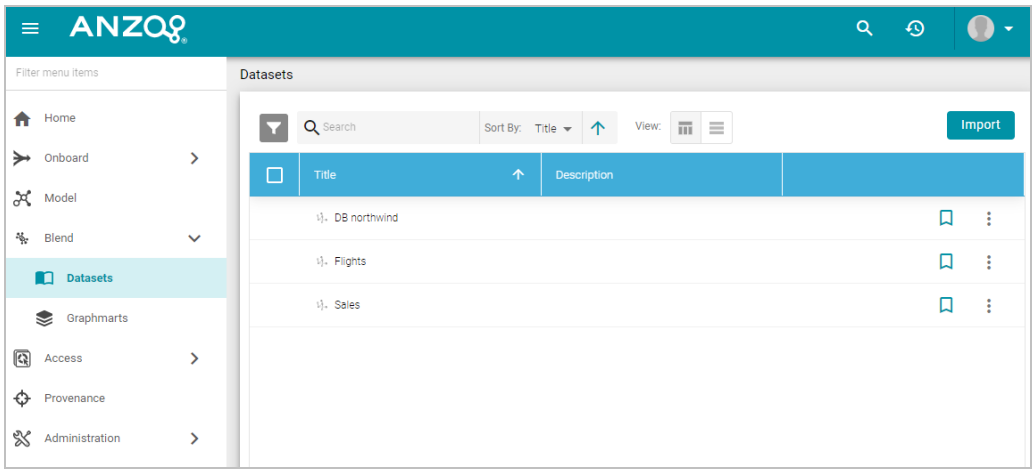
- Generating Graph Data Metrics
- Creating a Graphmart
- Adding a Data Set to a Graphmart
- Introduction to Data Layers
- Adding Data Layers to Graphmarts
- Adding Steps to Data Layers
- Masking Data in Data Layers
- Data Layer Hi-Res Analytics Settings Reference
- Managing Graphmart, Data Layer, and Step Security
- Blending Data with the Data Toolkit Service

Generating Graph Data Metrics

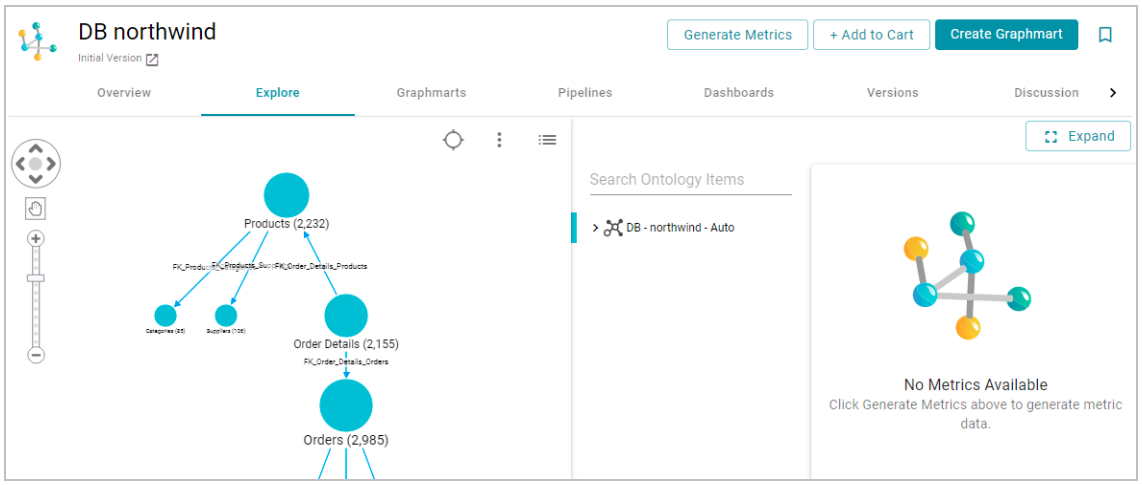
Similar to generating metrics for a data source schema (see Generating Source Data Metrics), Anzo provides the ability to calculate metrics for a graph data set in its final format. When metrics are generated for graph data, Anzo profiles the entire data set and reports metrics for the classes and properties in the model as well as statistics about the values for the properties. Generating graph data metrics helps users perform data discovery, assess the quality of the onboarded data, and decide whether to use the data set in a particular graphmart. The report can also assist users in determining the types of data layers to create and writing the queries to include in the steps.

Important To generate graph data metrics, AnzoGraph must be online. If you have dynamic AnzoGraph deployments enabled, Anzo will provision AnzoGraph automatically when metrics are generated.

1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Datasets screen, which lists the catalog of data sets. For example:

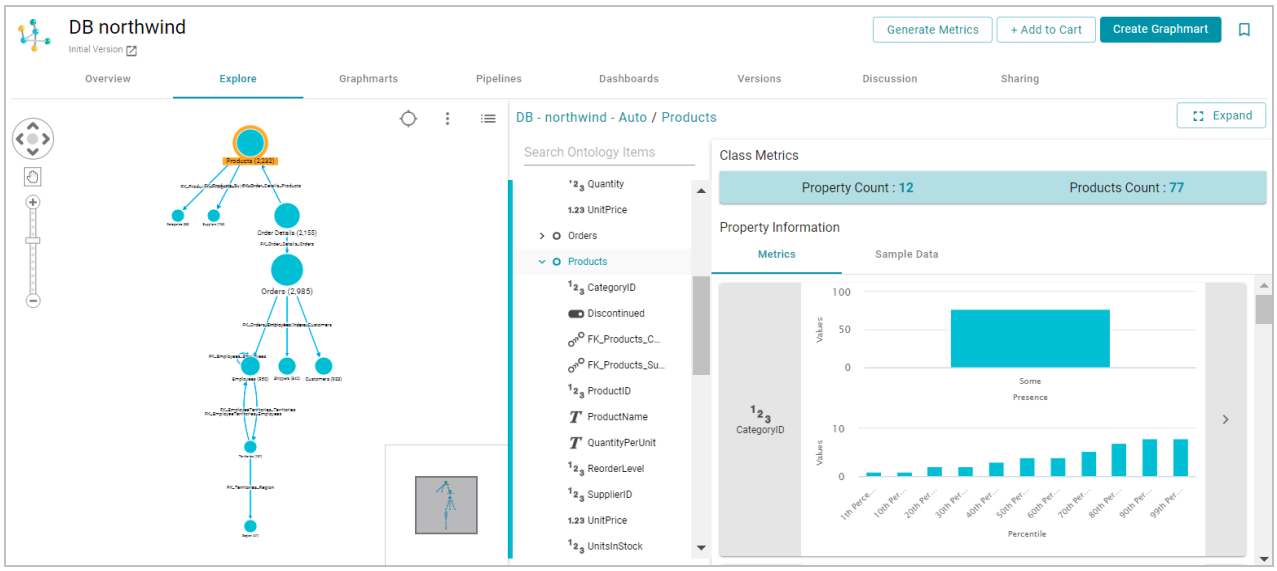


2. On the Datasets screen, click the name of the data set for which you want to generate metrics. Anzo displays the Explore screen for the source. For example:

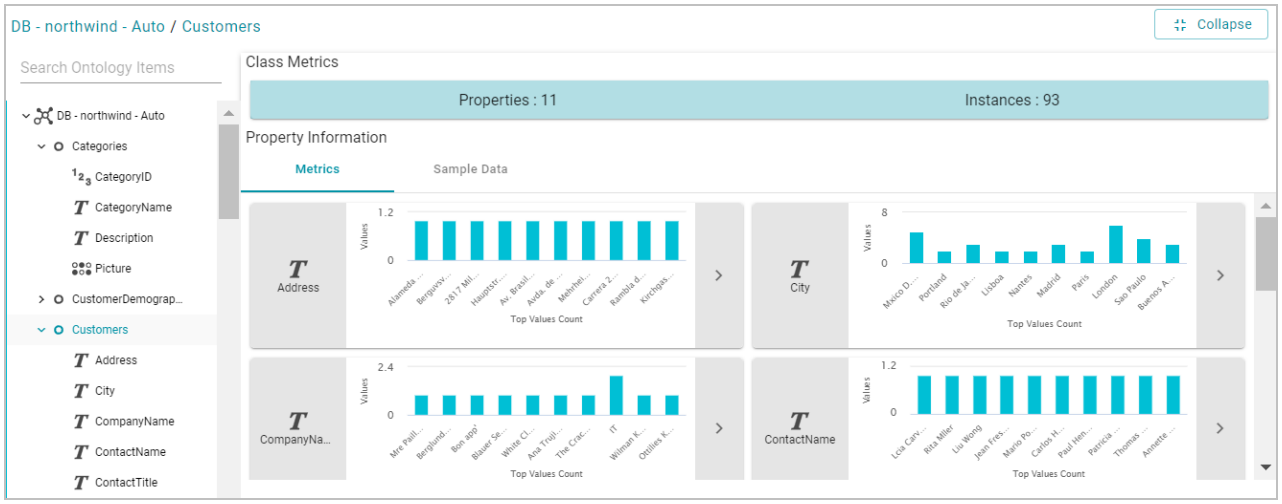


Note The instance counts for the classes in the graph view on the left side of the screen are the initial, non-unique counts from the ETL engine. Most likely the data has not yet been deduplicated. After generating metrics, the instance counts may change.

3. Click the **Generate Metrics** button at the top of the screen. Anzo provisions a temporary graphmart and loads the data into AnzoGraph. AnzoGraph computes the data profiling metrics. The process may take several minutes. Once the metrics are generated, Anzo removes the graphmart from AnzoGraph and the new information becomes available to explore in Anzo.
4. To display the metrics, select a node or edge in the graph view on the left side of the screen or expand the model in the middle of the screen and select a class or property. For example:



You can click the **Expand** button on the right side of the screen to collapse the graph view and expand the metrics view. For example:



Select any class or property to view its metrics. When a class is selected, Anzo displays the number of properties and total number of instances as well as one or more of the following metrics for each property in the class. The metrics that are calculated depend on the data type of the property:

- **Percentile Metric**: This metric presents the data distribution for a property in percentiles.
- **Top Value Counts Metric**: This metric displays the count (as a histogram) of the 10 most frequently occurring values for a property.
- **Presence Metric**: This metric displays the number of values present and not present for a property.

When a property is selected, Anzo displays the metrics described above and one or more of the following metrics, depending on the data type of the property:

- **Extrema Metric**: Shows the smallest and largest values.
- **Geometric Mean Metric**: Shows the geometric mean of the values.
- **Median Metric**: Shows the middle value.
- **Mode Metric**: Shows the value that appears most often.
- **Std Deviation Metric**: Shows the standard deviation in the set of values.
- **Unique Values Metric**: Shows the number of unique values.

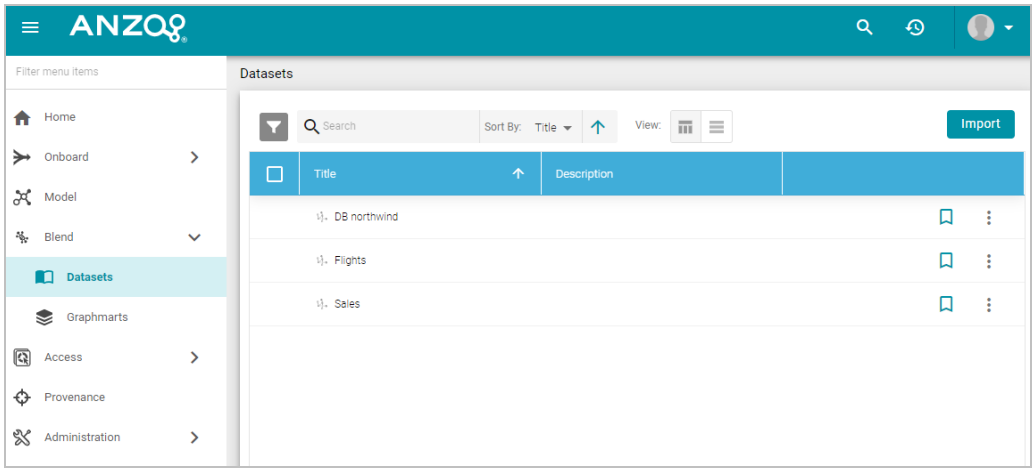
Related Topics

Creating a Graphmart

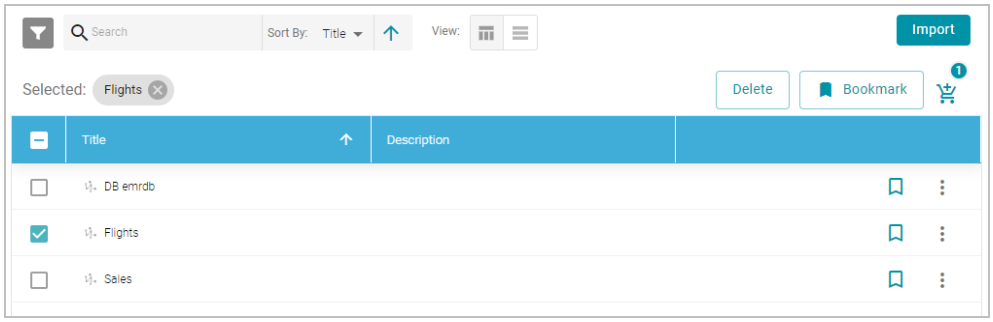
Creating a Graphmart

This topic provides instructions for creating graphmarts and loading them to AnzoGraph.

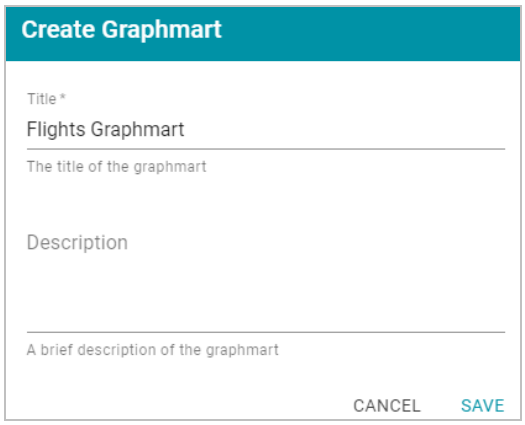
1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Datasets screen, which lists the catalog of data sets. For example:



2. In the dataset catalog, click the checkbox next to each data set that you want to add to the graphmart. Hover the pointer over an item to display the checkbox in the left column. Anzo adds the data sets to the shopping cart and additional icons become available at the top of the screen. For example:

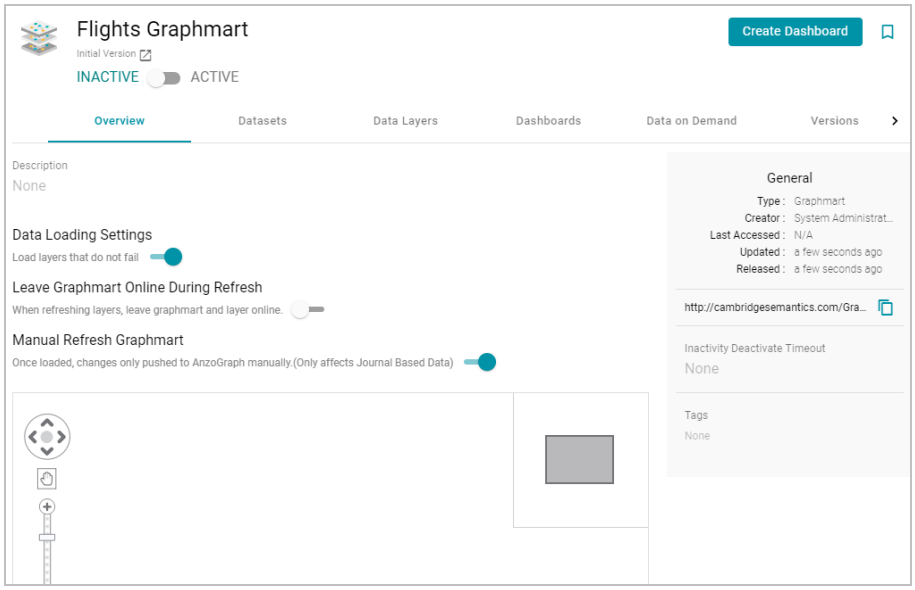


3. Click the shopping cart icon ( ) at the top of the screen. Anzo displays the Create Graphmart screen. For example:



Anzo populates the Title field by appending "Graphmart" to the data set name.

4. On the Create Graphmart screen, you have the option to edit the title and type an optional description. Click **Save** when you are ready to create the graphmart. Anzo creates the graphmart and displays the Graphmart Overview screen for the new graphmart. For example:



Note that the graph view of the data at the bottom of the screen will remain blank until the graphmart is activated.

5. If necessary, modify any of the following data load and graphmart settings:
    - **Load layers that do not fail**: This setting is enabled by default and controls whether failed data layers abort the graphmart load entirely or whether to abort the load if any data layer in the graphmart has an error or skip the layer that fails and load the layers that succeed. When enabled (the default setting), the graphmart is configured to load all layers that succeed and skip any layers that fail.

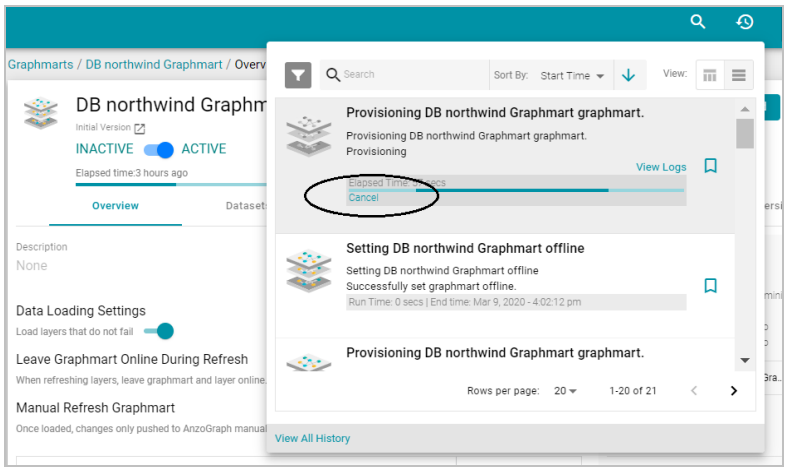
- **Leave Graphmart Online During Refresh**: This setting is disabled by default and controls whether to leave a graphmart and its data layers online while the graphmart is being refreshed in AnzoGraph. When **Leave Graphmart Online During Refresh** is enabled, Anzo copies the graphmart and data layers into temporary graphs so that the data remains online while the original graphmart is refreshed. When the refresh is complete, the temporary graphs are deleted.
- **Manual Refresh Graphmart**: This setting is enabled by default and is only relevant for journal-based data sets, such as system metadata or a data model. If the graphmart has a Load Data Step that loads system metadata or another journal-based linked data set, this setting controls whether changes to that data set are automatically deployed to AnzoGraph without requiring a manual refresh or reload of the graphmart. When **Manual Refresh Graphmart** is enabled, changes to the journal-based data set are only deployed to AnzoGraph when the graphmart is reloaded or refreshed. When **Manual Refresh Graphmart** is disabled, changes to the journal-based data set are automatically loaded to AnzoGraph without requiring a manual refresh.

6. You can add any number of data layers to enhance the data in the graphmart. For information, see Adding Data Layers to Graphmarts.
7. When you are ready to load the graphmart to AnzoGraph, slide the slider at the top of the screen from Inactive to **Active**.

   Note If you have more than one static AnzoGraph engine configured or you have a Cloud Location configured for dynamic AnzoGraph deployments, Anzo displays a Select an AnzoGraph Query Engine dialog box. Click the drop-down list to select the engine to load the graphmart to, or select **Spin up new AnzoGraph** (if available) to deploy a new instance. Then click **OK**.

   AnzoGraph loads the graphmart into memory and executes the data layer steps.

Tip If you want to cancel graphmart activation while data is loading, open the Activity Log by clicking the Activity Log icon ( ) in the main menu bar, and then click **Cancel** for the **Provisioning...graphmart** activity. For example:

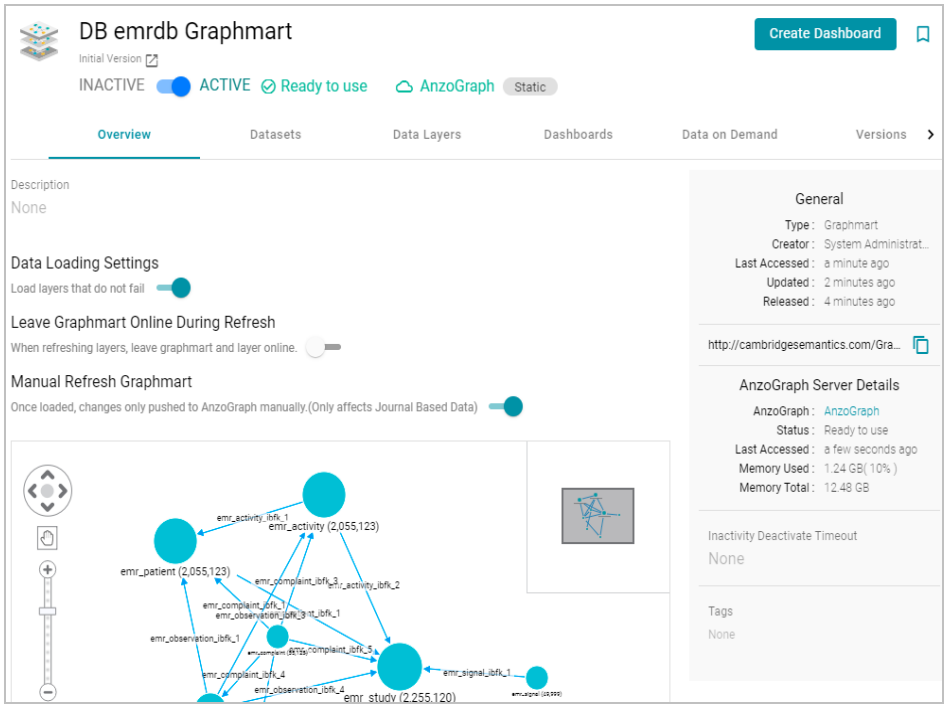


Once the graphmart is loaded into AnzoGraph, the data is available to access and analyze. For more information, see Accessing and Analyzing Data.

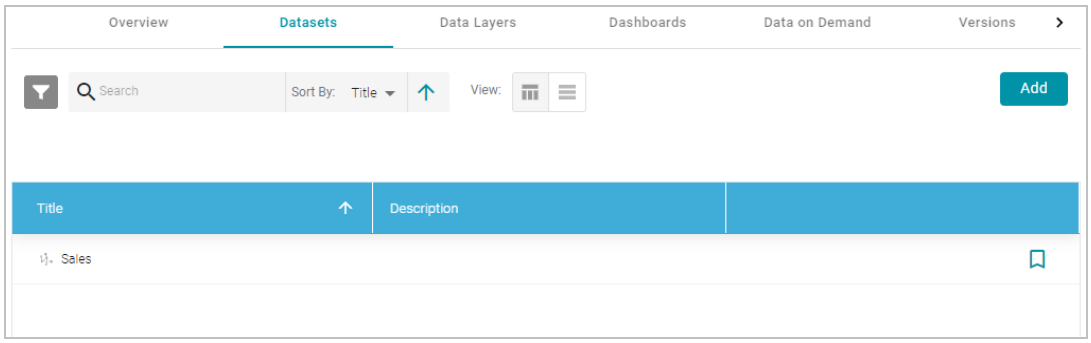
Adding a Data Set to a Graphmart

This topic provides instructions for quickly adding a new data set to an existing graphmart from the Datasets tab for the graphmart. You can also add a data set by creating a data layer step that loads the data. For more information, see Loading a Data Set.

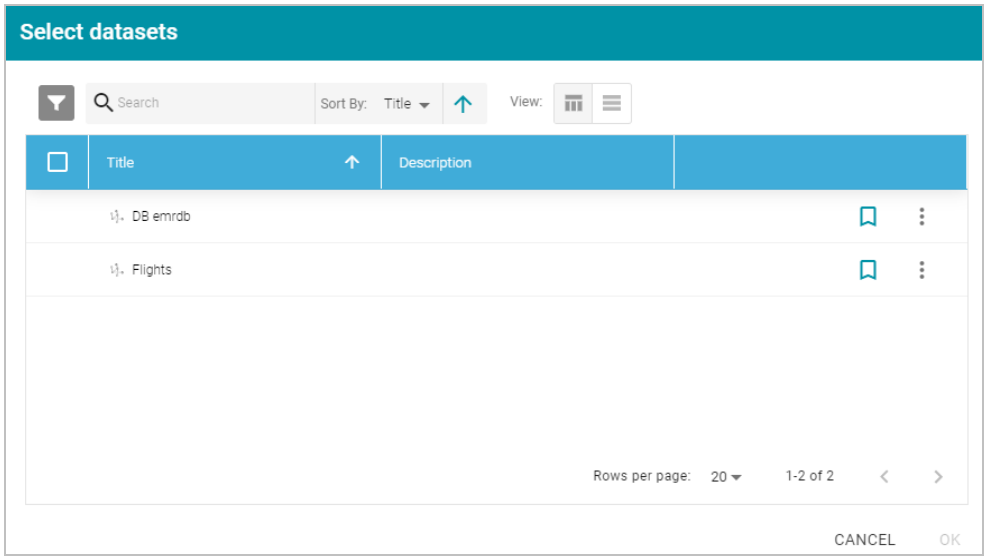
1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**.
2. On the Graphmarts screen, click the name of the graphmart that you want to add data to. Anzo displays the details for the graphmart. For example:



3. Click the **Datasets** tab. The screen lists the data sets in the graphmart. For example:

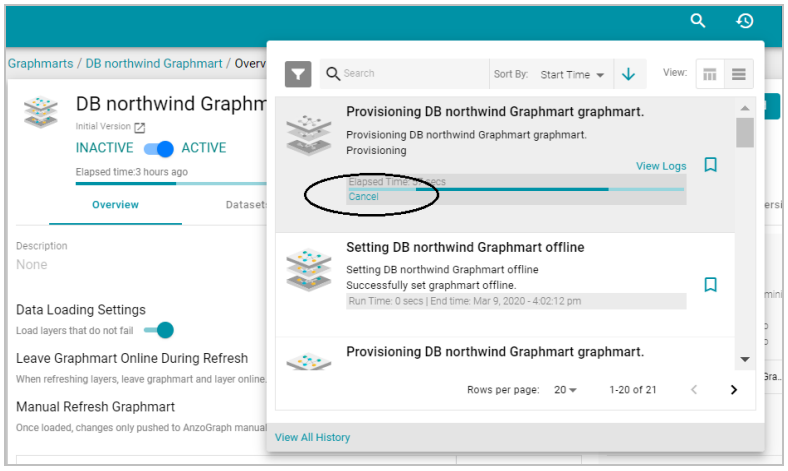


4. Click the **Add** button. Anzo opens the Select datasets dialog box.



5. In the dialog box, select the data set that you want to add to the graphmart, and then click **OK**. Anzo adds the data set to the graphmart and creates a new data layer with a Load Data Step that loads the data set.
6. To reload the graphmart and add the new data set to AnzoGraph, click the **Data Layers** tab, and then click the **Reload** button ( ).

Tip If you want to cancel graphmart activation while data is loading, open the Activity Log by clicking the Activity Log icon ( ) in the main menu bar, and then click **Cancel** for the **Provisioning...graphmart** activity. For example:



The new data set becomes available to explore in Hi-Res Analytics dashboards. For more information, see Analyzing Data with Hi-Res Analytics.

Related Topics

Creating a Graphmart

Accessing Data from Data on Demand Endpoints

Adding Data Layers to Graphmarts

Introduction to Data Layers

The Anzo Data Layers feature enables you to enhance graphmarts dynamically by creating layers that can load additional data sets, mask certain data, infer new data automatically, or run SPARQL queries to create, clean, conform, transform, or validate data. You can enable or disable layers any time and Hi-Res Analytics users can dynamically turn the layers on and off in dashboards.

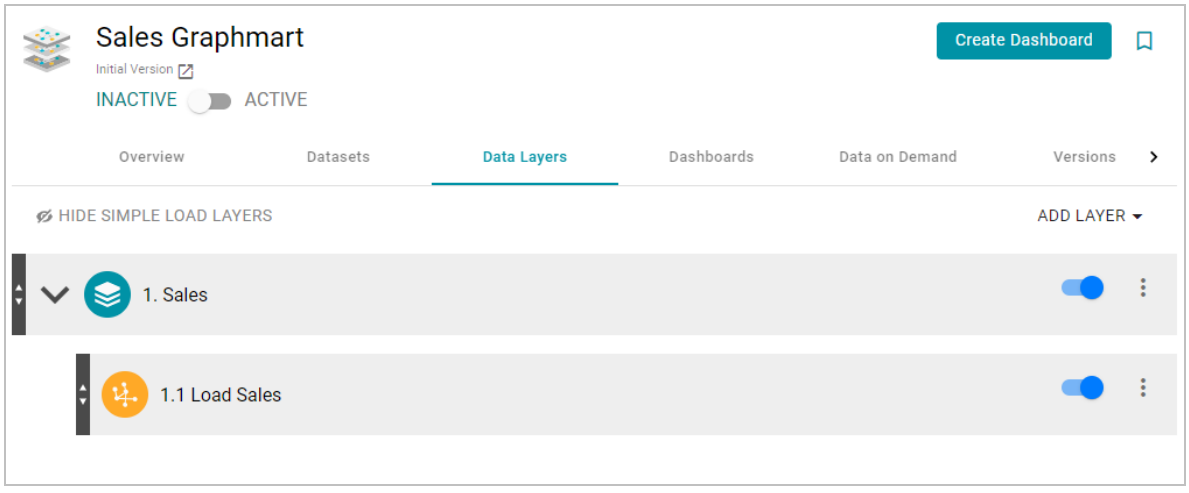
This topic introduces the fundamental concepts and vocabulary to know when working with data layers.

Layers

A layer is a container for one or more steps. The steps in a layer perform any data set loads or data creation and transformation.

- You can create any number of layers in a graphmart and control which user roles have access to the layers.
- Users can toggle data layers on and off in Hi-Res Analytics.
- You can configure masking on a layer to hide sensitive information.
- You cannot share data layers between graphmarts, but you can clone layers and include a copy in multiple graphmarts.
- You control the source data for steps in a layer. Layers can build upon the data generated by steps in previous layers or can be self-contained, applying changes only to the data defined in the layer.

When you create a graphmart (as described in Creating a Graphmart), Anzo automatically creates a data layer with a Load Data Step. For example, in the Sales Graphmart shown below, Anzo created a default **Sales** data layer that contains a **Load Sales** step that loads the data set for this graphmart to AnzoGraph.



For instructions on creating data layers, see Adding Data Layers to Graphmarts.

Steps

The steps in a layer perform the operations that you define, such as loading a data set or transforming the data. You can add any number of steps to a layer and can create the following types of steps:

- **Export Step**: Exports the graphmart data in memory to a file-based linked data set (FLDS).
- **Load Data Step**: Loads a data set from the Anzo Dataset catalog into a data layer in the graphmart.
- **Pre-compile Query Step**: Runs the included query immediately after a graphmart is loaded so that the query is pre-compiled by AnzoGraph. Pre-compiling a query reduces execution time when an end-user runs that query for the first time.
- **Query Driven Templated Step**: Enables users to create reusable query-driven templates for quickly creating additional query steps. Unlike the Templated Step, where users define each key-value pair, this step runs a query to identify all of the key-value pairs. Then the template query is run for each key-value solution from the first query.
- **Query Step**: Provides a SPARQL query template that you can use for writing a query that creates, cleans, conforms, or transforms data in the data layer.
- **RDFS+ Inference Step**: Uses RDFS and OWL rules to generate new data in a layer based on the vocabularies in the existing data.
- **Templated Step**: Enables users to create reusable templates for quickly creating additional query steps. The query in a Templated Step uses parameters to represent key-value pairs. When reusing the step, users modify the values for the keys rather than rewriting the query.
- **Validation Step**: Enables users to write a query that validates the data in a data layer.
- **View**: Enables advanced users to write a SPARQL CONSTRUCT query that defines a view of the data but does not alter the source data or create new data unless you choose to materialize the data.

For information about creating steps, see Adding Steps to Data Layers.

Masking

When configuring data layers and steps, Anzo provides an option to quickly mask or hide sensitive information. For example if you load data that includes social security numbers but do not want to make those values available in Hi-Res Analytics, you can simply mask the social security number property in the data layer. For more information, see Masking Data in Data Layers.

Related Topics

Adding Data Layers to Graphmarts

Adding Steps to Data Layers

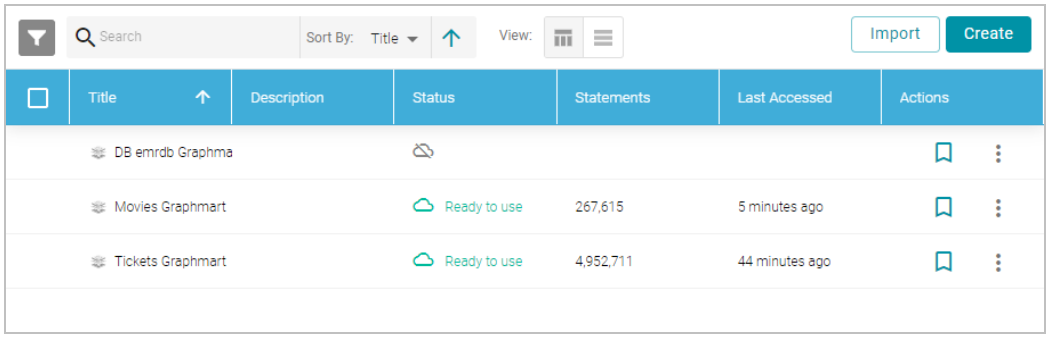
Masking Data in Data Layers

Data Layer Hi-Res Analytics Settings Reference

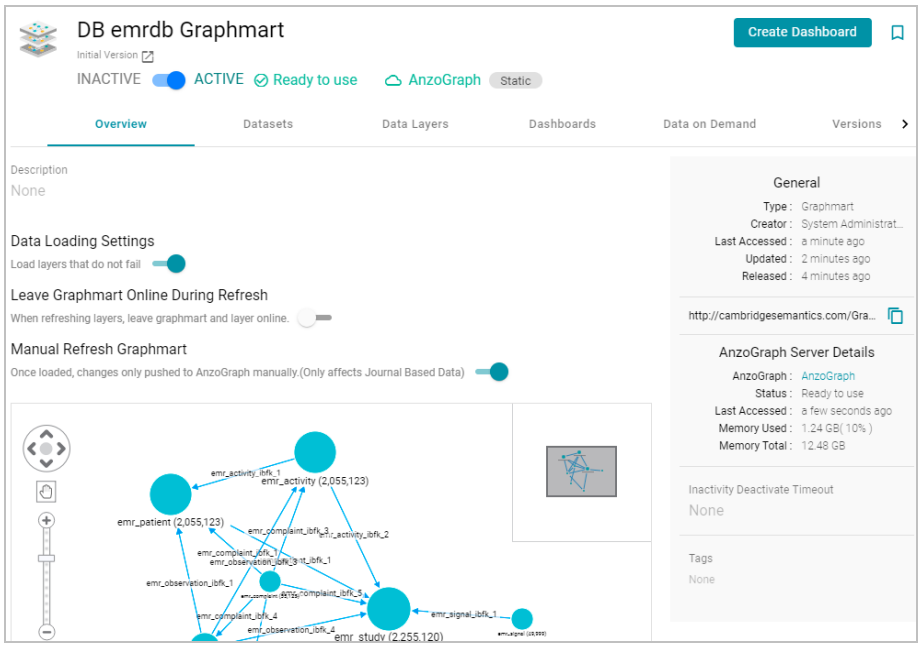
Adding Data Layers to Graphmarts

This topic provides instructions for adding data layers to graphmarts by creating a new layer or copying an existing layer. For conceptual information about data layers, see Introduction to Data Layers.

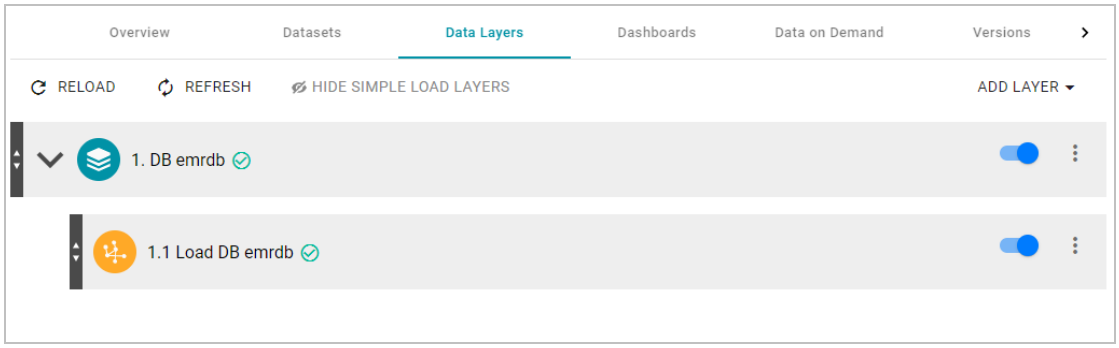
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



2. On the Graphmarts screen, click the name of the graphmart for which you want to add a data layer. Anzo displays the graphmart details. For example:



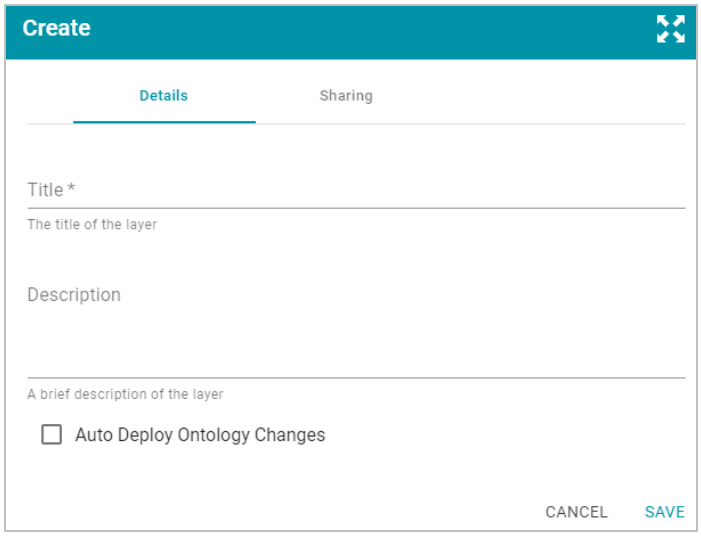
3. Click the **Data Layers** tab. Anzo displays the existing data layers. For example:



4. Follow the appropriate steps below, depending on whether you want to create a new layer from scratch or copy an existing layer to reuse. Click **Create a New Layer** or **Copy an Existing Layer** to expand the text and view the steps for that option:

Create a New Layer

a. To create a new layer, click **Add Layer** and select **New Layer**. Anzo displays the Create data layers details screen.



b. Specify a name for the layer in the **Title** field and an optional description in the **Description** field.

c. Specify how to control changes to the layer's dependent data models:

- If you want Anzo to automatically deploy to AnzoGraph any changes to the related models without having to manually refresh the layer or graphmart, select the **Auto Deploy Ontology Changes** checkbox.

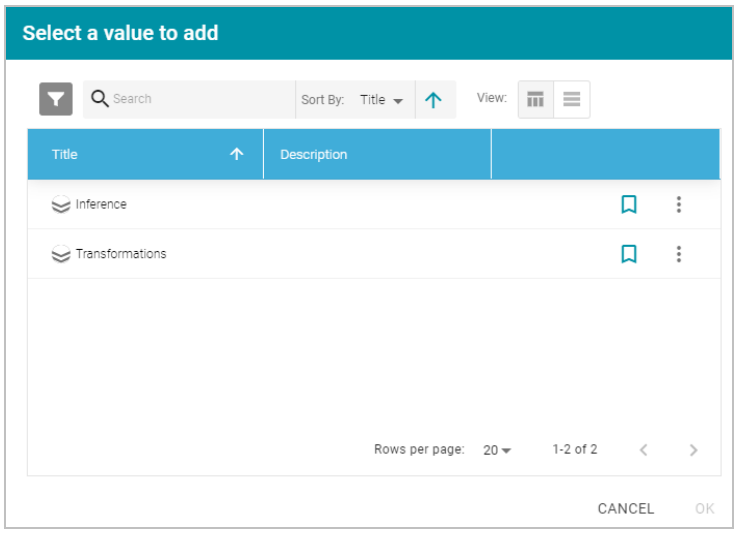
  Important: The **Manual Refresh Graphmart** setting on the graphmart must be **disabled** for automatic deployment of models to work. See Creating a Graphmart for information about graphmart settings.

- If you want data model changes to be deployed to AnzoGraph only when the layer or graphmart is manually refreshed or reloaded, leave the **Auto Deploy Ontology Changes** checkbox empty (disabled).

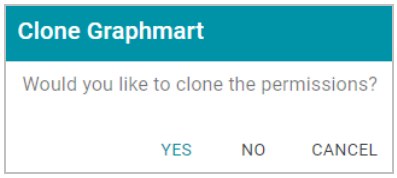
d. Click **Save** to add the new layer to the graphmart and return to the Data Layers screen.

Copy an Existing Layer

a. If you want to clone an existing layer, click **Add Layer** and select **Add Existing**. Anzo opens the Select a value to add dialog box, which lists the existing layers for all graphmarts. For example:

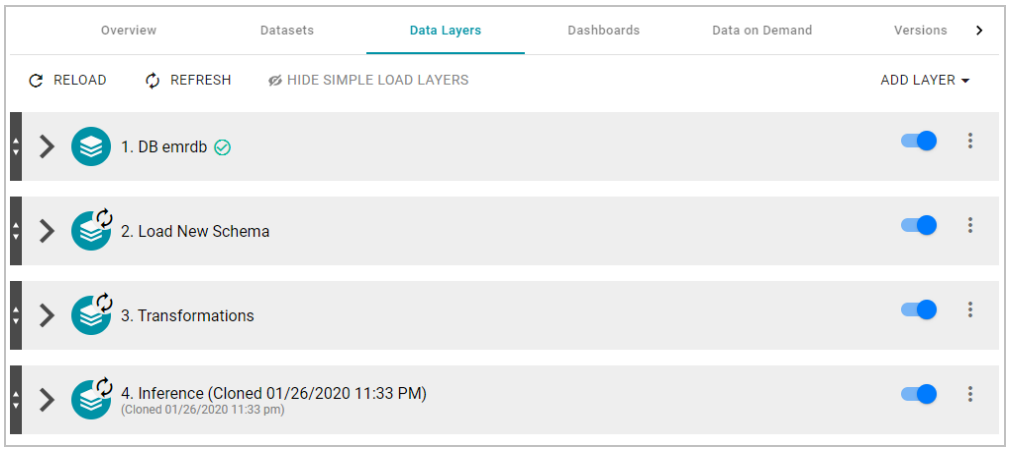


b. Select the layer that you want to copy and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the access control list (ACL) from the existing layer.



c. On the Clone dialog box, click **Yes** to copy the ACL configuration from the existing layer or click **No** to copy the layer without the ACL configuration. Anzo clones the layer and any steps that the layer contains, adds the copy to the graphmart, and returns to the Data Layers screen.

5. Anzo adds the new layer as the last layer in the graphmart. If you added a new layer by making a copy of an existing layer, Anzo adds "Cloned" to the data layer title and description with a timestamp that indicates when the layer was copied. For example:



If you want to edit the title or description, click the menu icon on the right of the layer and select **Edit**. Modify the title or description values and click **Save**. If you want to change the order of the layers in the graphmart, you can click the black bar on the left side of a layer and drag the layer up or down. Data layers in a graphmart are processed from top to bottom.

6. Next, add steps to the layer that will perform the data processing operations, such as loading, creating, deleting, or changing the data. See Adding Steps to Data Layers for instructions.

   Note The Refresh icon ( ) on the new layer indicates that the layer is out of sync with the graphmart data that is in AnzoGraph. Once you add data processing steps to the layer, you can click the **Reload** button ( ) at the top of the screen to reload the entire graphmart, or you can click the **Refresh** button ( ) to process only the layer or layers that are out of sync.

Related Topics

Introduction to Data Layers

Adding Steps to Data Layers

Masking Data in Data Layers

Managing Graphmart, Data Layer, and Step Security

Data Layer Hi-Res Analytics Settings Reference

Adding Steps to Data Layers

The steps in a data layer perform the data operations, such as loading, creating, deleting, or changing data. You can add any number of steps to a layer. The topics in this section provide instructions for adding steps to data layers. The list below describes each type of step:

- **Export Step**: Exports the data in memory to a file-based linked data set (FLDS). See Exporting Data to an FLDS for instructions on creating this type of step.

- **Load Data Step**: Loads a data set from the Anzo Catalog into a data layer in the graphmart. See Loading a Data Set for instructions on creating this type of step.
- **Pre-compile Query Step**: Runs the included query immediately after a graphmart is loaded so that the query is pre-compiled by AnzoGraph. See Pre-Compiling a Query for instructions on creating this type of step.
- **Query Driven Templated Step**: Creates a reusable query template by identifying all of the key-value pairs in the source data and then running the specified query for each key-value pair. See Creating a Query-Driven Template for instructions on creating this type of step.
- **Query Step**: Provides a SPARQL query template that you can use for writing a query that creates, cleans, conforms, or transforms data in the data layer. See Running a Query for instructions on creating this type of step.
- **RDFS+ Inference Step**: Uses RDFS and OWL rules to generate new data in a layer based on the patterns it finds in that layer's data. See Adding an RDFS+ Inference Step for instructions on creating this type of step.
- **Templated Step**: Creates a reusable query template with user-defined parameters and key-value pairs. See Creating a Query Template for instructions on creating this type of step.
- **Validation Step**: Enables users to write a query that validates the data to ensure that it conforms to expectations. See Validating the Data for instructions on creating this type of step.
- **View**: Enables advanced users to write a SPARQL query that defines a view of the data but does not alter the source data or create new data unless users choose to materialize the data. See Creating a View of the Data for instructions on creating this type of step.

Related Topics

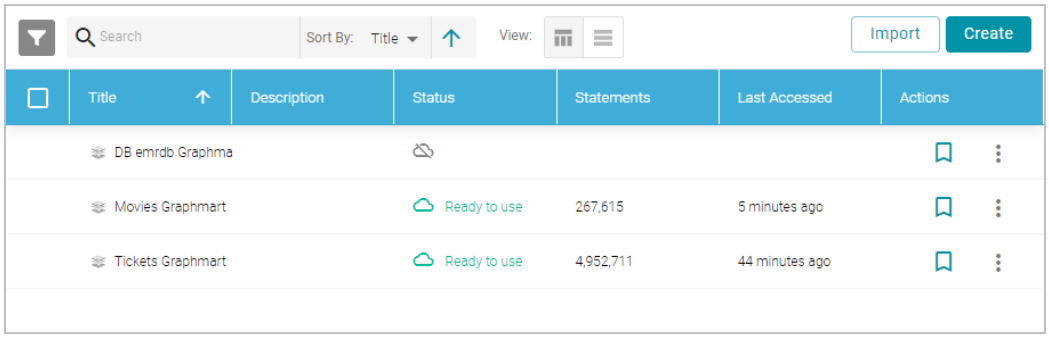
Introduction to Data Layers

Adding Data Layers to Graphmarts

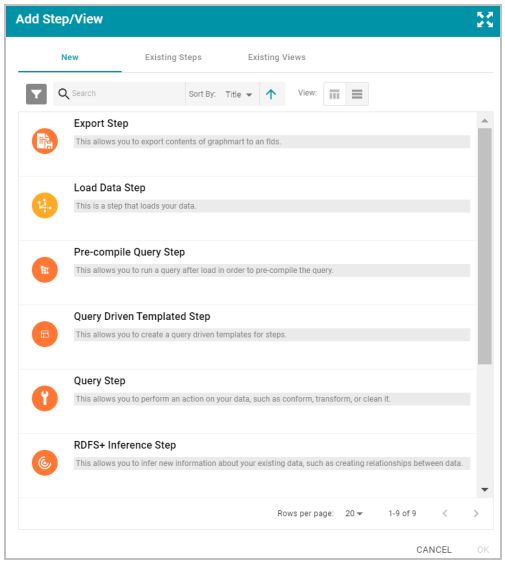
Exporting Data to an FLDS

Follow the instructions below to add a step to a data layer that exports the data in memory to a file-based linked data set (FLDS) on the file store.

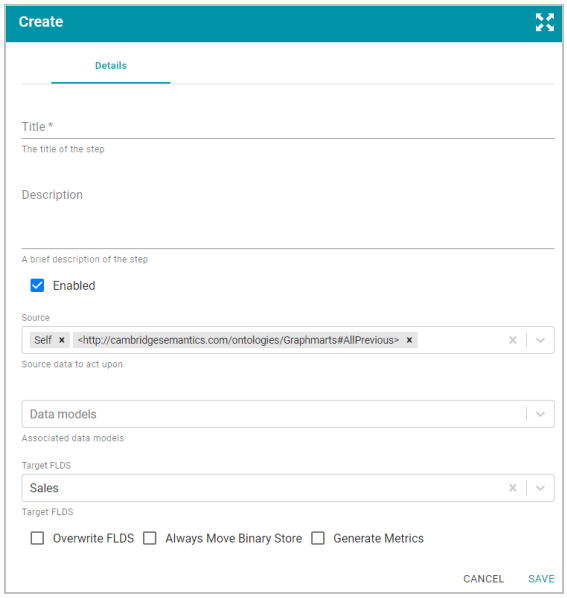
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.

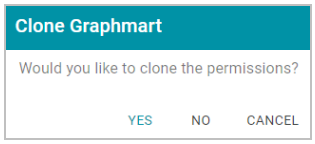


5. Follow one of the options below, depending on whether you want to create a step from scratch or clone an existing export step for reuse:
   - To create a new step, select **Export Step** and then click **OK**. Anzo opens the Create Export step screen. Proceed to the next step.



   - If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:

a. Select the export step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit export step screen. Proceed to the next step.

6. Under Details, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. If necessary, click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:
   - **Self**: This option is selected by default and means that Anzo exports only the data that is generated in the layer to which this step belongs.
   - **All Previous Layers Within Graphmart**: Choosing this option means that Anzo exports data that is generated by all of the layers in the graphmart that precede this layer.
   - **Previous Layer Within Graphmart**: Choosing this option means that Anzo exports only the data that is generated by the one layer that precedes this layer.
   - ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to export only the data that is generated by that layer.

     You can remove any of the source options by clicking the X to the left of the option name.
9. Click the **Data Models** drop-down list and select the model or models to export with the data.
10. If the graphmart contains one FLDS, the Target FLDS value defaults to that FLDS. If the graphmart contains multiple FLDSes, click the **Target FLDS** field and select the target FLDS for this export step.
11. At the bottom of the screen, enable or disable any of the following options as appropriate:
    - **Overwrite FLDS**: Controls whether the existing FLDS is replaced with the exported files or whether the exported files are added to the existing FLDS.
      - If you want Anzo to replace the current FLDS, select the **Overwrite FLDS** checkbox. When Overwrite FLDS is enabled, Anzo archives the existing files in a new timestamped directory under the

`archives` directory at the same level as the FLDS. The FLDS will contain only the exported data.

- If you want Anzo to add the exported files to the existing FLDS, leave the **Overwrite FLDS** checkbox unchecked. When Overwrite FLDS is disabled, Anzo adds the exported files to a new timestamped directory under the `rdf.ttl` directory in the FLDS. The FLDS will contain the original files as well as the new exported files.

- **Always Move Binary Store**: This option usually applies only to exports of unstructured data and controls whether the binary store is moved or copied during the export. Since the binary store can be large and have a nested structure, copying the data can take a very long time. Since moving the binary store is almost instantaneous, however, enabling **Always Move Binary Store** can reduce the time it takes to complete the export.
  - If you want Anzo to copy the binary store to the location specified by the Overwrite FLDS setting, leave **Always Move Binary Store** disabled (unchecked).
  - If you want Anzo to move the binary store to the location specified by the Overwrite FLDS setting, select the **Always Move Binary Store** checkbox to enable it.
- **Generate Metrics**: Controls whether graph data metrics are calculated before the data is exported. Since the data must be loaded in AnzoGraph to compute the metrics, you have the option to generate them during the export. If you load the exported files in the future, the graph data metrics will become available in the Dataset catalog. For more information, see Generating Graph Data Metrics.

12. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Related Topics

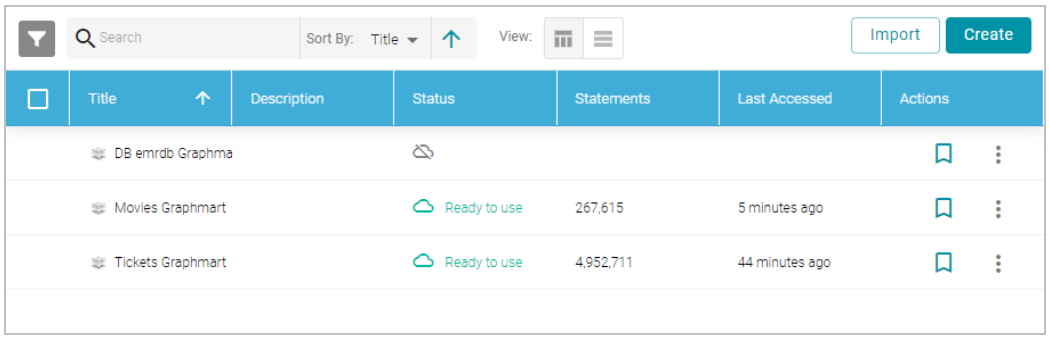
Adding Steps to Data Layers

Loading a Data Set

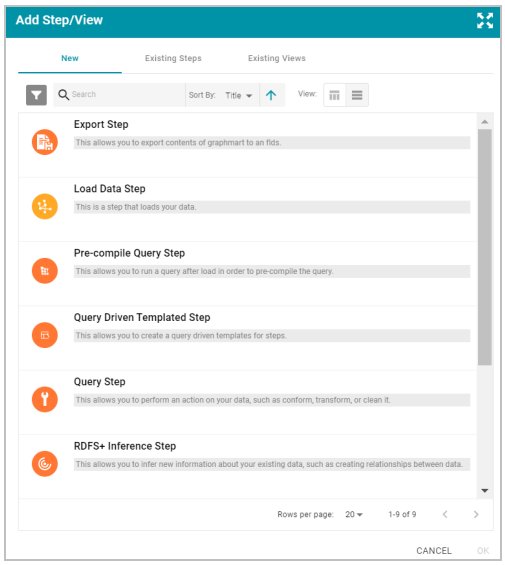
Follow the instructions below to add a step that loads a new data set into a data layer in a graphmart. The data set to load must be available in the Dataset catalog.

Note You cannot clone and reuse existing Load Data Steps. The steps below guide you through creating new Load Data Steps.

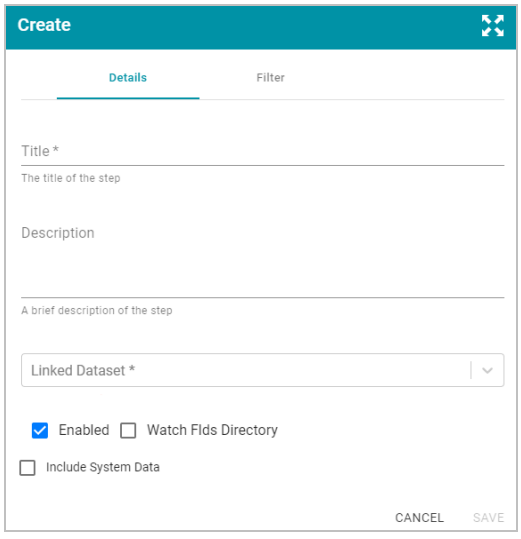
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Select **Load Data Step** on the Add screen, and then click **OK**. Anzo opens the Create load data step screen.



6. On the Create screen, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. Click the **Linked Dataset** drop-down list and select the data set that you want this step to load. The list displays all of the data sets in the Dataset catalog. If you want to choose a system data set, select the **Include System Data** checkbox. The Linked Dataset drop-down list will display the system data sets in addition to the data sets in the catalog.
8. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
9. If you want this step to watch the FLDS directory and indicate when any of the load files change, select the **Watch FLDS Directory** checkbox. When Watch FLDS Directory is enabled, Anzo will indicate that this step (and data layer) need to be refreshed if any of the files in the FLDS directory are changed.
10. **Optional**: Filter the load data. If you want to load all of the statements in the linked data set, proceed to step 9. If you do not want to load all of the data in the data set, follow the instructions in this step to filter the load data.

    Anzo provides two options for filtering data. To view the instructions for an option, click the **How do I...** text for that option:

    1. Exclude certain triples from the load by selecting predicates to filter out (masked predicates).

       How do I mask predicates?

       a. Click the **Filter** tab at the top of the load data step screen. Anzo displays the filter options:

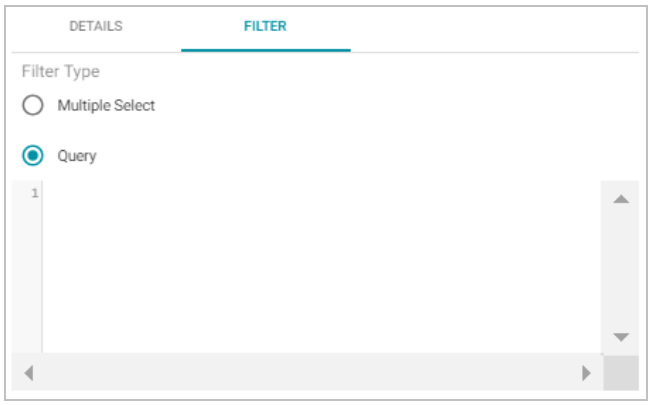
       b. Select the **Multiple Select** radio button.
       c. Click the **Masked Predicate** drop-down list and select a predicate to add it to the Masked Predicate field. Repeat this step to mask additional predicates. You can remove a property from the masked list by clicking the X to the right of the predicate name.

    2. If the data set is a graph source (file-based linked data set), you can hand-pick the data to load by writing a query that inserts specific values or filters out certain values.

       How do I include a load filter query?

a. Click the **Filter** tab at the top of the load data step screen. Anzo displays the filter options:

b. Select the **Query** radio button. Anzo displays a text box under the Query field.



c. Type a SPARQL INSERT query in the Query box. For example, you can use the following format to filter out properties from the files.

Note: Including the **${targetGraph}** and **${usingSources}** parameters are required.

```
INSERT {
  GRAPH ${targetGraph}{
    ?s ?p ?o.
  }
}
${usingSources}
WHERE {
  ?s ?p ?o .
  FILTER EXISTS { ?s a ?type . }
  FILTER(?type = <URI>)
}
```

Important In load filter queries, URIs are not supported in the object position. To specify a URI as an object, include the standard ?s ?p ?o triple pattern in the WHERE clause and then apply FILTER statements with URIs as needed. URIs are supported in the subject or predicate position.

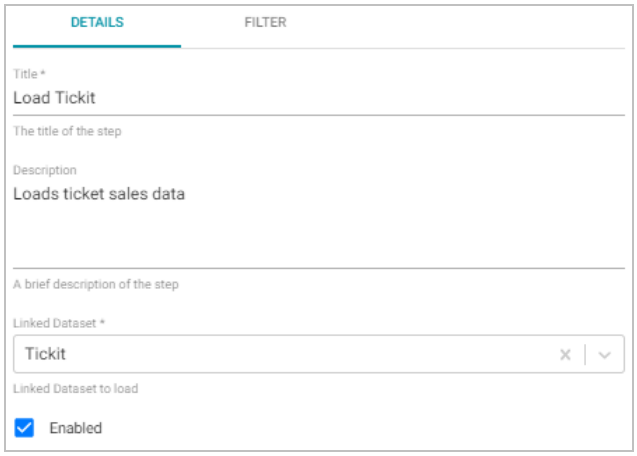
For example, the following query filters the data in a sample data set that includes information about people and the events they buy tickets for. The WHERE clause filters the data to load only the triples that are related to person1 (personid=1):

```
INSERT { GRAPH ${targetGraph} {
  ?s ?p ?o
 }
}
${usingSources}
WHERE {
  ?s ?p ?o ;
  <http://cambridgesemantics.com/ont/autogen/Fu/Tickit_
Data#tickit_users_personid> ?id .
  FILTER (?id=1)
}
```

11. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Example Load Data Step

The example below creates a step that loads an additional "Tickit" data set to the Movie Data graphmart:



The example excludes triples that include user's credit card numbers from the load by masking the "card" predicate:

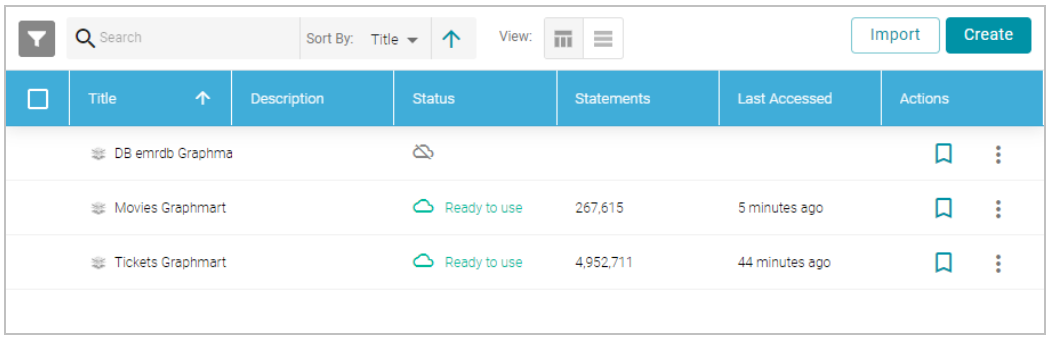
Related Topics

Adding Steps to Data Layers

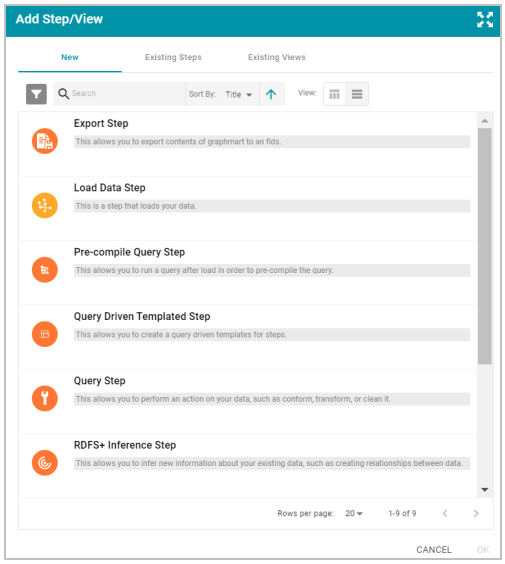
Pre-Compiling a Query

The first time a user runs an analytic query against AnzoGraph, AnzoGraph performs a code compilation process to generate the code for running that query. It then executes the query using that compiled code, and the same code is reused for subsequent runs of the query. If you determine that a particular query has a long code compilation time, you can add that query to a Pre-Compile Query Step. That way the query is run during the graphmart load and the compiled code is available before an end-user runs that query. Follow the instructions below to add a step that pre-compiles a query.

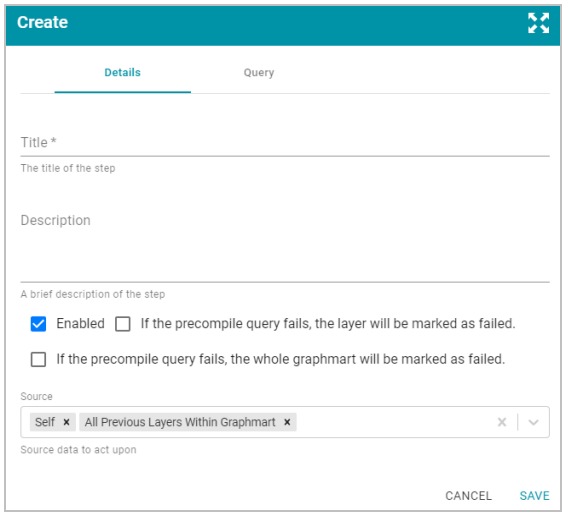
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



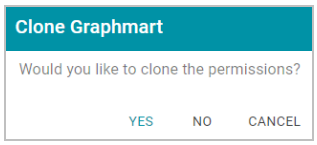
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing step for reuse:
    - If you want to create a new step, select **Pre-compile Query Step**, and then click **OK**. Anzo opens the Create step screen. Proceed to the next step.



    - If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:
        a. Select the pre-compile query step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



        b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

           Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

        c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit load data step screen. Proceed to the next step.

6. On the Details tab, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. Specify what action, if any, you want Anzo to take if this step fails. The step includes the following two settings that control how Anzo treats the layer or graphmart if the query fails:
    - **If the precompile query fails, the layer will be marked as failed**: Select this option if you want Anzo to abort the load of the data layer if this step fails. The graphmart and other successful data layers

continue to load.

- **If the validation query fails, the whole graphmart will be marked as failed**: Select this option if you want Anzo to abort the load of the entire graphmart if this step fails.

If you want Anzo to proceed to load the data layer if this step fails, leave both options blank.

9. Click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:
   - **Self**: This option is selected by default and means that the query runs against only the data that is generated in the layer to which this step belongs.
   - **All Previous Layers Within Graphmart**: Choosing this option means that the query runs against the data that is generated by all of the layers in the graphmart that precede this layer.
   - **Previous Layer Within Graphmart**: Choosing this option means that the query runs against only the data that is generated by the one layer that precedes this layer.
   - ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to run the query against only the data that is generated by that layer.

     You can remove any of the source options by clicking the X to the left of the option name.
10. Click the **Query** tab to add the query that this step will run. The tab includes the syntax for writing a SPARQL SELECT query.

```
SELECT *
${fromSources}
WHERE{
}
```

    The template includes the source graph parameter (`${fromSources}`). Using the configured Source data options from the Details tab, Anzo automatically populates the query with the appropriate source graph URIs when the query runs. Edit the template to add the query that you want to pre-compile.
11. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Related Topics

Adding Steps to Data Layers

SPARQL Query Templates and Best Practices

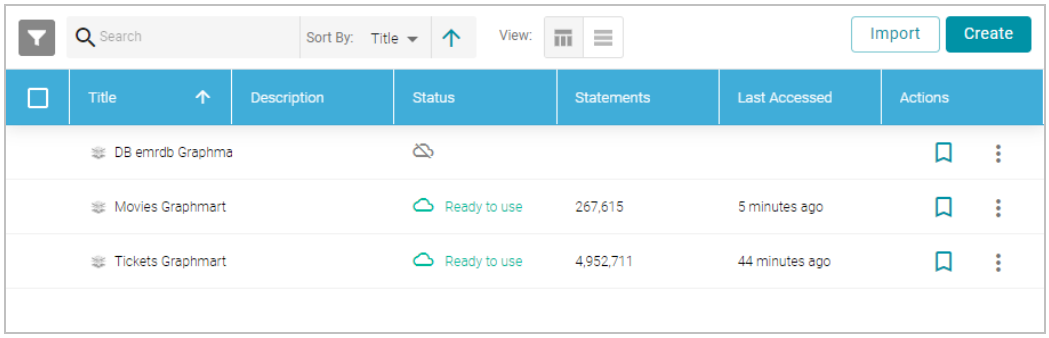
Creating a Query-Driven Template

Query-Driven Template steps are similar to Templated steps in that they provide a way to create query templates that use parameters to represent key-value pairs. The queries are reusable across data sets because, rather than

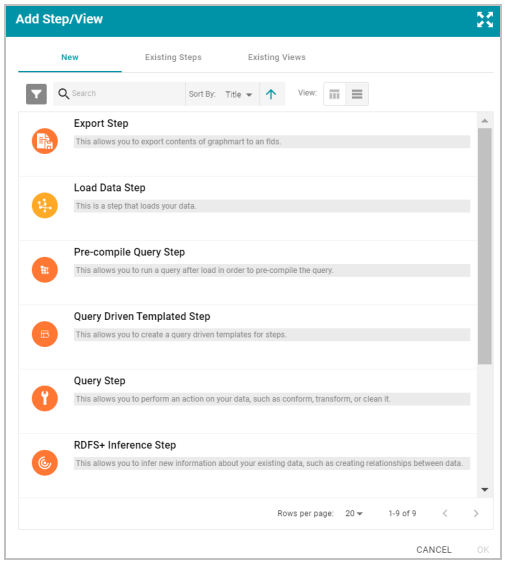
rewriting the query, the existing parameters can be substituted for alternate key-value pairs. The difference between the two types of steps is that the key-value pairs for Templated steps must be user-defined. In Query-Driven Template steps, a parameter query is run that automatically generates the key-value pairs. Then the defined template query is run for each key-value solution from the parameter query.

For more information about Templated steps with manually created key-value pairs, see Creating a Query Template.

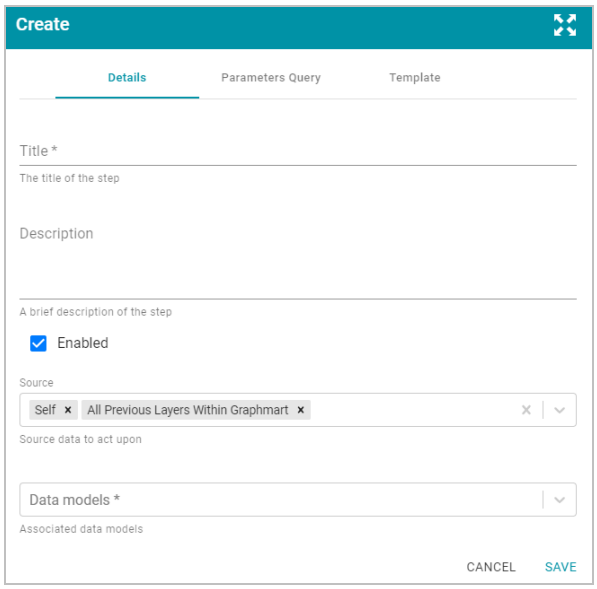
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



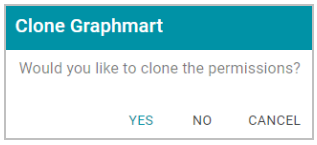
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing step for reuse:
   - If you want to create a new step, select **Query Driven Templated Step**, and then click **OK**. Anzo opens the Create step screen. Proceed to the next step.



- If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:
    a. Select the query-driven templated step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



    b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

    Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

    c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit load data step screen. Proceed to the next step.

6. On the Details tab, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. Click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:
    - **Self**: This option is selected by default and means that the query runs against only the data that is generated in the layer to which this step belongs.

- **All Previous Layers Within Graphmart**: Choosing this option means that the query runs against the data that is generated by all of the layers in the graphmart that precede this layer.
- **Previous Layer Within Graphmart**: Choosing this option means that the query runs against only the data that is generated by the one layer that precedes this layer.
- ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to run the query against only the data that is generated by that layer.

  You can remove any of the source options by clicking the X to the left of the option name.

9. Click the **Data models** drop-down list and select the model or models to create the template against.
10. Click the **Parameters Query** tab to view the parameter query that is used to determine the key-value pairs for the selected source. The tab includes the syntax for writing a SPARQL SELECT query.

```
SELECT DISTINCT ?param1 ?param2 ?param3
${fromSources}
WHERE{
?param1 ?param2 ?param3.
}
```

    Edit the query as needed. The template includes the source graph parameter (`${fromSources}`). Using the configured Source data options from the Details tab, Anzo automatically populates the query with the appropriate source graph URIs when the query runs.

11. Click the **Template** tab to use the provided template to write the query that will be run for each of the key-value pairs identified by the Parameter Query. The template includes the syntax for writing SPARQL DELETE and INSERT queries and includes source and target graph parameters that Anzo replaces at runtime. Edit the template text as needed.

```
DELETE{
  GRAPH ${targetGraph}{
 }
}
INSERT{
  GRAPH ${targetGraph}{
 }
}
${usingSources}
WHERE{
  ${param1} ${param2} ${param3} .
}
```

12. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Related Topics

Adding Steps to Data Layers

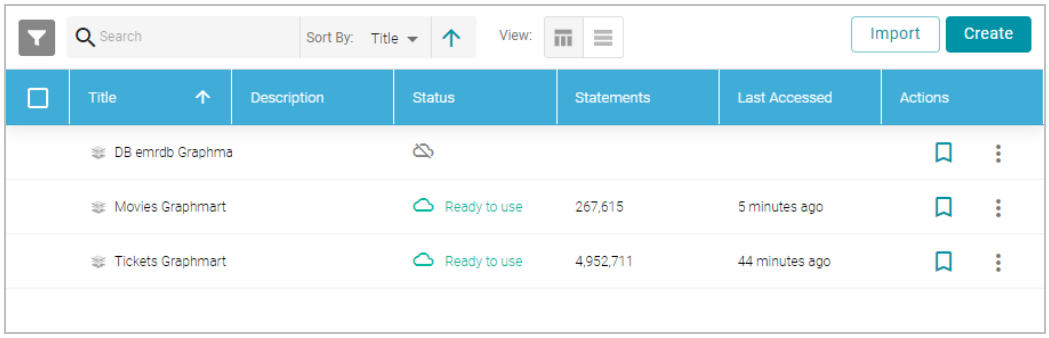
SPARQL Query Templates and Best Practices

Running a Query

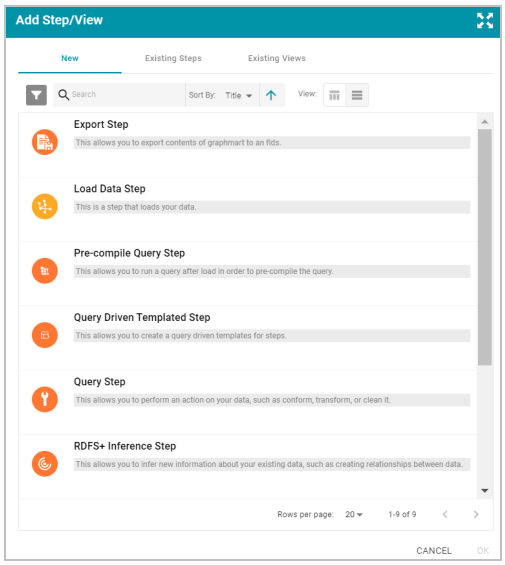
Follow the instructions below to add a step that runs a custom query to create, clean, conform, or transform data in a data layer.

Tip When writing queries for Query Steps, Cambridge Semantics recommends that you open an additional instance of the Anzo console so that you can use the Models manager to view the ontology for the data set that the query runs against. Viewing the model enables you to review the classes and properties and copy any URIs to use in the query.

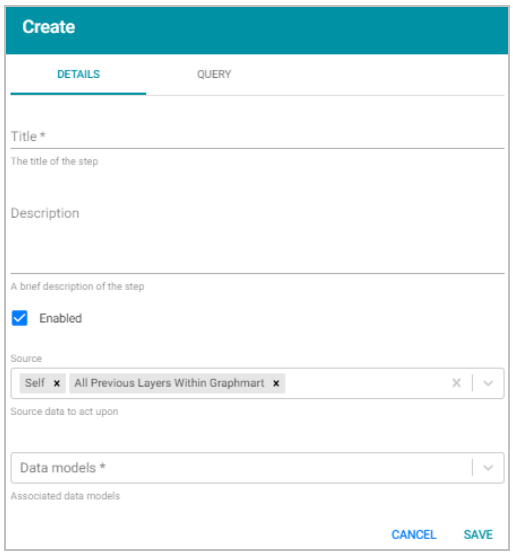
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



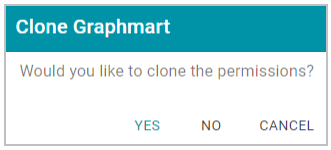
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing query step for reuse:
    - If you want to create a new step, select **Query Step**, and then click **OK**. Anzo opens the Create query step screen. Proceed to the next step.



    - If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:
        a. Select the query step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

   Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit load data step screen. Proceed to the next step.

6. On the Details tab, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. Click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:
   - **Self**: This option is selected by default and means that the query runs against only the data that is generated in the layer to which this step belongs.
   - **All Previous Layers Within Graphmart**: Choosing this option means that the query runs against the data that is generated by all of the layers in the graphmart that precede this layer.
   - **Previous Layer Within Graphmart**: Choosing this option means that the query runs against only the data that is generated by the one layer that precedes this layer.
   - ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to run the query against only the data that is generated by that layer.

     You can remove any of the source options by clicking the X to the left of the option name.
9. Click the **Data models** drop-down list and select the model or models to run this query against.
10. Click the **Query** tab to compose the query that this step will run. The tab includes the syntax for writing SPARQL INSERT and DELETE queries. For example:

```
DELETE{
  GRAPH ${targetGraph}{
 }
}
INSERT{
  GRAPH ${targetGraph}{
 }
}
${usingSources}
WHERE{
}
```

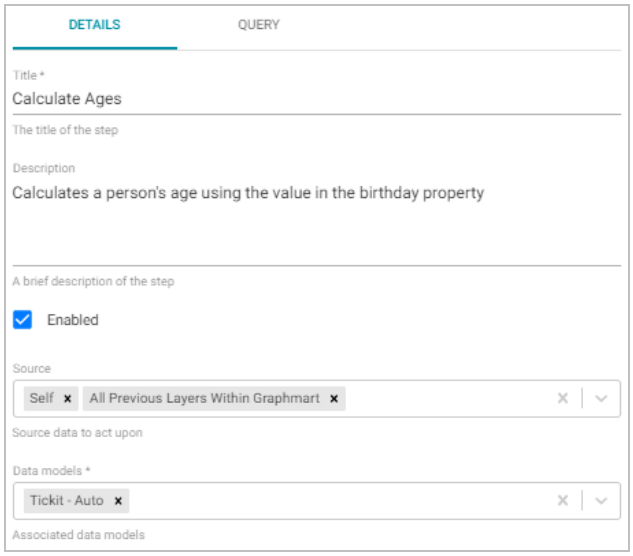
The template includes target and source graph parameters (`${targetGraph}` and `${usingSources}`). Using the configured Source data options from the Details tab, Anzo automatically populates the query with the appropriate target and source graph URIs when the query runs. Edit the template text as needed. See SPARQL Query Templates and Best Practices for more information. For information about the SPARQL syntax for INSERT and DELETE queries, see SPARQL 1.1 Update Language in the W3C SPARQL 1.1 Update specification.

For information about incorporating data from a remote endpoint, see Blending Data with the Data Toolkit Service.

11. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Example Query Step

The example below uses a data set that includes data about people, including a birthday property. The Query step uses the data that is generated by previous layers to calculate and insert the age of each person using their birthday values. The image below shows the details for the step:



The image below shows the query for this step. The query inserts triples for the age of each person, which is calculated by subtracting the year in the birthday date value from the current year.

Note The **p_Age** property shown below was manually added to the data model for the data set.

Running the example query step populates the values for the p_Age property.

Related Topics

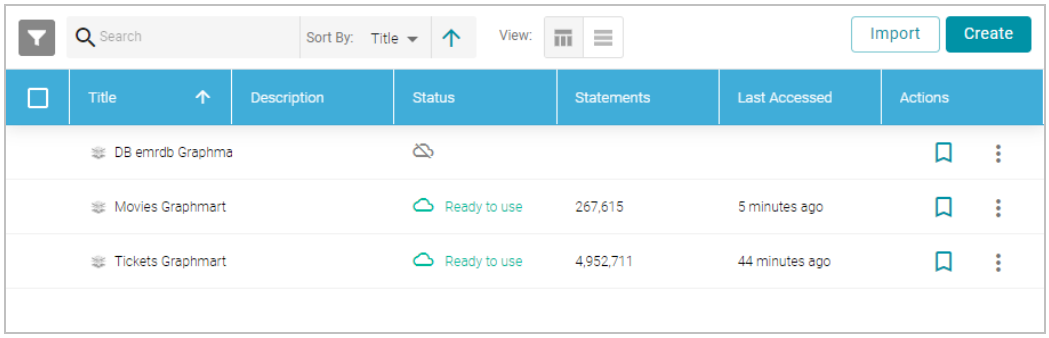
Adding Steps to Data Layers

SPARQL Query Templates and Best Practices

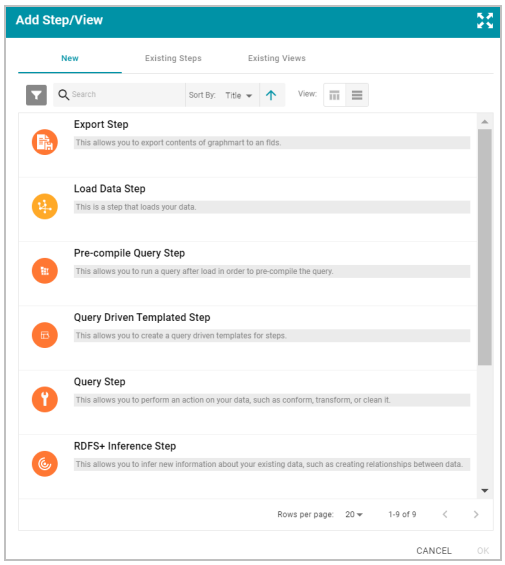
Adding an RDFS+ Inference Step

Follow the instructions below to add a step to a data layer that uses RDFS-plus and OWL rules to create new relationships based on the vocabularies in the existing data.

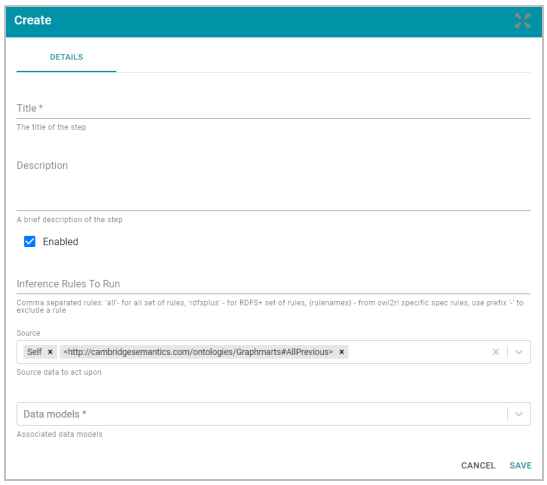
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



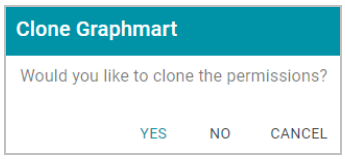
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing inference step for reuse:
    - Select **RDFS+ Inference Step**, and then click **OK**. Anzo opens the Create RDFS inference step screen. Proceed to the next step.



    - If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:
        a. Select the inference step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

   Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit inference step screen. Proceed to the next step.

6. Under Details, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. By default the step runs all of the RDFS-plus inference rules and a subset of the OWL 2 RL rules (see Inference Rule Reference below for specifics). If you want to customize the step to include or exclude certain rules, specify any combination of the following options in the **Inference Rules To Run** field. Specify multiple options in a comma-separated list:
   - **all**: Run all rules.
   - **rdfsplus**: Run only the RDFS-plus rules.
   - ***rule_names***: List specific rules to run only those rules. For a list of rule names, see Inference Rule Reference.
   - ***-rule_name***: Specify a hyphen (-) in front of a rule name to exclude that rule. For example, **-scm-svf2** excludes the scm-svf2 rule.

   For example, the following value runs all of the inference rules except prp-fp and prp-ifp:

   ```
   all,-prp-fp,-prp-ifp
   ```

   Note Certain inference rules are coupled. Specifying either of the rules in the pair automatically runs the coupled rule. The list below describes the paired rules:
   - scm-dom1 and scm-rng1
   - scm-dom2 and scm-rng2
   - prp-inv1 and prp-inv2

   In addition, running scm-eqc1 or cax-sco also runs cax-eqc1 and cax-eqc2. And running scm-eqp1 or prp-spo1 also runs prp-eqp1 and prp-eqp2.

9. Click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:
   - **Self**: This option is selected by default and means that inferences rules run against only the data that is generated in the layer to which this step belongs.

- **All Previous Layers Within Graphmart**: Choosing this option means that inference runs against the data that is generated by all of the layers in the graphmart that precede this layer.
- **Previous Layer Within Graphmart**: Choosing this option means that inference runs against only the data that is generated by the one layer that precedes this layer.
- ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to run inferences against only the data that is generated by that layer.

  You can remove any of the source options by clicking the X to the left of the option name.

10. Click the **Data Models** drop-down list and select the model or models to use for this layer to run inference against.
11. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Inference Rule Reference

This topic provides reference information for the RDFS-plus rules and the subset of OWL 2 RL rules that inference steps run.

RDFS-Plus Rules

The tables below define the RDFS-plus inference rules.

Semantics of Class Axioms

Rule	Description	IF	THEN
cax-eqc1	Two classes are synonymous.	T(?c1, owl:equivalentClass, ?c2) T(?x, rdf:type, ?c1)	T(?x, rdf:type, ?c2)
cax-eqc2	Two classes are synonymous.	T(?c1, owl:equivalentClass, ?c2) T(?x, rdf:type, ?c2)	T(?x, rdf:type, ?c1)
cax-sco	Members of a subclass are also members of the superclass.	T(?c1, rdfs:subClassOf, ?c2) T(?x, rdf:type, ?c1)	T(?x, rdf:type, ?c2)

Semantics of Axioms about Properties

Rule	Description	IF	THEN
prp-dom	Infer the subject's type from the predicate's domain.	T(?p, rdfs:domain, ?c) T(?x, ?p, ?y)	T(?x, rdf:type, ?c)
prp-eqp1	Two properties are synonymous.	T(?p1, owl:equivalentProperty, ?p2) T(?x, ?p1, ?y)	T(?x, ?p2, ?y)
prp-eqp2	Two properties are synonymous.	T(?p1, owl:equivalentProperty, ?p2) T(?x, ?p2, ?y)	T(?x, ?p1, ?y)
prp-fp	If predicate p is a functional property, then a subject can be related to only one specific object by p.	T(?p, rdf:type, owl:FunctionalProperty) T(?x, ?p, ?y1) T(?x, ?p, ?y2)	T(?y1, owl:sameAs, ?y2)
prp-ifp	If predicate p is an inverse functional property, then a specific object can be related to only one subject by p.	T(?p, rdf:type, owl:InverseFunctionalProperty) T(?x1, ?p, ?y) T(?x2, ?p, ?y)	T(?x1, owl:sameAs, ?x2)
prp-inv1	Two properties are the inverse of each other.	T(?p1, owl:inverseOf, ?p2) T(?x, ?p1, ?y)	T(?y, ?p2, ?x)
prp-inv2	Two properties are the inverse of each other.	T(?p1, owl:inverseOf, ?p2) T(?x, ?p2, ?y)	T(?y, ?p1, ?x)
prp-rng	Infer the object's type from the predicate's range.	T(?p, rdfs:range, ?c) T(?x, ?p, ?y)	T(?y, rdf:type, ?c)
prp-spo1	Relationships that are described by a subproperty also hold for the superproperty.	T(?p1, rdfs:subPropertyOf, ?p2) T(?x, ?p1, ?y)	T(?x, ?p2, ?y)

prp-symp	The inverse is true for a property.	T(?p, rdf:type, owl:SymmetricProperty) T(?x, ?p, ?y)	T(?y, ?p, ?x)
prp-trp	Chains of relationships collapse into a single relationship.	T(?p, rdf:type, owl:TransitiveProperty) T(?x, ?p, ?y) T(?y, ?p, ?z)	T(?x, ?p, ?z)

Semantics of Schema Vocabulary

Rule	Description	IF	THEN
scm-cls	Every class is its own subclass and equivalent class, and it is a subclass of owl:Thing.	T(?c, rdf:type, owl:Class)	T(?c, rdfs:subClassOf, ?c) T(?c, owl:equivalentClass, ?c) T(?c, rdfs:subClassOf, owl:Thing) T(owl:Nothing, rdfs:subClassOf, ?c)
scm-dom1	A property with domain c also has domain c's superclasses.	T(?p, rdfs:domain, ?c1) T(?c1, rdfs:subClassOf, ?c2)	T(?p, rdfs:domain, ?c2)
scm-dom2	A subproperty inherits the domains of the superproperties.	T(?p2, rdfs:domain, ?c) T(?p1, rdfs:subPropertyOf, ?p2)	T(?p1, rdfs:domain, ?c)
scm-eqc1	Equivalent classes are subclasses of each other.	T(?c1, owl:equivalentClass, ?c2)	T(?c1, rdfs:subClassOf, ?c2) T(?c2, rdfs:subClassOf, ?c1)
scm-eqc2	If two classes are subclasses, they are also equivalent classes.	T(?c1, rdfs:subClassOf, ?c2) T(?c2, rdfs:subClassOf, ?c1)	T(?c1, owl:equivalentClass, ?c2)

scm-eqp1	Equivalent properties are subproperties of each other.	T(?p1, owl:equivalentProperty, ?p2)	T(?p1, rdfs:subPropertyOf, ?p2) T(?p2, rdfs:subPropertyOf, ?p1)
scm-eqp2	If two properties are subproperties, they are also equivalent properties.	T(?p1, rdfs:subPropertyOf, ?p2) T(?p2, rdfs:subPropertyOf, ?p1)	T(?p1, owl:equivalentProperty, ?p2)
scm-rng1	A property with range c also has range c's superclasses.	T(?p, rdfs:range, ?c1) T(?c1, rdfs:subClassOf, ?c2)	T(?p, rdfs:range, ?c2)
scm-rng2	A subproperty inherits the ranges of its superproperties.	T(?p2, rdfs:range, ?c) T(?p1, rdfs:subPropertyOf, ?p2)	T(?p1, rdfs:range, ?c)
scm-sco	owl:subClassOf relationships are transitive	T(?c1, rdfs:subClassOf, ?c2) T(?c2, rdfs:subClassOf, ?c3)	T(?c1, rdfs:subClassOf, ?c3)
scm-spo	owl:subPropertyOf relationships are transitive.	T(?p1, rdfs:subPropertyOf, ?p2) T(?p2, rdfs:subPropertyOf, ?p3)	T(?p1, rdfs:subPropertyOf, ?p3)

Note The scm-dp and scm-op schema vocabulary rules are not run. Those rules add significant compute overhead but do not result in meaningful inference results.

OWL 2 RL Rules

The tables below define the subset of OWL 2 RL inference rules that inference steps run.

Semantics of Equality

Rule	Description	IF	THEN

eq-rep-o	Describes the replacement property of the owl:sameAs axiom.	T(?o, owl:sameAs, ?o') T(?s, ?p, ?o)	T(?s, ?p, ?o')
eq-rep-p	Describes the replacement property of the owl:sameAs axiom.	T(?p, owl:sameAs, ?p') T(?s, ?p, ?o)	T(?s, ?p', ?o)
eq-rep-s	Describes the replacement property of the owl:sameAs axiom.	T(?s, owl:sameAs, ?s') T(?s, ?p, ?o)	T(?s', ?p, ?o)
eq-sym	Describes the symmetric property of the owl:sameAs axiom.	T(?x, owl:sameAs, ?y)	T(?y, owl:sameAs, ?x)
eq-trans	Describes the transitive property of the owl:sameAs axiom.	T(?x, owl:sameAs, ?y) T(?y, owl:sameAs, ?z)	T(?x, owl:sameAs, ?z)

Semantics of Schema Vocabulary

Rule	Description	IF	THEN
scm-svf1	A property restriction c1 is a subclass of c2 if they are both someValuesFrom restrictions on the same property and c1's target class is a subclass of c2's target class.	T(?c1, owl:someValuesFrom, ?y1) T(?c1, owl:onProperty, ?p) T(?c2, owl:someValuesFrom, ?y2) T(?c2, owl:onProperty, ?p) T(?y1, rdfs:subClassOf, ?y2)	T(?c1, rdfs:subClassOf, ?c2)

scm-svf2	A property restriction c1 is a subclass of c2 if they are both someValuesFrom restrictions on the same class where c1's target property is a subproperty of c2's target property.	T(?c1, owl:someValuesFrom, ?y) T(?c1, owl:onProperty, ?p1) T(?c2, owl:someValuesFrom, ?y) T(?c2, owl:onProperty, ?p2) T(?p1, rdfs:subPropertyOf, ?p2)	T(?c1, rdfs:subClassOf, ?c2)
scm-int		T(?c, owl:intersectionOf, ?x) LIST[?x, ?c1, ..., ?cn]	T(?c, rdfs:subClassOf, ?c1) T(?c, rdfs:subClassOf, ?c2) ... T(?c, rdfs:subClassOf, ?cn)

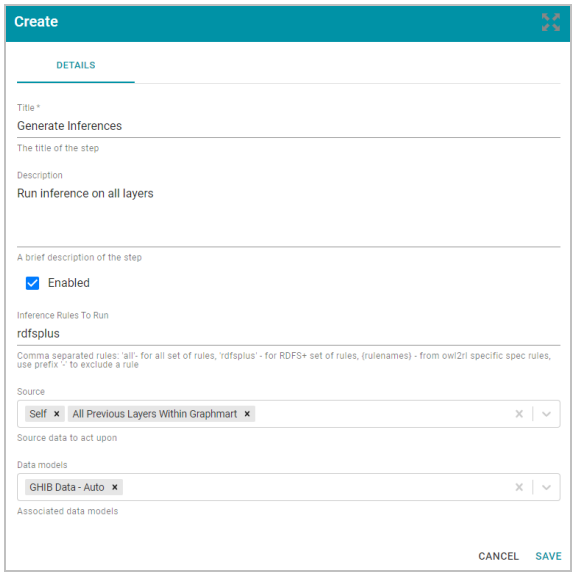
Semantics of Classes

Rule	Description	IF	THEN
cls-svf1	At least one object of a property is a member of the specified class.	T(?x, owl:someValuesFrom, ?y) T(?x, owl:onProperty, ?p) T(?u, ?p, ?v) T(?v, rdf:type, ?y)	T(?u, rdf:type, ?x)

cls-int1	An instance belongs to every one of the specified classes.	T(?c, owl:intersectionOf, ?x) LIST[?x, ?c1, ..., ?cn] T(?y, rdf:type, ?c1) T(?y, rdf:type, ?c2) ... T(?y, rdf:type, ?cn)	T(?y, rdf:type, ?c)

Example RDFS+ Inference Step

The following example inference step runs the RDFS-plus rules to generate inferences for the layers in a graphmart.



Related Topics

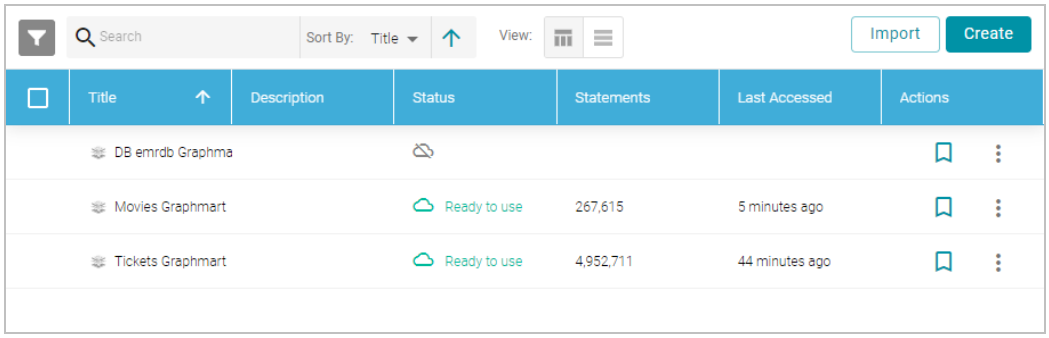
Adding Steps to Data Layers

Creating a Query Template

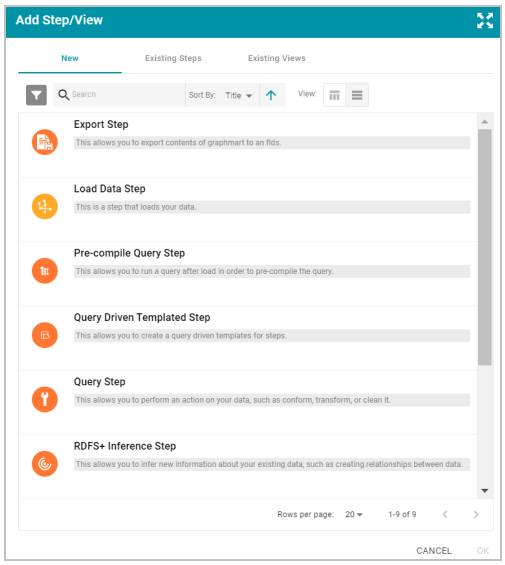
Templated steps enable users to create reusable templates for creating additional query steps in different layers or graphmarts. In templated queries, key-value pairs are represented by parameters in a query. When reusing the step, users do not need to rewrite the query to target the different data source. Instead, they modify the values for the keys. Follow the instructions below to add a reusable query template step.

Tip This type of query template step uses key-value pairs that are user-defined. Creating the key-value pairs requires familiarity with the data and properties defined in the model. To create a query template that enables you to run a query and automatically generate the key-value pairs, see Creating a Query-Driven Template.

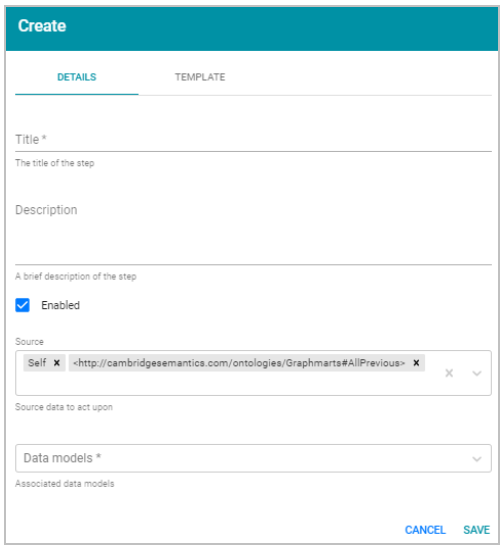
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



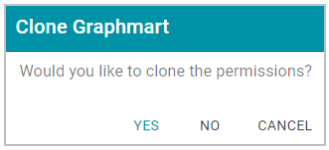
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing template step for reuse:
    - Select **Templated Step**, and then click **OK**. Anzo opens the Create templated step screen. Proceed to the next step.



- If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:
    a. Select the templated step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



    b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

    Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

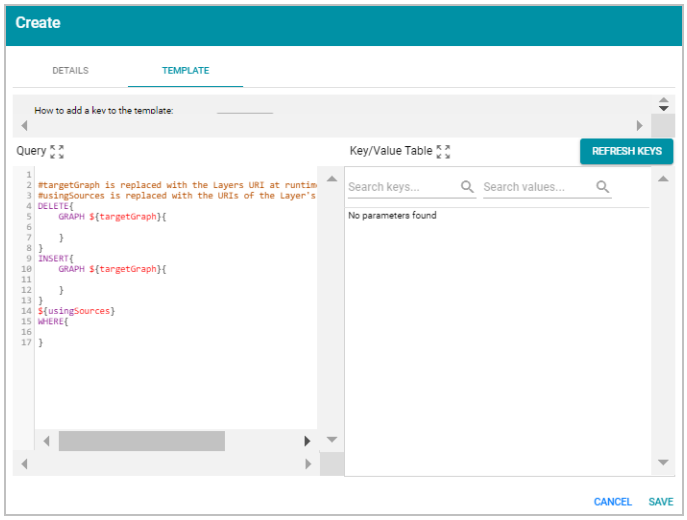
    c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit load data step screen. Proceed to the next step.

6. Under Details, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. Click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:
    - **Self**: This option is selected by default and means that the query runs against only the data that is generated in the layer to which this step belongs.
    - **All Previous Layers Within Graphmart**: Choosing this option means that the query runs against the data that is generated by all of the layers in the graphmart that precede this layer.

- **Previous Layer Within Graphmart**: Choosing this option means that the query runs against only the data that is generated by the one layer that precedes this layer.
- ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to run the query against only the data that is generated by that layer.

  You can remove any of the source options by clicking the X to the left of the option name.

9. Click the **Data models** drop-down list and select the model or models to use for this step to run against.
10. Click the **Template** tab. Anzo displays the Template screen.



11. Follow the steps below to create a template query that uses parameters to represent key-value pairs:
    a. On the left side of the screen, use the transformation template to write the query that this step will run. The template includes the syntax for writing SPARQL INSERT and DELETE queries and includes source and target graph parameters that Anzo replaces at runtime. Edit the template text as needed.

       In the query, include parameters in the format `${key_name}` that you intend to replace at runtime with the value that you define for the key. Anzo automatically adds the key to the Key/Value Table on the right side of the screen.

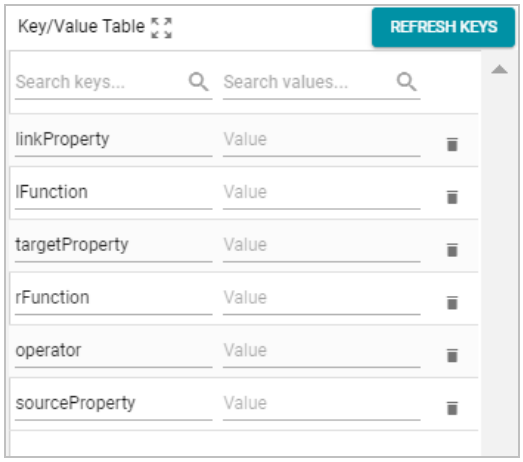
       For example, the following INSERT query includes several parameters that represent properties and functions:

```
INSERT {
  GRAPH ${targetGraph}{
    ?lsubj ${linkProperty} ?rsubj
  }
}
${usingSources}
WHERE {
  ?lsubj <${sourceProperty}> ?lobj .
```

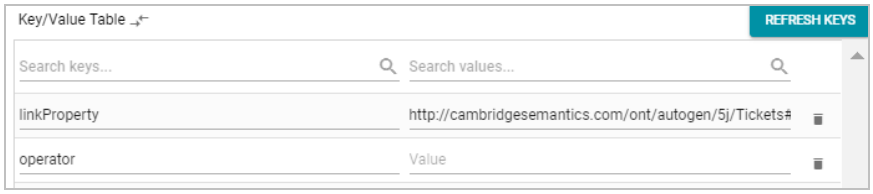
```
  ?rsubj <${targetProperty}> ?robj .
  FILTER (${lFunction}(?lobj) ${operator} ${rFunction}(?robj))
}
```

See SPARQL Query Templates and Best Practices for additional guidance on writing SPARQL queries.

b. On the right side of the screen above the key-value table, click **Refresh Keys**. Anzo adds each key in the template to the table. For example:



c. In each row, type the desired **Value** for each key. For example, the row below specifies an eventid property URI (http://cambridgesemantics.com/ont/autogen/5j/Tickets#tickit_sales_eventid) as the value for the linkProperty key:



To delete a key-value row, click the trashcan icon (🗑) to the right of the row.

12. Click **Save** to save the template and add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

The new Templated Step becomes available to clone into other data layers. Users can select the step from the Existing Steps tab when they add steps.

Related Topics

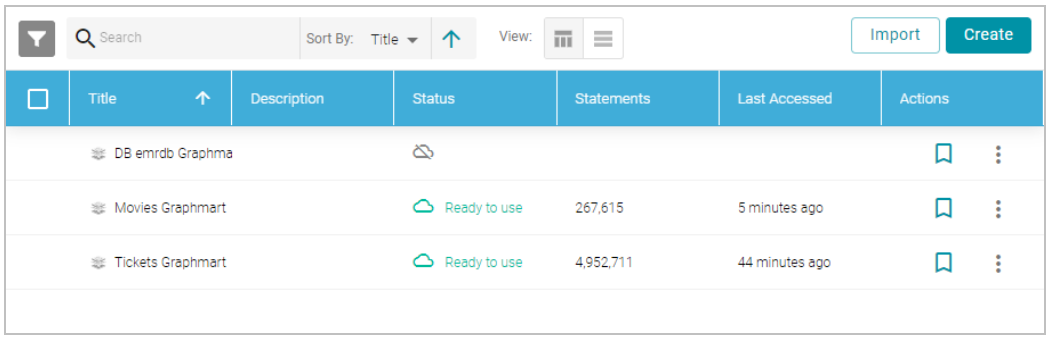
Adding Steps to Data Layers

SPARQL Query Templates and Best Practices

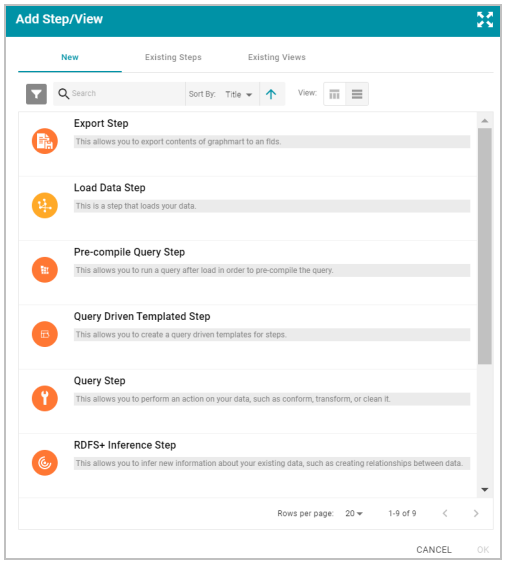
Validating the Data

Follow the instructions below to add a step that validates the data in a data layer.

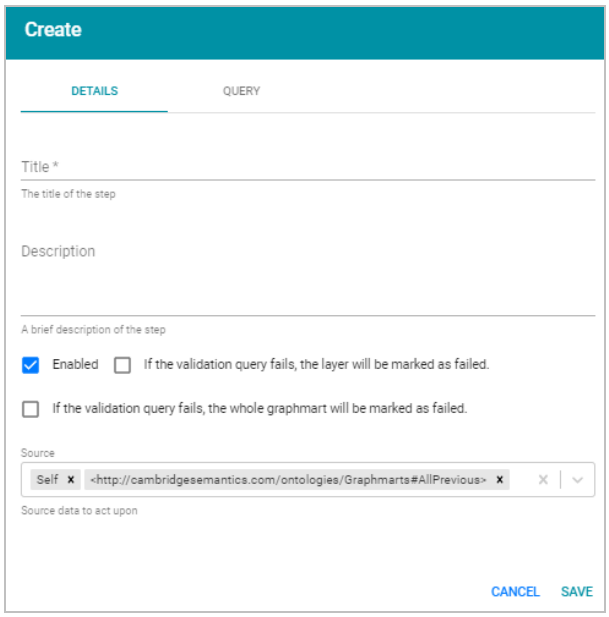
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



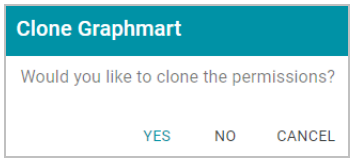
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing validation step for reuse:
   - Select **Validation Step**, and then click **OK**. Anzo opens the Create validation step screen. Proceed to the next step.



- If you want to clone an existing step and add it to this layer, click the **Existing Steps** tab and follow these steps:
    a. Select the validation step that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



    b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

    Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

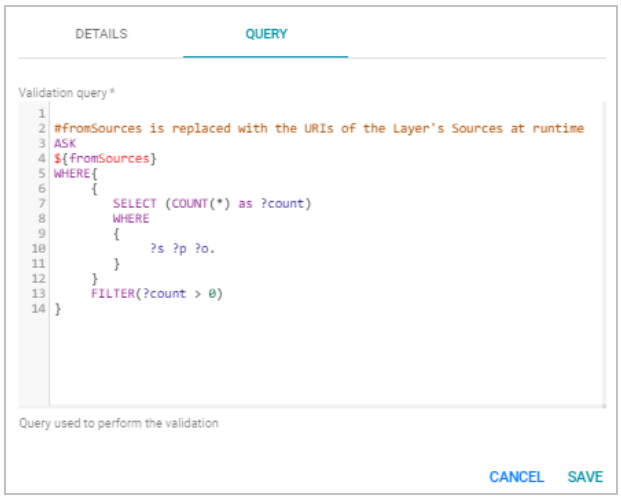
    c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit load data step screen. Proceed to the next step.

6. Under Details, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
8. Specify what action, if any, you want Anzo to take if this validation step fails. The step includes the following two settings that control how Anzo treats the layer or graphmart if the validation fails:
    - **If the validation query fails, the layer will be marked as failed**: Select this option if you want Anzo to abort the load of the data layer if this step fails. The graphmart and other successful data layers continue to load.

- **If the validation query fails, the whole graphmart will be marked as failed**: Select this option if you want Anzo to abort the load of the entire graphmart if this validation step fails.

If you want Anzo to proceed to load the data layer if the validation step fails, leave both options blank.

9. Click the **Query** tab to compose the query that this step will run. Anzo displays the Query screen.



The tab tab includes the syntax for writing a SPARQL ASK query, which is useful for determining whether a certain pattern exists in the data. ASK queries return "true" or "false" to indicate whether a solution exists. The template includes a source graph parameter (`${fromSources}`). Using the configured Source data options from the Details tab, Anzo automatically populates the query with the appropriate source graph URIs when the query runs.

10. Edit the template text as needed, and then click **Save** to save the validation query and add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

Related Topics

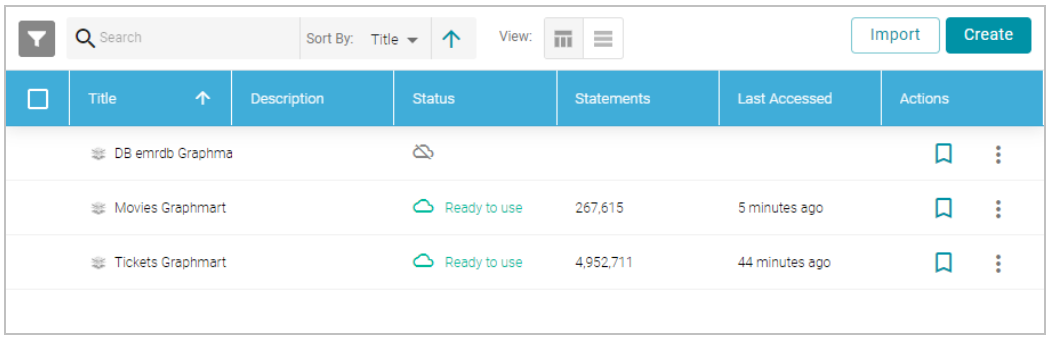
Adding Steps to Data Layers

Creating a View of the Data

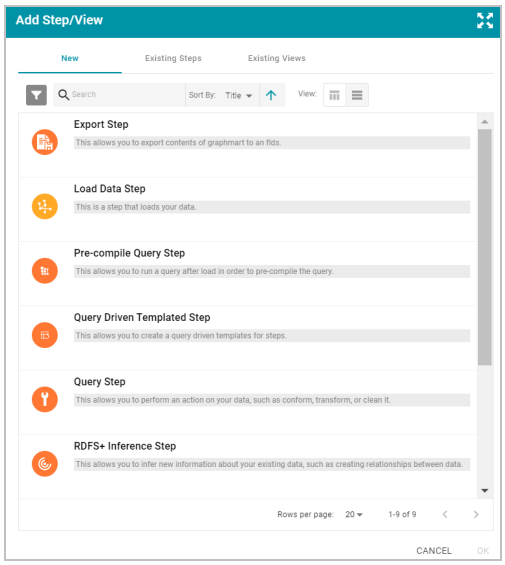
Follow the instructions below to add a step to a data layer that creates a custom view of the data but does not change the source or materialize new data by default. View steps use SPARQL CONSTRUCT queries to create a view definition in AnzoGraph.

Tip If you plan to write a view to hide or mask sensitive information in the data, Anzo also provides the option to quickly configure masking at the data layer level. See Masking Data in Data Layers for more information.

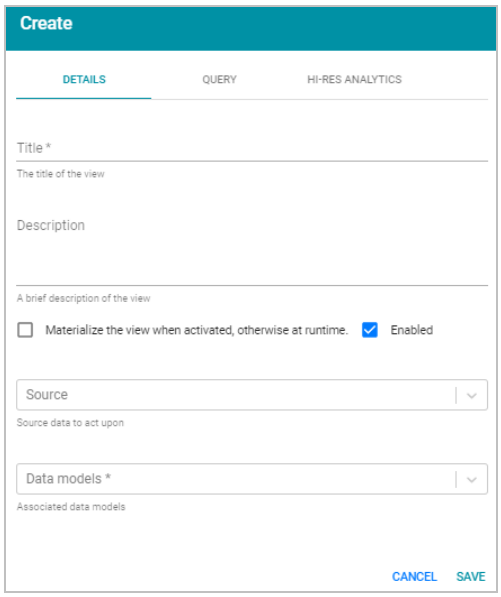
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



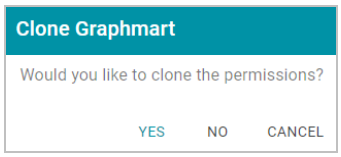
2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the menu icon (⋮) on the layer for which you want to create a step, and then select **Add Step/View**. Anzo opens the Add step dialog box.



5. Follow one of the options in this step, depending on whether you want to create a step from scratch or clone an existing view step for reuse:
    - If you want to create a new step, select **View**, and then click **OK**. Anzo opens the Create view step screen. Proceed to the next step.



- If you want to clone an existing step and add it to this layer, click the **Existing Views** tab and follow these steps:
    a. Select the view that you want to clone and click **OK**. Anzo displays the Clone dialog box, which asks if you want to copy the permissions from the existing step.



    b. On the Clone dialog box, click **Yes** to copy the permission configuration from the existing step or click **No** to copy the step without the permission configuration.

    Anzo clones the step, adds the copy to the layer, and returns to the Data Layers screen.

    c. On the Data Layers screen, click the menu icon (⋮) on the cloned step and select **Edit**. Anzo opens the Edit load data step screen. Proceed to the next step.

6. On the Details tab, type a name for the step in the **Title** field and add an optional description in the **Description** field.
7. If you want to store a copy of the data that the view creates (materialize the data), select the **Materialize the view when activated...** check box. When this option is disabled Anzo creates a virtual view where only the view definition is stored in memory and not a copy of the data.
8. By default the **Enabled** option is selected, indicating that the step is enabled and will run when the layer is loaded. If you want to disable the step so that it is not processed, clear the Enabled check box.
9. Click the **Source** drop-down list and configure the source data for this step. Steps can build upon the data generated by steps in other layers or can be self-contained, applying changes that relate only to the data defined in the layer that contains this step. You can select any number of the following options:

- **Self**: This option is selected by default and means that the query runs against only the data that is generated in the layer to which this step belongs.
- **All Previous Layers Within Graphmart**: Choosing this option means that the query runs against the data that is generated by all of the layers in the graphmart that precede this layer.
- **Previous Layer Within Graphmart**: Choosing this option means that the query runs against only the data that is generated by the one layer that precedes this layer.
- ***Layer Name***: The Source drop-down list also includes options for specific layer names. You can choose a specific layer to run the query against only the data that is generated by that layer.

  You can remove any of the source options by clicking the X to the left of the option name.

10. Click the **Data models** drop-down list and select the model or models to run this query against.
11. In the **Query** field, compose the CONSTRUCT query that creates the view of the data that you want to see. You can use the following syntax as a template for the query:

```
CONSTRUCT {
}
${fromSources}
${fromNamedSources}
WHERE {
  GRAPH ?graph {
 }
}
```

    Do not include a GRAPH keyword in the CONSTRUCT clause as Anzo uses the view's URI as the graph URI for the constructed triples. In addition, Anzo uses the configured Source data options to automatically replace the `${fromSources}` and `${fromNamedSources}` variables with the appropriate FROM clauses when the query runs.

    For more information about CONSTRUCT queries, see CONSTRUCT in the W3C SPARQL 1.1 Query Language specification.

12. Click **Save** to add the step to the data layer. Anzo adds the step as the last step in the layer. If you want to change the order of the steps, click the black bar on the left side of a step and drag it up or down.

The **Hi-Res Analytics** tab for view steps contains advanced settings that control how the layer is exposed to and affects Hi-Res Analytic queries. Changing these settings can have unexpected consequences. Cambridge Semantics recommends that you do not modify the Hi-Res Analytics settings unless you understand the repercussions. To learn about the advanced settings, see Data Layer Hi-Res Analytics Settings Reference.

Related Topics

Adding Steps to Data Layers

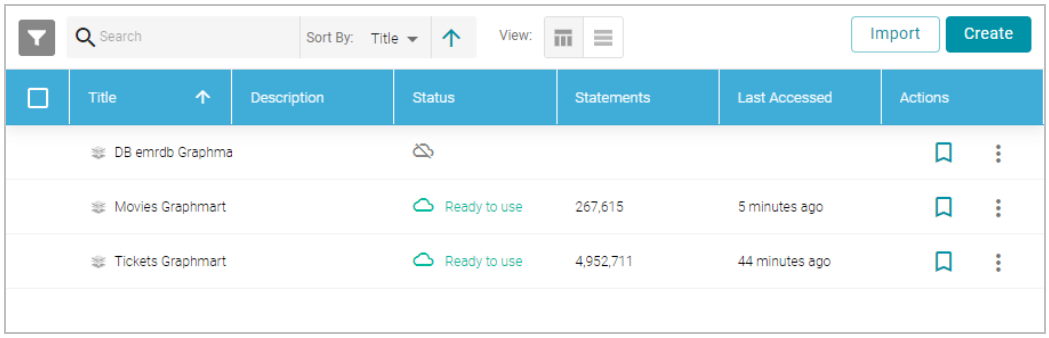
SPARQL Query Templates and Best Practices

Masking Data in Data Layers

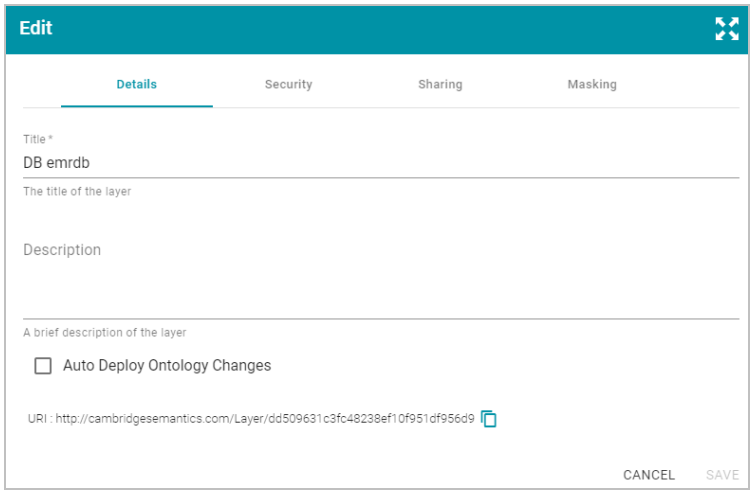
Anzo data layers offer a solution that enables users to hide or mask sensitive information by selecting properties or predicates to hide in the layer. When you mask predicates at the data layer level, Anzo still loads the triples associated with those predicates so that other steps and layers in the graphmart can use that data in calculations. The triples are excluded from Hi-Res Analytics, however. This topic provides instructions for configuring a data layer to mask data.

Note To configure masking, the data layer must include at least one Load Data Step or Query Step. You cannot configuring masking on data layers that contain only View or RDFS+ Inference steps. For information about adding steps to layers, see Adding Steps to Data Layers.

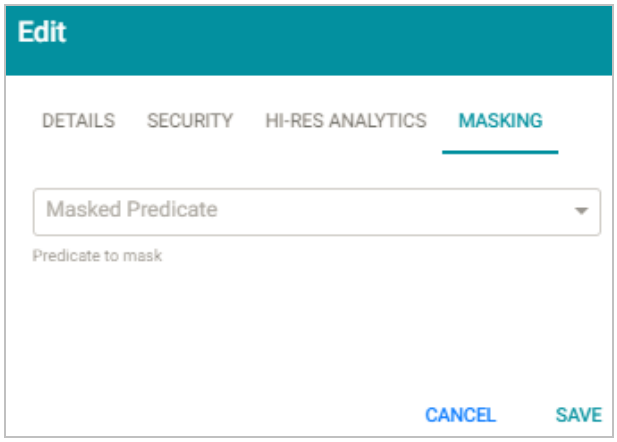
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



2. On the Graphmarts screen, click the name of the graphmart that you want to change.
3. Click the **Data Layers** tab. Anzo displays the existing data layers.
4. Click the ellipses (⋮) on the layer for which you want to mask data, and then select **Edit**. Anzo opens the Edit data layer screen. For example:



5. Click the **Masking** tab:



6. On the Masking screen, click the **Masked Predicate** drop-down list. The list includes the predicates or properties from the ontologies selected in the data layer's steps. Select a property to add it to the Masked Predicate field. Repeat this step to mask additional properties. You can remove a property from the masked list by clicking the X to the right of the property name.

   For example, the following image shows a data layer that masks users' credit card values. When users view Hi-Res Analytics that include this layer, the "card" property is not available to display.

7. Click **Save** to save the masking configuration and return to the Data Layers screen.

Related Topics

Introduction to Data Layers

Adding Data Layers to Graphmarts

Adding Steps to Data Layers

Managing Graphmart, Data Layer, and Step Security

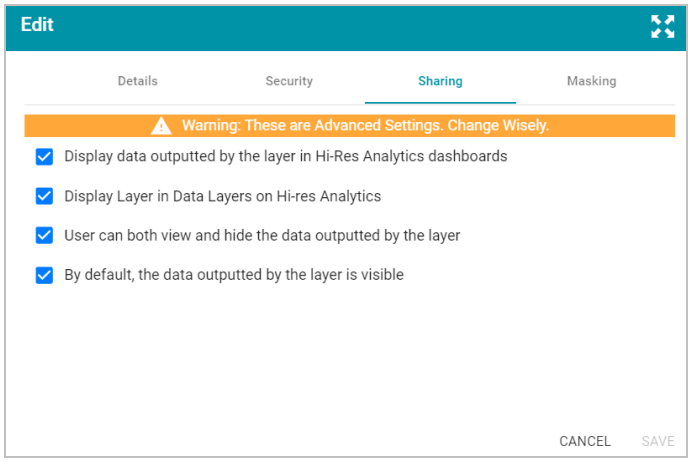
Data Layer Hi-Res Analytics Settings Reference

Data Layer Hi-Res Analytics Settings Reference

This topic provides reference information about the advanced data layer Hi-Res Analytics settings that control how a layer is exposed to and affects Hi-Res Analytic queries.

Important Changing these settings can have unexpected consequences.

The Hi-Res Analytics settings are available on the **Sharing** tab when you edit a data layer:



The sections below describe each of the available settings:

- Display data outputted by the layer in Hi-Res Analytics dashboards
- Display Layer in Data Layers in Hi-Res Analytics
- User can both view and hide the data outputted by the layer
- By default, the data outputted by the layer is visible

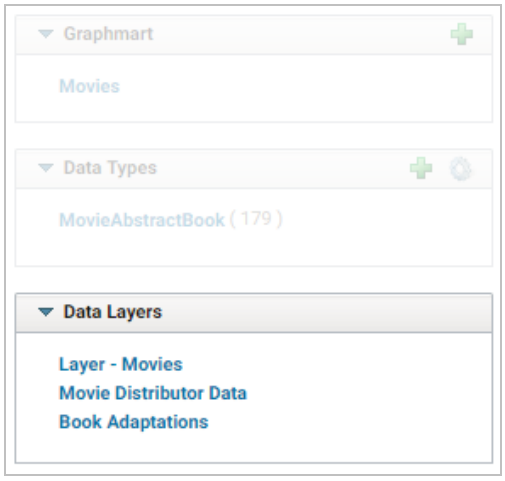
Display data outputted by the layer in Hi-Res Analytics dashboards

This setting controls whether the data generated by the steps is available to query and display in Hi-Res Analytics:

- When the setting is **enabled** (the default value), the layer's data is available to Hi-Res Analytics.
- When the setting is **disabled**, other data layers in the graphmart can use the layer's data, but the data is not available to Hi-Res Analytics.

Display Layer in Data Layers in Hi-Res Analytics

This setting controls whether Anzo displays the layer in the Data Layers panel on Hi-Res Analytics dashboards. The image below shows an example Data Layers panel:



- When the setting is **enabled** (the default value), the layer is listed in the Data Layers panel in Hi-Res Analytics.
- When the setting is **disabled**, the layer's data is always used in Hi-Res Analytics but users do not see the layer listed in the Data Layers panel.

User can both view and hide the data outputted by the layer

This setting controls whether users have the option to show and hide the layer in the Data Layers panel on Hi-Res Analytics dashboards:

- When the setting is **enabled** (the default value), the layer is listed in the Data Layers panel in Hi-Res Analytics and users have the option to show and hide the layer.
- When the setting is **disabled**, whether the layer shows up in the Data Layers list depends on the **By default, the data outputted by the layer is visible** setting. If the layer is visible in the Data Layers panel ("By default, the data outputted by the layer is visible" is enabled), users cannot toggle it on and off.

By default, the data outputted by the layer is visible

This setting controls whether the data generated by the steps in the layer is visible in Hi-Res Analytics:

- When the setting is **enabled** (the default value), the layer is listed in the Data Layers panel in Hi-Res Analytics and is selected by default.
- When the setting is **disabled**, the layer shows up in the Data Layers panel but is not selected. To include the layer's data in Hi-Res Analytic queries, users must select the layer.

Related Topics

Introduction to Data Layers

Adding Data Layers to Graphmarts

Adding Steps to Data Layers

Masking Data in Data Layers

Managing Graphmart, Data Layer, and Step Security

Managing Graphmart, Data Layer, and Step Security

Security for graphmarts is primarily managed by configuring user, group, and role permissions at the graphmart, data layer, and step level. Together, the permissions defined for each component control the data that a user can access, whether they can view or modify a component, and whether they can view or modify a component's metadata.

This topic provides details about the permissions for each component and includes instructions for configuring permissions for each type of component.

- Graphmart, Layer, and Step Permissions Reference
- Configuring Graphmart, Layer, or Step Permissions

Graphmart, Layer, and Step Permissions Reference

In the Anzo application, graphmarts, data layers, and steps offer the same predefined permission sets to apply and use the same mechanism for assigning permissions, but the privileges granted with a permission set differ depending on the component:

- **Graphmart** permissions control a user's ability to activate, deactivate, and reload or refresh a graphmart, view or modify a graphmart and its metadata, and view, create, or modify data layers.
- **Data Layer** permissions control which users can access or modify the data that is output from a layer, i.e., which users can enable or disable layers, edit, create, and delete layers, or change layer metadata, such as security settings.
- **Step** permissions also control which users can access or modify the data that is output from a layer, i.e., which users can enable and disable steps, add, edit, and delete steps, and view or modify step metadata.

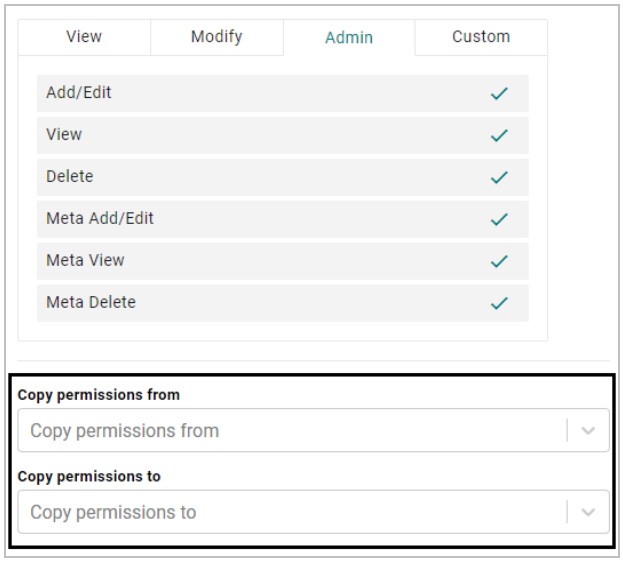
This section provides information about the predefined permission sets and default permissions for each component.

Permission Inheritance

When assigning permissions at the graphmart, data layer, or step level, you can configure that component to inherit the permissions from another component or pass on its permissions to other components. For example, you can configure one graphmart to pass its permissions to other graphmarts. Inheritance transmits all of a component's permissions for all users, groups, and roles.

Note **By default, data layers and steps inherit their permissions from the parent graphmart**. That means graphmart permissions supersede the permissions set at the data layer or step level by default. For simplicity and to avoid unexpected outcomes, Cambridge Semantics recommends that you manage all permissions at the graphmart level.

The inheritance settings are displayed below the permissions table on the graphmart Sharing tab or the Security tab for data layers and steps.



Since graphmarts pass permissions to layers and steps, by default, the **Copy permissions from** field is empty for graphmarts. And the **Copy permissions to** field is populated with the names of the data layers and steps in the graphmart. For data layer and step permissions, the **Copy permissions from** field is populated with the parent graphmart name, and the **Copy permissions to** field is empty.

Graphmart Level Permissions Reference

Graphmart level permissions control a user's ability to view, activate and deactivate, reload or refresh a graphmart, modify a graphmart's content, or view or modify its metadata. There are three predefined graphmart permission sets that include a combination of six permissions that can be assigned to an Anzo user, group, or role. You also have the option to customize the set of permissions that are applied to a user, group, or role.

The table below lists the predefined permission sets and describes the privileges that are granted for each permission that is part of the predefined set:

Set	Applied Permission	Allows a user to:
View	**View (Graphmart)**	• See the graphmart in the Anzo application. • Copy the graphmart URI from the Overview tab. • Copy data layer URIs from the Data Layers tab. • See the existing Data on Demand endpoints on the Data on Demand tab. • Reload and refresh the graphmart. • Create and import graphmart versions.
	Meta View (Sharing Tab)	• The Meta View permission relates only to the graphmart Sharing tab. If a user has this permission, they can see the Sharing tab, but they cannot modify, add, or remove permissions.
Modify	In addition to the **View** and **Meta View** permissions described above, the **Modify** set includes the **Add/Edit** and **Delete** permissions described below.	
	Add/Edit (Graphmart)	• Rename the graphmart and edit the description. • Create Data on Demand endpoints. • Add data sets to the graphmart. • Enable, disable, or add and edit data layers and steps. • Activate and deactivate the graphmart.
	Delete (Graphmart)	• Remove data sets from the graphmart. • Delete data layers and steps from the graphmart. • Delete the graphmart.
Admin	In addition to the **View**, **Meta View**, **Add/Edit**, and **Delete** permissions described above, the **Admin** set includes the **Meta Add/Edit** and **Meta Delete** permissions described below.	
	Meta Add/Edit (Sharing Tab)	• The Meta Add/Edit permission relates only to the graphmart Sharing tab. If a user has this permission, they can modify graphmart access on the Sharing tab by adding permissions to a user, group, or role.
	Meta Delete (Sharing Tab)	• The Meta Delete permission relates only to the graphmart Sharing tab. If a user has this permission, they can modify graphmart access by removing permissions from a user, group, or role.

Default Graphmart Permissions

The table below lists the predefined permission sets that are applied by default when a new graphmart is created. Besides the sysadmin user, the graphmart creator is granted **Admin** privileges by default. The Everyone role is granted **View** privileges by default. No other users, groups, or roles have graphmart permissions assigned by default.

Anzo User/Role	Applied Permission Set
Sysadmin User	Admin
Graphmart Creator	Admin
Everyone Role	View

Note The default graphmart permission configuration is controlled by the default access policy for the Graphmarts registry. For information about default access policies, see Managing Default Access Policies.

Data Layer Level Permissions Reference

Data layer level permissions control a user's ability to view, enable and disable, and edit, create, and delete a data layer or view or modify its metadata.

Note Data layer permissions also depend on the permissions assigned for the parent graphmart. By default, all data layers and steps in a graphmart inherit their permissions from the graphmart. To navigate to a data layer, a user needs to have **View** permissions for the parent graphmart. To activate or deactivate the graphmart that contains the data layer of interest, or to create a new data layer, a user needs **Modify** permissions for the graphmart.

There are three predefined data layer permission sets that include a combination of six permissions that can be assigned to an Anzo user, group, or role. You also have the option to customize the set of permissions that are applied to a user, group, or role.

The table below lists the predefined permission sets and describes the privileges that are granted for each permission that is part of the predefined set:

Set	Applied Permission	Allows a user to:
View	**View (Data Layer)**	• See the layer on the Data Layers tab in the Anzo application. • Make a copy of the layer and copy the layer URI. • Make a copy of the steps in the layer and copy the step URIs. • View the data that is output by the layer.
	Meta View (Security Tab)	• The Meta View permission relates only to the layer Security tab. If a user has this permission, they can see the Security tab, but they cannot modify, add, or remove permissions.
Modify	In addition to the **View** and **Meta View** permissions described above, the **Modify** set includes the **Add/Edit** and **Delete** permissions described below.	
	Add/Edit (Data Layer)	• Modify the data layer.
	Delete (Data Layer)	• Delete the data layer.
Admin	In addition to the **View**, **Meta View**, **Add/Edit**, and **Delete** permissions described above, the **Admin** set includes the **Meta Add/Edit** and **Meta Delete** permissions described below.	
	Meta Add/Edit (Security Tab)	• The Meta Add/Edit permission relates only to the layer Security tab. If a user has this permission, they can modify layer access by adding permissions to a user, group, or role.
	Meta Delete (Security Tab)	• The Meta Delete permission relates only to the layer Security tab. If a user has this permission, they can modify layer access by removing permissions from a user, group, or role.

Default Data Layer Permissions

The table below lists the predefined permission sets that are applied by default when a new layer is created. Besides the sysadmin user, the layer creator is granted **Admin** privileges by default. The Everyone role is granted **View** privileges by default. No other users, groups, or roles have layer permissions assigned by default.

Anzo User/Role	Applied Permission Set
Sysadmin User	Admin
Layer Creator	Admin
Everyone Role	View

Step Level Permissions Reference

Step level permissions control a user's ability to view, enable and disable, and edit, create, and delete a step or view or modify its metadata.

Note Step level permissions also depend on the permissions assigned for the parent data layer and graphmart. By default, all data layers and steps in a graphmart inherit their permissions from the graphmart. To navigate to a step, a user needs to have **View** permissions for the parent graphmart and layer. To enable, disable, or edit and delete steps, a user needs **Modify** permissions for the data layer.

There are three predefined step permission sets that include a combination of six permissions that can be assigned to an Anzo user, group, or role. You also have the option to customize the set of permissions that are applied to a user, group, or role.

The table below lists the predefined permission sets and describes the privileges that are granted for each permission that is part of the predefined set:

Set	Applied Permission	Allows a user to:
View	**View (Step)**	• See the step on the Data Layers tab in the Anzo application. • Make a copy of the step and copy the step URI. • View the data that is output by the step.
	Meta View (Security Tab)	• The Meta View permission relates only to the step Security tab. If a user has this permission, they can see the Security tab, but they cannot modify, add, or remove permissions.

Set	Applied Permission	Allows a user to:
Modify	In addition to the **View** and **Meta View** permissions described above, the **Modify** set includes the **Add/Edit** and **Delete** permissions described below.	
	Add/Edit (Step)	• Modify the step.
	Delete (Step)	• Delete the step.
Admin	In addition to the **View**, **Meta View**, **Add/Edit**, and **Delete** permissions described above, the **Admin** set includes the **Meta Add/Edit** and **Meta Delete** permissions described below.	
	Meta Add/Edit (Security Tab)	• The Meta Add/Edit permission relates only to the step Security tab. If a user has this permission, they can modify step access by adding permissions to a user, group, or role
	Meta Delete (Security Tab)	• The Meta Delete permission relates only to the step Security tab. If a user has this permission, they can modify step access by removing permissions from a user, group, or role.

Default Step Permissions

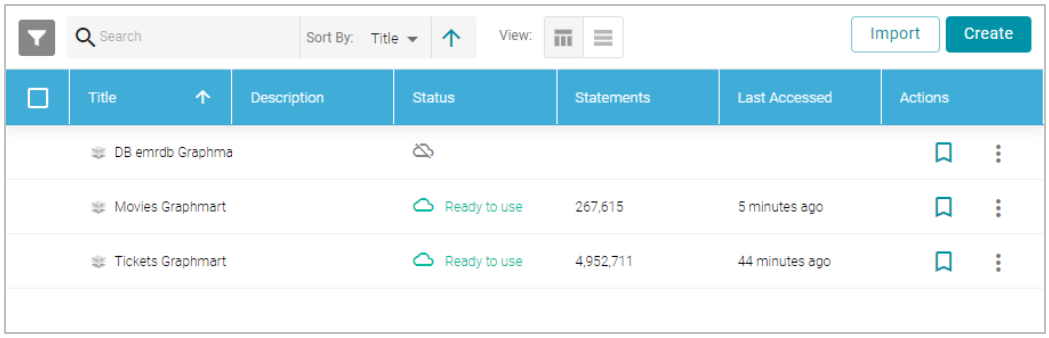
The table below lists the predefined permission sets that are applied by default when a new step is created. Besides the sysadmin user, the step creator is granted **Admin** privileges by default. The Everyone role is granted **View** privileges by default. No other users, groups, or roles have step permissions assigned by default.

Anzo User/Role	Applied Permission Set
Sysadmin User	Admin
Step Creator	Admin
Everyone Role	View

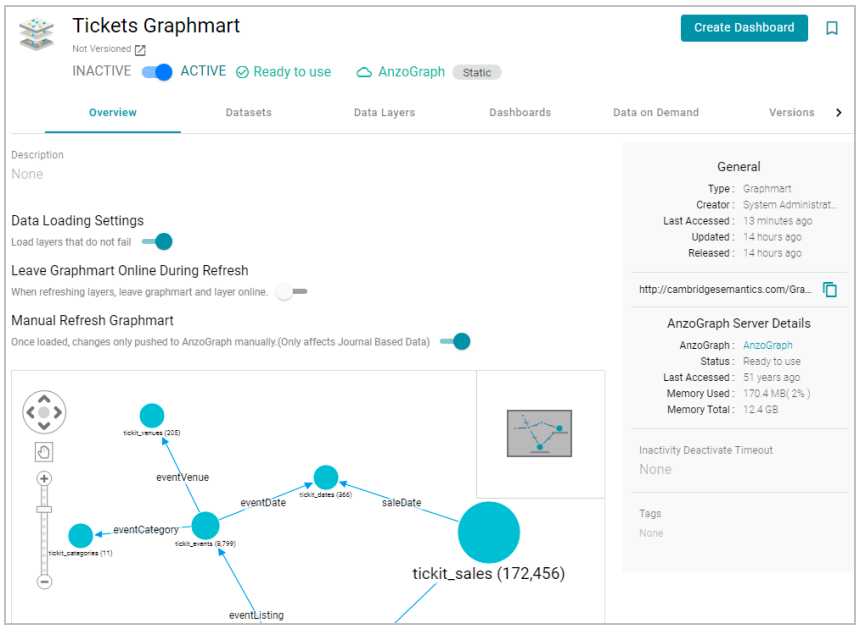
Configuring Graphmart, Layer, or Step Permissions

Follow the instructions below to configure permissions at the graphmart, data layer, or step level. For details about the predefined permission sets and associated privileges, see the Graphmart, Layer, and Step Permissions Reference above.

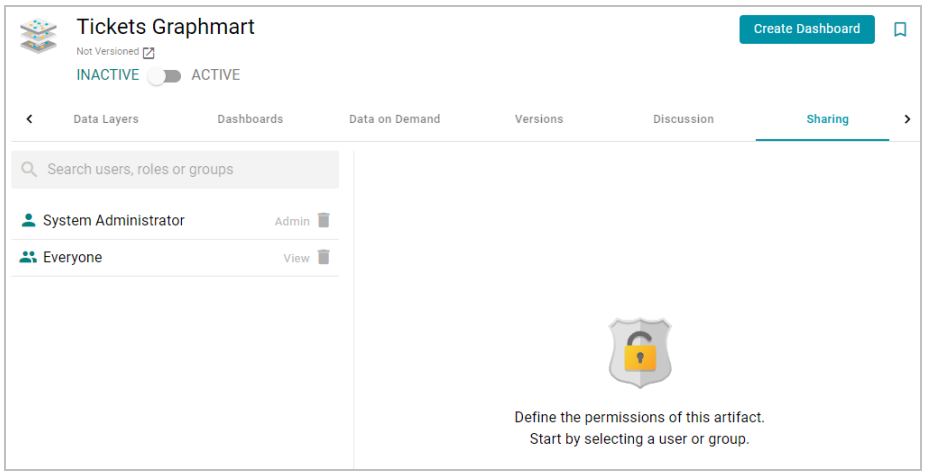
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



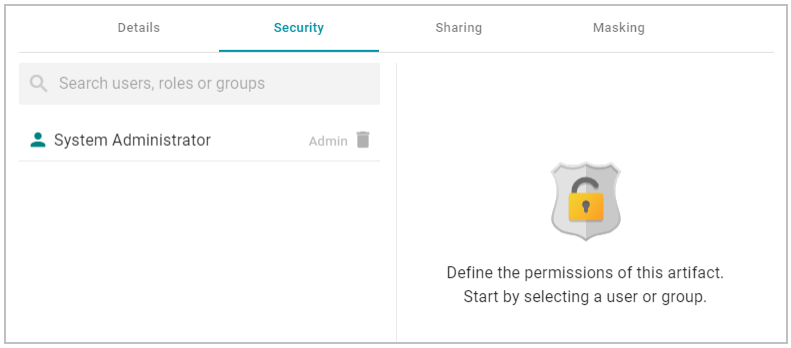
2. On the Graphmarts screen, click the name of the graphmart for which you want to configure permissions. Anzo displays the graphmart details. For example:



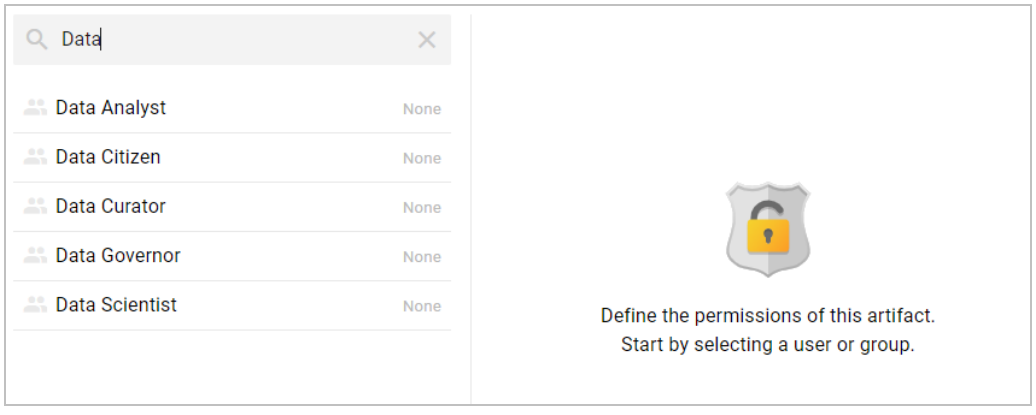
3. Follow the appropriate instructions below, depending on whether you want to configure permissions at the graphmart level or for a layer or step in the graphmart:
   - To configure permissions at the graphmart level, click the **Sharing** tab. The security screen is displayed. For example:



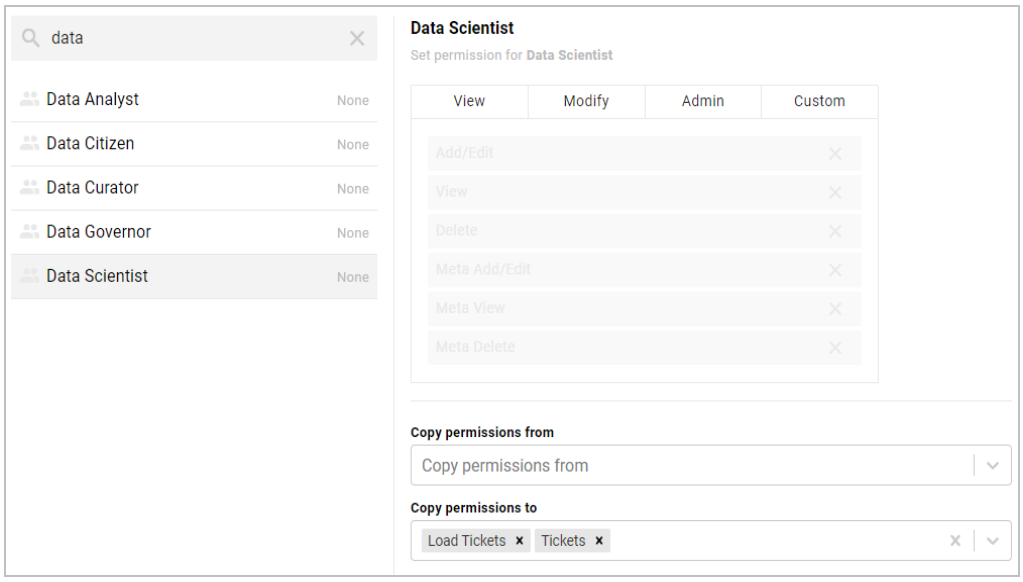
- To configure permissions for a data layer or step in the graphmart, click the **Data Layers** tab. On the Data Layers screen, find the layer or step that you want to configure. Then click the menu icon (⋮) for that layer or step and select **Edit**. On the Edit screen, click the **Security** tab. The security screen is displayed. For example:



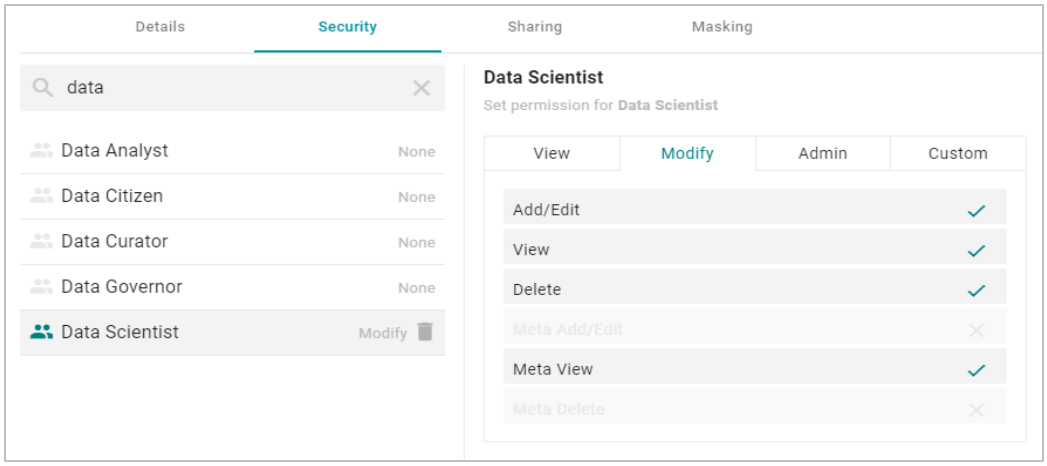
4. On the security screen, type a value in the **Search users, roles or groups** field to find and display a user, role, or group. The resulting list shows the current permission level that is set for each user, role, or group. For example, the image below shows a list of roles and their current permissions (None):



5. On the left side of the screen, select the user, role, or group for which you want to configure permissions. The permissions settings are displayed on the right side of the screen. For example:



6. To assign a predefined set of permissions, click the **View**, **Modify**, or **Admin** radio button to assign that level of access to the selected user, role, or group. Refer to the Graphmart, Layer, and Step Permissions Reference above for details about the permissions sets. For example, the image below gives Modify permissions to users with the Data Scientist role:



   If you want to customize the permissions, click the **Custom** radio button and then select or deselect the permissions checkboxes. To clear permissions for a user, role, or group, click the trashcan icon ( ) next to the user, role, or group name.

7. If you want to change the inheritance for the component, use the fields below the permissions table. For details about inheritance, see Permission Inheritance above. To apply all of the permissions from another component to this component, select the component to inherit from in the **Copy permissions from** field. To pass this component's permissions to other components, select the components to pass permissions to in the **Copy permissions to** field. For example, the image below shows the inheritance configuration for a graphmart:

8. When you are finished configuring permissions, click **Save** to save the changes.

Changes to graphmart, data layer, and step permissions take effect immediately. Users do not need to log out and log back in, and affected graphmarts do not need to be reloaded or refreshed.

Related Topics

Creating a Graphmart

Adding a Data Set to a Graphmart

Adding Data Layers to Graphmarts

Adding Steps to Data Layers

Blending Data with the Data Toolkit Service

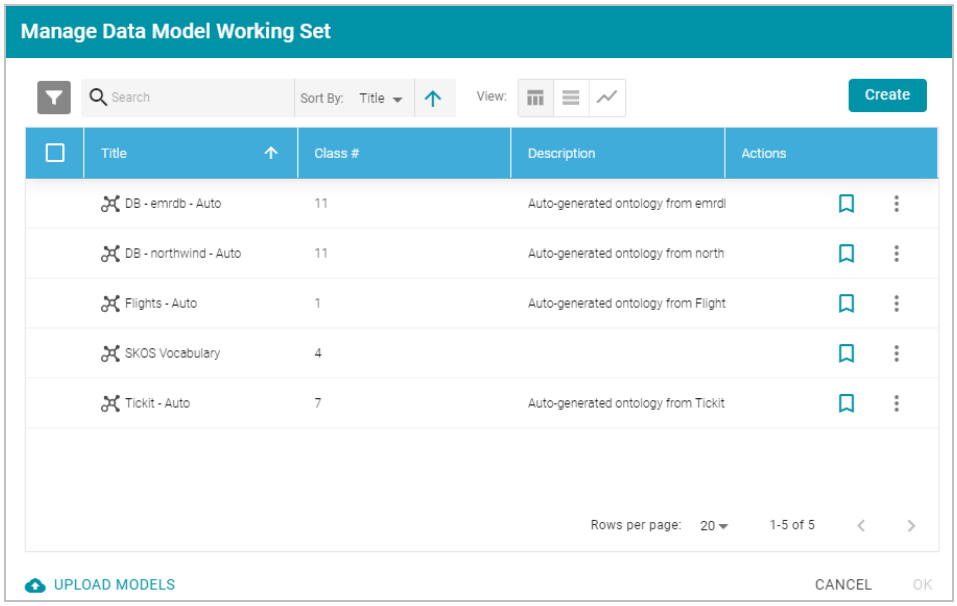
The Anzo Data Toolkit SPARQL service is a flexible tool that enables users to query data from remote endpoints that are accessible over JDBC or HTTP. Using the data toolkit, you can access data that is stored in databases, web servers, or files. You can also use the toolkit to invoke other scripts. The data that you retrieve can be incorporated into a data layer to augment the data that is stored in Anzo.

Note Data Toolkit Service calls are processed by AnzoGraph using a data toolkit Java plugin. Before running Data Toolkit Service queries, make sure that the AnzoGraph cluster is configured to use the toolkit. For more information, see the Pre- and Post- Installation configuration instructions in Installing AnzoGraph on a Single Server or Installing AnzoGraph on a Cluster.

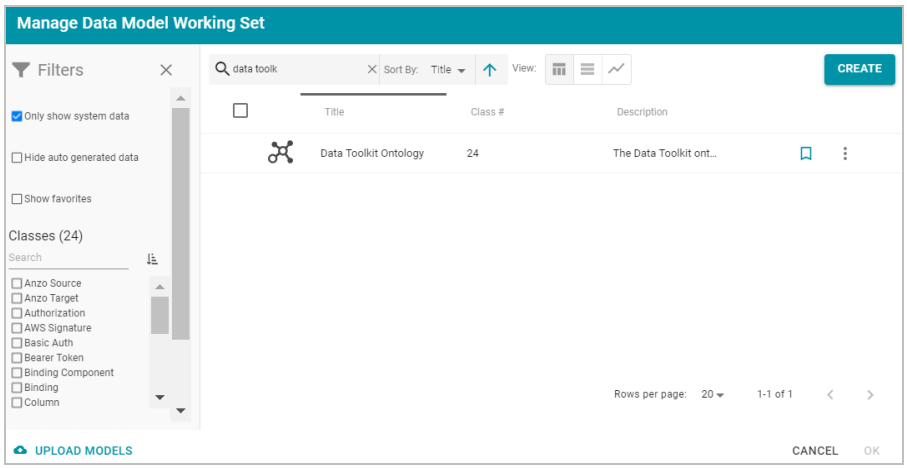
Getting Familiar with the Data Toolkit

One way to learn about the capabilities of the Data Toolkit is to explore the Data Toolkit Ontology, which is available as a system model. Exploring the classes that the model contains enables you to view details such as the properties that are available for queries, the types of data sources you can target, and how to supply credentials. This section provides instructions for viewing the model.

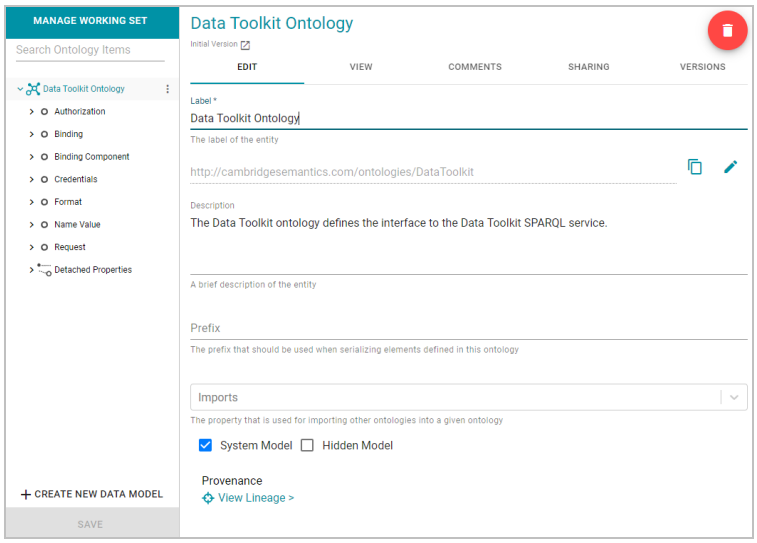
1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



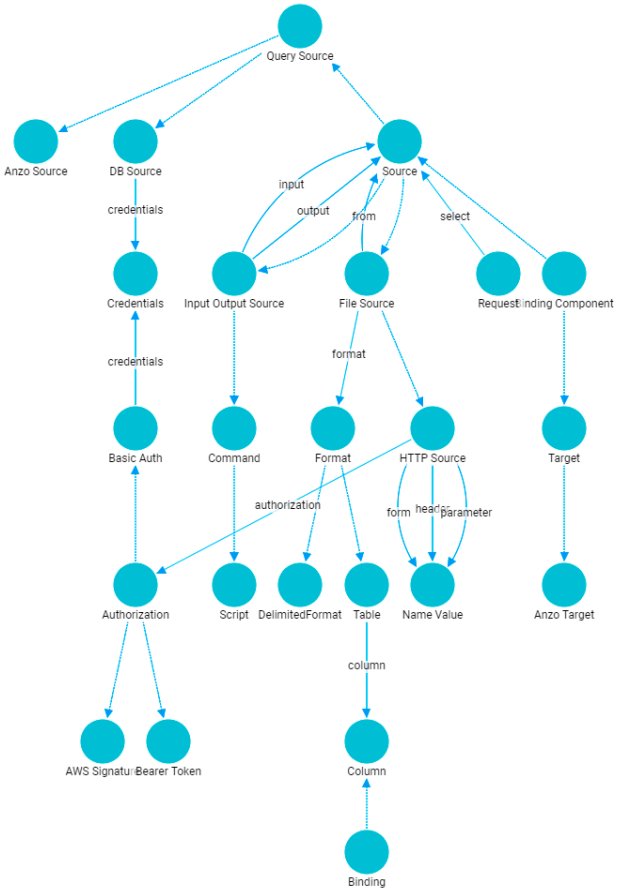
2. Open the Filters panel by clicking the filter icon ( ) in the top left corner of the screen.
3. In the Filters panel, select the **Only show system data** checkbox. The Manage Working Set screen is refreshed to show only the system models.
4. In the search field at the top of the screen, search for "data toolkit." The working set screen displays the model.



5. To open the model, select the checkbox to the left of the model and click **OK**. Anzo opens the model in the viewer:



6. You can click the **View** tab to see a graph view of the model. For example, the image below shows a hierarchical view of the model.



Using the Toolkit in a Data Layer

To invoke the Data Toolkit service in a data layer, add a **Query Step** to the layer. When writing the query to run in the Query Step, include a SERVICE call that specifies the remote endpoint and defines the data to retrieve from the endpoint. For instructions on creating a Query Step, see Running a Query. For details about writing data toolkit SERVICE statements, see Writing a Data Toolkit SERVICE Query below.

Writing a Data Toolkit SERVICE Query

The following WHERE clause syntax can be used as a template for writing queries with SERVICE calls that invoke the Data Toolkit service:

```
WHERE
{
  [ SERVICE TOPDOWN <http://openanzo.org/localsparql>
    { ]
      SERVICE <http://cambridgesemantics.com/services/DataToolkit>
        {
          triple_patterns
        }
  [ } ]
}
```

The SERVICE TOPDOWN clause is optional. It typically calls the local Anzo SPARQL endpoint, and it indicates that the rest of the query produces values that are input to the SERVICE call. When TOPDOWN is excluded, the query is run "bottom up;" the SERVICE clause is run first, and those results become input to the rest of the query. In general, if the SERVICE call needs input based on the results from the rest of the query, it is a TOPDOWN query.

Top Down Query Scenarios

The list below describes two scenarios for using a top-down query:

- A graphmart contains data about a company's office locations. A user wants to query the graphmart to return all of the office locations and then send those results to a service to retrieve the current temperature for each location.
- A graphmart contains data about employees. A user wants to query the graphmart to return the top 10 employees and then send the results to a service to retrieve those employees' personal information from an LDAP server. If the query excluded the TOPDOWN clause, the service would be called first and return the LDAP records for every employee.

Bottom Up Query Scenario

This example describes a scenario for using a bottom-up query: A graphmart contains data about a company's office locations. A user wants to query a service to find the city with highest income and then use that result to query the graphmart and see if the company has an office in that city.

Examples

The query below invokes the Data Toolkit service to retrieve weather forecast data for specific cities from Dark Sky API.

```
PREFIX s: <http://cambridgesemantics.com/ontologies/DataToolkit#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX anzo: <http://openanzo.org/ontologies/2008/07/Anzo#>
PREFIX zowl: <http://openanzo.org/ontologies/2009/05/AnzoOwl#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT
  ?name ?state ?latitude ?longitude ?time ?summary ?temperature ?rainChance
  ?rainIntensity ?nearestStormDistance ?nearestStormBearing ?visibility
WHERE
  {
    SERVICE <http://cambridgesemantics.com/services/DataToolkit>
    {
      ?data a s:HttpSource ;
      s:url "https://api.darksky.net/forecast/bdbe3f638eb908c9b94919537dad5945/
{{?latitude}},{{?longitude}}" ;
      s:selector "currently" ;
      ?time () ;
      ?summary () ;
      ?temperature () ;
      ?rainChance ( "precipProbability" ) ;
      ?rainIntensity ( "precipIntensity" ) ;
      ?nearestStormDistance () ;
      ?nearestStormBearing () ;
      ?visibility () .
    }
  }
  VALUES( ?name ?state ?latitude ?longitude )
  {
    ( "Lakeway" "TX" 30.374563 -97.975892 )
    ( "Boston" "MA" 42.358043 -71.060415 )
    ( "Seatle" "WA" 47.590720 -122.307053 )
    ( "Chicago" "IL" 41.837741 -87.823296 )
    ( "Hilo" "HI" 19.702040 -155.090312 )
  }
```

Additional examples are in progress. The Data Toolkit plugin JAR file, **com.cambridgesemantics.anzo.datatoolkit.jar**, in *Anzo_install_dir*/Server/plugins/ contains several example queries.

Related Topics

Running a Query

Accessing and Analyzing Data

Once data has been onboarded, modeled, and blended into the dataset catalog and graphmarts, users have several options for accessing and analyzing the data. Anzo provides the Hi-Res Analytics application where users can create dashboards for exploring and visualizing the data without needing to have specialized query knowledge. The Query Builder in the user interface enables users to find specific statements or write and run SPARQL queries. Users can also access data remotely from the Anzo SPARQL endpoint, HTTP client interface, or by using the Data on Demand service to generate data feeds for third-party business intelligence tools. The topics in this section provide information about the ways to access data in Anzo.

- Analyzing Data with Hi-Res Analytics
- Accessing Data with the Query Builder
- Accessing Data from Data on Demand Endpoints
- Accessing Data via the SPARQL Endpoint
- Accessing Data via the HTTP Client Interface
- SPARQL Query Templates and Best Practices

Analyzing Data with Hi-Res Analytics

Anzo enables business users to ask and answer both ad-hoc and pre-determined questions using custom user dashboards. Automated query generation eliminates the need to have specialized query knowledge. Users can traverse even the most complicated multi-dimensional data to build exploratory charts, filters, tables, and network views.

The topics in this section provide guidance on getting started with Hi-Res Analytics and include instructions for creating and modifying dashboards and dashboard components. This section also includes reference information about the available filters and lenses as well as the supported functions you can use for calculating the values to display on dashboards.

- Introduction to Hi-Res Analytics
- Getting Started: Exploring and Visualizing Data
- Creating a Dashboard
- Creating a Lens
- Creating a Dashboard Filter
- Combining Data from Multiple Classes
- Calculating Values in Filters and Lenses
- Managing Dashboard and Lens Security
- Searching for Text in Unstructured Documents
- Exporting a Lens

- Deleting a Lens
- Supported Functions and Formulas
- Filter Type Reference
- Lens Type Reference

Introduction to Hi-Res Analytics

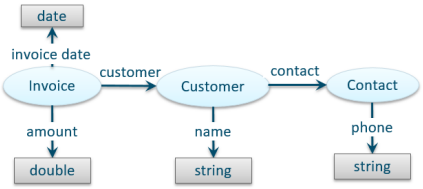
Anzo Hi-Res Analytics enables you to define and create visual data representations using the latest in powerful web technologies. This introduction defines the fundamental concepts of working with Hi-Res Analytics.

To fully leverage the advanced capabilities of Hi-Res Analytics, it helps to have skills working with Excel functions and formulas, SPARQL, and JavaScript and HTML. You can use Hi-Res Analytics without these skills but may not be able to take advantage of all functions.

- Concepts and Vocabulary
- General Interface Elements
- Dashboard Interface

Concepts and Vocabulary

Term	Description
Dashboard	Dashboards enable you to view, edit, and share your data. You view data through lenses, such as tables, charts, or web pages, which format the data for display. You can apply filters to dashboards to refine the results. There are two types of dashboards: **Dashboard** and **Graphmart dashboard**. The type that you choose depends on whether you want to display data that is stored in a local Anzo volume or AnzoGraph. Select the Dashboard type when working with data in a local Anzo volume, or select Graphmart dashboard when working with graphmarts stored in AnzoGraph. For more information, see Creating a Dashboard.
Data Layer	A graphmart can have any number of data layers that load additional data sets, mask certain data, infer new data, or create, clean, conform, or transform data. Users can choose to include or exclude the data from certain layers when creating or viewing Hi-Res Analytics. For more information, see Introduction to Data Layers.
Lens	Lenses are the structures that display your data. You must have at least one lens to view any of your data. You can reuse existing lenses or create new ones. For more information, see Creating a Lens.

Term	Description
Filter	Filters narrow and further define the data to display. Dashboard-level filters apply globally to all lenses in a dashboard. Lens-level filters apply only to a specific lens. You can also create subfilters to refine data based on additional criteria. For more information, see Creating a Dashboard Filter.
Property	A data property contains instances that can consist of different data types. The data types determine functional aspects within Hi-Res Analytics. For example, certain filters act only on dates or numbers. Relative paths are transitional elements that point you to another class.
Path	Paths are sequences of properties in an ontology that lead to certain values. For example, in an invoice you can find a phone number for the invoiced customer by following the customer > contact > phone path: 
Functions and formulas	Each lens can use functions and formulas to determine what data is presented. Available functions depend on the property's data type. For more information, see Calculating Values in Filters and Lenses.

General Interface Elements

This section describes the general interface elements in the Hi-Res Analytics application.

Windows

Objects such as lenses and filters use separate windows so that you can configure individual dashboard objects. Close windows by moving the pointer over them to display the X in the top right corner.

Tabs

Hi-Res Analytics uses dashboard-level and object-level tabs. You can open multiple dashboards or objects at once and the tabs allow for navigation.

Object-level tabs control navigation between open lenses and filters in the center pane. All open lenses and filters with orientation set to Center appear as tabs in the center pane, under the appropriate dashboard tab. By default, a new lens appears in the center pane. New filters appear in the left pane.

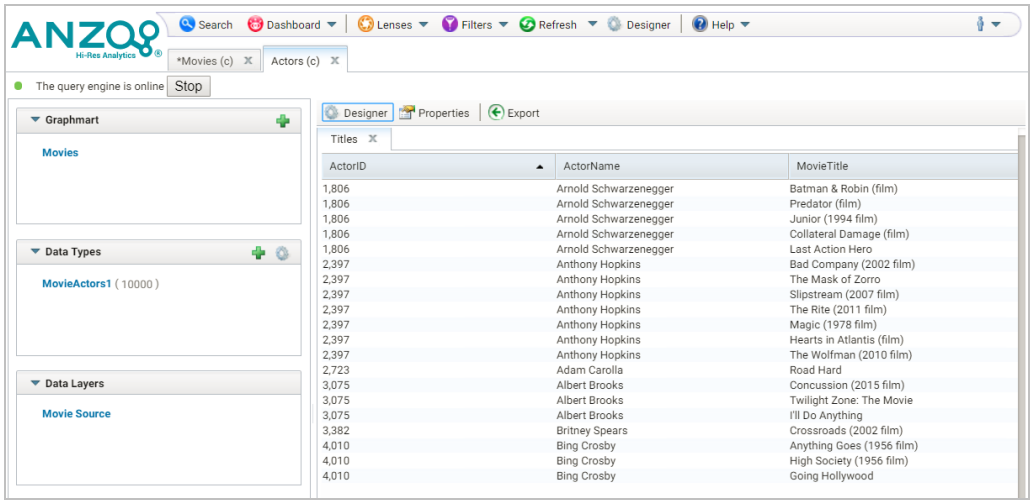
When you change a dashboard, an asterisk appears on the dashboard name tab. Save the dashboard to preserve the changes.

Designer

Designer windows allow initial and further configuration of dashboards, lenses, and filters. Click the cog icon ( ) in the main toolbar to open the Designer for a dashboard, or click the cog icon in an object window to open the Designer for that object. With some exceptions, all settings available during creation are available for reconfiguration.

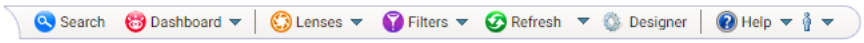
Dashboard Interface

The images in this section show the administrator views. Some options are not available to users with lower permission levels.



Main Toolbar

The toolbar at the top of the screen provides the following options:

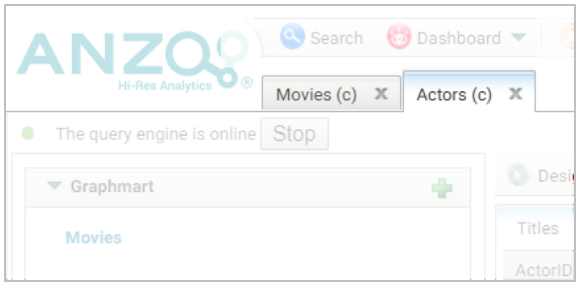


- **Search**: Enables you to search for dashboard-level objects, such as linked data sets, dashboards, and system data sets.
- **Dashboard**: Accesses dashboard functions, including Save.
- **Lenses**: Creates, opens lenses.
- **Filters**: Creates or manages selected filters.
- **Refresh**: Accesses the automatic refresh check box. Select this box to refresh data automatically. New data will appear and change according to changes elsewhere.
- **Designer**: Controls dashboard layout and design.

- **Help**: Opens help options.
- **User**: Click to sign out as current user.

Dashboard Tabs

The dashboard tabs under the main toolbar display the open dashboards and enable you to click to view different dashboards.

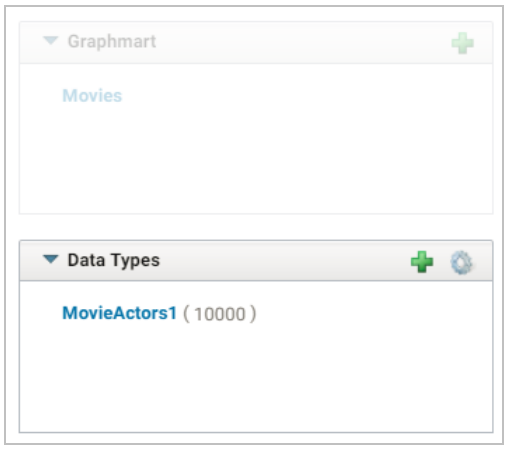


Graphmart

The Graphmart panel displays the selected graphmart for the dashboard.

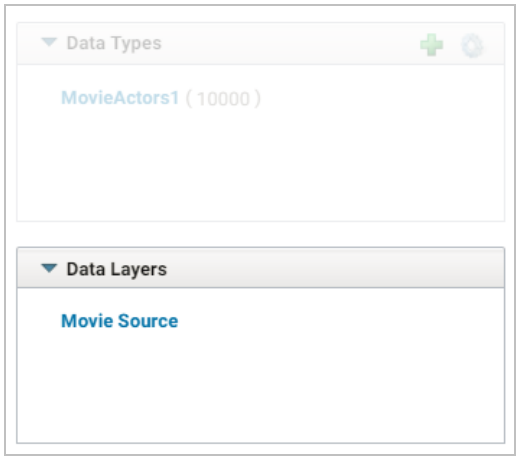
Data Types

The Data Types panel displays the selected data types for the dashboard's graphmart.



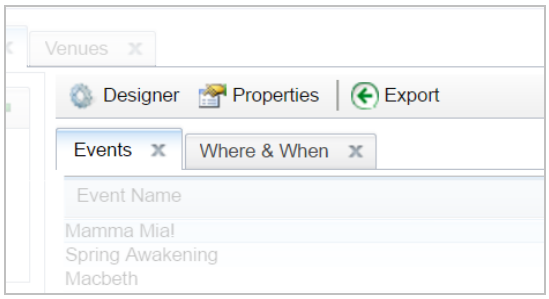
Data Layers

The Data Layers panel displays the data layers for the dashboard's graphmart.



Object Toolbar and Tabs

The object toolbar and tabs enable you to manage the lenses and filters in the selected dashboard. The tabs display the open objects, and the toolbar enables you work with the object properties.



Filters

By default, filters that you create appear in the left column of the dashboard.

Related Topics

Getting Started: Exploring and Visualizing Data

Creating a Dashboard

Creating a Lens

Creating a Dashboard Filter

Combining Data from Multiple Classes

Getting Started: Exploring and Visualizing Data

When you start to build a new dashboard, you might not know what data exists in the knowledgebase, which values in that data you ultimately want to display, and the most pertinent way to visualize the results. This topic introduces the available lenses and filters and provides guidance on getting started by using the Anzo Hi-Res Analytics tools to perform data discovery. By experimenting with simple objects, you can explore the data, determine which questions you want to answer, and start to visualize the end result.

To get started:

1. Create a New Dashboard
2. Explore the Data
3. Create Visualizations of the Data

Create a New Dashboard

1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**.
2. On the Graphmarts screen, click the name of the graphmart for which you want to create a dashboard.
3. On the Graphmart details screen, click the **Create Dashboard** button. Hi-Res Analytics opens and displays the New Dashboard dialog box.
4. Type a **Title** for the dashboard and enter an optional **Description**.
5. Leave the default **Graphmart dashboard** value in the Type field and then click **OK**.
6. In the main toolbar, click the **Dashboard** button and select **Save**.

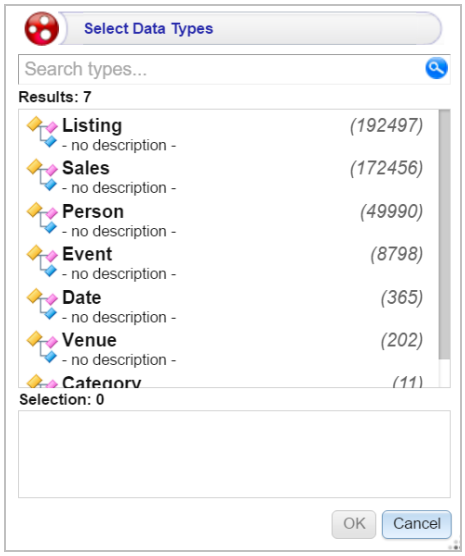
Tip You might want to create multiple dashboards so that you can click between dashboards and view multiple classes of data at the same time.

Explore the Data

Once you create a new dashboard, you can experiment with Hi-Res Analytics objects and tools to get to know the data and decide the best way to display it.

Decide What Type (Class) of Data You Want to See

First, review the types of data or classes that exist in the data set: on the What can I do next? tab, click **Specify the types of data you would like to see**. The Select Data Types dialog box displays the available data types. The value in parentheses shows the total number of instances of that type exist in the data set:



Choose one data type. The property that you choose determines the fields that become available to filter on.

Note Though you must choose one base data type for a dashboard, you can leverage the relationships in the graph to access and integrate data from additional classes. See Combining Data from Multiple Classes for more information.

To explore the properties that relate to your selection, select a data type and then click **OK** to close the Select Data Types dialog box. In the main toolbar, click **Filters** > **Create a Filter**. In the Create Filter dialog box, click the **Fields** field and review the properties that are available. You can repeat this process to review all of the related properties for each data type.

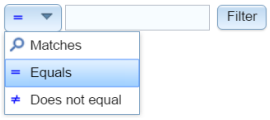
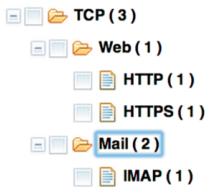
Create Filters to See the Values for Properties

To dive deeper into the data and quickly determine what values exist for the class of properties you selected, you can start adding filters to the dashboard. Filters reveal the values associated with fields and help you learn the data set specifics such as whether data exists for certain properties and whether the data includes many duplicate or unique values. Learning more about the details enables you to start making decisions about what properties to group on, for example, what properties have relationships, and what results you want to visualize on the dashboard.

To create a filter, click **Filters** in the main toolbar and select **Create a Filter**. In the Create Filter dialog box, click the **Fields** field and select the property or property path to filter on. Then click the **Filter** field to select the filter type.

The list of available choices depends on the data type of the property you selected in Fields. The table below describes each filter type. For more information about the filters, see Filter Type Reference.

Filter	Description
Cloud Armenia Australia Britain Briton Canada China Czech Republic Denmark England Finland France Germany Hungary Iran Ireland Israel Italy Japan Korea Mexico Netherlands New Zealand Poland Portugal Romania Russia Scotland Senegal South Africa South Korea 37 choices | more	Cloud filters display values in term clouds where each term is written in a font size that represents the number of results for that value. Unlike list filters, which enable you to select and filter on multiple values at once, cloud filters allow you to filter on one value at a time.
List ☑ Wales (10) ☑ United States (614) ☑ United Kingdom (61) ☑ Turkey (1) ☐ Taiwan (2)	List filters display selections in a list and allow you to select and filter on multiple values at a time. The number in parentheses indicates the total number of results for that value.
Single Select List Wales (10) United States (614) United Kingdom (61) Turkey (1) Taiwan (2)	Single Select List filters are similar to List filters but, like Cloud filters, only allow you to select and filter on one value at a time.
Limit Include the 10 Largest Largest Smallest	Limit filters enable you to limit the results to the specified number of largest or smallest values. You can use limit filters for any data type. For strings, results are ordered alphabetically. Largest orders by the last letters in the alphabet and Smallest orders by the first letters in the alphabet.
Date Range 12/31/2007 4:00 PM - 2/29/2008 4:00 PM (1489) 2/29/2008 4:00 PM - 4/29/2008 4:00 PM (1459) 4/29/2008 4:00 PM - 6/28/2008 4:00 PM (1416) 6/28/2008 4:00 PM - 8/27/2008 4:00 PM (1416)	Date Range filters are available for date and time data types and enable you to define date ranges and group the results into those ranges.
Numeric Range 1 - 3 (132605) 3 - 5 (39190) 5 - 7 (555) 7 - 9 (106)	Numeric Range filters are similar to Date Range filters but are available for numeric data types and enable you to define numeric ranges and group the results into those ranges.

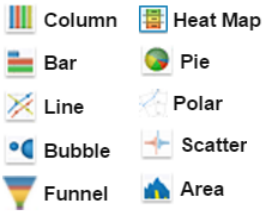
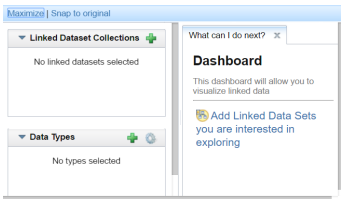
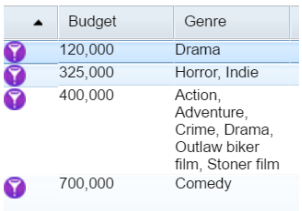
Filter	Description
Range Slider  Min: 2/17/1765 Max: 8/25/1994	Range Slider filters display a slider control that enables you to filter results by a range that you specify by setting a minimum and maximum value.
Relative Time  Last years	Relative Time filters enable you to search for records that fall into the specified time increment. You can filter on increments from years down to milliseconds.
Search  Filter Matches Equals Does not equal	Search filters enable you to filter for records that contain a partial match, exact match, or do not equal the value that you specify.
Presence Exists (910) Does not exist (238)	Presence filters indicate whether a specified value exists. This filter is useful for finding records that exclude a particular value. Presence filters are available for relative paths and properties of all data types.
Hierarchy  TCP (3) Web (1) HTTP (1) HTTPS (1) Mail (2) IMAP (1)	Hierarchy filters enable you to view parent and child relationships and filter data based on those relationships. Hierarchy filters are only available for relative paths, indicated by a path icon ( ) in the Create Filter dialog box.
Types  Movie Person (141) Director (141)	Types filters enable you to filter on each type of child property that is related to the specified parent property. Types filters are only available for relative paths, indicated by a path icon ( ) in the Create Filter dialog box.

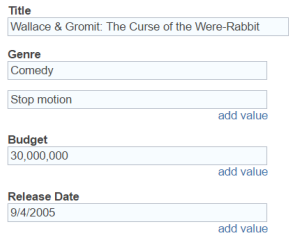
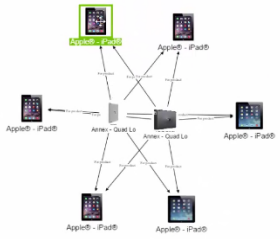
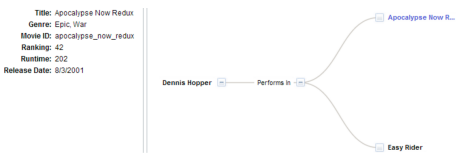
Create Visualizations of the Data

Once you have a good understanding of the values and relationships that exist in the data set, you can experiment with the Hi-Res Analytics lenses and decide on the most appropriate way to display the data. Creating a Table lens is a quick way to view the data that you filtered. This section provides instructions for creating a table lens and describes each of the lenses available in Hi-Res Analytics.

To create a table lens, click **Lenses** in the main toolbar and select New. In the Create Lens dialog box, select the **Table** lens and click **Next**. Type a **Title** for the lens, and then click **Finish**. The Designer window opens. In the Designer, click the **Auto-generate columns** icon ( ) to add all available columns to the table. Then click **Save**. The new lens displays as a new sub-tab on the dashboard and displays the data according to the data type and filter or filters that you created. Now that you can view a summary of the data in a table, it can help you determine how to further narrow or expand the results by adding, changing, or removing filters. In addition, you can experiment by adding other lenses to the dashboard to find the ideal way to display the data to answer the questions that you have.

The table below describes each type of lens. For more information about each lens, see Lens Type Reference.

Lens	Description
AnzoKO Web Page	AnzoKO Web Page lenses include the Knockout JavaScript framework and enable you to display data on a web page that you create using HTML, CSS, and JavaScript.
Chart 	Anzo offers several types of Chart lenses. These lenses are useful for displaying large amounts of complex data and have the widest format range of any lens type. The ability to add an axis enables you to compare data, such as for comparing monthly sales data for multiple stores.
Dashboard 	Dashboard lenses display a dashboard within a dashboard.
Drill Down 	Drill Down lenses create clickable data points that enable you to drill down to view additional details. You can specify mutliple lenses within the Drill Down lens so that clicking a data point presents the data in a different view.

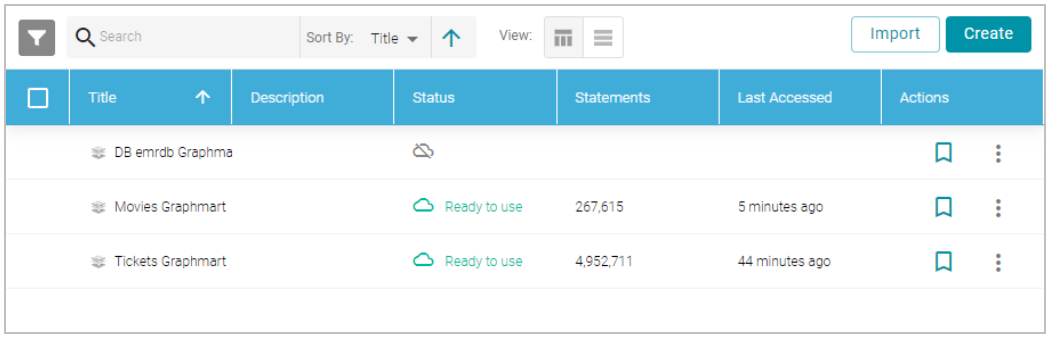
Lens	Description
Form 	Form lenses create an interactive form that enables you to drag and drop and configure objects into a widget that appears as part of your dashboard. You can also nest objects in other objects.
List 	List lenses display results in a list layout, similar to the Microsoft Windows® Explorer interface. The lens enables you to add icons for each data value, and results are grouped onto pages according to the Page Size value that you specify.
Network Navigator 	Network Navigator lenses provide interactive graph visualizations for viewing and exploring relationships across your entire network of data. The lens enables you to quickly generate a standard graph or hierarchical view of the data and then customize the visualization to target the relationships and information that interests you.
Query	Query lenses enable you to retrieve data using a custom SPARQL query and display the results by writing basic HTML and CSS. You can use a Query lens to access data from external sources. Query lenses do not bind directly to the linked data set, data type, or filters defined on the dashboard.
Resource Tree Navigator 	Resource Tree Navigator lenses display results in a hierarchical tree view. You can click parent data points to open the successive child data points. This lens is useful for presenting small amounts of data; each discrete group appears on a separate page in the dashboard. You can also click certain objects to view the object's data properties in the left panel.

Lens	Description
Table  Event Name ▲ | Date | Venue .38 Special | 9/10/2008 5:00 PM | RBC Center .38 Special | 8/26/2008 5:00 PM | Nationwide Arena .38 Special | 1/20/2008 4:00 PM | Shea Stadium .38 Special | 5/29/2008 5:00 PM | Quicken Loans Arena .38 Special | 4/28/2008 5:00 PM | US Airways Center .38 Special | 12/10/2008 4:00 PM | Turner Field .38 Special | 5/4/2008 5:00 PM | Qwest Field	Table lenses present results in a basic table grid consisting of rows and columns. Table lenses are useful for presenting data aggregates or summaries.
Web Page	Web Page lenses enable you to display results on a web page that you create using HTML, CSS, and JavaScript.

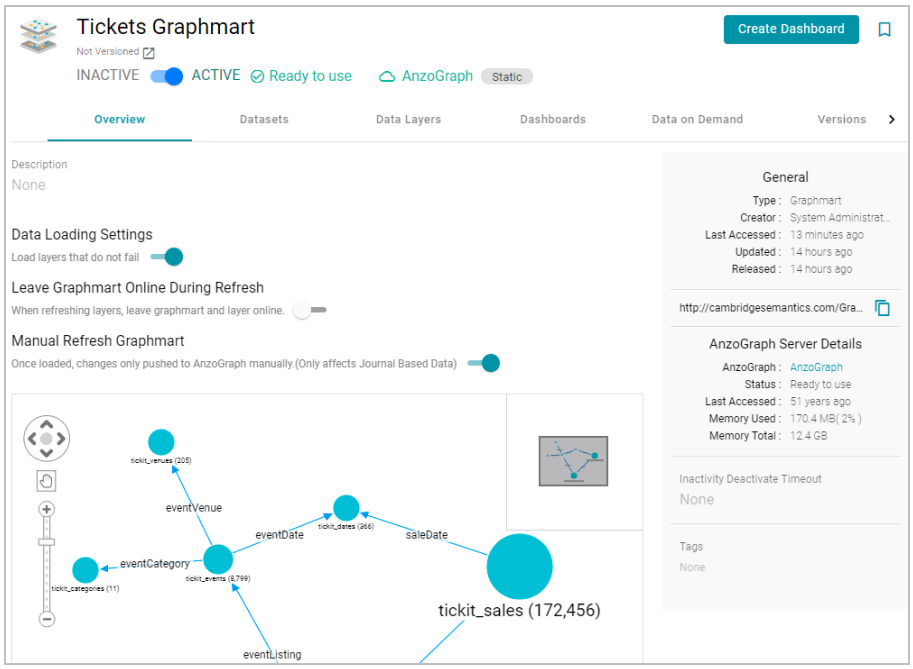
Creating a Dashboard

This page provides instructions for creating a new Hi-Res Analytic dashboard for a graphmart.

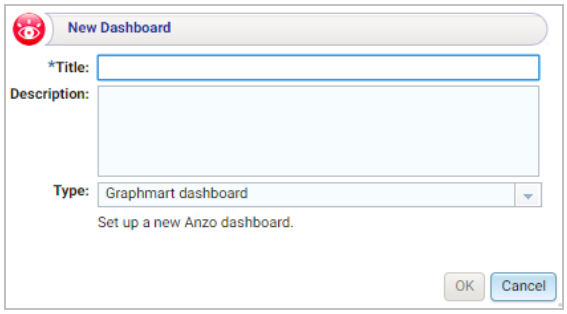
1. In the Anzo application, expand the **Blend** menu and click **Graphmarts**. Anzo displays a list of the existing graphmarts. For example:



2. On the Graphmarts screen, click the name of the graphmart for which you want to create a dashboard. Anzo displays the details for the graphmart. For example:



3. On the Graphmart overview screen, click the **Create Dashboard** button. The Hi-Res Analytics application opens and displays the New Dashboard dialog box.



4. Type a **Title** for the dashboard and enter an optional **Description**.
5. Leave the default **Graphmart dashboard** value in the **Type** field and then click **OK**.

   The new dashboard appears as a new tab on the screen and contains a sub-tab titled **What can I do next?**. This tab acts as a wizard to guide you through the initial dashboard creation.

6. On the What can I do next? tab, click **Specify the type of data you would like to see**. The Select Data Types dialog box appears.
7. In the Select Data Types dialog box, select the data type or class of data that you want to display on the dashboard. Anzo uses the type, along with any filters, to populate the visualizations (lenses) that you add to the dashboard.

   Note Though you must choose one base data type for a dashboard, you can leverage the relationships in the graph to access and integrate data from additional classes. See Combining Data from Multiple Classes for more information.

8. Click **OK** to close the Select Data Types dialog box. The data type that you chose appears in the Data Types panel on the left side of the dashboard.
9. In the main toolbar, click the **Dashboard** button and select **Save**.

Now that the dashboard basics are defined, see Creating a Lens and Creating a Dashboard Filter for instructions on adding lenses and filters to the dashboard.

Related Topics

Combining Data from Multiple Classes

Managing Dashboard and Lens Security

Creating a Lens

Creating a Dashboard Filter

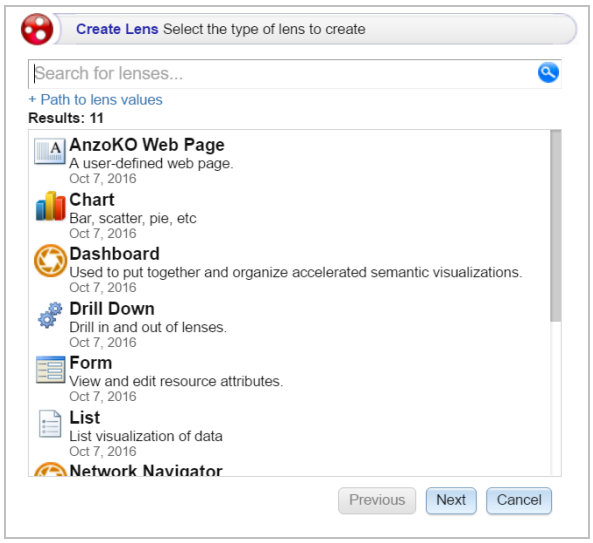
Creating a Lens

Lenses define the data's visual presentation. Each type of lens represents a unique method for displaying data. For instance, in a column chart, you can present multiple data series for comparison. You can also apply custom formats such as fonts and colors to any lens. This section provides instructions for creating and formatting a new lens.

- Creating a New Lens
- Cloning a Lens

Creating a New Lens

1. Open the dashboard that you want to add a lens to.
2. In the Hi-Res Analytics main toolbar, click **Lenses** and select **New**. The Create Lens window opens.



3. Determine the type of lens that you want to create. The following list describes each lens type. For more information about each lens type, see Lens Type Reference.
    - **AnzoKO Web Page**: Includes the Knockout JavaScript framework and displays data on a web page that you create using HTML, CSS, and JavaScript.
    - **Chart**: Displays results in rectangular columns, 3D bubbles, scatter charts, heat maps, or other chart types. A chart lens has the widest format range of any lens type. These lenses are useful for displaying large amounts of complex data. The ability to add an axis enables you to compare data, such as comparing monthly sales data for multiple stores.
    - **Dashboard**: Dashboard lenses display a dashboard within a dashboard.
    - **Drill Down**: Creates clickable data points that enable you to drill down to view additional details. You can specify multiple lenses within the Drill Down lens so that clicking a data point presents the data in a different view. When you select a Drill Down lens, the Designer opens and prompts you to select the lenses that you want to use for the drill down functionality. The top lens in the Designer becomes the primary lens with the clickable data points. You can drag lenses to re-order them. You then separately configure each of the lenses that you selected. There is no further configuration for the drill down.
    - **Form**: Enables you to create an editable or read-only form on the dashboard. Form lenses can be useful for displaying many details about a record. Form lenses are read-only when used with graphmarts.
    - **List**: Displays results as icons in a folder view, similar to the Microsoft Windows® Explorer interface. List lenses enable you to add images for each data value.
    - **Network Navigator**: Provides interactive graph visualizations for viewing and exploring relationships across your entire network of data. These lenses enable you to quickly generate a standard graph or hierarchical view of the data and then customize the visualization to target the relationships and information that interests you.
    - **Query**: Retrieve data using a custom SPARQL query and display the results by writing basic HTML and CSS. You can use a Query lens to access data from external sources. Query lenses do not bind directly to the data set, data type, or filters defined on the dashboard.
    - **Resource Tree Navigator**: Displays results in a hierarchical tree view. Click parent data points to open the successive child data points. You can also click certain objects to view the object's data properties in the left panel. This lens is useful for presenting small amounts of hierarchical data.
    - **Table**: Presents results in a basic table grid consisting of rows and columns. Table lenses are useful for presenting data aggregates or summaries.
    - **Web Page**: Displays results on a custom web page that you create using HTML, CSS, and JavaScript.
4. On the Create Lens dialog box, select the type of lens that you want to add to the dashboard, and then click **Next**.
5. Type a **Title** and optional **Description** for the lens.
6. Click **Finish**. The lens Designer dialog box opens to enable you to configure the lens.

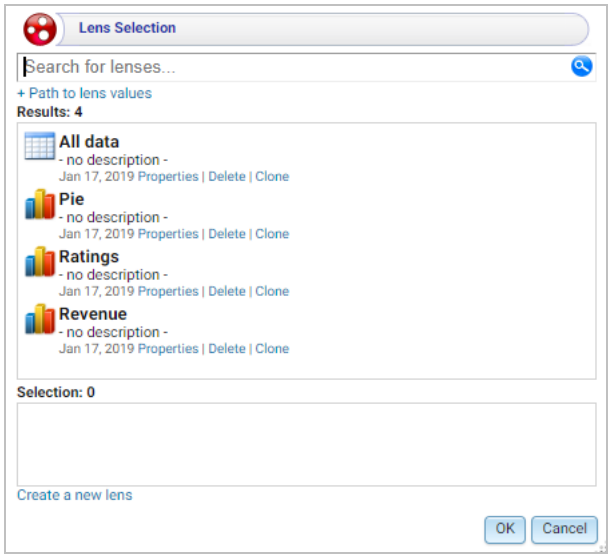
7. Configure the lens and then click **Save** to save the configuration and add the lens to the dashboard. For information about using formulas to compute the values to display in the lens, see Calculating Values in Filters and Lenses.

Cloning a Lens

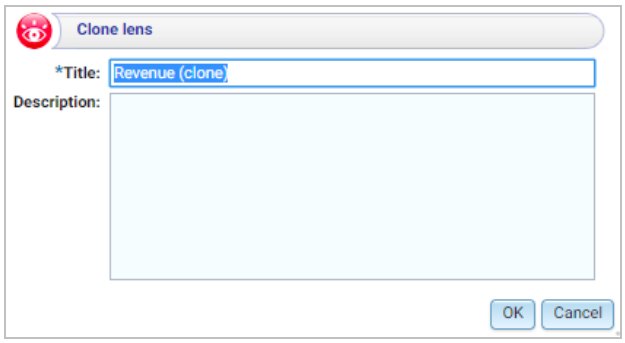
Cloning a lens makes a copy of the lens that can be changed without affecting the original lens or other dashboards.

Note You can only clone lenses from dashboards that you have permission to modify. If you open a dashboard with read-only access, the Open Lens and Clone options are not available. To clone a lens from a read-only dashboard, save a copy of the dashboard so that you become the owner. To save a copy, click the **Dashboard** button in the main Hi-Res Analytics toolbar and select **Save As**. Then follow the procedure below to clone a lens into the dashboard that you own.

1. Open a dashboard in the Hi-Res Analytics application, then click **Lenses** in the main toolbar and select **Open**. Anzo opens the Lens Selection dialog box, which lists the lenses that are available to open. For example:



2. Click the **Clone** link for the lens that you want to clone. Anzo displays the Clone lens dialog box, and populates the Title field with the existing lens name and "(clone)." For example:



3. Modify the **Title** to name the new copy of the lens, and add or change the **Description** if necessary. Then click **OK**.
4. Anzo adds the new copy of the lens to the Lens Selection dialog box and selects it. Click **OK** to add the lens to the dashboard.

Related Topics

Combining Data from Multiple Classes

Managing Dashboard and Lens Security

Exporting a Lens

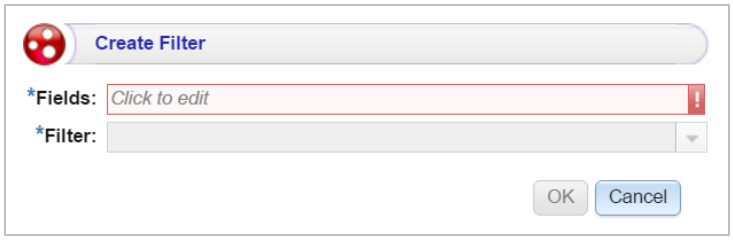
Deleting a Lens

Creating a Dashboard Filter

Filters narrow the data presented in a dashboard. You can define filter criteria using Microsoft Excel-like functions such as AVG, SUM, or UPPER, or groupings such as a date range or aggregation. Though you can also filter data in some lens objects, such as a column, a dashboard filter applies across the entire dashboard. When you add a filter, all lenses on the dashboard update simultaneously based on your filter selection. Unlike lenses, filters cannot be shared by other users or dashboards and must be created for each dashboard.

Follow the instructions below to create a filter.

1. Open the dashboard that you want to add a filter to.
2. In the Hi-Res Analytics application main toolbar, click **Filters** and select **Create a filter**. The Create Filter dialog box opens.



3. Click in the **Fields** field to open the drop-down list and select a property or relative path to filter on. The list of available properties depends on the selected data type for the dashboard. The list below describes the icons and options that are available when choosing a property:
   - The join symbol ( ) denotes a property that is not linked to another class. A path icon ( ) denotes a path; the property is linked to another class. Paths specify what data to display relative to the selected property. Selecting a property that is linked to another class navigates to the next class so that you can select one of the linked properties.
   - When you navigate to a new property, the breadcrumbs at the top of the dialog box show you the property path. You can click the eraser icon ( ) to clear the path.

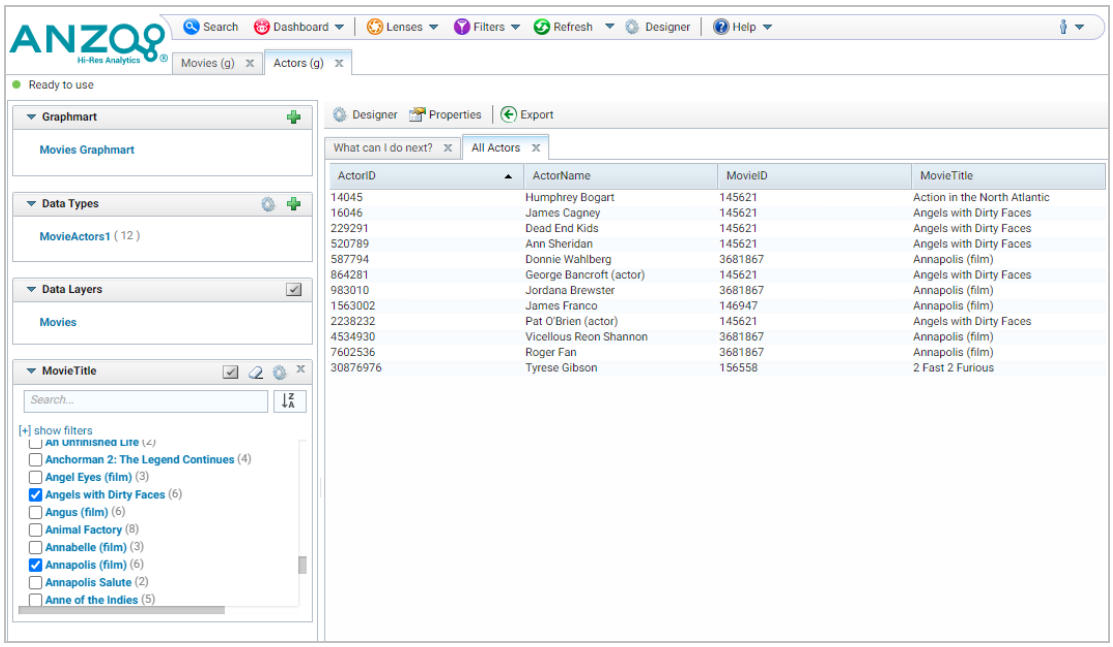
- If you want to use a function or formula to determine the values that the filter returns, click the function button () and select a function from the drop-down list. The functions that become available in the list depend on the data type of the selected property. To choose a more advanced function or type a formula, click **Advanced**. The Calculated Value dialog box opens and enables you to choose additional properties and functions as well as type your own calculation. For more information, see Calculating Values in Filters and Lenses.

4. When you have selected a Field for the filter, click the **Close** link to close the list and return to the Create Filter dialog box. Additional options are displayed on the screen.
5. Click the **Filter** field and select the type of filter to create. The list of available filters depends on the data type of the property selected in Fields. For example, the Date Range filter only appears for date properties. The list below provides a short description of each filter type. For more information about the filter types, see Filter Type Reference.
    - **Cloud**: Cloud filters display values in term clouds where each term is written in a font size that represents the number of results for that value. Unlike list filters, which enable you to select and filter on multiple values at once, cloud filters allow you to filter on one value at a time. The cloud filter is available for all data types but cannot be with used relative paths, which are indicated by a path icon () in the Create Filter dialog box.
    - **Date Range**: Date Range filters enable you to define date ranges and group the results into those ranges. Date Range filters are available for properties with date and time data types.
    - **Hierarchy**: Hierarchy filters data into hierarchical categories to display parent and child relationships. Hierarchy filters are available only for relative paths (indicated by the path icon  in the Create Filter dialog box) and not properties.
    - **Limit**: Limit filters enable you to limit the results to the specified number of largest or smallest values. You can use limit filters for any data type. For strings, results are ordered alphabetically. Largest orders by the last letters in the alphabet and Smallest orders by the first letters in the alphabet.
    - **List**: List filters display results in a list of distinct values and allow you to select and filter on multiple values at a time. The list filter is available for properties of all data types.
    - **Numeric Range**: Numeric Range filters enable you to define numeric ranges and group the results into those ranges. Numeric Range filters are similar to Date Range filters but are available for properties with numeric (integer or double) data types. You can also perform a function on a property so that it results in a number value, such as using the COUNT function.
    - **Presence**: Presence filters indicate the presence or absence of a selected property. Presence filters are useful for finding records that exclude a particular value. They are available for relative paths and properties of all data types.

- **Quartile**: Quartile filters group and rank the values for a property into four equal ranges. This filter requires a property with a numeric or date data type and is not available for relative paths.
- **Range Slider**: Range Slider filters display a slider control that enables you to filter results by a range that you specify by setting a minimum and maximum value. The Range Slider filter requires a property with numeric or date data type, or a function resulting in a number, such as COUNT
- **Relative Time**: Relative Time filters enable you to filter for records that fall into the specified time increment relative to the current time. Relative Time filters are available for properties with date data types.
- **Search**: Search filters are available for all data types and enable you to search for values in the selected property. For unstructured data, use the Full Text Search filter.
- **Single Select List**: Single Select List filters are similar to List filters but only allow you to select and filter on one value from the list at a time. This filter is available for properties of all data types but is not available for relative paths.
- **Types**: Types filters enable you to filter data according to the classes defined by a relative path. This filter is available only for relative paths (indicated by the path icon  in the Create Filter dialog box) and not properties.

6. (Optional) In Filter Properties, add a **Title**. If you do not type a title, Anzo uses the property or path as the title.
7. (Optional) Modify additional filter options as needed. Depending on the selected property and filter types that you selected, one or more of these options are available for configuration:
   - **Label Field**: The property to show as the value for each list item in the filter if you want it to differ from the value that results from the property or relative path you chose in the **Fields** field.
   - **Exclude**: Removes the selected property from the results.
   - **Show Bars**: Displays the total values for the selected property as a bar graphic in the background of the filter.
   - **Show Blanks**: Displays any null values for the selected property by including a “Blank” option in the filter.
   - **Show counts**: Indicates whether the results of this filter change based on selections in other filters on the dashboard.
   - **Respond to other filters**: Displays the number of results for the value.
   - **Interval Unit**: Defines the unit of time for the Interval value.
   - **Interval**: Defines the length of time in each grouping. For example, for a date field with an Interval Unit of "Decade," an Interval value of 2 creates groups of two-decade increments.
   - **Format Type**: Enables you to change the format type for a date or number property.
   - **Create Filter**: Enables you to specify a subfilter to further refine the results in the filter.
8. When you finish configuring the filter, click **OK**. The new filter appears in the left-hand column of the dashboard and displays the values that are available for filtering the displayed data. For example, the image below shows

a List filter for a dashboard that displays data about movie actors. The property for the filter is MovieTitle and selecting one or more titles in the filter narrows the scope of the dashboard to the actors who star in the selected movies.



9. Save the dashboard to save the filter.

Related Topics

Creating a Lens

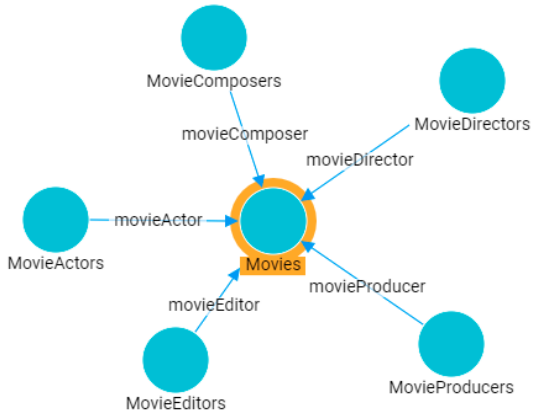
Combining Data from Multiple Classes

Filter Type Reference

Combining Data from Multiple Classes

Though you must choose one base data type (or class) for each Hi-Res Analytics dashboard, selecting a data type with connections to other classes enables you to configure lenses and filters that combine the data from those classes. This powerful capability can help surface the semantic relationships in your data and enable you to leverage those relationships to access and integrate all of the data in the graph. When choosing the base data type for a dashboard, it helps to consider all of the desired filters.

For example, consider the following data model for a movie data set:

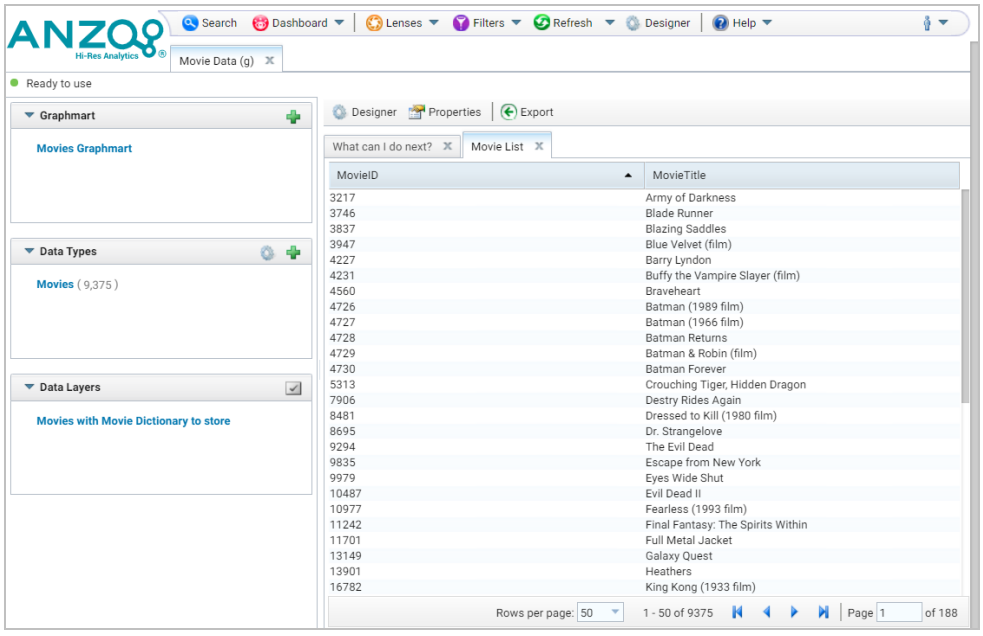


By creating a dashboard that specifies **Movies** as the base data type, the lenses and filters in the dashboard can navigate the paths to properties in the other classes, MovieActors, MovieDirectors, MovieComposers, and so on. This topic provides guidance on accessing data from multiple classes in filters and lenses.

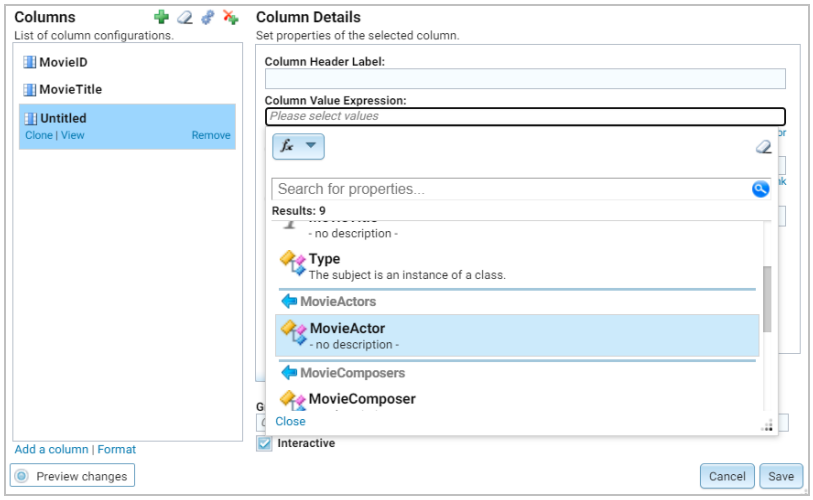
- Combining Classes in a Lens
- Filtering on Multiple Classes

Combining Classes in a Lens

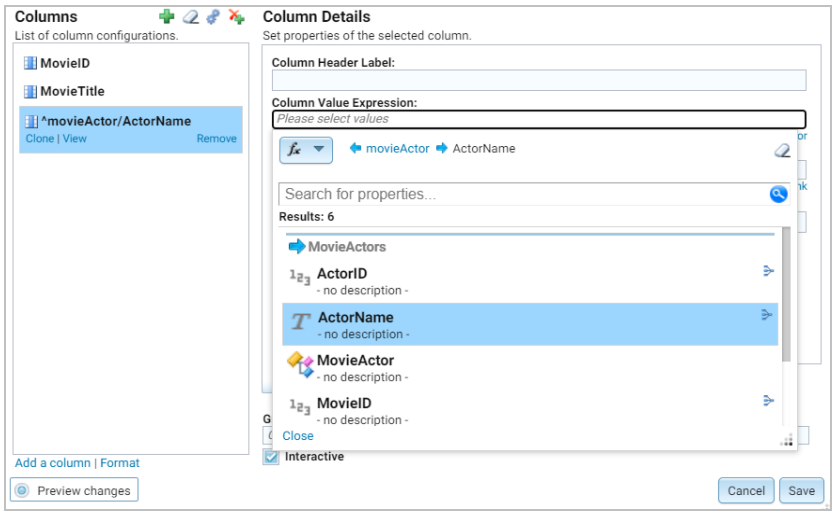
The image below shows a dashboard that accesses the graph for the above model. The specified Data Type is **Movies**, and a table lens displays all of the columns/properties in the Movies class:



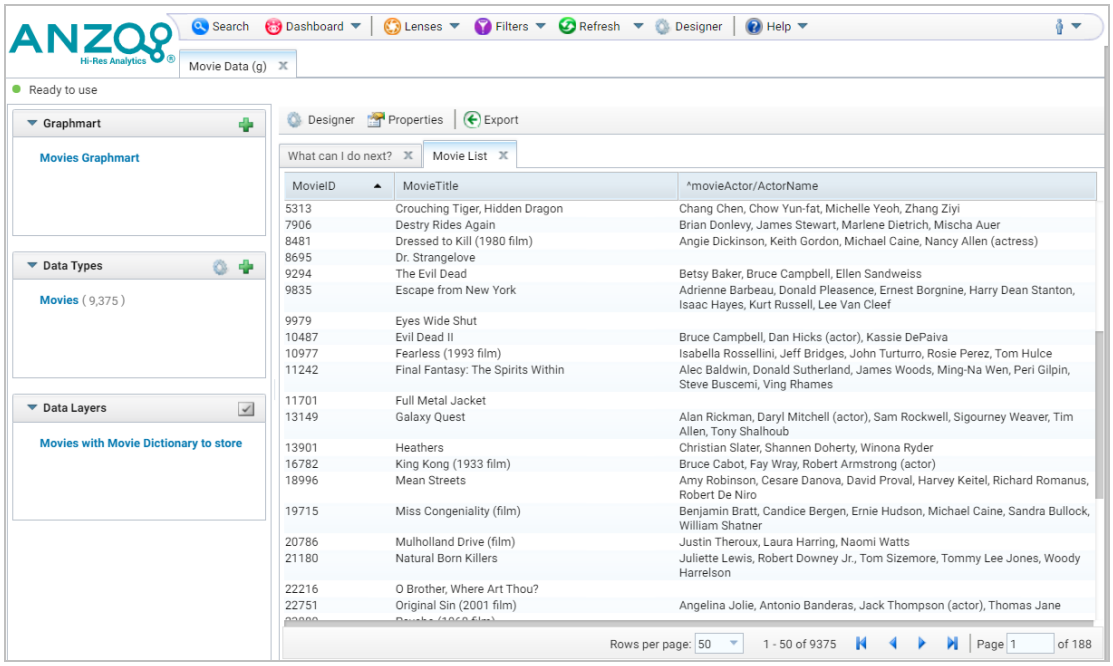
Lenses and filters can be configured to leverage the relationships from the base class to the connected classes. For example, adding a column that navigates the **movieActor** path to access the **MovieActors** class could be used to display values such as the names of the actors who starred in the movies. To navigate the relationship in the lens Designer, the MovieActor path is selected for the new column:



Once the path is chosen, the properties from the **MovieActor** class are displayed:

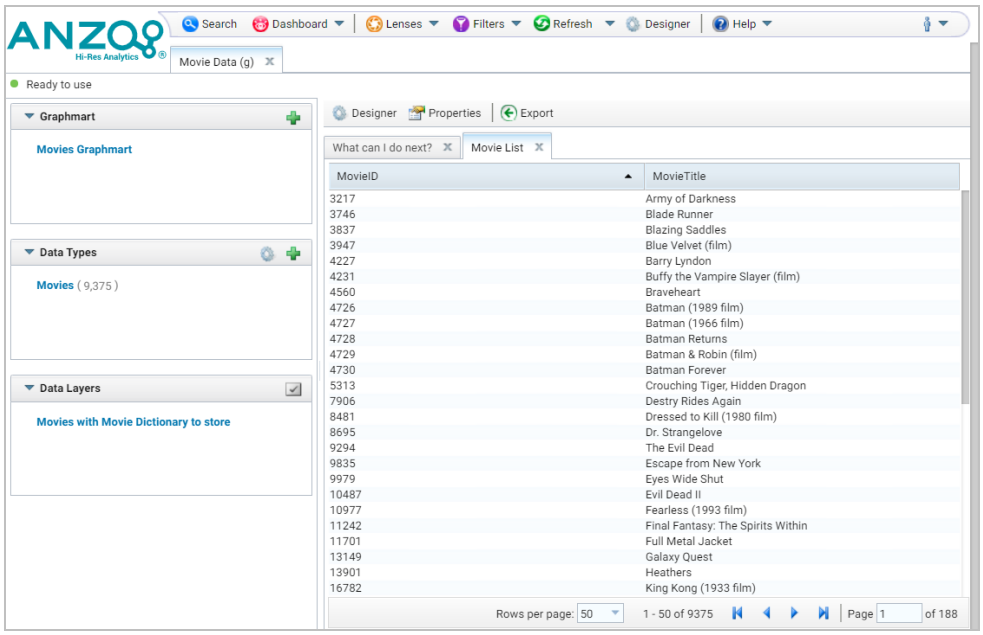


Selecting **ActorName** adds the column to the dashboard. The actors from each movie are now integrated into the lens even though the actor name values are not in the base class.

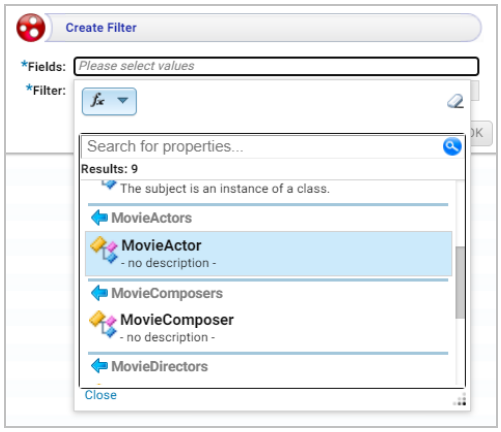


Filtering on Multiple Classes

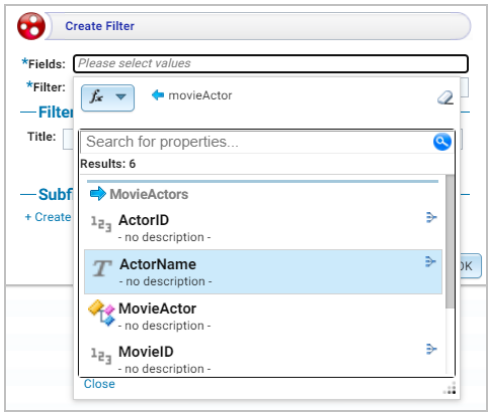
In addition to combining classes in lenses, you can also apply filters across classes. Like the example above, the image below shows a dashboard where the specified Data Type is **Movies**, and a table lens displays all of the columns/properties in the Movies class:



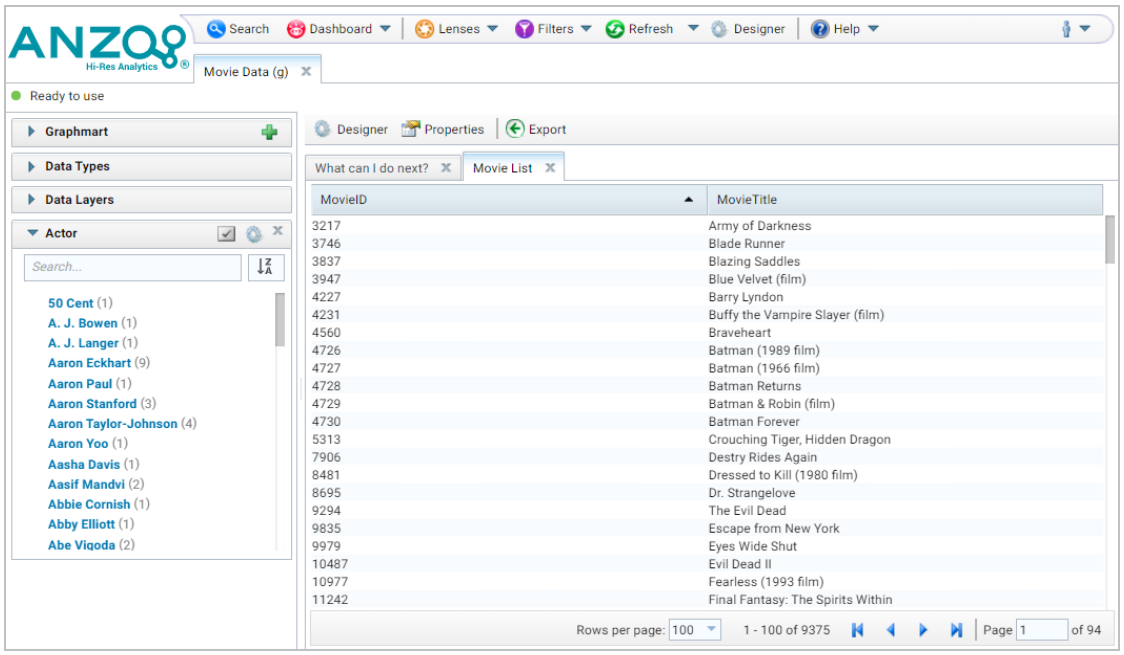
A filter can be configured to leverage the relationships from the base class to the connected classes. For example, adding a filter that navigates the **movieActor** path to access the **MovieActors** class could be used to display, and filter on, the names of the actors who starred in each movie. To navigate the relationship in the Create Filter , the MovieActor path is selected for the Field to filter on:



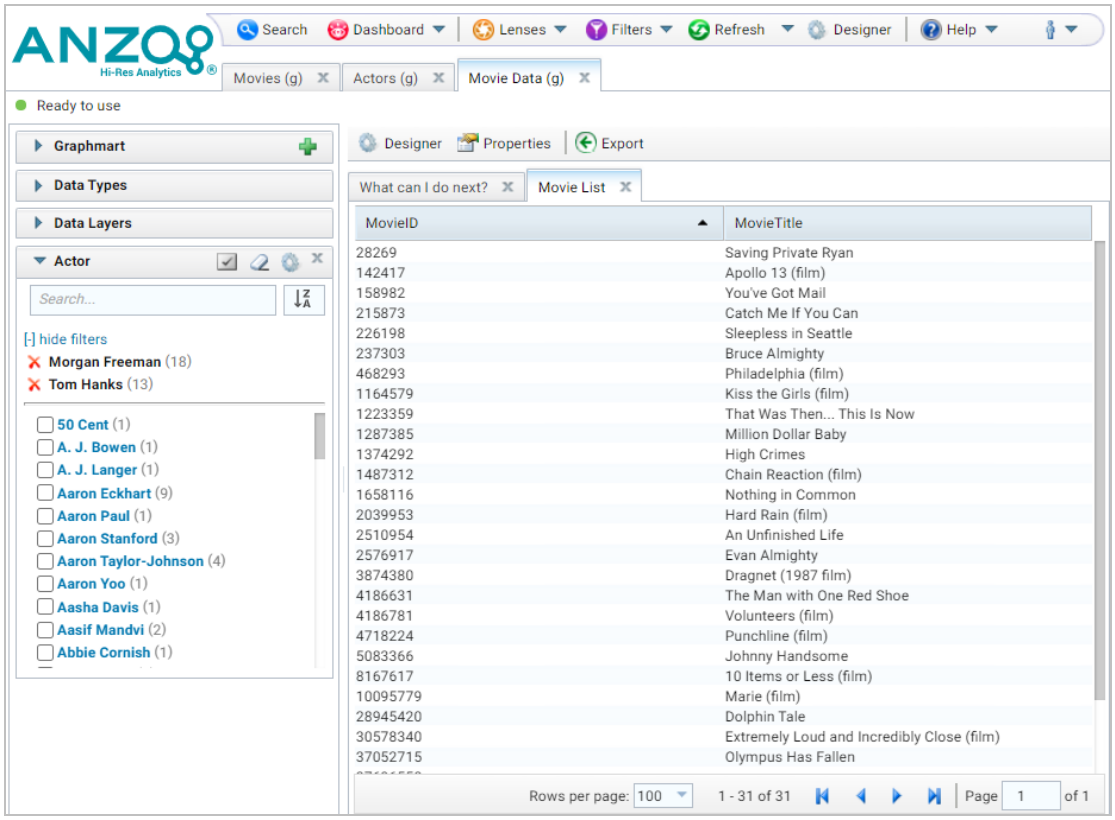
Once the path is chosen, the properties from the **MovieActor** class are displayed:



Selecting the **ActorName** property and choosing **List** as the type of filter adds a filter to the dashboard that lists all of the actors in the graph. Users can select particular actor names to filter the lens so that it only shows the movies that include one or more of the selected actors.



For example, selecting Morgan Freeman and Tom Hanks refreshes the lens to display only the movies that included one or both of those actors.



Using the same path traversal, filters could also be created to narrow the data to certain directors or producers by following the relationships to the MovieDirectors or MovieProducers classes. For more information about creating or editing lenses and filters, see Creating a Lens and Creating a Dashboard Filter.

Related Topics

Creating a Dashboard

Creating a Lens

Creating a Dashboard Filter

Managing Dashboard and Lens Security

Calculating Values in Filters and Lenses

Anzo provides many standard and advanced functions that you can use to compute the values that are displayed on a dashboard. When selecting properties and paths for lenses and filters, you can add calculations by selecting functions from a list or by writing your own formula. Hi-Res Analytics enables you to save your formulas as computed properties that can be reused on other dashboards, lenses, and filters.

This section provides instructions for using functions and formulas to calculate displayed values, saving formulas as computed properties, and reusing computed properties.

Computations in Filters and Lenses

When you apply formulas to properties in filters, Anzo performs the calculation across all of the values that exist for the selected property and then groups the results into the list of values that the calculations return. For multiple value properties, all values for that property are included in the calculations.

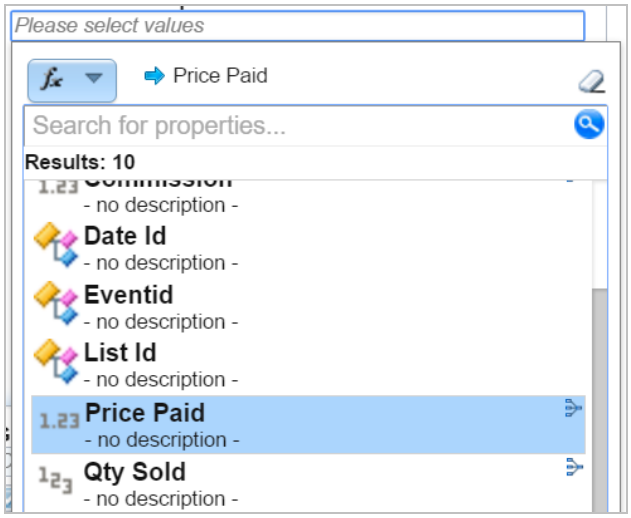
When you apply formulas to properties in lenses, the calculation results depend on the data type of the dashboard or lens. If the property belongs to a class that allows multiple values, Anzo performs the calculation on each set of multiple values and returns the results as one record in the lens. If the class includes single value properties, the calculation is performed separately for each single value.

- Applying Functions and Formulas to Properties
- Saving Formulas for Reuse
- Reusing Computed Properties

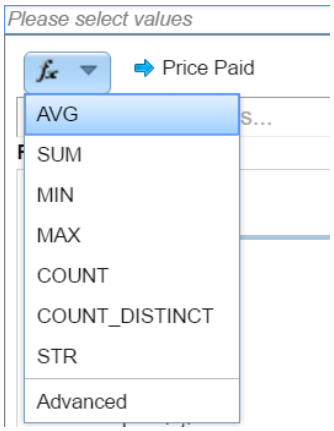
Applying Functions and Formulas to Properties

Follow these instructions to use a function or formula to compute the values in a lens or filter.

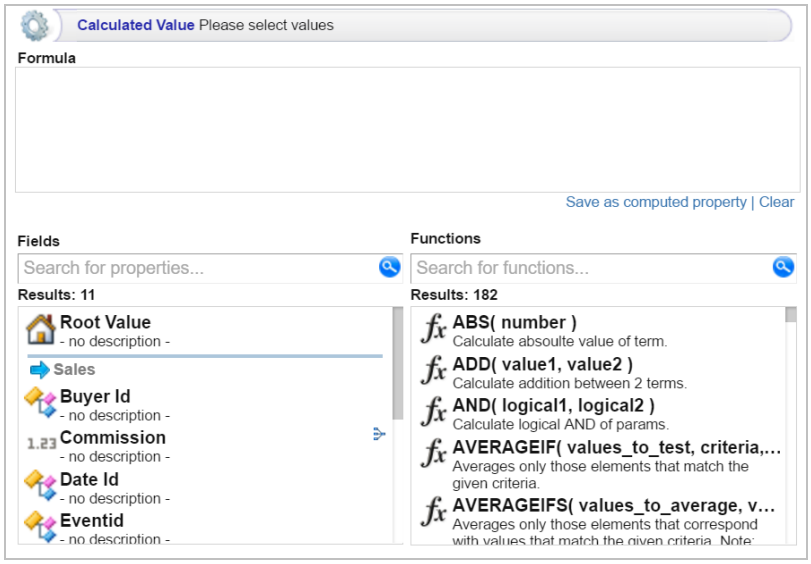
1. Create a new lens or filter or open the Designer for an existing lens or filter.
2. In the drop-down list for selecting properties or fields, select the property or path for which you want to compute the values. For example:



3. Click the function button ( ) to display the list of standard functions. The list varies depending on the data type of the selected property.



4. Click a function to apply it to the property that you chose. For information about each of the available functions, see Supported Functions and Formulas.
5. To choose a more advanced function or type a formula, click **Advanced**. The Calculated Value dialog box opens and enables you to choose additional properties and functions as well as type your own calculation.



Tip To create an advanced formula, it might help to get started by viewing the functions listed in the Functions column on the right side of the screen. Each function includes the syntax to use for creating a formula that uses that function.

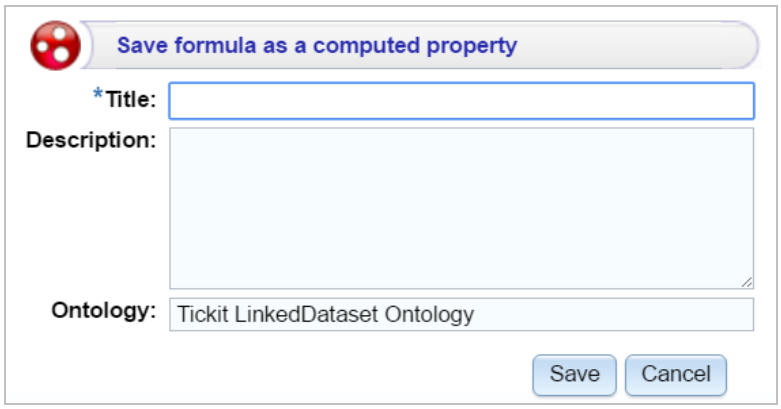
6. In the Functions column, double-click a function to add it to the Formula field at the top of the dialog box. For information about each of the available functions, see Supported Functions and Formulas.
7. Place your cursor in the Formula where you want to insert the property to perform the calculation on (for example, inside the parentheses) and then double-click the property in the Fields column. If the syntax for the function includes characters such as commas, type the characters in the appropriate location in the formula. You can click the **Clear** link on the bottom right of the Formula field any time to clear that field and start over.

8. When you are finished writing a formula, you have two options:
    - If you want to use the formula now without saving it for later use, click **Close** to close the Calculated Value dialog box. Then complete the lens or filter configuration.
    - If you want save the formula for reuse, click the **Save as computed property** link and follow the instructions below in Saving Formulas for Reuse.

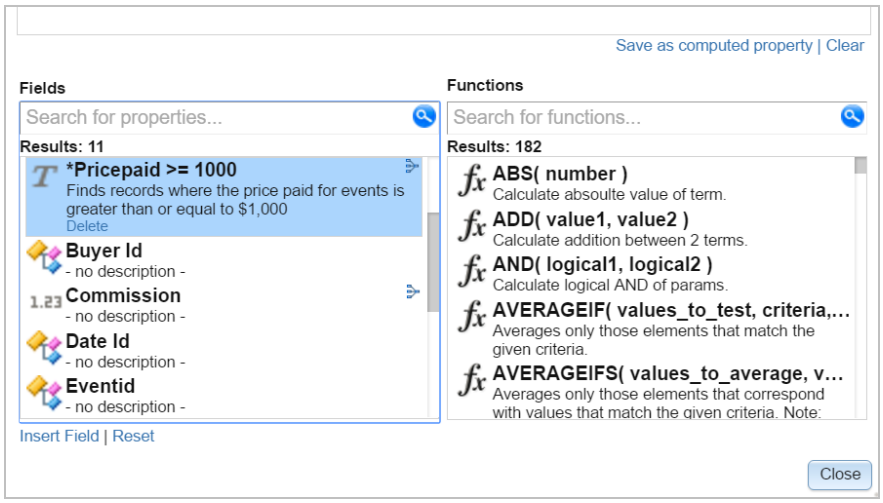
Saving Formulas for Reuse

Follow these instructions to save a formula as a computed property that you can use in other lenses and filters that target the same class of data.

1. When you have finished writing a formula in the Calculated Value dialog box, click the **Save as computed property** link below the Formula field. The Save formula as computed property dialog box opens.
2. In the **Title** field, type a name for the new computed property.



3. Type a description of the new property in the **Description** field.
4. If necessary, click in the **Ontology** field to choose another ontology to save the property in. If you want to save this property in multiple ontologies, you can click the **Save as computed property** link again after saving the property in the current ontology.
5. Click **Save**. Anzo saves the new property and labels it with an asterisk (*). The property becomes available in the Fields column in the Calculated Value dialog box.



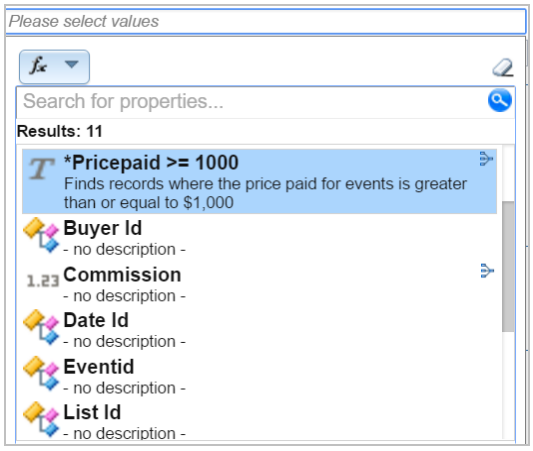
6. Click **Close** to close the dialog box.

Reusing Computed Properties

When an ontology contains computed properties, any other dashboards, lenses, and filters that use that ontology can also use the computed properties as long as they also use the same data type or class of data that the computed property is saved in.

To use a computed property:

1. Open the Designer for the filter or lens where you want to apply the computed property.
2. Click in the **Fields** or **Column Value Expression** field to open the property selection drop-down list. The drop-down list includes any computed properties that are available for use with the selected data type. Computed properties are labeled with an asterisk (*).



3. To use the property as-is, select the property and then close the drop-down list.

4. If you want to make changes to the formula and save it as a different computed property, select the property and then click the function button () to open the Calculated Value dialog box. Follow the instructions in Calculating Values in Filters and Lenses above to edit the formula and then save a new computed property.

Managing Dashboard and Lens Security

This topic includes reference information about dashboard and lens permissions and provides instructions for configuring permissions.

- Dashboard Level Permissions Reference
- Lens Level Permissions Reference
- Configuring Dashboard or Lens Permissions

Dashboard Level Permissions Reference

Dashboard level permissions affect a user's ability to view, modify, delete, design, or configure dashboards and dashboard permissions. There are three predefined dashboard permission sets that can be assigned to an Anzo user or group. You also have the option to customize the set of permissions that are applied to a user or group.

The table below lists the predefined permission sets and describes the privileges that are granted for each permission that is part of the predefined set:

Set	Applied Permission	Allows a user to:
View	**Read**	• Search for and open accessible dashboards. • Save As a new dashboard. • Share the dashboard. • View dashboard Properties. • View lens Properties. • Export lenses.
Modify	In addition to the **Read** permission described above, the **Modify** set includes the **Write** and **Delete** permissions described below.	
	Write	• Use the dashboard Designer to change the dashboard. • Clone lenses.
	Delete	• Delete the dashboard.

Set	Applied Permission	Allows a user to:
Admin	In addition to the **Read**, **Write**, and **Delete** permissions described above, the **Admin** set includes the **Manage** permission described below.	
	Manage	• The Manage permission relates only to the Security tab. If a user has this permission, they can modify dashboard access by changing permissions for a user, group, or role.

Default Dashboard Permissions

The table below lists the predefined permission sets that are applied by default when a new dashboard is created. Besides the sysadmin user, the dashboard creator is granted **Admin** privileges by default. The Everyone role is granted **View** privileges by default. No other users, groups, or roles have dashboard permissions assigned by default.

Anzo User/Role	Applied Permission Set
Sysadmin User	Admin
Dashboard Creator	Admin
Everyone Role	View

Lens Level Permissions Reference

Lens level permissions affect a user's ability to view, modify, delete, design, or configure lenses and lens permissions. There are three predefined lens permission sets that can be assigned to an Anzo user or group. You also have the option to customize the set of permissions that are applied to a user or group. While dashboard level permissions can affect a user's ability to clone a lens, the appropriate lens level permissions are required to be able to perform functions such as deleting or redesigning a lens.

The table below lists the predefined permission sets and describes the privileges that are granted for each permission that is part of the predefined set:

Set	Applied Permission	Allows a user to:
View	**Read**	• Search for and open accessible lenses. • View lens Properties. • Export lenses.

Set	Applied Permission	Allows a user to:
Modify	In addition to the **Read** permission described above, the **Modify** set includes the **Write** and **Delete** permissions described below.	
	Write	• Use the lens Designer to change the lens. • Rename the lens. • Clone the lens.
	Delete	• Delete the lens.
Admin	In addition to the **Read**, **Write**, and **Delete** permissions described above, the **Admin** set includes the **Manage** permission described below.	
	Manage	• The Manage permission relates only to the Security tab. If a user has this permission, they can modify lens access by changing permissions for a user, group, or role.

Default Lens Permissions

The table below lists the predefined permission sets that are applied by default when a new lens is created. Besides the sysadmin user, the lens creator is granted **Admin** privileges by default. The Everyone role is granted **View** privileges by default. No other users, groups, or roles have lens permissions assigned by default.

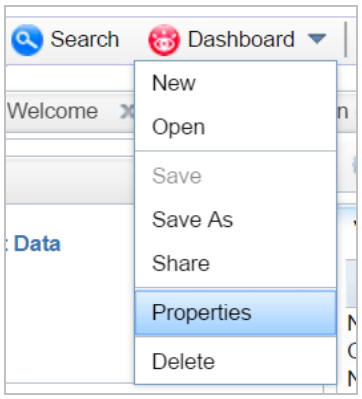
Anzo User/Role	Applied Permission Set
Sysadmin User	Admin
Lens Creator	Admin
Everyone Role	View

Configuring Dashboard or Lens Permissions

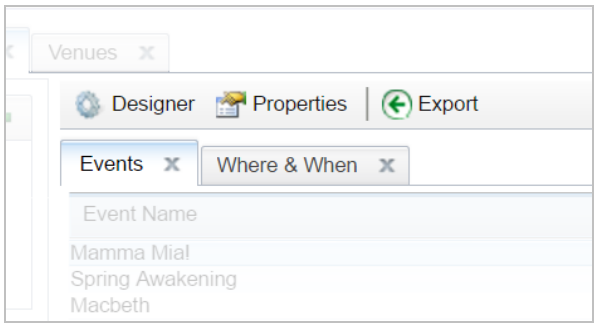
This section provides instructions for modifying dashboard or lens properties to grant or restrict access to your dashboards and lenses.

Note Data can be restricted at a higher level than a dashboard. Though users might have access to view your dashboards and lenses, data permissions determine whether they can view the data that the dashboard displays.

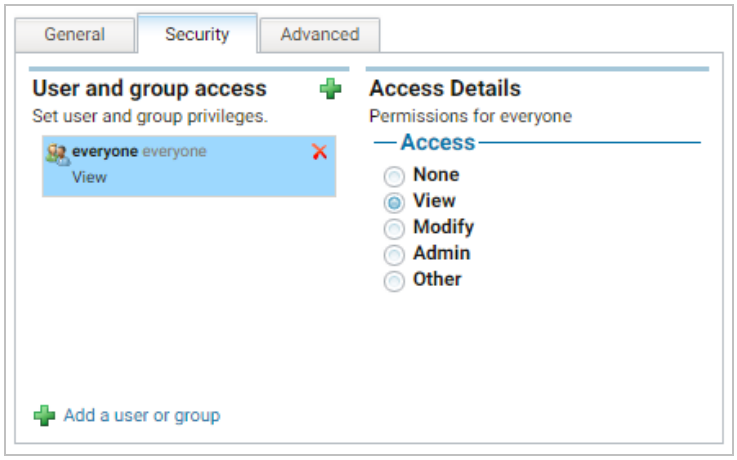
1. In the Anzo console, expand the **Access** menu, and click **Hi-Res Analytics**.
2. In the list of analytics, click the dashboard for which you want to modify access. Anzo opens the dashboard in the Hi-Res Analytics application.
3. Open the Properties dialog box for the either dashboard or for a specific lens. **Note**: Sharing a dashboard automatically shares the lenses in that dashboard.
    - To change access at the dashboard level, click **Dashboard** and select **Properties** in the main toolbar.



    - To change access for a lens in the dashboard, click the lens to display it and then click the **Properties** button in the object toolbar.



4. In the Properties dialog box, click the **Security** tab. This tab lists the available groups and users who can view this dashboard or lens.



5. Select a user or user group to manage, and then modify any of the following options:
    - **Remove a user or group**: Click the delete icon ( ) next to the user or group.
    - **Add a user or group**: Click to add a user or group. On the User/Group Browser dialog box, select the users or groups that you want to add. Then click **OK**.
    - **Access Details**: Select the access level for the selected user or group. Refer to Dashboard Level Permissions Reference or Lens Level Permissions Reference above for details about each of the access options.
        - **None**: No permissions set for the selected dashboard or lens.
        - **View**: Grants the **View** predefined permission set for the selected dashboard or lens.
        - **Modify**: Grants the **Modify** predefined permission set for the selected dashboard or lens.
        - **Admin**: Grants the **Admin** predefined permission set for the selected dashboard or lens.
        - **Other**: Enables you to set custom access levels for the selected dashboard or lens. Select the checkboxes to enable any combination of the following permissions: Read, Write, Delete, or Manage (administrator permissions).
6. Click **Save** to save the changes.

To get a URL to your dashboard that you can send to users, click **Dashboard** in the main toolbar and select **Share**. The Share Dashboard dialog box opens and displays a URL for the dashboard. You can copy the link and send it to users.

Related Topics

Creating a Dashboard

Deleting a Lens

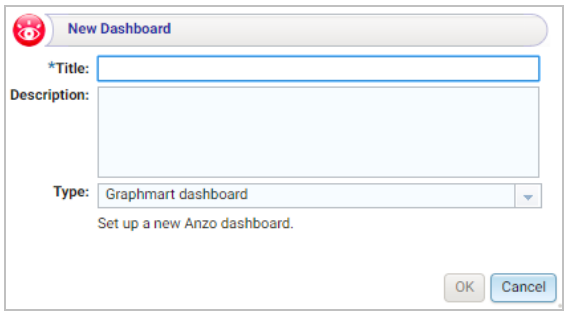
Creating a Lens

Searching for Text in Unstructured Documents

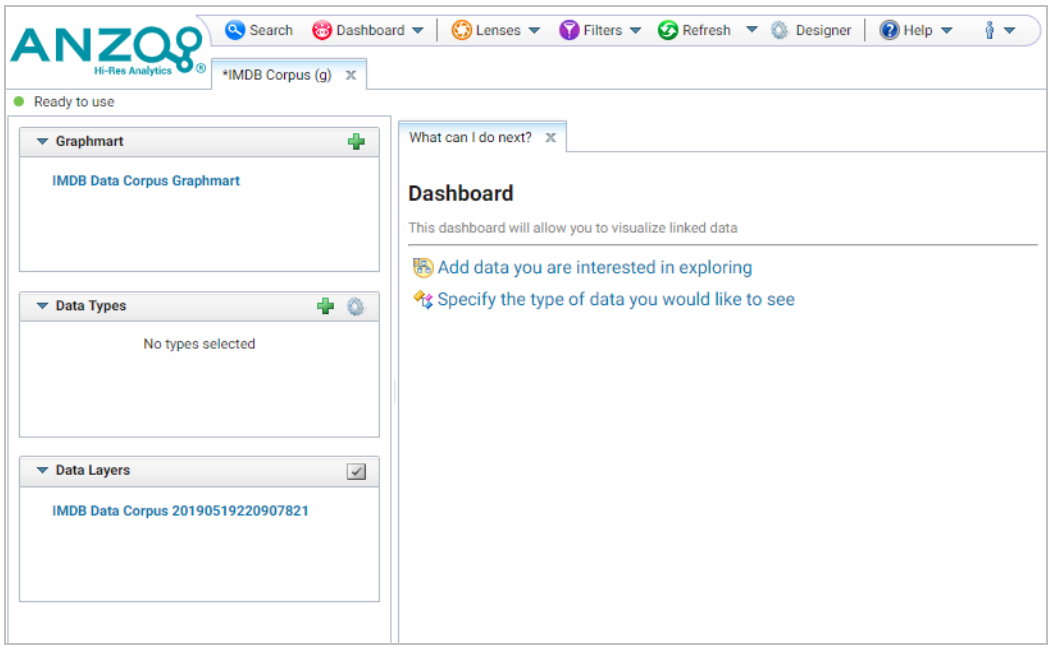
Anzo Hi-Res Analytics incorporates the Elasticsearch search engine to enable you to perform full text searches on unstructured documents. This topic provides instructions for creating a dashboard with text search capability and running a search across unstructured documents.

For information about running a pipeline to create an unstructured document data set, see Onboarding Unstructured Data.

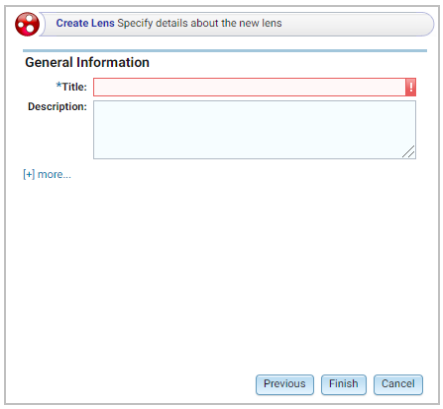
1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**.
2. On the Graphmarts screen, click the name of the graphmart that contains the unstructured documents. Anzo displays the graphmart overview screen.
3. On the Graphmart overview screen, click the **Create Dashboard** button. The Hi-Res Analytics application opens and displays the New Dashboard dialog box.



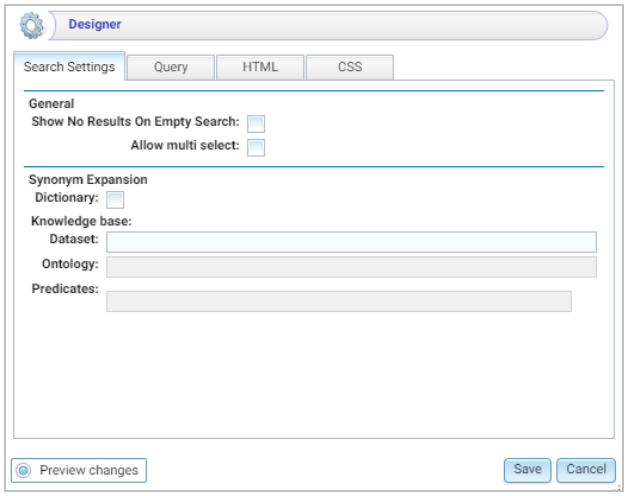
4. Type a name for the dashboard in the **Title** field and enter an optional **Description**. Then click **OK**. Anzo creates the dashboard and populates the Graphmart and Data Layers panels. For example:



5. In the Data Types panel, click the green plus icon (**+**) to open the Select Data Types dialog box. In the dialog box, select **Unstructured Document** and then click **OK**. Anzo adds the data type to the Data Types panel.
6. Next, click the **Lenses** button in the main toolbar and select **Open** from the drop-down list. Anzo opens the Len Selection dialog box.
7. In the Lens Selection dialog box, select **Document Search** and then click **Next**. Anzo displays the Create Lens dialog box.

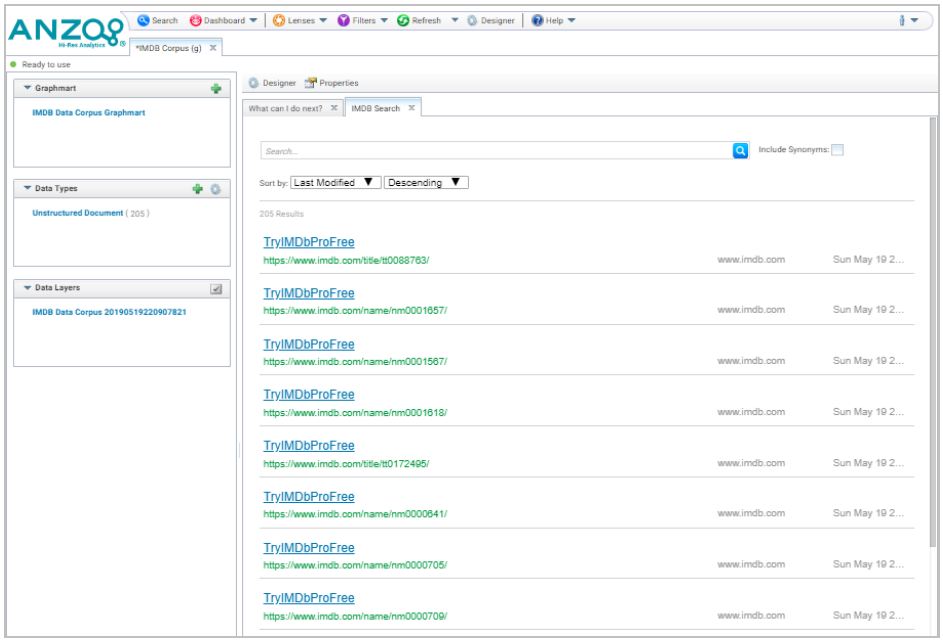


8. Type a name for the lens in the **Title** field and include an optional **Description**. Then click **Finish**. Anzo opens the Document Search Designer where you can configure the search settings or customize the style sheet, query, and HTML, if necessary. For example:

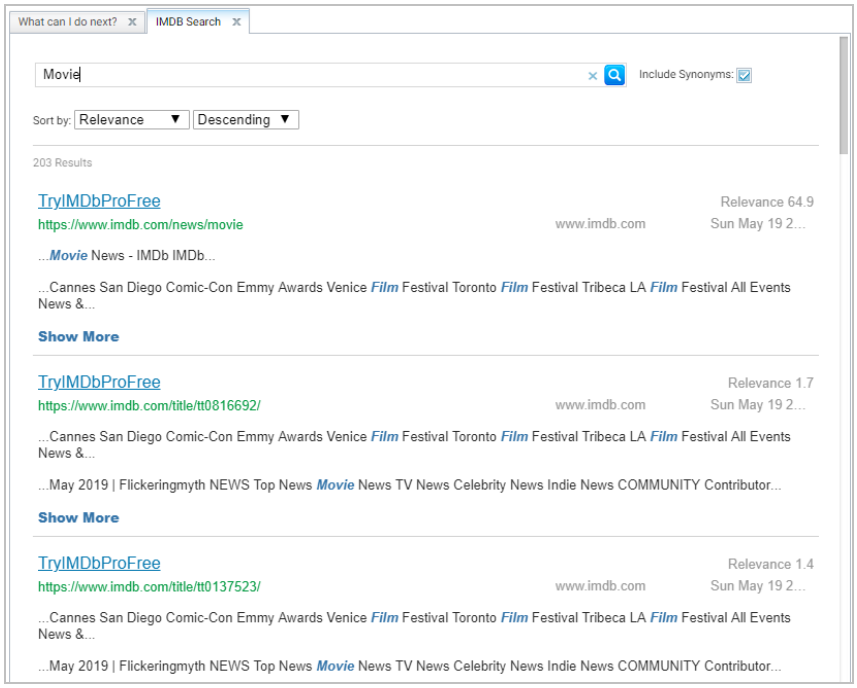


9. In the Designer, change the optional search settings as needed. The list below describes each option:
    - **Show No Results on Empty Search**: Determines whether documents are listed in the search results before a search is run. When enabled, the Document Search lens remains blank until a search is run.
    - **Allow Multi Select**: Determines whether a user can select multiple documents at a time in the results. When enabled, multiple documents can be selected by holding the Shift key and clicking documents in the results.
    - **Synonym Expansion Dictionary**: Determines whether to display an option for including synonyms in text searches. When enabled, the lens displays an **Include Synonyms** checkbox next to the Search field.
    - **Knowledge Base Dataset**: Enables you to include a knowledge base in the search if one exists. Click the field to select an available knowledge base.
    - **Ontology**: Enables you to select a data model to use for the search.
    - **Predicates**: Enables you to select specific predicates from the model.

10. Click **Save**. Anzo add the lens to the dashboard. Depending on the search settings, the lens displays the list of documents. For example:

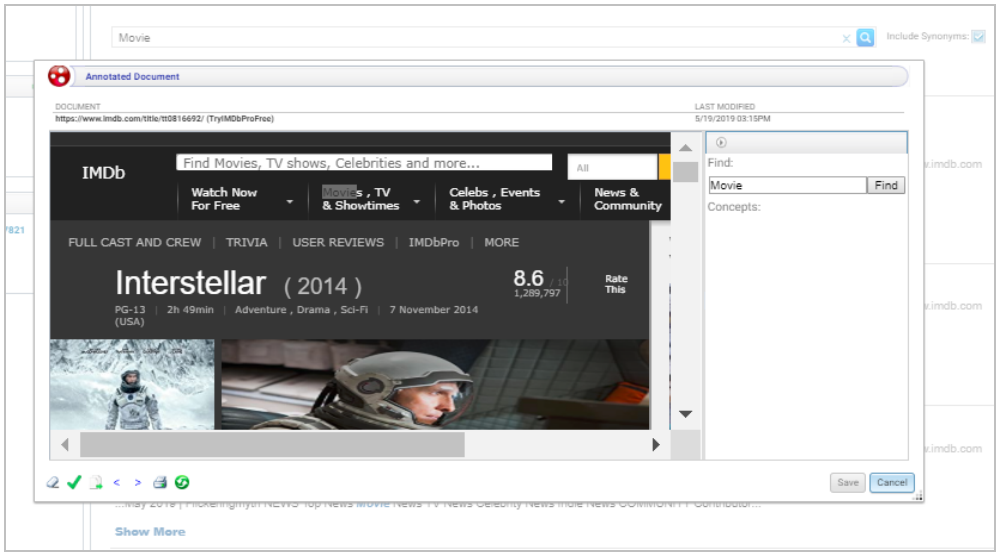


11. To run a search, type the text to find in the **Search** field and press **Enter**. See the Supported Search Syntax section below for information about supported search syntax. Anzo finds documents that include the search value and displays the documents, snippets of text to show the context of where the matches were found, and the Elasticsearch relevance score for the match. For information about how the relevance score is calculated, see What Is Relevance? in the Elasticsearch documentation. For example:



Clicking **Show More** expands the result to display additional matches. For example:

12. To refine the search, alter the text in the **Search** field and press **Enter** again. You can also click highlighted terms in the search results to open a dialog box that shows the full annotated document where the match was found. For example:



Supported Search Syntax

This section describes the keyword search syntax that Anzo supports.

Wildcard Characters: ? and *

- **?**: Use a question mark (?) to represent a single wildcard character. For example, in the search**co?l**, the resulting documents will include terms like "cool" or "coal."
- *: Use an asterisk (*) to represent multiple wildcard characters. For example, in the search**col***, the resulting documents will include terms like "collect" or "color."

Boolean Operators: +, -, OR, AND, NOT

- **+**: Use a plus (+) character to indicate mandatory matches. For example, in the search **flight +New York**, the resulting documents can include "flight" as an optional match and must include "New York."
- **-**: Use a minus (-) character to indicate a term that must not match. For example, in the search **flight +New York -Los Angeles**, the resulting documents can include "flight" as an optional match, must include "New York," and must not include "Los Angeles."
- **OR**: In the search **New York OR Los Angeles**, the resulting documents will include a match for either "New York" or "Los Angeles."
- **AND**: In the search **New York AND Los Angeles**, the resulting documents must include matches for both "New York" and "Los Angeles."
- **NOT**: In the search **New York NOT Los Angeles**, the resulting documents must include "New York" and cannot contain "Los Angeles."
- Grouping operators: In the search**(flight AND New York) OR Los Angeles**, the resulting documents will include "flight" and "New York" and optionally include "Los Angeles."

Fuzzy Matches: ~*n*

To search for a fuzzy match, use a tilde (~) character followed by a number to represent the number of fuzzy or incorrect characters. For example, in the search **Flgth~3**, the resulting documents could include the term "Flight."

Regular Expressions

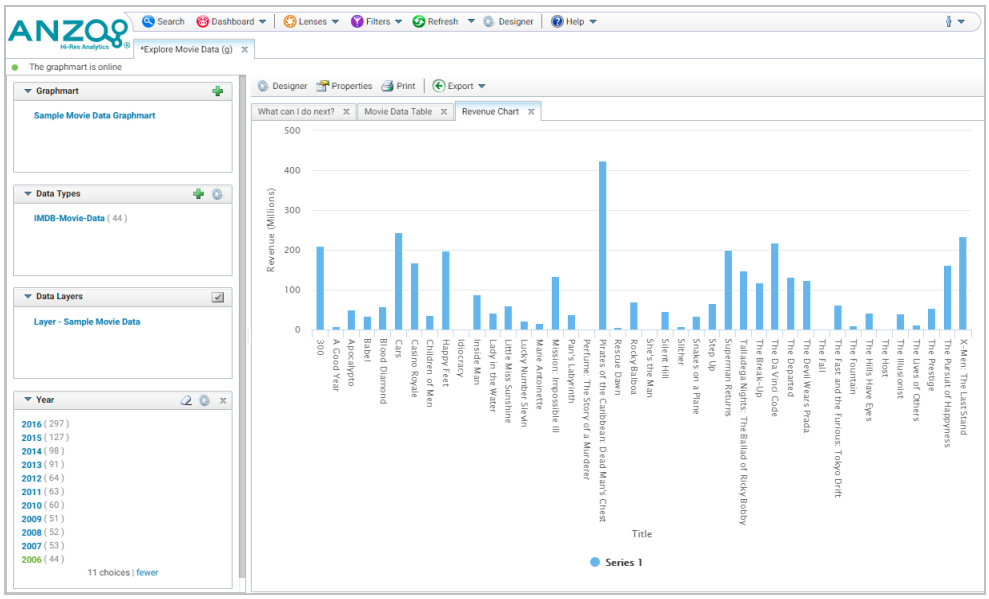
For example, the following search expression matches email addresses: **/([a-zA-Z0-9_\-\.]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})**/.

For more information about the regular expression syntax that Elasticsearch supports, see Regular expression syntax in the Elasticsearch documentation.

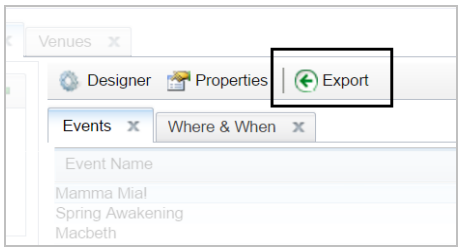
Exporting a Lens

If you have dashboards with Table and Chart lenses, you can export those lenses from the Hi-Res Analytics application. Charts can be exported as images in JPEG, PNG, or SVG format, and tables can be exported to CSV or JSON files. Follow the instructions below to export a lens.

1. Open the dashboard that contains the lens that you want to export.
2. If necessary click the tab for the lens to make it active. For example, the image below shows a chart lens.



3. In the object toolbar for the lens, click the **Export** button.



4. If the lens is a chart, select the one of the image types from the drop-down list. Anzo creates the image as that type and downloads the file to your computer.
5. If the lens is a table, the Export Options dialog box is displayed:

6. In the Export Options dialog box, specify the following file options:
   - **File name**: Specify a name for the file. Do not specify the file type extension.
   - **Format**: Click the Format field and select **CSV** to create a .csv file or **JSON** to create a .json file.

- **Multi Valued Column Separator**: For CSV files, click this column to select the character to use as a separator in the file. This option does not apply to JSON files.
- **Export Headers**: Indicates whether to include column headers in the file. Clear the checkbox to exclude headers from the file. This option does not apply to JSON files.

7. Click **OK** to download the file to your computer.

Related Topics

Creating a Lens

Deleting a Lens

Deleting a Lens

This topic provides information about the permissions that are required for deleting lenses as well as instructions for deleting a lens.

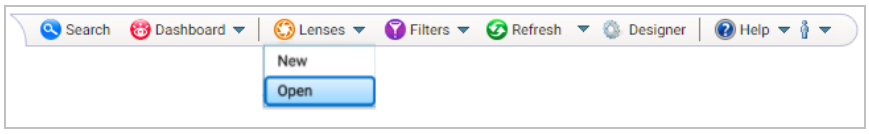
Required Permissions for Deleting Lenses

By default, only the **sysadmin** user and **lens creator** have permission to delete a lens. To delete a lens, a user must have the **Manage** permission assigned for that lens. The Manage permission is included in the **Admin** predefined lens permission set. If lens permissions have not been changed since the lens was created, the sysadmin user and the lens creator are the only users who have permission to delete that lens. The Manage permission is also required to change lens security settings and grant privileges to other users. Users who have read access to a lens (granted through the View, Modify, or Admin lens permission sets) can view the lens security settings to identify which non-sysadmin users have permission to delete the lens. For more information, see Managing Dashboard and Lens Security.

Deleting a Lens

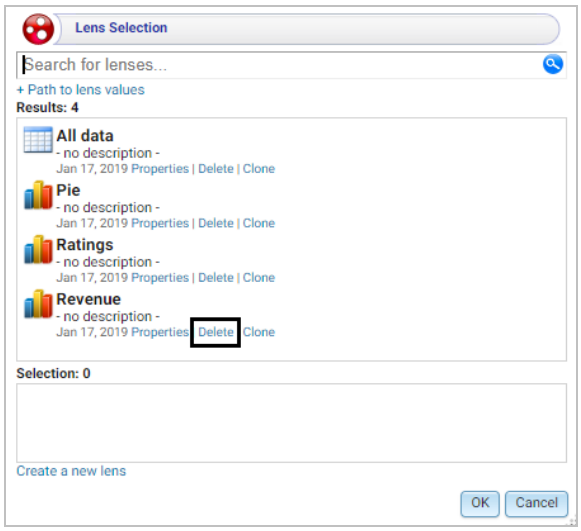
Follow the steps below to delete a lens.

1. In the Hi-Res Analytics application, click the **Lenses** menu in the main toolbar and select **Open**.



   The Lens Selection dialog box is displayed.

2. In the Lens Selection dialog box, find the lens that you want to delete and then click the **Delete** link for that lens. For example:



3. The application presents a confirmation message. Click **Yes** to delete the lens. The lens is removed from the Lens Selection dialog box, and you can repeat this process to delete additional lenses for which you have the required privileges.

Related Topics

Managing Dashboard and Lens Security

Creating a Lens

Exporting a Lens

Supported Functions and Formulas

This section describes the standard and advanced functions that are available when working with Hi-Res Analytics. For information about using functions in dashboards, see Calculating Values in Filters and Lenses.

- Functions on Strings
- Functions on RDF Terms
- Functions on Numerics
- Functions on Dates, Times, and Durations
- Functions on Boolean Values
- Window Aggregate Functions

Functions on Strings

The table below details the Anzo functions for string data types.

Function (syntax) & Argument Data Types	Description	Return Type
BUSINESS_ENTITY_ EXCLUDER(text) text: string	Removes from strings suffixes that represent business entities.	String
CONCATURL(text, ...) text: string	Concatenates the values for the specified properties or expressions and returns the concatenation as an xsd:anyURI value.	URI
CONCATENATE(text, ...) text: string	Concatenates the values for the specified properties or expressions and returns the concatenation as an xsd:string value.	String
CONTAINS(text, pattern) text: string pattern: string	Determines if the values for a property contain the specified string. Results are grouped under True or False.	Boolean
ENCODE_FOR_URI(text) text: string	Returns results encoded as URIs.	URI
ESCAPEHTML(text) text: string	Escapes the specified string for use in HTML.	String
FIND(find_text, within_text, start_num) find_text: string within_text: string start_num: integer	Determines if the specified text exists in another text string.	Integer

Function (syntax) & Argument Data Types	Description	Return Type
GROUPCONCAT(separator, valueSeparator, serialize, valueLimit, rowLimit, delimitBlanks, text) separator: string valueSeparator: string serialize: boolean valueLimit: integer rowLimit: integer delimitBlanks: boolean text: string	Performs a string concatenation all of the values that are bound to a property.	String
LANG(value) value: string	Returns any language tags that exist for the specified property's literal values and groups the results under any language tags or "blank" if a language tag does not exist for a record.	String
LANGMATCHES(language_tag, language_range) language_tag: string language_range: string	Determines whether any of the values for a property contain a language tag from the specified range of tags.	Boolean
LCASE(value) value: string literal	Converts string values to lower case in the filter.	String
LEFT(text, number_of_characters) text: string number_of_characters: integer	Returns the specified number of characters starting from the left of the string.	String
LEN(value) value: string	Calculates the length of the string values.	Integer

Function (syntax) & Argument Data Types	Description	Return Type
LEVENSHTEIN_DIST (value1, value2) value1: string value2: string	Calculates the Levenshtein distance or measure of similarity between the specified strings. The distance is the number of edits required to transform the first string into the second string.	Integer
LOWER(text, language, country, variant) text: string language: string country: string variant: string	Converts string values to lower case letters.	String
MD5(value) value: term	Calculates the MD5 hash of string values.	String
MID(text, start_num, num_chars) text: string start_num: integer num_chars: integer	Returns the specified number of characters from a string, starting from the chosen position in the string.	String
REGEX(text, pattern, [flags]) text: string pattern: string flags: string	Determines whether the specified string matches a regular expression pattern. You can use the optional flags argument to include one or more modifier flags that further define the pattern. For information about flags, see the Flags section of the W3C XQuery 1.0 and XPath 2.0 Functions and Operators specification.	Boolean
REPLACE(text, pattern, replacement, flags) text: string pattern: string replacement: string flags: string	Extends the REGEX function to provide the ability to take a replacement pattern and return the replaced string.	String

Function (syntax) & Argument Data Types	Description	Return Type
RIGHT(text, num_chars) text: string num_chars: integer	Returns the specified number of characters, starting at the end of a string.	String
RULE_BASED_LOCALITY_SENSITIVE_HASH(text) text: string	Transforms the specified string by normalizing across spacing and characters, removing punctuation and special characters, and cleaning common English affixes.	String
SEARCH(text, pattern, required, wildcard, remove, escape) text: string pattern: string required: string wildcard: string remove: string escape: string	Uses text search semantics to determine whether the specified text matches a regular expression pattern.	Boolean
STRAFTER(text, pattern) text: string pattern: string	Returns the part of a string that comes after the pattern that you specify.	String
STRBEFORE(text, pattern) text: string pattern: string	Returns the part of a string that comes before the pattern that you specify.	String
STRDT(value, *URI*("data_type")) value: string data_type: string	Casts a string value to the specified data type. A URI function, such as TOURI, IRI, or URI, is required to specify the data type, which is a URI. For example, the following formula casts a regionkey column from a string to an integer: `STRDT([{region}regionkey], TOURI("xsd:int"))`	Term

Function (syntax) & Argument Data Types	Description	Return Type
STRENDS(text, pattern) text: string pattern: string	Determines whether the specified string ends with the given pattern.	Boolean
STRLANG(text, language) text: string language: string	Constructs a literal value with the specified language tag.	String
STRSTARTS(text, pattern)	Determines whether the specified string value starts with the given pattern.	Boolean
STRUUID()	Returns a string that is the result of generating a Universally Unique Identifier (UUID).	String
SUBSTITUTE(text, old_text, new_text, instance_num) text: string old_text: string new_text: string instance_num: integer	Substitutes new text for old text in a string.	String
TOURI(value) value: string	Casts a string value to a URI.	URI
TRIM(text) text: string	Removes all spaces from values except for single spaces between words.	String
UPPER(text, language, country, variant) text: string language: string country: string variant: string	Converts all lower case letters to upper case letters.	String

Functions on RDF Terms

The table below details the Anzo functions for RDF term types: literal values, URIs, and blank nodes.

Function (syntax) & Argument Data Types	Description	Return Type
ADD(term1, term2) term1: term term2: term	Adds the results from the expressions that you specify.	Term
AVERAGEIF(values_to_test, criterion, values_to_average) values_to_test: term criterion: term values_to_average: integer	Calculates the averages of the values that meet the specified criterion.	Integer
AVERAGEIFS(values_to_ average, values_to_test, criteria, ...) values_to_average: numeric values_to_test: term criteria: term	Similar to the AVERAGEIF function but enables you to specify multiple criteria.	Integer
BNODE(term) term: term	For use with Presence, Hierarchy, and Types filters to determine whether blank nodes exist for properties. You can also perform the BNODE function on literal values.	Term
BOOLEAN(term) term: term	Creates an xsd:boolean type based on label of the input term.	Boolean
BOUND(term) term: term	Determines which records include a value for the specified property and returns "True" for records that include a value or "False" for records that do not include a value.	Boolean
CASE(value, criteria, ..., result, ..., default) value: term criteria: term result: term default: term	Enables you to add IF/THEN logic. CASE expressions evaluate a series of conditions for the properties that you specify and return results when the test returns true. The optional "default" argument is a default value to return if none of the tests pass.	Term

Function (syntax) & Argument Data Types	Description	Return Type
CEILING(number) number: term	Calculates the ceiling (the next whole number up from the value if the value has a fractional part) of the values that exist for the selected property and then groups the results into the list of ceiling values. CEILING returns the value itself if it is a whole number.	Term
CHOOSE_BY_MAX(test, value) test: term value: term	Calculates the maximum values from the first expression or property and returns the values from the second expression or property that correspond to the maximum values. For example, in an imaginary sales data set, the following formula returns the IDs for the buyers who spent the most: `CHOOSE_BY_MAX ([{Sales}Price Paid], [{Sales}Buyer Id])`	Term
CHOOSE_BY_MIN(test, value) test: term value: term	Calculates the minimum values from the first expression or property and returns the values from the second expression or property that correspond to the minimum values. For example, in an imaginary sales data set, the following formula returns the IDs for the buyers who spent the least: `CHOOSE_BY_MIN ([{Sales}Price Paid], [{Sales}Buyer Id])`	Term
COALESCE(value, ...) value: term	Evaluates any number of expressions and returns the results for the first expression that does not raise an error. Errors occur if an expression evaluates to an unbound variable or a non-RDF term.	Term
COUNT(value) value: term	Counts the number of values for the selected property.	Integer
COUNT_DISTINCT(value) value: term	Counts the number of unique values for the selected property.	Integer
COUNTIF(value, criterion) value: term criterion: term	Calculates the counts of the values that meet the specified criterion.	Integer

Function (syntax) & Argument Data Types	Description	Return Type
COUNTIF(value, criteria, ...) value: term criteria: term	Similar to the COUNTIF function but enables you to specify multiple criteria.	Integer
DATATYPE(term) term: literal value	For use with Presence, Hierarchy, and Types filters.	URI
DATEVALUE(date_text) date_text: term	Groups results under the specified literal date value.	Date
EQUAL(value1, value2) value1: term value2: term	Determines whether value1 is equal to value2.	Boolean
GE(value1, value2) value1: term value2: term	Performs a greater than or equal to (>=) comparison between value1 and value2.	Boolean
GT(value1, value2) GT functions on numerics, booleans, dateTimes, and terms in this priority order	Performs a greater than (>) comparison between value1 and value2.	Boolean
IF(test, value_if_true, value_if_false, value_if_error) test: boolean value_if_true: term value_if_false: term value_if_error: term	Evaluates one expression and returns a second expression depending on the answer.	Term
IFERROR(value, value_if_error, ...) value: term value_if_error: term	Synonym for COALESCE.	Term

Function (syntax) & Argument Data Types	Description	Return Type
IN(value, test_value, ...) value: term test_value: term	Determines whether any of the values for the first property are found in the other specified expressions or properties. Anzo groups the results under True or False.	Boolean
ISBLANK(value) value: term	Determines whether the property has blank node values and groups the results under True or False.	Boolean
ISDATATYPE(value, data_type) value: term data_type: URI	Determines whether the values for a property are the specified data type and groups the results under True or False.	Boolean
ISERROR(value) value: term	Determines whether the argument evaluates to an error and groups the results under True or False.	Boolean
ISIRI(value) ISURI(value) value: term	Determines whether the argument is an IRI. ISIRI and ISURI return true if the value is an IRI or URI (and is not blank) and false if it is not.	Boolean
ISLITERAL(value) value: term	Determines whether the property has literal values.	Boolean
ISNUMERIC(value) value: term	Determines whether the property has numeric values.	Boolean
LE(value1, value2) value1: term value2: term	Performs a less than or equal to (<=) comparison between value1 and value2.	Boolean
LOCALNAME(URI)	Returns only the local name portion of a URI.	String
LONG(value) value: term	Displays numeric values in xsd:long format.	Long

Function (syntax) & Argument Data Types	Description	Return Type
LT(value1, value2) value1: term value2: term	Performs a less than (<) comparison between value1 and value2.	Boolean
MAX(value, ...) value: term	Aggregate function that calculates the maximum values for each aggregate group.	Term
MAXVAL(value, ...) value: literal	Computes the maximum values for the specified arguments.	Literal
MD5(value) value: term	Calculates the MD5 hash of string values.	String
METADATAGRAPHURI(URI)	Returns the metadata graph URI for the specified input URI.	URI
MIN(value, ...) value: term	Aggregate function that calculates the minimum values	Term
MINVAL(value, ...) value: literal	Computes the minimum values for the specified arguments.	Literal
MODE(value) value: term	Aggregate function that returns the number that occurs most frequently in each aggregate group.	Numeric
NAMESPACE(URI)	Returns the namespace for the specified URI values.	String
NOT(value) value: boolean	Performs logical negation on the specified expression.	Boolean
NOT_EQUAL(value1, value2) value1: term value2: term	Performs a not equal (!=) comparison between value1 and value2.	Boolean

Function (syntax) & Argument Data Types	Description	Return Type
NOT_IN(value, test_value, ...) value: term test_value: term	Tests whether the value is not found in the test_value list of expressions.	Boolean
OR(logical1, logical2) logical1: boolean logical2: boolean	Calculates the logical OR of the input values.	Boolean
PARTITIONINDEX(value, start, interval) value: literal start: literal interval: literal	Returns the zero-based index of the bucket in which the value falls. The buckets start at the specified start and are sized according to the specified interval. The first bucket is (start, start+interval): closed on the low end and open on the high end. PARTITIONINDEX returns less than 0 if the value does not fall into any bucket, such as when the value is less than start or if the comparison is indeterminate for date and time data types.	Integer
SAMETERM(term1, term2) term1: term term2: term	Determines whether the specified RDF terms are the same.	Boolean
SAMPLE(term) term: term	Returns an arbitrary value from the group to represent the given variable.	Term
SERIALIZE(term) term: term	Returns the string representation of the specified term.	String
SHA1(term)	Calculates the SHA1 digest of the specified term.	String
SHA224(term)	Calculates the SHA224 digest of the specified term.	String
SHA256(term)	Calculates the SHA256 digest of the specified term.	String
SHA384(term)	Calculates the SHA384 digest of the specified term.	String

Function (syntax) & Argument Data Types	Description	Return Type
SHA512(term)	Calculates the SHA512 digest of the specified term.	String
STR(term)	Returns a string representation of the values for the selected property.	String
STRLEN(term)	Calculates the length of the specified term.	Integer
SUBSTR(term, start, [length]) term: term start: integer length: integer	Returns a substring of the specified term. The start argument indicates the character position to start the substring with. The first character in the term is position 1. The optional length argument specifies the number of characters to return.	String
SUMIF(values_to_test, criterion, values_to_sum) values_to_test: term criterion: term values_to_sum: numeric	Calculates the sums of the values that match the specified criterion.	Integer
SUMIFS(values_to_sum, values_to_test, criteria, ...) values_to_sum: numeric values_to_test: term criteria: term	Similar to the SUMIF function but enables you to specify multiple criteria.	Integer
TEXT(value, format) value: term format: string	Formats a term value as text.	String
UCASE(term) term: term	Returns the specified term as an uppercase string value.	String
UNBOUND()	Returns an unbound term	Term
UUID()	Generates a new IRI from the Universally Unique Identifier (UUID) Uniform Resource Name (URN) namespace.	URI

Functions on Numerics

The table below details the Anzo functions for numeric data types. "Term" indicates an RDF term type value: a literal value, URI, or blank node.

Function (syntax) & Argument Data Types	Description	Return Type
ABS(number) number: numeric value	Calculates the absolute values that exist for the selected property.	Numeric
ADD(term1, term2) term1: term term2: term	Adds the results from the expressions that you specify.	Term
AVERAGEIF(values_to_test, criterion, values_to_average) values_to_test: term criterion: term values_to_average: integer	Calculates the averages of the values that meet the specified criterion.	Integer
AVERAGEIFS(values_to_average, values_to_test, criteria, ...) values_to_average: numeric values_to_test: term criteria: term	Similar to the AVERAGEIF function but enables you to specify multiple criteria.	Integer
AVG(number) number: numeric value	Calculates the averages of the values that exist for the selected property.	Numeric
CEILING(number) number: term	Calculates the ceiling (the next whole number up from the value if the value has a fractional part) of the values that exist for the selected property and then groups the results into the list of ceiling values. CEILING returns the value itself if it is a whole number.	Term

Function (syntax) & Argument Data Types	Description	Return Type
CHOOSE_BY_MAX(test, value) test: term value: term	Calculates the maximum values from the first expression or property and returns the values from the second expression or property that correspond to the maximum values. For example, in an imaginary sales data set, the following formula returns the IDs for the buyers who spent the most: `CHOOSE_BY_MAX ([{Sales}Price Paid], [{Sales}Buyer Id])`	Term
CHOOSE_BY_MIN(test, value) test: term value: term	Calculates the minimum values from the first expression or property and returns the values from the second expression or property that correspond to the minimum values. For example, in an imaginary sales data set, the following formula returns the IDs for the buyers who spent the least: `CHOOSE_BY_MIN ([{Sales}Price Paid], [{Sales}Buyer Id])`	Term
COS(angle) angle: double	Calculates the cosines of the values that exist for the selected property.	Double
DECIMAL(value) value: term	Returns numeric results in decimal format.	Numeric
DIVIDE(value1, value2) value1: numeric value2: numeric	Divides the values for the first property or expression by the values for the second property or expression and groups the results into the list of division values.	Numeric
DOUBLE(value) value: term	Displays the results of the specified numeric property in xsd:double format.	Double
EXP(number) number: double	Raises the results to the power of the specified number.	Double
FACT(number) number: integer	Calculates the factorial of the results by the specified number.	Integer
FLOAT(value) value: term	Returns numeric results in float format and groups the results into the list of float values	Float

Function (syntax) & Argument Data Types	Description	Return Type
FLOOR(number) number: term	Calculates the floor (the closest whole number down from the value if the value has a fractional part) of the values that exist for the selected property and then groups the results into the list of floor values. FLOOR returns the value itself if it is a whole number.	Term
FORMATDATE(value, format) value: term format: string	Formats a numeric or date value into date text.	String
FORMATFRACTION(value, tolerance, seperate_whole_ number) value: term tolerance: double separate_whole_number: boolean	Returns results in fraction format rather than decimal format.	String
FORMATNUMBER(value, format) value: term format: numeric format	Formats a numeric value into text in the specified format.	
GE(value1, value2) value1: term value2: term	Performs a greater than or equal to (>=) comparison between value1 and value2.	Boolean
GT(value1, value2) GT functions on numerics, booleans, dateTimes, and terms in this priority order	Performs a greater than (>) comparison between value1 and value2.	Boolean

Function (syntax) & Argument Data Types	Description	Return Type
HAMMING_DIST(value1, value2) value1: long value2: long	Calculates the hamming distance between two values.	Integer
HAVERSINE_DIST(lat1, lon1, lat2, lon2) lat1: double lon1: double lat2: double lon2: double	Computes the haversine distance between two latitude and longitude values.	Double
INTEGER(value) value: term	Returns numeric results in integer format.	Integer
LE(value1, value2) value1: term value2: term	Performs a less than or equal to (<=) comparison between value1 and value2.	Boolean
LN(value) value: double	Calculates the natural logarithm of numeric values.	Double
LOG(number, base) number: double base: double	Calculates the specified base logarithm of numeric values.	Double
LONG(value) value: term	Displays numeric values in xsd:long format.	Long
LT(value1, value2) value1: term value2: term	Performs a less than (<) comparison between value1 and value2.	Boolean
MAX(value, ...) value: term	Aggregate function that calculates the maximum values for each aggregate group.	Term

Function (syntax) & Argument Data Types	Description	Return Type
MAXVAL(value, ...) value: literal	Computes the maximum values for the specified arguments.	Literal
MEDIAN(value) value: numeric	Aggregate function that calculates the median value for each aggregate group.	Numeric
MIN(value, ...) value: term	Aggregate function that calculates the minimum values	Term
MOD(number, divisor) number: integer divisor: integer	Calculates the modulo or remainder of the division between two numeric values.	Integer
MODE(value) value: term	Aggregate function that returns the number that occurs most frequently in each aggregate group.	Numeric
MODEPERCENT(value) value: numeric	Aggregate function that calculates the percentage of the values that belong to the mode.	Numeric
MULTIPLY(value1, value2) value1: numeric value2: numeric	Multiplies value1 by value2.	Numeric
NPV(rate, year, value) rate: numeric year: numeric value: numeric	Calculates the net present value of an investment by using a discount rate and a series of future payments (negative values) and income (positive values).	Numeric
PI()	Returns the value for PI.	Double
POWER(number, power) number: numeric power: numeric	Raises the specified number to the specified power.	Double

Function (syntax) & Argument Data Types	Description	Return Type
QUOTIENT(numerator, denominator) numerator: numeric denominator: numeric	Calculates the quotient for the specified values.	Integer
RAD(angle) angle: double	Converts degrees to radians.	Double
RAND()	Returns a random double value between 0 and 1.	Double
RANDBETWEEN(bottom, top) bottom: numeric top: numeric	Returns a random integer between the specified values (inclusive). If the input values are decimal types, Anzo returns a random integer between the ceil(bottom) and floor(top).	Integer
ROUND(number) number: double	Rounds a numeric value to the nearest integer.	Integer
ROUNDDOWN(number, num_digits) number: numeric num_digits: integer	Rounds a numeric value down by the specified number of digits.	Numeric
ROUNDUP(number, num_digits) number: numeric num_digits: integer	Rounds a numeric value up by the specified number of digits.	Numeric
SIN(angle) angle: double	Calculates the sine of the specified value.	Double
SQRT(number) number: double	Calculates the square root of the specified number.	Double

Function (syntax) & Argument Data Types	Description	Return Type
STDEV(number) number: numeric	Calculates the standard deviation of a group of numbers.	Numeric
STDEVP(number) number: numeric	Calculates the standard deviation product of a group of numbers.	Numeric
SUM(number) number: numeric	Calculates the sums of the values that exist for the selected property.	Integer
SUMIF(values_to_test, criterion, values_to_sum) values_to_test: term criterion: term values_to_sum: numeric	Calculates the sums of the values that match the specified criterion.	Integer
SUMIFS(values_to_sum, values_to_test, criteria, ...) values_to_sum: numeric values_to_test: term criteria: term	Similar to the SUMIF function but enables you to specify multiple criteria.	Integer
SUMPRODUCT(number) number: numeric	Calculates the sum of the product of the specified numeric values.	Numeric
SUMSQ(number) number: numeric	Calculates the square root of each number in the group and adds them all together.	Numeric
TAN(angle) angle: double	Calculates the tangent of the specified angle.	Double
TIME(hour, minute, second) hour: integer minute: integer second: integer	Converts the specified hour, minute, and second integer values as a time value.	Time

Function (syntax) & Argument Data Types	Description	Return Type
VAR(number) number: numeric	Calculates the variance for a group of numbers, i.e., how widely the values vary from the average of the values.	Numeric
VARP(number) number: numeric	Calculates the variance for a sample group of numbers, i.e., how widely the values vary from the average of the values.	Numeric

Functions on Dates, Times, and Durations

The table below details the Anzo functions for date, time, and duration data types.

Function (syntax) & Argument Data Types	Description	Return Type
DATE(year, month, day) year: integer month: integer day: integer	Groups results under the date (year, month, day) that you type.	Date
DATEPART(date_value) date_value: date or dateTime	Returns the date portion of a dateTime value.	Date
DATETIME(value) value: datetime, string (the string is parsed to datetime), or long (time in milliseconds since epoch)	Returns the appropriate dateTime based on the specified input value.	Date
DATEVALUE(date_text) date_text: term	Groups results under the specified literal date value.	Date
DAY(date_value) date_value: date or dateTime	Returns as an integer (1-31) the day portions of the values that exist for the selected property.	Integer
DAYSFROMDURATION(value) value: duration or numeric	Returns the day portion of duration values.	String

Function (syntax) & Argument Data Types	Description	Return Type
DUR_TO_MILLIS(value) value: date or dateTime	Displays date or date time values as the time in milliseconds.	Long
DURATION(number) number: long	Displays the specified values in duration format (PnYnMnDTnHnMnS).	Duration
DURATIONFORMAT(millis, format) millis: numeric format: duration	Displays the specified values in duration format and groups the results into the list of durations. This function enables you to specify the duration format to use. The default format is H:mm:ss.SSS.	String
DURATIONPERIODFORMAT (start, end, format) tart: duration or numeric end: duration or numeric format: duration	Calculates the duration between the specified start and end values. This function also enables you to specify the duration format. The default format is PYYYYMMDDThhmmss.SSS.	String
FORMATDATE(value, format) value: term format: string	Formats a numeric or date value into date text.	String
GT(value1, value2) GT functions on numerics, booleans, dateTimes, and terms in this priority order	Performs a greater than (>) comparison between value1 and value2.	Boolean
HOUR(value) value: time or dateTime	Returns the hour portions of the values that exist for the selected property.	Integer

Function (syntax) & Argument Data Types	Description	Return Type
MASKEDDATETIME(value, year, month, day, hour, minute, second, millis) value: dateTime year: boolean month: boolean day: boolean hour: boolean minute: boolean second: boolean millis: boolean	Displays date or datetime values as the specified date and time.	DateTime
MILLIS(value) value: date or dateTime	Displays date or datetime values as the time in milliseconds.	Date
MINUTE(value) value: time	Returns the minute portions of the values that exist for the selected property.	Integer
MONTH(value) value: date	Returns as an integer (1-12) the month portions of the values that exist for the selected property.	Integer
NOW(timezone) timezone: string	Returns the current date and time.	DateTime
NOWMILLIS()	Returns the current date and time in epoch milliseconds.	Long
PARSEDATETIME(date_string, output_type) date_string: string output_type: URI	Returns the specified string or literal value as a date, time, or datetime value.	DateTime
SECOND(time_value) time_value: date or dateTime	Returns the second portions of the values that exist for the selected property.	Integer

Function (syntax) & Argument Data Types	Description	Return Type
TIME(hour, minute, second) hour: integer minute: integer second: integer	Converts the specified hour, minute, and second integer values as a time value.	Time
TIMEPART(value) value: string, time, or dateTime	Returns the appropriate time based on the input value.	Time
TODAY()	Returns today's date.	Date
WEEKDAY(date_value, return_type) date_value: date return_ype: integer	Returns the day of the week that corresponds to the specified date.	Integer
WEEKNUM(date_value, return_type) date_value: data return_type: integer	Returns the week of the year that the specified date occurs in.	Integer
YEAR(date_value) date_value: date or dateTime	Returns as an integer (1900-9999) the year portions of the values that exist for the selected property.	Integer
YEARMONTH(date_value) date_value: date	Returns the year-month of the specified date.	DateTime

Functions on Boolean Values

The table below details the Anzo functions for boolean data types.

Function (syntax) & Argument Data Types	Description	Return Type
NOT(value) value: boolean	Performs logical negation on the specified expression.	Boolean

Function (syntax) & Argument Data Types	Description	Return Type
OR(logical1, logical2) logical1: boolean logical2: boolean	Calculates the logical OR of the input values.	Boolean

Window Aggregate Functions

Window aggregates operate on a particular partition or window of the result set. Unlike grouped aggregate functions that group the result set and return a single row, window aggregates retain the resulting rows and return a value for each row.

Except for WINDOW_NTILE, WINDOW_PERCENTILE, and WINDOW_QUARTILE, use the following syntax for window aggregates:

```
WINDOW_FUNCTION(value, partition_over, order_by, order, start_frame,
start_frame_type, start_frame_value, end_frame_type, end_frame_value)
```

The table below lists the supported window aggregates and provides the syntax for the WINDOW_NTILE, WINDOW_PERCENTILE, and WINDOW_QUARTILE functions.

Function	Description	Return Type
WINDOW_AVG	Returns the average of the input values.	Numeric
WINDOW_COUNT	Returns the count of the specified values.	Integer
WINDOW_MAX	Returns the maximum of the input values.	Numeric
WINDOW_MIN	Returns the minimum of the input values.	Numeric
WINDOW_NTILE(ntile, value, order_by, partition_over)	Divides the rows in the partition into the specified number of ranked groups and returns the group that each value belongs to.	Numeric
WINDOW_PERCENTILE (value, order_by, partition_ over)	Like using NTILE(100), this function divides the rows in the partition into 100 ranked groups and returns the group that each value belongs to.	Numeric

Function	Description	Return Type
WINDOW_PRODUCT	Returns the product of the input values.	Numeric
WINDOW_QUARTILE(value, order_by, partition_over)	Like using NTILE(4), this function divides the rows in the partition into 4 ranked groups and returns the group that each value belongs to.	Numeric
WINDOW_SUM	Returns the sum of the input values.	Numeric

Filter Type Reference

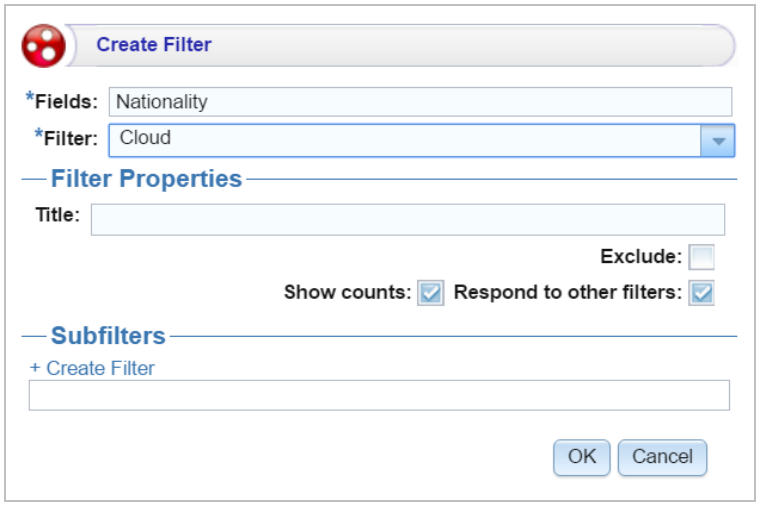
The topics in this section provide reference information for each type of filter that is available in Hi-Res Analytics dashboards:

- Cloud Filter
- Date Range Filter
- Hierarchy Filter
- Limit Filter
- List Filter
- Numeric Range Filter
- Presence Filter
- Quartile Filter
- Range Slider Filter
- Relative Time Filter
- Search Filter
- Single Select List Filter
- Types Filter

Cloud Filter

Cloud filters display values in term clouds where each term is written in a font size that represents the number of results for that value. Unlike list filters, which enable you to select and filter on multiple values at once, cloud filters allow you to filter on one value at a time. The cloud filter is available for all data types but cannot be with used relative paths, which are indicated by a path icon () in the Create Filter dialog box.

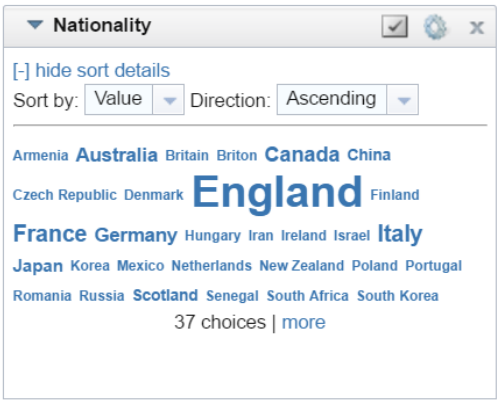
After selecting Cloud from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Exclude	Removes the selected property from the results.
Show counts	Displays the number of results for the term when you hover the pointer over a term in the cloud.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard. You can click a value in the cloud to display only those data points in the lens.

Example

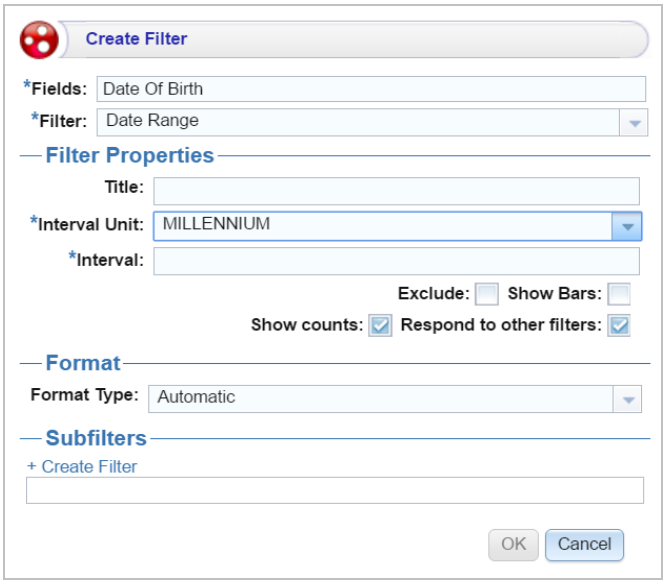


Field	Description
[+] show sort details	Reveals the following fields: **Sort by**: Select **Value** to sort string values alphabetically, or select **Count** to order results according to the total number of results for each value. **Direction**: Select **Ascending** to order results starting at the top. For strings, the alphabet starts at the top. Or select **Descending** to order results starting at the bottom.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Date Range Filter

Date Range filters are available for properties with date and time data types and enable you to define date ranges and group the results into those ranges.

After selecting the appropriate property and choosing Date Range from the Filter drop-down, configure the following properties as needed:



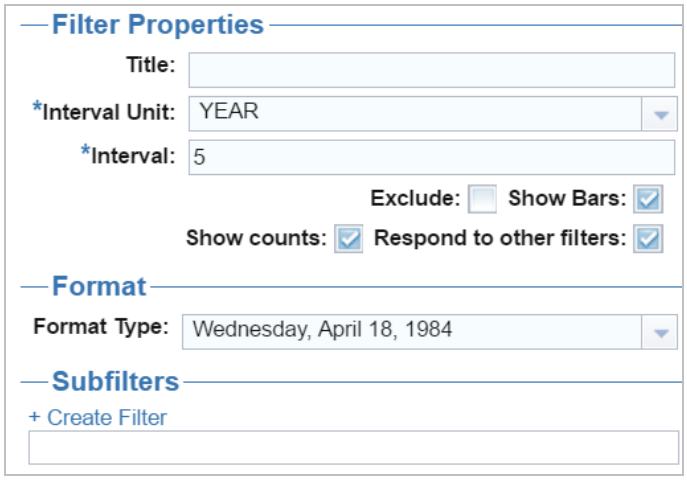
Field	Description
Title	Defines the filter title.

Field	Description
Interval Unit	Defines the unit of time for the Interval value: Millennium, Century, Decade, Year, Month, Week, or Day.
Interval	Defines the length of time in each grouping. For example, for a date field with an Interval Unit of "Decade," an Interval value of 2 creates groups of two-decade increments.
Exclude	Removes the selected property from the results.
Show bars	Displays the total values for the selected property as a bar graphic in the background of the filter.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Format Type	Defines the date format: Automatic (default), 4/18/1984, Apr 18, 1984, April 18, 1984, or Wednesday, April 18, 1984.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

This example uses the following Filter Property and Format settings:



These settings result in the following filter:

The blue bars represent the number of values for each of the grouped intervals. Each interval is a five-year period.

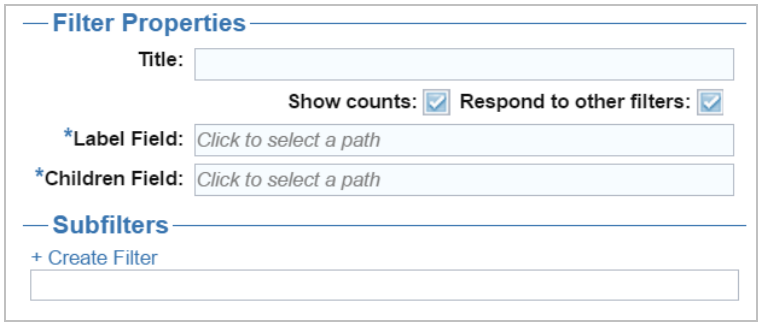
This filter has the following options:

Field	Description
[+] show sort details	Reveals the following fields: **Sort by**: Select **Value** to sort string values alphabetically, or select **Count** to order results according to the total number of results for each value. **Direction**: Select **Ascending** to order results starting at the top. For strings, the alphabet starts at the top. Or select **Descending** to order results starting at the bottom.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Hierarchy Filter

Hierarchy filters display parent and child relationships and are available for relative paths (indicated by the path icon in the Create Filter dialog box) and not properties.

After selecting Hierarchy from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Label Field	Defines a label for the values. Typically a child attribute.
Children Field	Defines the relationship or property that populates the hierarchy.
Create filter	Creates a subfilter for this filter.

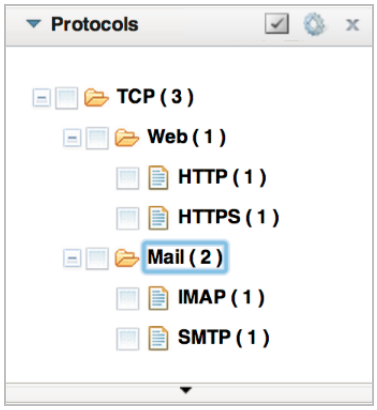
When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

This example uses an internet protocol data set with the following properties defined:

- The **Fields** path is the internet protocols.
- **Label Field** is the Label data property which is an attribute of Protocols.
- **Children Field** is the Narrower property, which directs the Protocols property to the lowest hierarchical level.

These settings result in the following hierarchy filter:

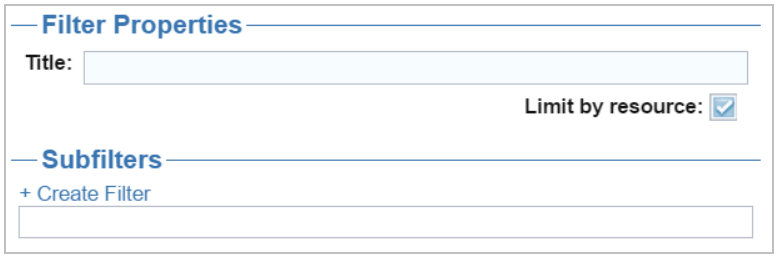


This hierarchical structure enables you to view parent and child relationships and filter the dashboard data based on these relationships.

Limit Filter

Limit filters enable you to limit the results to the specified number of largest or smallest values. You can use limit filters for any data type. For strings, results are ordered alphabetically. Largest orders by the last letters in the alphabet and Smallest orders by the first letters in the alphabet.

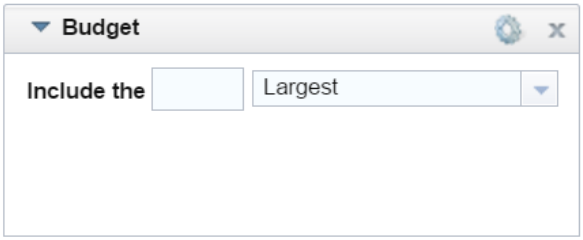
After selecting Limit from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Limit by resource	
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example



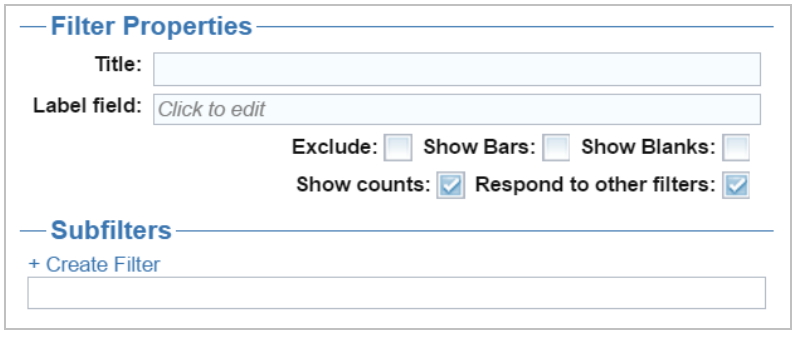
This filter has the following options:

Field	Description
Include the	Sets the number of results to filter for. Type a number and press **Enter** to filter the values.
Limit definition drop-down	Select to filter on the **Largest** or **Smallest** value of the selected property.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

List Filter

List filters display results in a list and allow you to select and filter on multiple values at a time. The list filter is available for all data types.

After selecting List from the Filter drop-down, configure the following properties as needed:

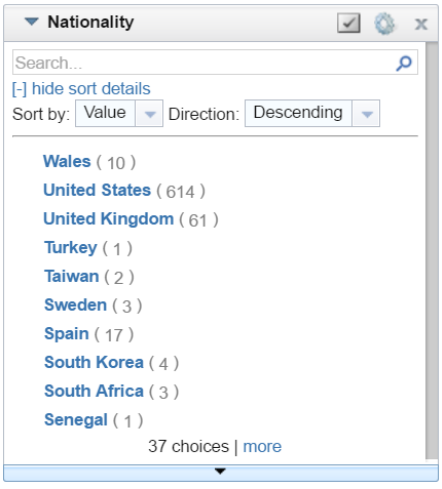


Field	Description
Title	Defines the filter title.

Field	Description
Label field	The property to show as the value for each list item in the filter if you want it to differ from the value that results from the property or relative path you chose in the Fields field.
Exclude	Removes the selected property from the results.
Show bars	Displays the total values for the selected property as a bar graphic in the background of the filter.
Show Blanks	Displays any null values for the selected property by including a “Blank” option in the filter.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

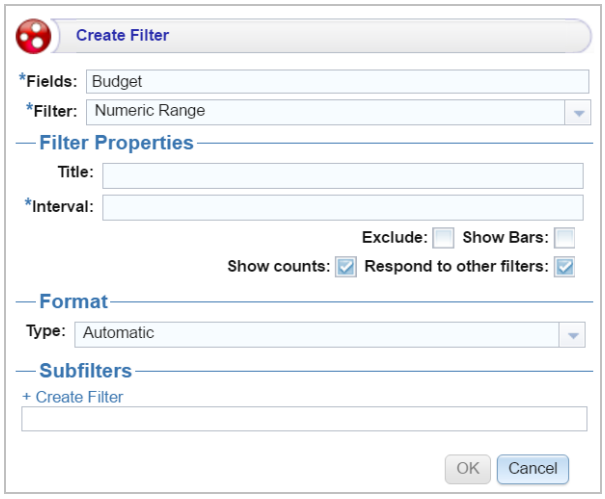


Field	Description
[+] show sort details	Reveals the following fields: **Sort by**: Select **Value** to sort string values alphabetically, or select **Count** to order results according to the total number of results for each value. **Direction**: Select **Ascending** to order results starting at the top. For strings, the alphabet starts at the top. Or select **Descending** to order results starting at the bottom.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Numeric Range Filter

Numeric Range filters are similar to Date Range filters but are available for properties with numeric (integer or double) data types. You can also perform a function on a property so that it results in a number value, such as using the COUNT function. These filters enable you to define numeric ranges and group the results into those ranges.

After selecting the appropriate property and choosing Numeric Range from the Filter drop-down, configure the following properties as needed:



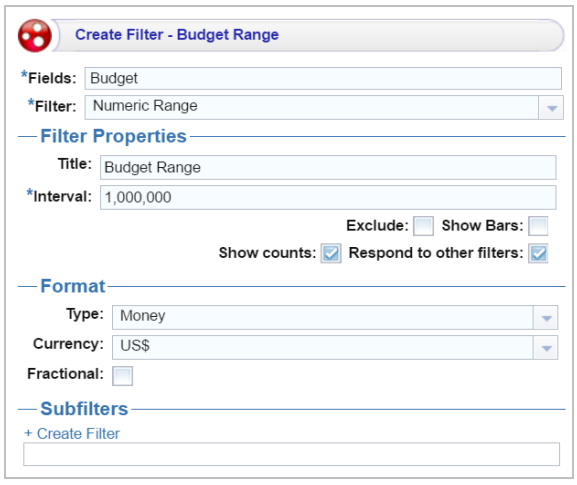
Field	Description
Title	Defines the filter title.

Field	Description
Interval	Defines the number of values in each grouping.
Exclude	Removes the selected property from the results.
Show bars	Displays the total values for the selected property as a bar graphic in the background of the filter.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Format Type	Defines the date format: Automatic (default), 4/18/1984, Apr 18, 1984, April 18, 1984, or Wednesday, April 18, 1984.
Create filter	Creates a subfilter for this filter.

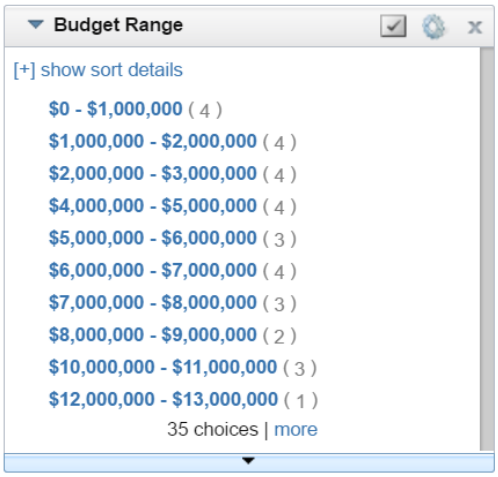
When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

This example uses the following Filter Property and Format settings:



These settings result in the following filter. Each interval is $1,000,000:



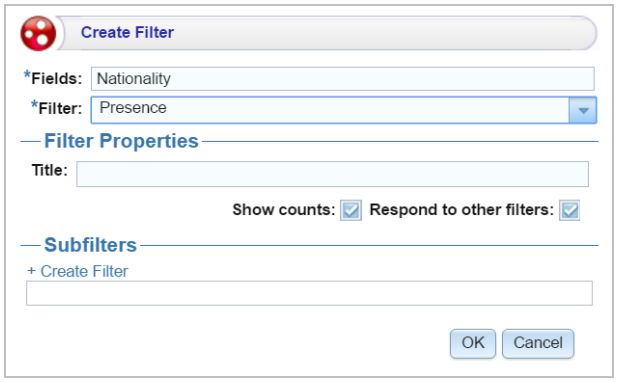
This filter has the following options:

Field	Description
[+] show sort details	Reveals the following fields: **Sort by**: Select **Value** to sort string values alphabetically, or select **Count** to order results according to the total number of results for each value. **Direction**: Select **Ascending** to order results starting at the top. For strings, the alphabet starts at the top. Or select **Descending** to order results starting at the bottom.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Presence Filter

Presence filters indicate whether a specified value exists. This filter is useful for finding records that exclude a particular value. Presence filters are available for relative paths and properties of all data types.

After selecting a property and choosing Presence from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

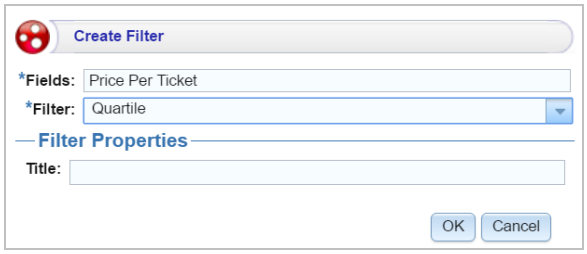
Most presence filters look like the following example:

Field	Description
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Quartile Filter

Quartile filters group and rank the values for a property into four equal ranges. This filter requires a property with a numeric or date data type and is not available for relative paths.

After selecting the appropriate property and choosing Quartile from the Filter drop-down, Quartile filters do not require additional configuration.



You can type a title in the Title field, and then click **OK** to create the filter. The new filter appears on the dashboard.

Example

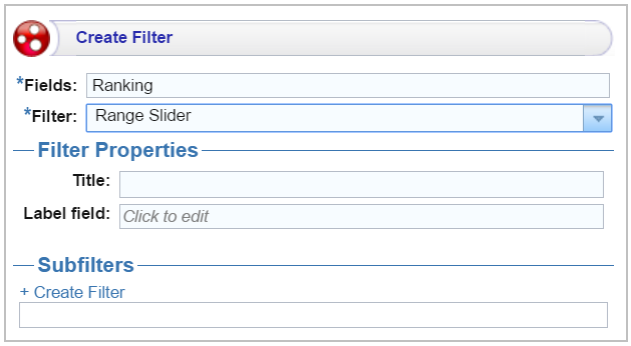
The example below shows the quartiles for the Price Per Ticket property. Anzo groups values into equal ranges by rank, from the most expensive tickets to the least expensive.

Field	Description
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Range Slider Filter

Range Slider filters display a slider control that enables you to filter results by a range that you specify by setting a minimum and maximum value. The Range Slider filter requires a property with numeric or date data type, or a function resulting in a number, such as COUNT.

After selecting the appropriate property and choosing Range Slider from the Filter drop-down, configure the following properties as needed:

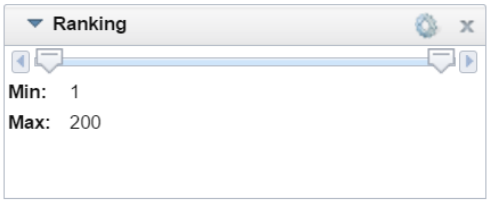


Field	Description
Title	Defines the filter title.
Label field	The property to show as the value for each list item in the filter if you want it to differ from the value that results from the property or relative path you chose in the Fields field.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

The following example shows the Range Slider filter that results from the settings shown above. To further narrow the possible results from this filter, you can drag the left-hand slider to the right to choose a new minimum value and drag the right-hand slider to the left to choose a new maximum value.



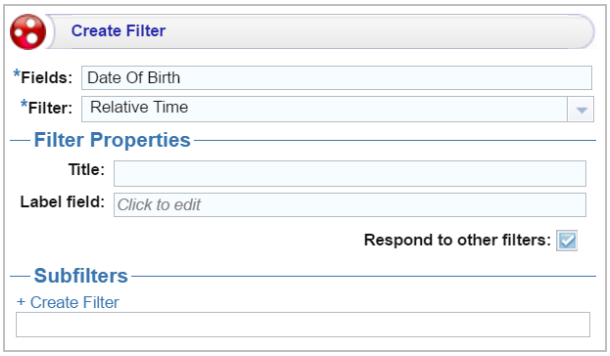
Field	Description
Minimum slider	Drag the left-hand slider to specify the minimum value, the beginning of the range.
Maximum slider	Drag the right-hand slider to specify the maximum value, the end of the range.
Designer ( )	Click to open the Designer and reconfigure the filter.

Field	Description
Close (x)	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Relative Time Filter

Relative Time filters are available for properties with date data types and enable you to filter for records that fall into the specified time increment relative to the current time.

After selecting the appropriate property and choosing Relative Time from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Label field	The property to show as the value for each list item in the filter if you want it to differ from the value that results from the property or relative path you chose in the Fields field.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

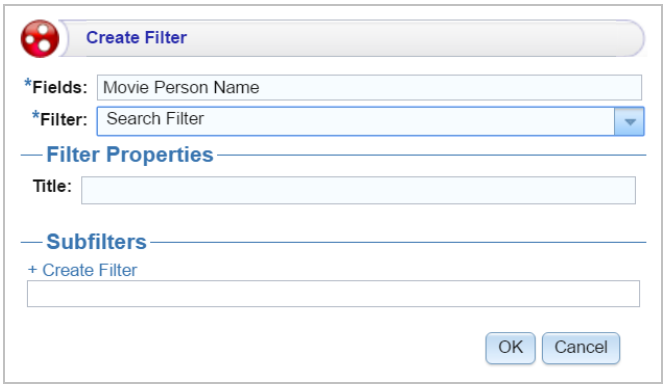
Example

Field	Description
Last or Next drop-down	Select the relative time direction: Last or Next.
Number field	Specify a number to represent the amount of time.
Time measurement	Select the time increment: years, quarters, months, weeks, days, hours, minutes, seconds, or milliseconds.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Search Filter

Search filters are available for all data types and enable you to search for values in the selected property.

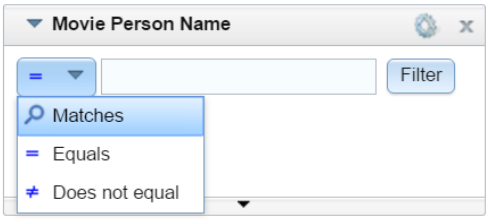
After selecting the desired property and choosing Search from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example

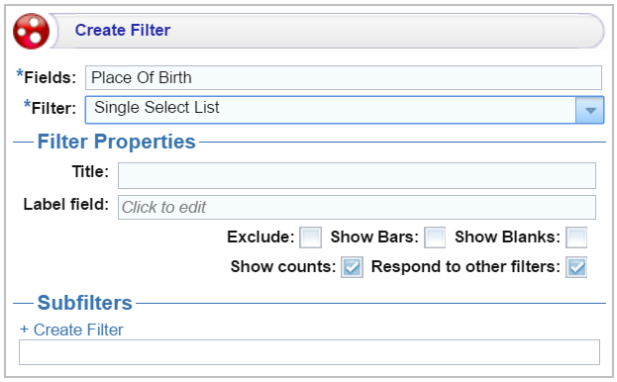


Field	Description
Search type	The type of match to perform: Matches (including partial matches), Equals (exact match), or Does not equal.
Search criteria	The value to search for.
Filter	Click to perform the search.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Single Select List Filter

Single Select List filters are similar to List filters but only allow you to select and filter on one value from the list at a time. This filter is available for properties of all data types but is not available for relative paths.

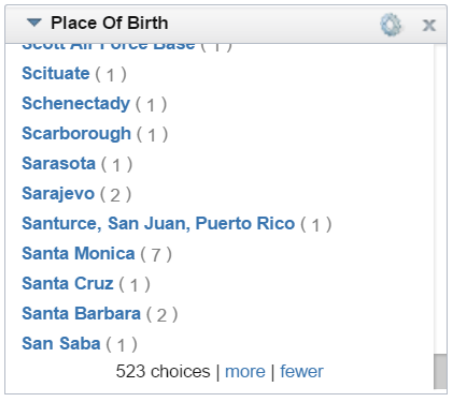
After selecting the appropriate property and choosing Single Select List from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Label field	The property to show as the value for each list item in the filter if you want it to differ from the value that results from the property or relative path you chose in the Fields field.
Exclude	Removes the selected property from the results.
Show bars	Displays the total values for the selected property as a bar graphic in the background of the filter.
Show blanks	Displays any null values for the selected property by including a “Blank” option in the filter.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example



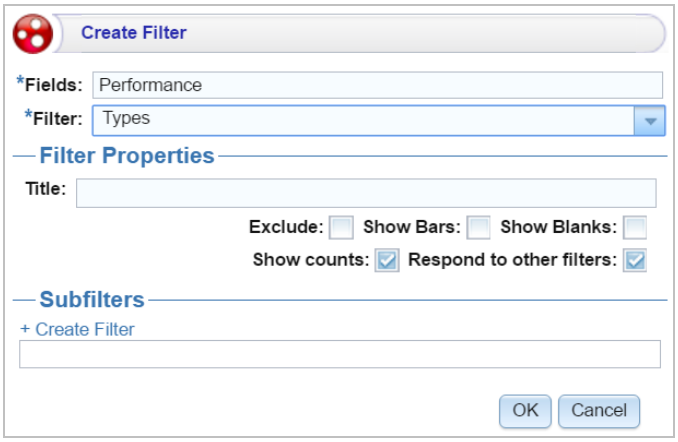
Field	Description
Designer ( )	Click to open the Designer and reconfigure the filter.

Field	Description
Close (x)	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Types Filter

Types filters enable you to filter data according to the classes defined by a relative path. These filters are available only for relative paths (indicated by the path icon in the the Create Filter dialog box) and not properties.

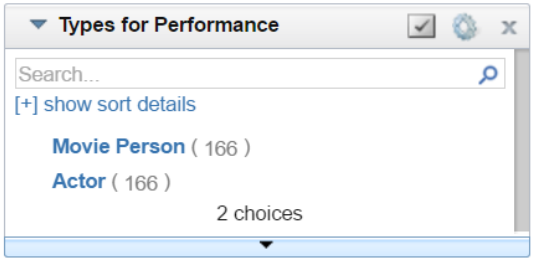
After choosing the appropriate path and selecting Types from the Filter drop-down, configure the following properties as needed:



Field	Description
Title	Defines the filter title.
Exclude	Removes the selected property from the results.
Show bars	Displays the total values for the selected property as a bar graphic in the background of the filter.
Show blanks	Displays any null values for the selected property by including a “Blank” option in the filter.
Show counts	Displays the number of results for the value.
Respond to other filters	Indicates whether the results of this filter change based on selections in other filters on the dashboard.
Create filter	Creates a subfilter for this filter.

When you have completed the configuration, click **OK** to create the filter. The new filter appears on the dashboard.

Example



Field	Description
Search	Enables you to search for a value in the resulting list.
[+] show sort details	Reveals the following fields: **Sort by**: Select **Value** to sort string values alphabetically, or select **Count** to order results according to the total number of results for each value. **Direction**: Select **Ascending** to order results starting at the top. For strings, the alphabet starts at the top. Or select **Descending** to order results starting at the bottom.
Designer ( )	Click to open the Designer and reconfigure the filter.
Close ( )	Click to close the filter and remove it from the dashboard. This action cannot be undone.

Lens Type Reference

The topics in this section provide reference information for each type of lens that is available in Hi-Res Analytics dashboards.

- AnzoKO Web Page Lens
- Chart Lenses
- Dashboard Lens
- Drill Down Lens
- Form Lens
- List Lens
- Network Navigator Lens
- Query Lens

- Resource Tree Navigator Lens
- Table Lens
- Web Page Lens

AnzoKO Web Page Lens

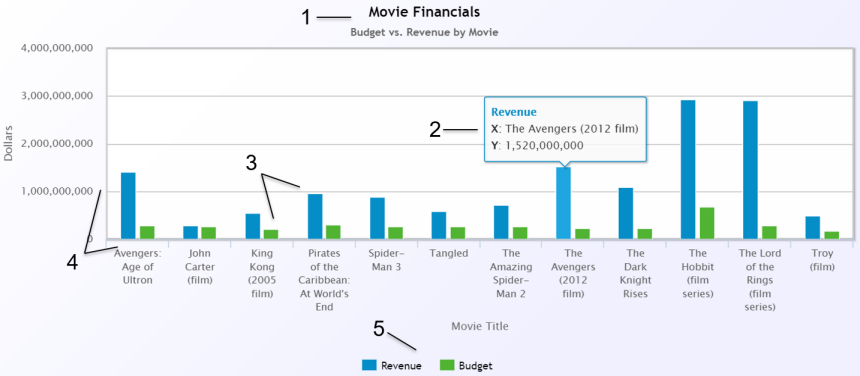
The AnzoKO Web Page lens includes the Knockout JavaScript framework and enables you to create visualizations of RDF resources and metadata using knockout.js-like syntax without needing to write additional JavaScript to declare which parts of the data to render in which sections of the HTML.

Chart Lenses

Anzo Hi-Res Analytics employs the Highcharts API to provide interactive chart lenses. This section provides information about chart concepts and describes the general, shared chart settings.

Chart Concepts

This section describes the high-level, basic chart options that you can configure.

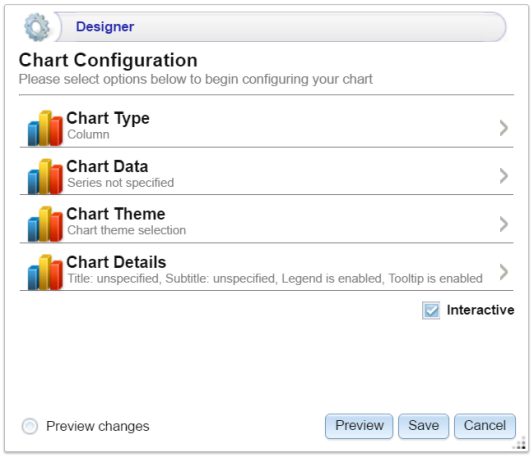


1. **Title and Subtitle**: You can configure titles and subtitles for all charts.
2. **Tooltips**: Configurable tooltips display details when users place their cursor over an item in the chart.
3. **Series**: A series is a set of properties and formulas used to display data on the chart. You can include multiple data series and configure each series individually.
4. **Axes**: The X and Y axes define the horizontal and vertical coordinates for displaying the data.
5. **Legend**: The legend differentiates each series in the chart. You can also click a series in the legend to show or hide that series in the chart.

General Chart Configuration

When creating a chart, select the chart type that best suits your intended data presentation. All charts allow you to add multiple data series and configure each series individually.

The Chart Configuration screen is the initial screen in the Designer window that appears after you name and create the chart lens.



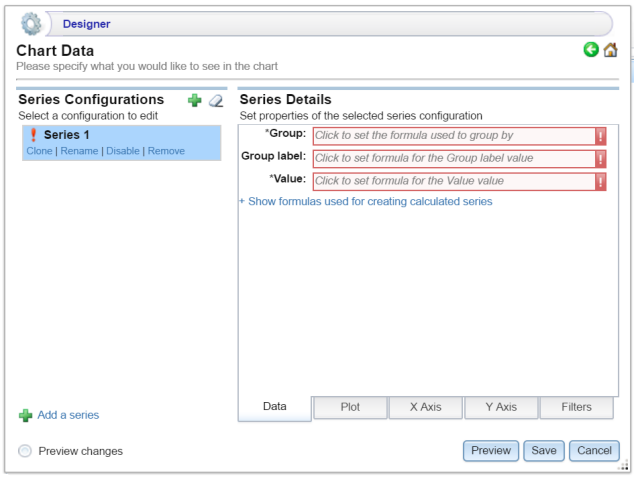
- **Chart Type**: Enables you to select the type of chart that you want to display, such as column, pie, or line.
- **Chart Data**: Enables you to specify the data that will populate the chart.
- **Chart Theme**: Enables you to select a theme or color scheme for the chart.
- **Chart Details**: Enables finer-grained customization than the Chart Theme settings. You can further customize the chart design by adding details such as a chart title and subtitle and modifying styles, fonts, legend, and tool-tip formats.

Chart Designer Interface Functions

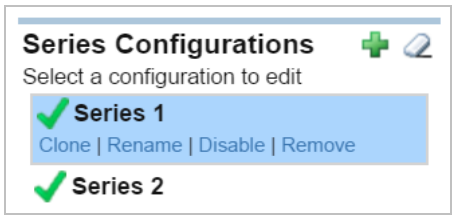
Icon	Description
	Click the eraser icon to erase all series. This action cannot be undone.
	Click the back icon to return to the previous screen.
	Click the home icon to return to the Chart Configuration screen.
	Click the plus icon to add a series.

Chart Data

The Chart Data screen allows you to configure the Properties and formulas that populate your chart. Most charts share the same settings.



Series Configurations



The Series Configurations section contains settings to manage the data series. Click a series to select it. The Series Details appear in the section to the right. The Series Configuration options are:

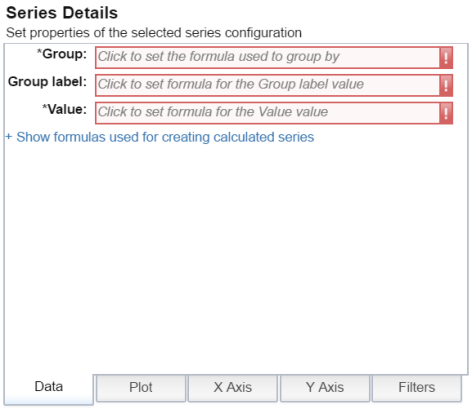
- **Clone**: Creates a new series with the same settings.
- **Rename**: Renames the series.
- **Disable**: Removes series data from the chart without deleting it.
- **Remove**: Deletes the series. You must have at least one active series for a functioning chart. Removing a series cannot be undone.

Series Details

Depending on the chart type, five tabs appear at the bottom of the Series Details screen:

Data

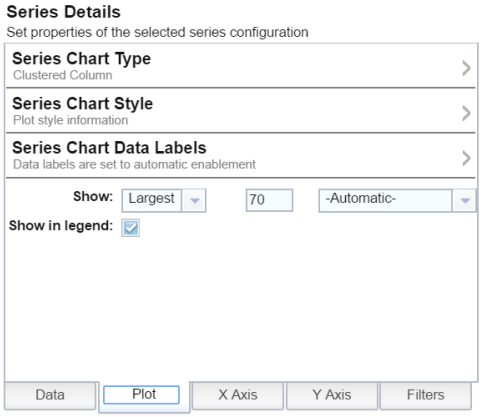
The Data tab defines properties and formulas used to populate the chart.



Field	Description
Group	Defines property and optional formulas for grouping data.
Group label	Typically the same as Group. Defines the properties and formulas to serve as the group label.
Value	Defines the property and optional forrmulas to populate the values in the chart.
Show formulas used for...	Creates a calculated series using the following fields: **Series Group**: Selects property and functions or formulas used to group data in addition to the Group setting. **Series Label**: Typically the same as Series Group. The property to use as the series label.

Plot

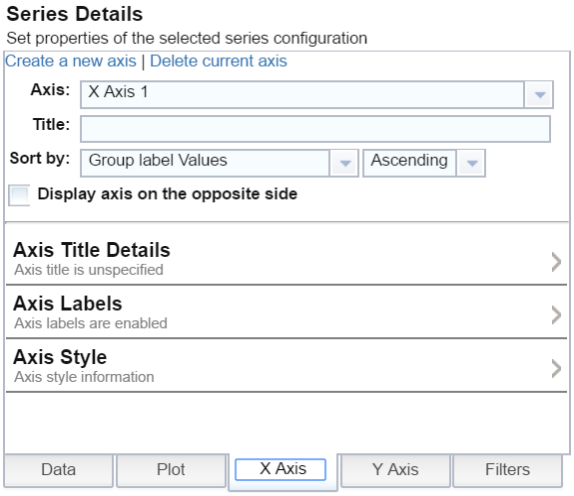
The Plot tab defines chart formatting, including data labels, legends, and other display options.



Field	Description
Series Chart Type	Enables you to select a chart type for the series.
Series Chart Style	Enables you to change chart formats such as fill colors and border lines.
Series Chart Data Labels	Enables you to change chart data label formats.
Show	Enables you to define a portion of the data to display based on the largest or smallest Group Labels or Values.
Show in legend	Indicates whether to show the series' name in the legend.

X-Axis

The X-Axis tab defines formats and labels for the X axis values.

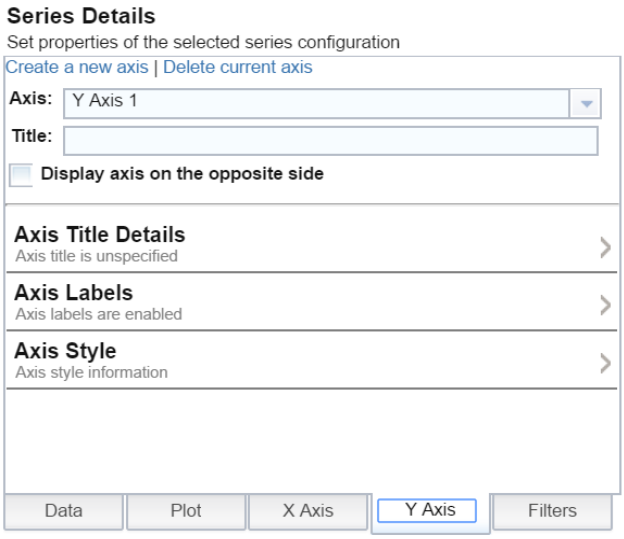


Field	Description
Axis	Enables you to select the X axis to use for the series if multiple axes exist.
Title	Defines the title for the X axis.
Sort by	Enables you to select the sort value (either Group label or Value) for string data types.
Display axis on the opposite side	Moves the X axis to the opposite side of the chart.

Field	Description
Axis Title Details	Enables you to change the format for the axis title.
Axis Labels	Enables you to change the format for axis labels.
Axis Style	Enables you to make axis style changes.

Y-Axis

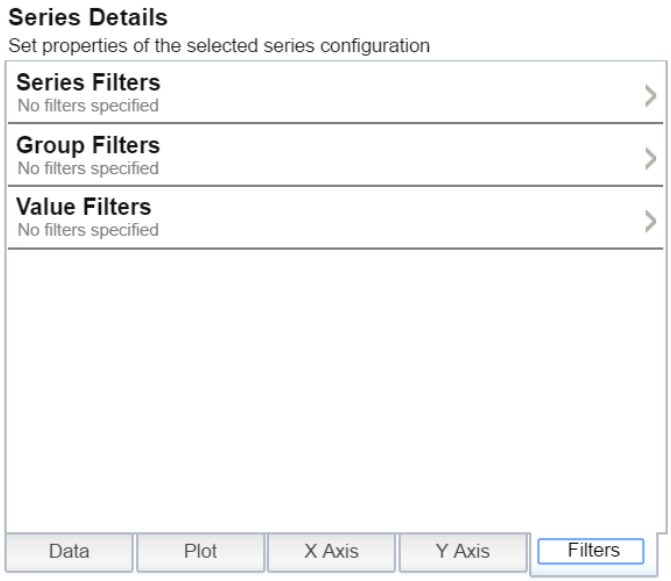
The Y-Axis tab defines formats and labels for the Y axis values.



Field	Description
Axis	Enables you to select the Y axis to use for the series if multiple axes exist.
Title	Defines the title for the Y axis.
Display axis on the opposite side	Moves the Y axis to the opposite side of the chart.
Axis Title Details	Enables you to change the format for the axis title.
Axis Labels	Enables you to change the format for axis labels.
Axis Style	Enables you to make axis style changes.

Filters

The Filters tab defines any filters to apply to this lens.



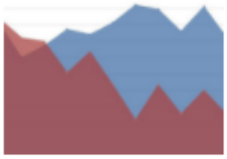
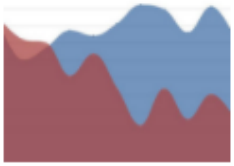
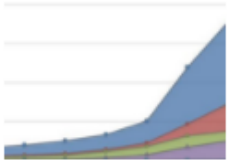
Field	Description
Series Filters	Enables you to define filters that apply to the entire series.
Group Filters	Enables you to define filters that apply only to the Group values.
Value Filters	Enables you to define filters that apply only to the Value values.

For more information about Series Details settings, see the documentation for specific lens types:

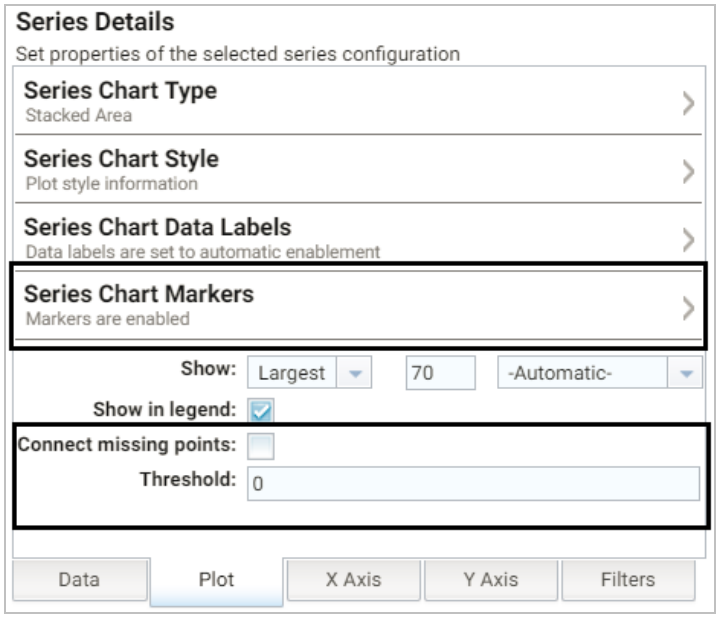
- Area Chart
- Bar Chart
- Bubble Chart
- Column Chart
- Funnel Chart
- Heat Map
- Line Chart

Area Chart

Area charts are useful for emphasizing trends. Area charts are similar to line charts but have the added ability to display stacked data series. There are five types of area charts:

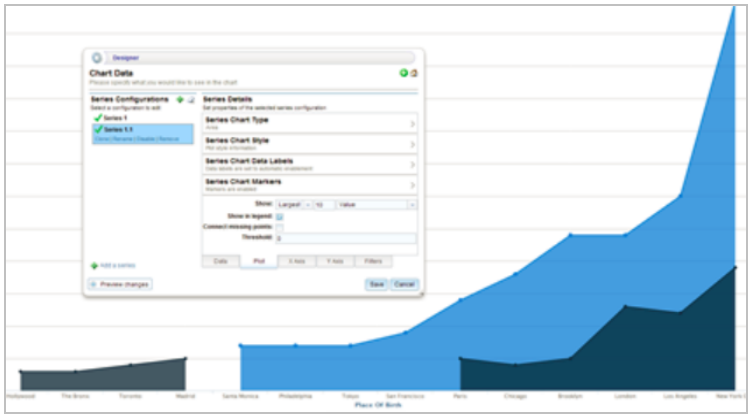
Example	Type	Description
	**Area**	Connects value points on the chart with straight lines and shades the area below the line.
	**Step Area**	Connects value points on the chart with short horizontal steps and shades below the line. This chart emphasizes the extent of value change by expanding the data points across the x axis.
	**Area Spline**	Connects value points on the chart with curved lines and shades the area below the line.
	**Stacked Area**	Connects value points on the chart with straight lines and shades the area below the line. Add a Series Group to define the groups within the totals. Hover the mouse pointer over a colored area to view the value.
	**100% Stacked Area**	Compares each value as a percentage of the total and shades the area below each series. Add a Series Group to define the groups presented as a percentage within the total.

In addition to the General Chart Configuration options, the Area Chart Designer includes the following area-chart-specific settings on the Plot tab:



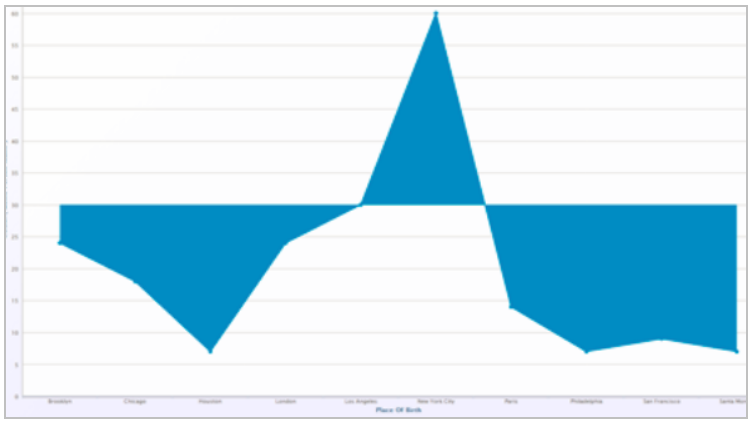
- **Series Chart Markers**: The options in this category enable you to customize the data points that appear on the series line:
  - **Enabled**: Enables or disables series chart markers.
  - **Symbol**: Selects a symbol to mark data points.
  - **Marker Radius**: Defines the marker size in pixels.
  - **Fill Color**: Defines the marker color.
  - **Outline Thickness**: Defines the thickness of the marker outline.
  - **Outline Color**: Defines the color of the marker outline.
- **Connect missing points**: Selecting this option connects the graph line across missing points.

  For example, selecting **Connect missing points** for the example below would connect the two dark blue areas by filling in the space between them.



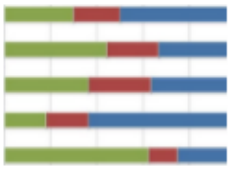
- **Threshold**: Defines the Y axis value to use as a base (starting point) for the shaded area. For example, a Threshold of 0 begins all shading at the value 0. A Threshold of 10 begins the shading at 10 and draws the area

chart above or below the threshold as required. For example, the image below shows an area chart with a threshold of 30.



Bar Chart

There are three types of bar charts:

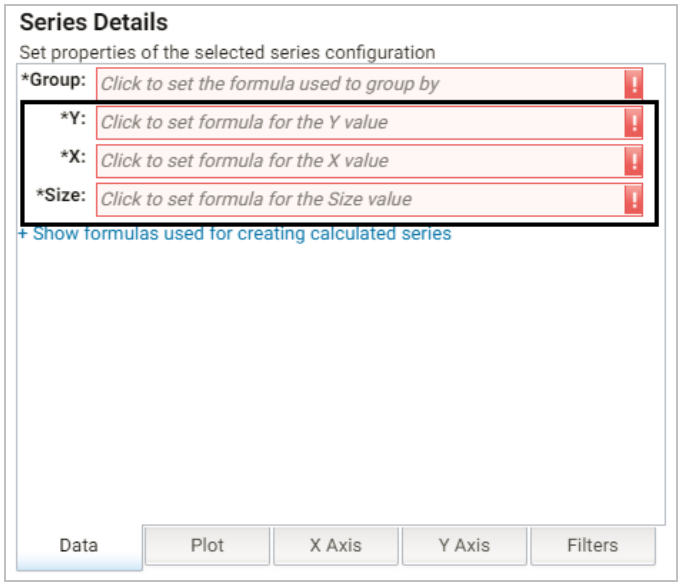
Example	Type	Description
	**Clustered Bar**	Compares values across categories.
	**Stacked Bar**	Compares the contribution of each value to a total across categories. Add a Series Group to define the groups within the totals. Hover the mouse pointer over a colored area to view the value.
	**100% Stacked Bar**	Compares each value as a percentage of the total. Add a Series Group to define the groups within the total.

The Bar Chart Designer uses the General Chart Configuration options.

Bubble Chart

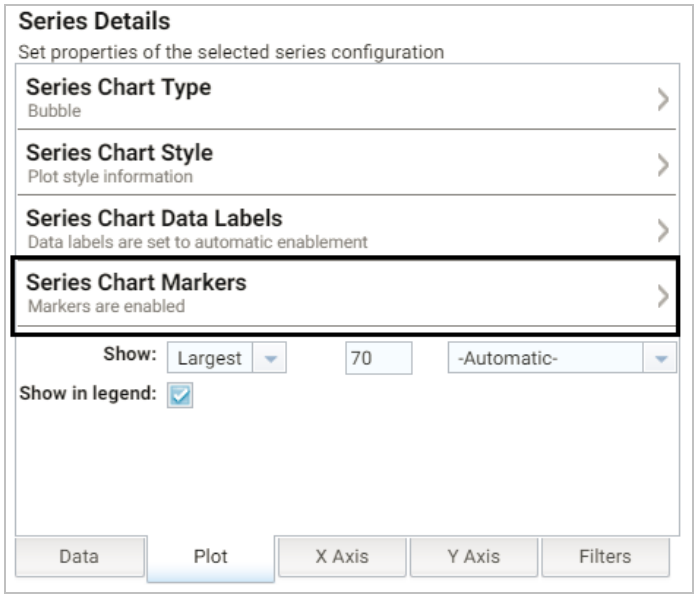
Bubble charts are useful for displaying data that has a third dimension. Bubble charts plot points for the X axis, Y axis, and represent relative size.

In addition to the General Chart Configuration options, the Bubble Chart Designer includes the following bubble-chart-specific settings on the Data tab:



- **Y**: Selects the Y axis values.
- **X**: Selects the X axis values.
- **Size**: Selects the property to use to determine the proportionate bubble size.

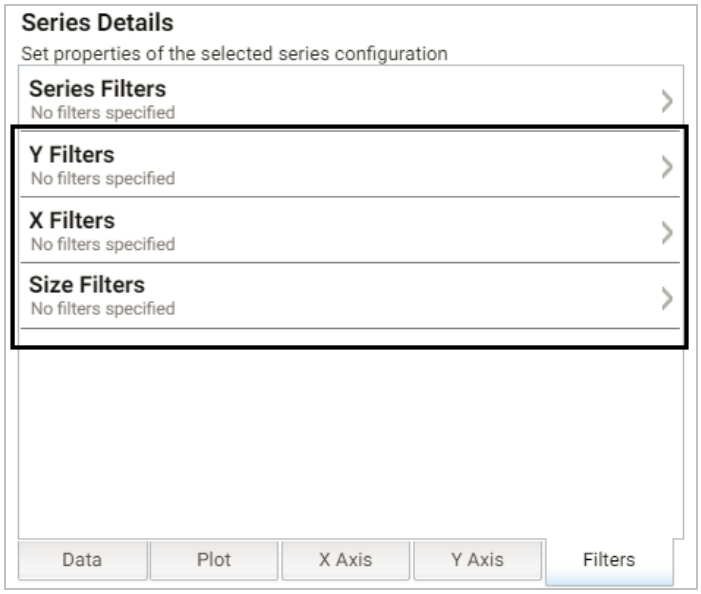
The Bubble Chart Designer also includes the following bubble-chart-specific settings on the Plot tab:



Series Chart Markers: The options in this category enable you to customize the data points, such as outlines, that appear on the bubbles:

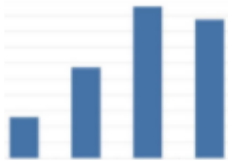
- **Enabled**: Enables or disables series chart markers.
- **Symbol**: Selects a symbol to mark data points.
- **Marker Radius**: Defines the marker size in pixels.
- **Fill Color**: Defines the marker color.
- **Outline Thickness**: Defines the thickness of the marker outline.
- **Outline Color**: Defines the color of the marker outline.

The Bubble Chart Designer enables you to create filters for the X, Y, and Z (Size) axes from the Filters tab.



Column Chart

There are three types of column charts:

Example	Type	Description
	**Clustered Column**	A basic column chart that compares values across categories.
	**Stacked Column**	Compares the contribution of each value to a total across categories. Add a Series Group to define the groups within the totals. Hover the mouse pointer over a colored area to view the value.
	**100% Stacked Column**	Compares each value as a percentage of the total. Add a Series Group to define the groups within the total.

The Column Chart Designer uses the General Chart Configuration options.

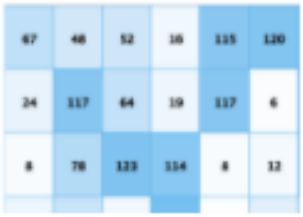
Funnel Chart

Funnel charts display a wide area at the top, with other data area proportionally smaller below it.

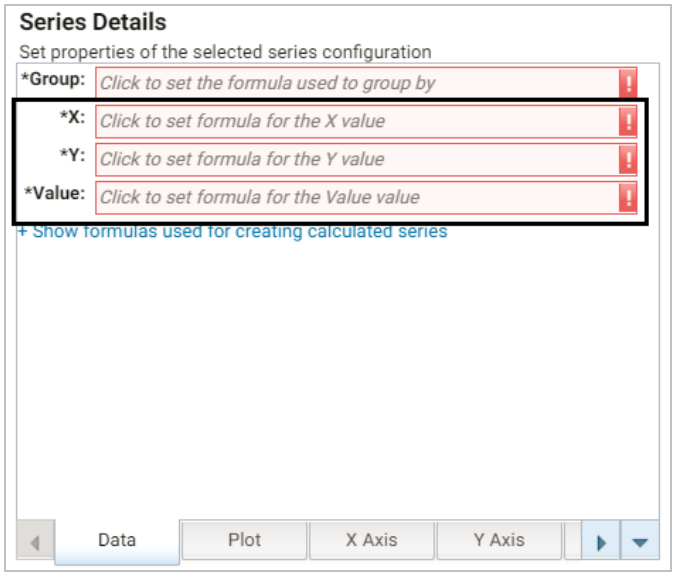
The Funnel Chart Designer uses the General Chart Configuration options.

Heat Map

Heat maps display data in tabular format within defined value ranges, such as low, medium, or high. Data points are rendered as a block of color depending on where they fall in the range.

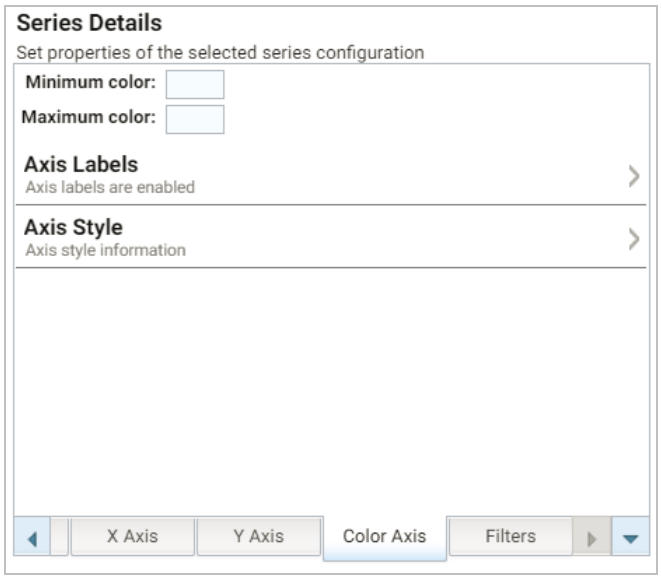


In addition to the General Chart Configuration options, the Heat Map Chart Designer includes the following heat-map-chart-specific settings on the Data tab:



- **X**: Selects the X axis values.
- **Y**: Selects the Y axis values.
- **Value**: Selects the property to use for the value range.

The Heat Map Designer also includes a Color Axis tab that enables you to customize the value range block colors and axis labels and styles.

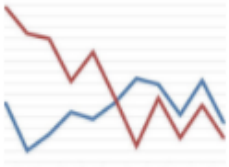
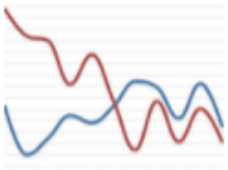
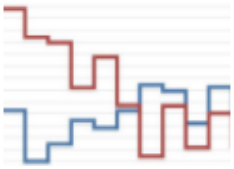


- **Minimum color**: The color to use for values in the minimum range.
- **Maximum color**: The color to use for values in the maximum range.

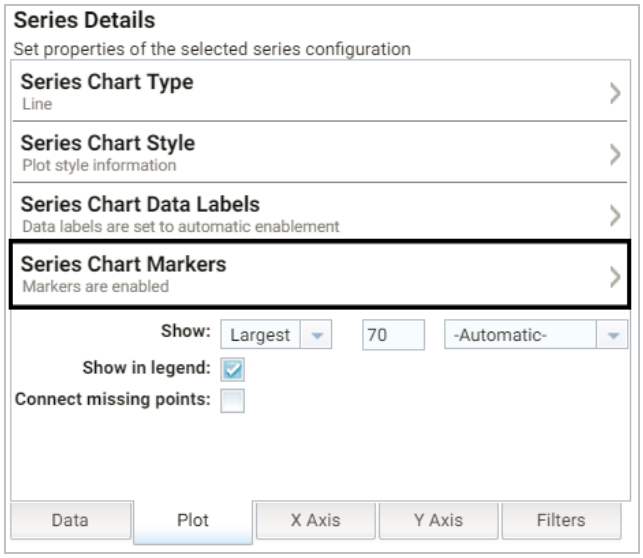
- **Axis Labels**: Enables you to customize the styles of the labels on the X and Y axes.
- **Axis Style**: Enables you to customize axis styles such as grid and tick lines.

Line Chart

There are three types of line charts:

Example	Type	Description
	**Line**	Connects value points with straight lines.
	**Spline**	Connects value points with curved lines.
	**Step Line**	Connects value points with short horizontal steps. This chart emphasizes the extent of value change by expanding the data points across the X axis.

In addition to the General Chart Configuration options, the Line Chart Designer includes the following line-chart-specific settings on the Plot tab:



Series Chart Markers: The options in this category enable you to customize the data points on the lines:

- **Enabled**: Enables or disables series chart markers.
- **Symbol**: Selects a symbol to mark data points.
- **Marker Radius**: Defines the marker size in pixels.
- **Fill Color**: Defines the marker color.
- **Outline Thickness**: Defines the thickness of the marker outline.
- **Outline Color**: Defines the color of the marker outline.

Dashboard Lens

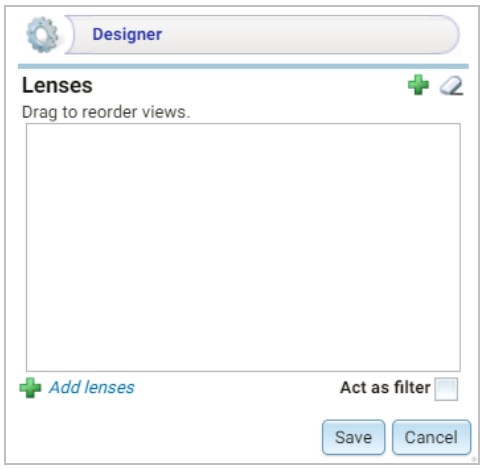
Dashboard lenses display a dashboard within a dashboard.

Drill Down Lens

Drill Down lenses combine other lenses into a hierarchical interface. Clicking on an object in one lens opens the next lens in successive order.

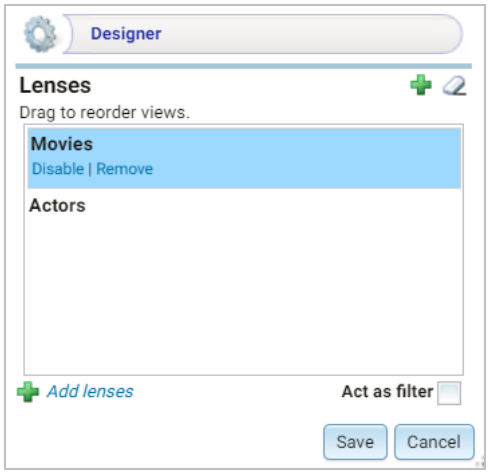
Drill Down Lens Configuration

The drill down lens Designer does not require any property selections or format configurations. Instead, you configure each of the lenses that you choose.



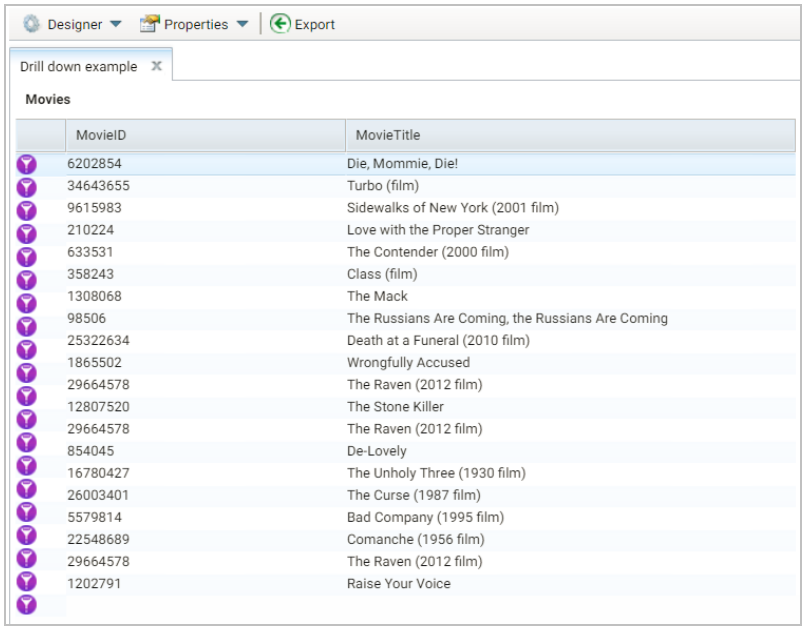
Click the plus icon ( ) at the top or bottom of the Designer to add lenses. When you finish adding lenses, click **Save**.

The lens listed first becomes the lens with the drill down functionality. Clicking a drill down icon takes you to the next lens. You can drag the lenses in the Designer to change the display order.



Anzo adds the drill down lens to the dashboard, and you can configure each lens using the Designer for that lens.

In this example, clicking the drill down icon ( ) next to a movie ID displays the Actors lens, which shows the actor for that movie:



Form Lens

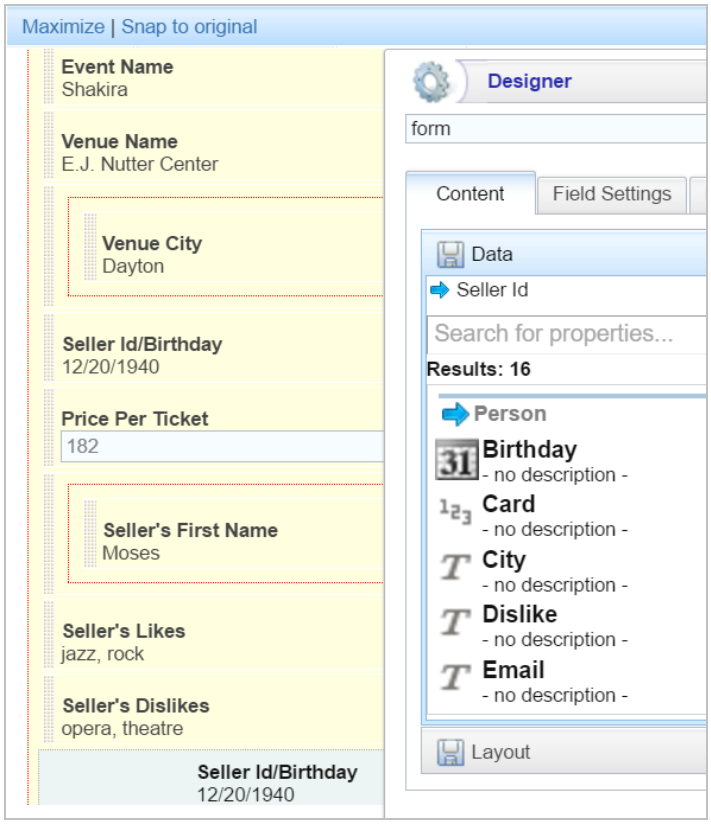
Form lenses enable you to create an editable or read-only form on the dashboard. Creating forms can be useful for displaying many details about each record instead of using a table where the large number of columns makes the data hard to read.

Note: On Graphmart dashboards, Form lenses are read-only.

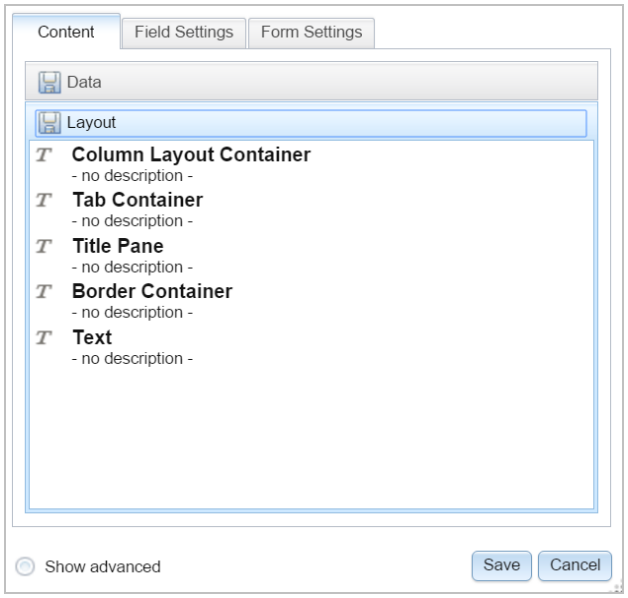
Form Lens Configuration

On the Content tab in the Designer, drag onto the dashboard each property or relative path that you want to appear as a field on the form. After adding objects, you can rearrange the form layout and use the Field Settings tab to further

configure each field.



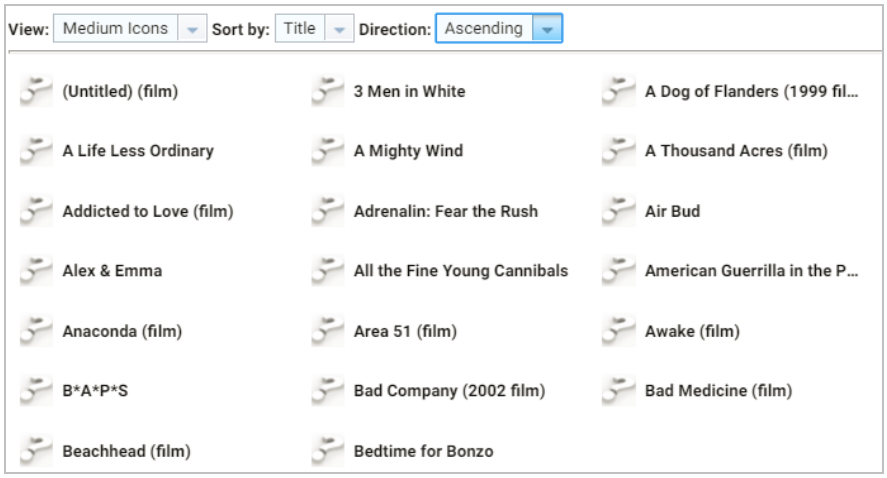
To arrange the fields in a different layout, such as a two-column layout, click **Layout** below the list of properties. The Designer displays the available layout containers.



Drag a container onto the form to create the layout template. You can then drag properties into the template.

List Lens

List lenses display each data property in a list layout with icons, similar a Microsoft Windows Explorer view.



Network Navigator Lens

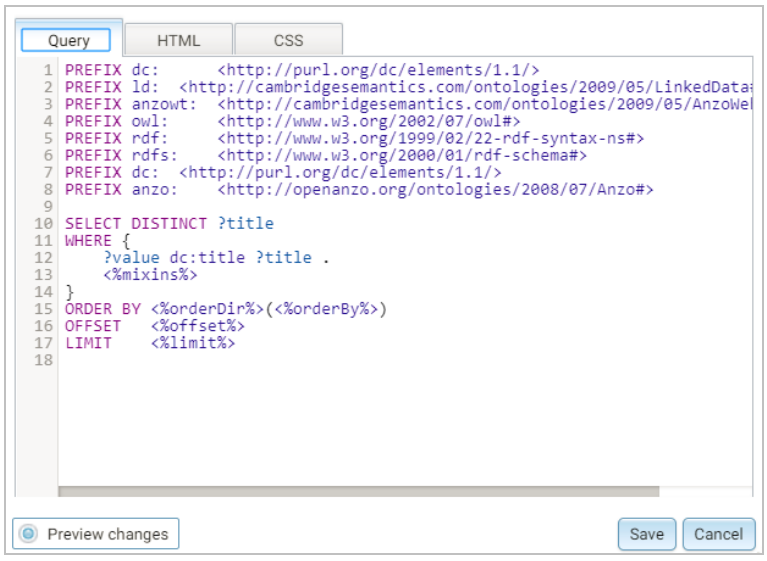
The Network Navigator lens is an interactive graph visualization tool that enables you to view and explore relationships across your entire network of structured, unstructured, internal, or external data. The lens includes preconfigured and customizable tools so that you can quickly generate a standard graph or hierarchical view of the data and then customize the visualization to target the relationships and information that interests you.

Query Lens

The query lens allows you to retrieve and display data using custom a SPARQL query. You format the query results using HTML and CSS. This lens can access external SPARQL-compatible data sources.

Query Lens Configuration

The Query lens Designer has three tabs:



- **Query**: This tab displays a SPARQL query template that you can use to write the query. Note the default code that reflects inherent Anzo functionality:

- **<%mixins%>**: Incorporates a filter function.
- **ORDER BY**: Incorporates a sort function.

  See SPARQL Query Templates and Best Practices for guidance on writing SPARQL queries.

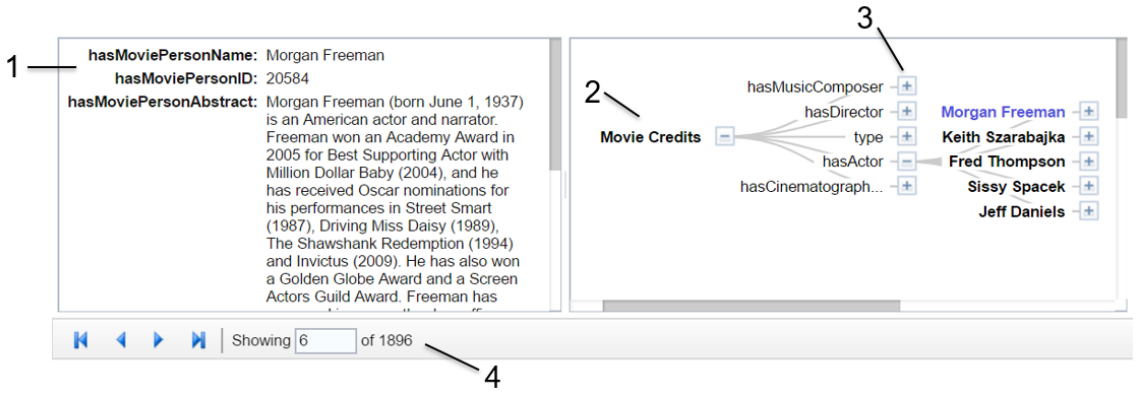
- **HTML**: This tab includes default HTML and basic JavaScript code with sample values. You can edit the content to design the results that the query returns. The default HTML code automatically adds returned query data to a table and organizes it so that new rows are created for each record. Make sure that the <option> elements correspond to the elements in your query.
- **CSS**: This tab enables you to create a cascading style sheet to format the HTML and define the look and feel of the lens. Cambridge Semantics recommends that you define all CSS classes as namespaces to avoid global format changes.

Related Topics

SPARQL Query Templates and Best Practices

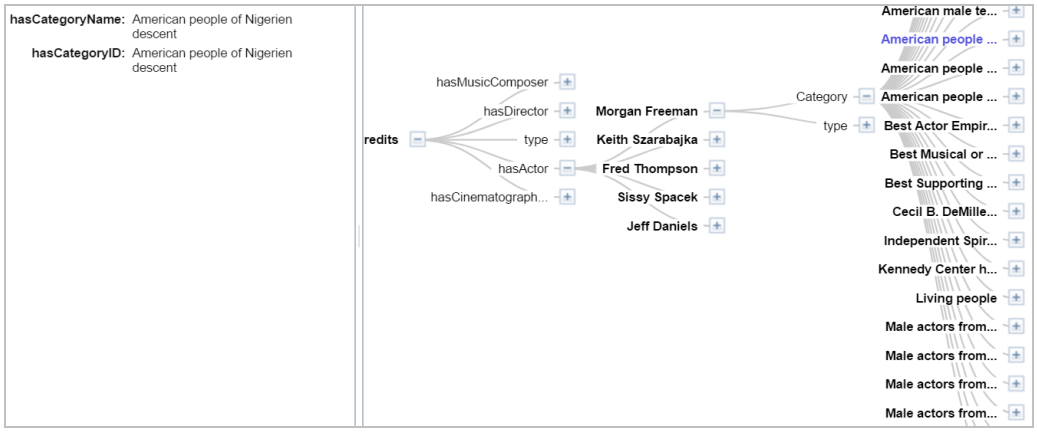
Resource Tree Navigator Lens

The resource tree navigator lens displays data in a tree format with points that you can click to open successive child data points.



1. **Related data**: Displays the class data related to the selected data property. Data changes when a data end point is selected.
2. **Class property**: Displays the label property of the target class as the initial (start) point of the resource tree. Expand the tree to view child properties by clicking the plus icon for a data point.
3. **Selected linked property**: Displays the initial selected property that links to other classes.
4. **Navigation tools**: Use the arrows to navigate to other pages. The Showing text box displays the current page number and total number of pages.

Click through to an end point and the data view changes to reflect the new class. The data point **American people of Nigerien descent** is selected, and the related class data appears on the left of the screen.



Resource Tree Navigator Lens Configuration

The Designer simply displays all of the properties that are linked to other classes. Select each property that you want the resource tree to include. Then click **Save**.

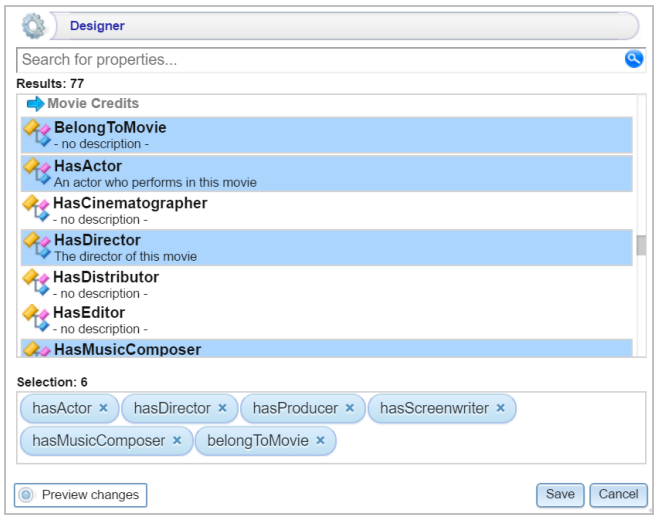


Table Lens

Table lenses display data in a standard row-and-column grid layout. You define each table column by selecting a property.

Table Lens Configuration

The Table lens Designer enables you to choose the properties to become columns as well as apply functions and filters to the data that the table displays.

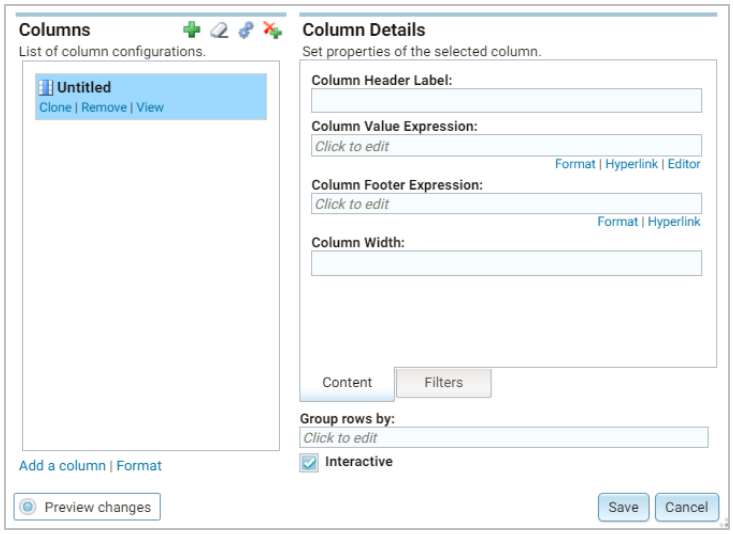


Table Designer Icons

The table below describes the functionality of the icons in the Table Designer.

Icon	Description
	Click the eraser icon to erase all columns. This action cannot be undone.
	Click the plus icon to add a column.
	Click the auto-generate columns icon to add all properties (for the data type selected on the dashboard) as columns.
	Click the Add delete button icon to add delete links to rows in the table. **Note**: Delete links do not appear on Graphmart dashboards since graphmart data cannot be edited.

Column Details

This section describes the fields that are available on the Content and Filters tabs in the Designer. Click a property or column on the left side of the screen to configure the options for that column.

- **Column Header Label**: (Optional) The column name to display. Overrides the Column Value Expression property name.
- **Column Value Expression**: The property name or calculation to use to populate the values in the column.
- **Column Footer Expression**: (Optional) The property to use for the table footer.
- **Column Width**: (Optional) The width of the column in pixels.
- **Group rows by**: (Optional) The property to use to group data on.

- **Filters Tab**: **Create filter**: Enables you to create a filter on the column. For more information, see Creating a Dashboard Filter.

Default Data Display Formats

This section describes the default display formats for date and numeric values in tables.

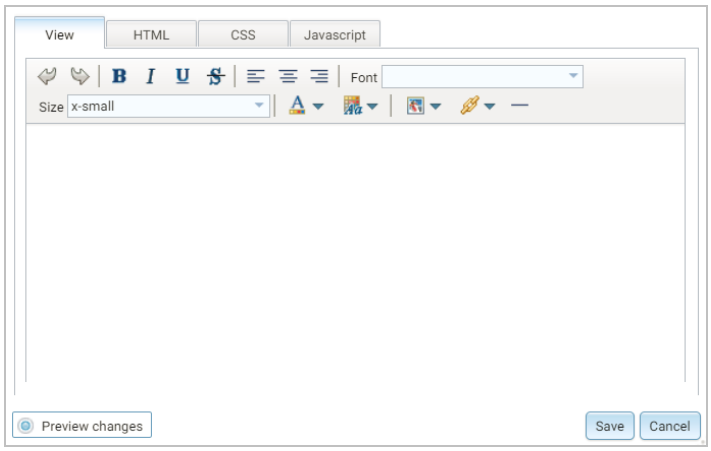
- **Date**: By default Anzo displays date values in "short" date format. The order of the month, day, and year depends on the location of your browser. For example, in the United States the default date format is MM/DD/YYYY. In Australia, the default date format is DD/MM/YYYY. Note that this is not dependent on the Anzo server location but on the location auto-detected by the browser.
- **Numeric**: Anzo displays the complete value without a limit on precision. Numeric formats are also dependent on the location of the browser. For example, in the United States the default format for a large number is 4,294,967,295.00 and in Canada the default format is 4 294 967 295,000. Note that this is not dependent on the Anzo server location but on the location auto-detected by the browser.

Web Page Lens

Web Page lenses enable you to display data by creating a web page using HTML, CSS, and JavaScript. This lens is for advanced users with coding skills in these areas. A powerful feature of the this lens is the ability to bind data to Anzo graphs so that updates reflect in real time.

Web Page Lens Configuration

The Web Page Designer has four tabs:



- **View**: Provides a rich text interface for viewing the page (WYSIWYG). Changes made to this page are reflected in the HTML code.
- **HTML**: This tab enables HTML coding and data binding. The example HTML image below shows code that defines text format as well as data binding using the anzowbind:innerhtml command.

For more information about data binding, see the Data Binding Example section below.

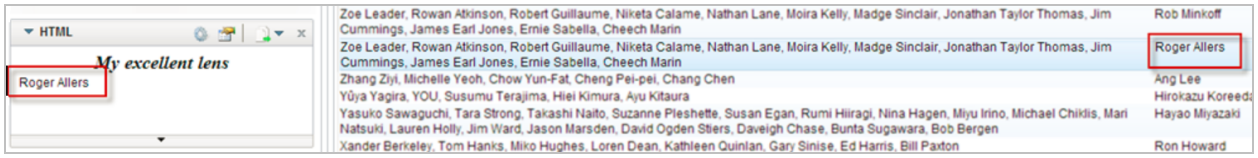
- **CSS**: This tab enables you to create a cascading style sheet to format the HTML and define the look and feel of the web page. Cambridge Semantics recommends that you define all CSS classes as namespaces to avoid global format changes.
- **Javascript**: This tab enables you to write JavaScript code to implement functions such as if statements, animations, or event notifications.

Data Binding Example

When data is bound to a web page lens using HTML code, the web page lens behaves as follows:

- The lens will reflects data changes in real time.
- If the lens is oriented to the left-hand column (using the Orientation drop-down), selecting data in an active lens prompts the web page lens to display the related data.

In the example below, the active table lens row is selected, prompting the web page lens on the left (“My excellent lens”) to display the corresponding data.



Accessing Data with the Query Builder

The Query Builder in the user interface provides options for accessing data in various data sources. The Query Builder includes a **Find** option that enables users to search for quads by specifying a single subject, object, predicate, or graph name. It also includes a **Query** option that enables users to write, run, and save SPARQL queries. The topics in this section provide information about accessing data using the Query Builder.

- Searching for Quads in the Query Builder
- Running SPARQL Queries in the Query Builder

Searching for Quads in the Query Builder

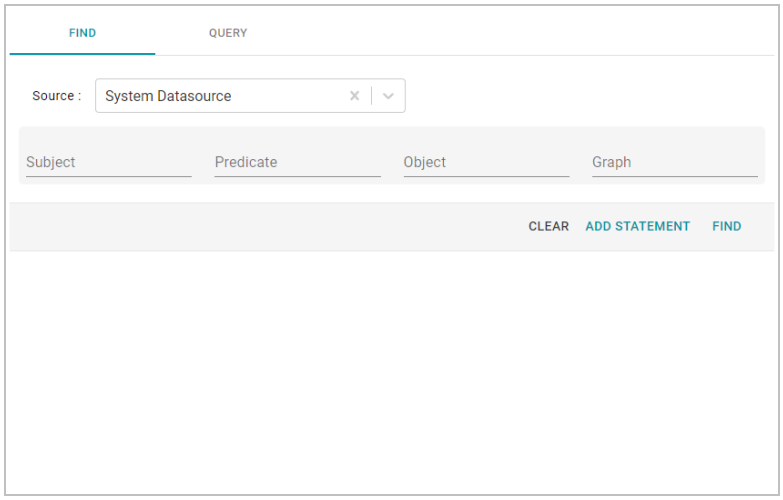
The Query Builder includes a Find tab for searching for data in Anzo by specifying a single subject, object, predicate, or graph name or any combination of those elements. Statements that match the search criteria are returned in quads,

and the screen includes quick filters that enable users to toggle filters on and off to show or hide any of the quad elements. The Find tab supports searches against the following data sources:

- Anzo System Data Source
- AnzoGraph
- Anzo System Tables
- Data Profiling Metrics
- LDAP Primary Data Source

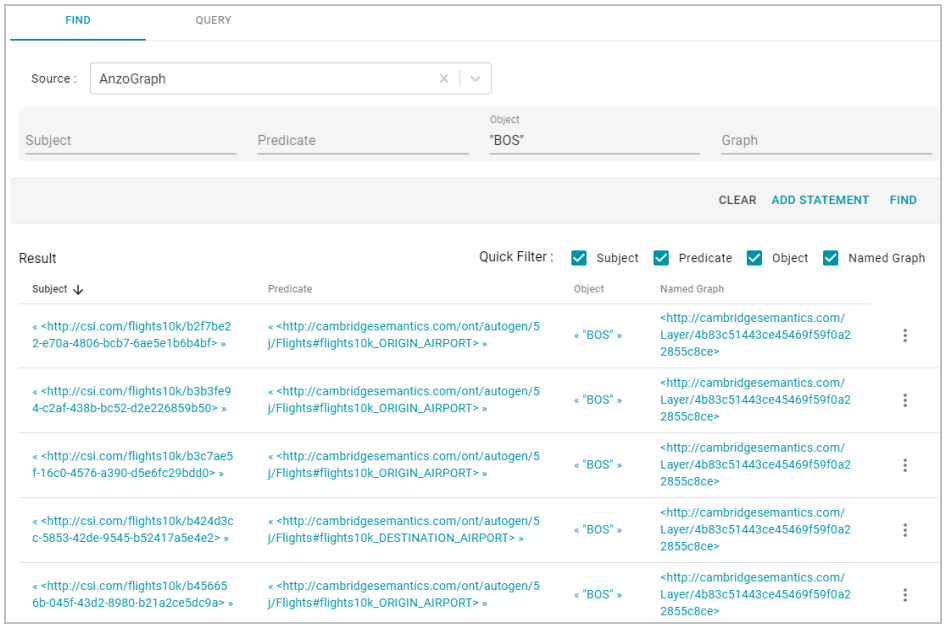
When finding data in the system data source, users have the option to modify or delete statements directly in the user interface. Follow the instructions below to find data in any of the supported data sources.

1. In the Anzo console, expand the **Access** menu and click **Query Builder**. Anzo displays the Find tab.



2. Click the **Source** drop-down list and select the data source that you want to search.
    - Select **System Datasource** to search the local Anzo volume.
    - Select the name of an AnzoGraph instance to search for data in graphmarts that are loaded to that instance.
    - Select **Data Profiling Metrics** to search the data metrics volume.
    - Select **LDAP Primary Datasource** to search the directory server.
    - Select **System Tables** to search Anzo system table data.
3. Follow the guidelines below to specify the data to find in the data source:
    - Specify any subject, predicate, object, or graph name in the appropriate field. You can specify a value for one field in the quad or any combination of fields.
    - Any URIs and/or literal values that you specify must match the value in the data. Partial values, wildcard characters, and regular expressions are not supported.
    - If you want to get a list of all of the statements in the data source, you can leave all of the fields blank.

4. Click **Find** to search for the statements that match the search criteria. Anzo displays the matching statements. For example:



5. The following options are available for working with the results:
   - To filter results by showing or hiding parts of the quads in the statements, you can select or clear the Quick Filter checkboxes above the results.

     Clearing a checkbox hides that part of the quad in the result list. You can display the item again by selecting the checkbox.
   - To modify the search parameters, you can click any of the graph, subject, predicate, or object values in the results. The search is automatically run again using only the value that you clicked.
   - If the source that you searched is the **System Datasource**, you can edit, delete, or add statements directly. See System Datasource Options below for details.

System Datasource Options

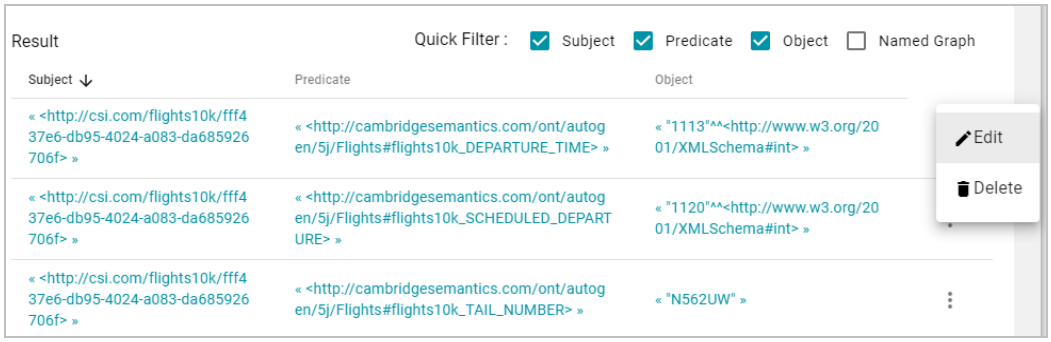
This section provides information about editing, deleting, and adding statements on the Find screen.

Note: Though the options described below are available for all data sources, adding, deleting, or editing statements is only successful when the data source is **System Datasource**.

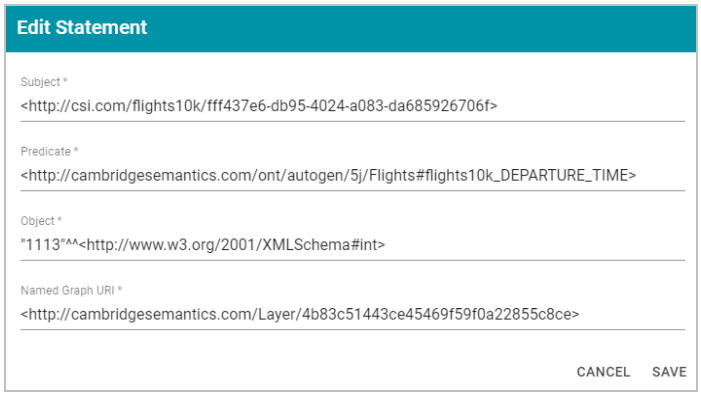
- Editing a Statement
- Deleting a Statement
- Adding a Statement

Editing a Statement

To edit a statement, click the menu icon (⋮) to the right of the statement and select **Edit**.



Anzo displays the Edit Statement dialog box. For example:

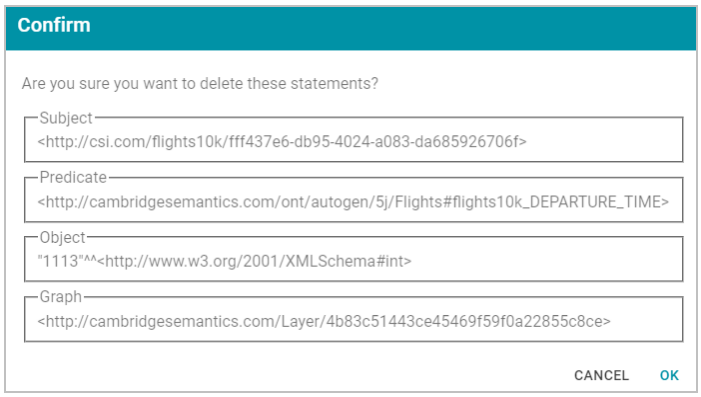


Change any of the quad values, and then click **Save**.

Important: If you edit URI values, make sure that the modified value is a valid URI.

Deleting a Statement

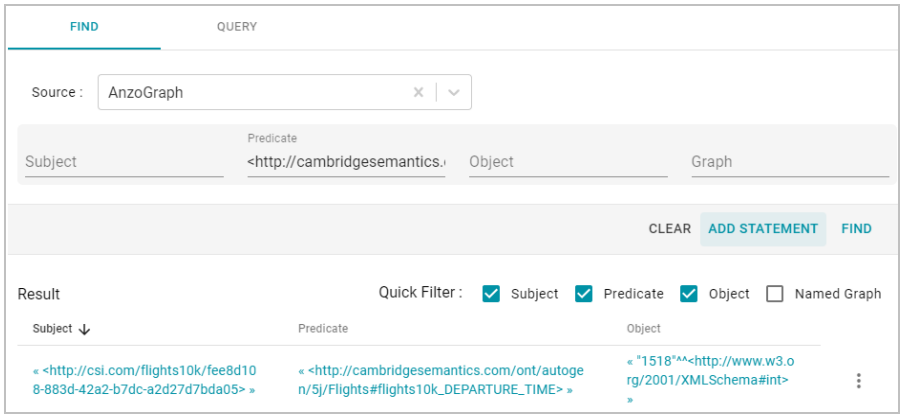
To delete a statement, click the menu icon (⋮) to the right of the statement and select **Delete**. Anzo displays the statement in a confirmation dialog box. For example:



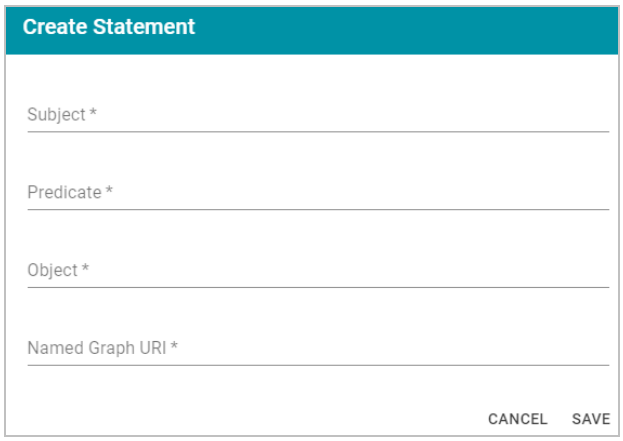
Click **OK** to remove the statement from the system data source.

Adding a Statement

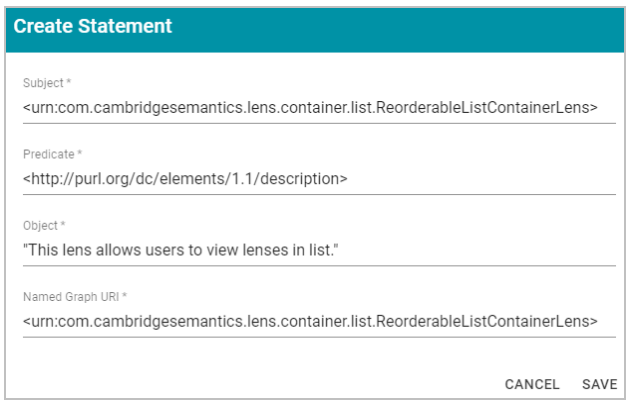
To add a quad to the data source, click **Add Statement** at the top of the result list. For example:



Anzo displays the Create Statement dialog box.



Specify the new quad by adding the subject, predicate, object, and named graph URI in the appropriate fields. Each field is required. URIs must be valid, and the Named Graph URI that you specify must be present in the data source. You cannot add a new named graph. For example:



Click **Save** to add the new quad to the data source.

Related Topics

Running SPARQL Queries in the Query Builder

Analyzing Data with Hi-Res Analytics

Running SPARQL Queries in the Query Builder

The Query Builder includes a Query tab for writing and running SPARQL queries. The query editor provides syntax assistance, type-ahead suggestions for model entity names, and automated prefix creation and query formatting for readability. It also includes the option to save queries for later use.

The Query tab supports running queries against the following data sources:

- Graphmarts and specific data layers within graphmarts
- Linked Data Sets
- Data sources: Anzo System Data Source, AnzoGraph, Anzo System Tables, Data Profiling Metrics, LDAP Primary Data Source

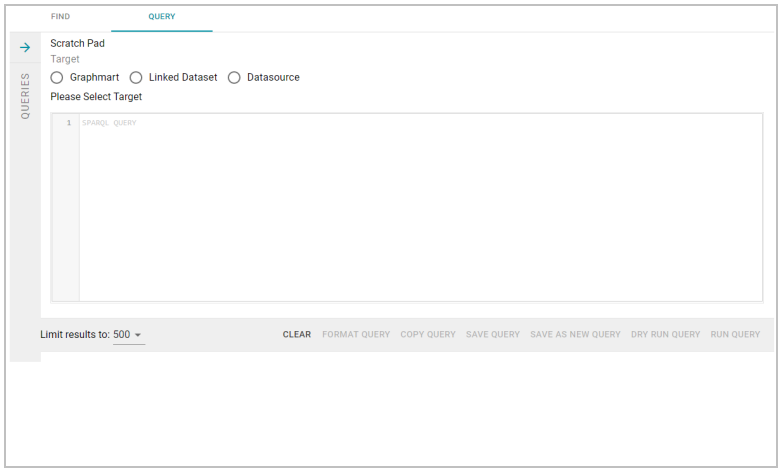
Note To ensure that queries perform well and do not consume too many resources on the system, keep the following guidelines in mind when developing and testing queries:

- Set a limit on the number of results to return.
- Avoid cross-product joins
- Consider using VALUES clauses instead of FILTER clauses.
- When retrieving a large number of values, use subqueries instead of OPTIONAL clauses.

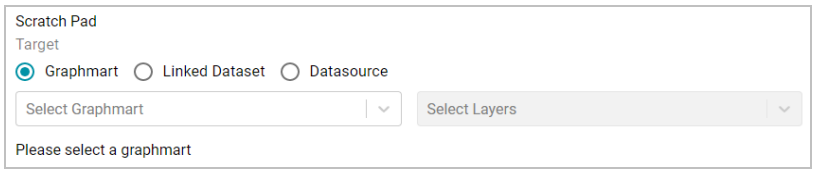
For query templates and additional details about best practices, see SPARQL Query Templates and Best Practices.

Follow the instructions below to write and run SPARQL queries against any of the supported data sources.

1. In the Anzo console, expand the **Access** menu and click **Query Builder**. Then click the **Query** tab. Anzo displays the query editor.

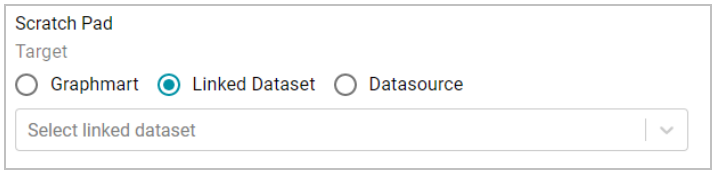


2. At the top of the screen, click the radio button for the target data source:
   - To query data that is in a graphmart, select the **Graphmart** radio button.



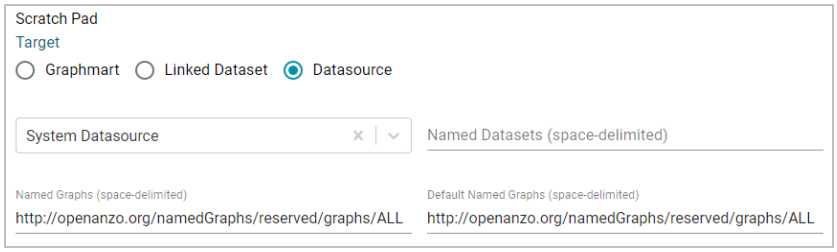
     Click the **Select Graphmart** drop-down list and select the graphmart to query. If you want to narrow the scope of the query by selecting one or more data layers in the graphmart, click the **Select Layers** drop-down list and select the data layer or layers to target.

   - To query data that is in a linked data set, select the **Linked Dataset** radio button.



     Click the **Select linked dataset** drop-down list and select the linked data set to query.

   - To run queries against the system data source, data metrics volume, Anzo system tables, LDAP server, or AnzoGraph, select the **Datasource** radio button.



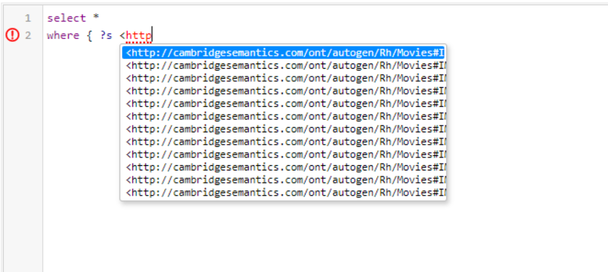
     Click the **Datasource** drop-down list and select the target source:

     - Select **System Datasource** to search the local Anzo volume.
     - Select the name of an AnzoGraph instance to search for data in graphmarts that are loaded to that instance.
     - Select **Data Profiling Metrics** to search the data metrics volume.
     - Select **LDAP Primary Datasource** to search the directory server.
     - Select **System Tables** to search Anzo system table data.

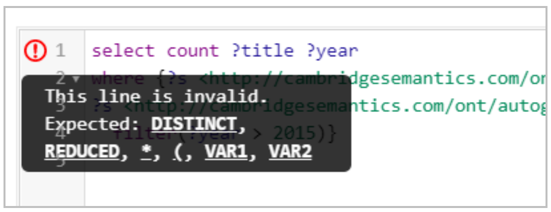
     By default, the Named Graphs and Default Named Graphs values are set to all named graphs (`http://openanzo.org/namedGraphs/reserved/graphs/ALL`). If you want to narrow the scope of the query, you can replace the values with specific graph URIs. To list multiple graphs, separate URIs with a space.

3. In the text box below the target, compose the SPARQL query. For information about the supported SPARQL functions, see Supported Functions and Formulas.

   Tip: When typing entity URIs, the query builder offers suggestions by listing the properties in the data source. You can click an item in the list to insert that entity. For example:

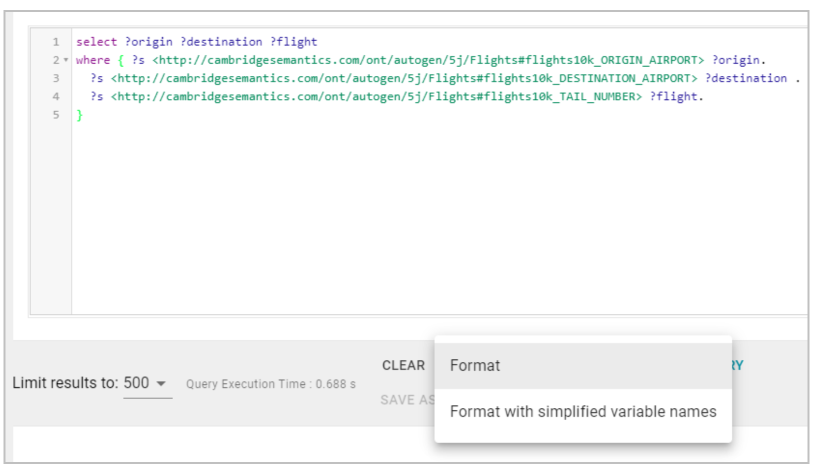


   When a red exclamation mark icon ( ) is displayed next to a line number, you can hover the pointer over the icon to view guidance on how to resolve the issue. For example:



4. If you want to format the query for readability, click **Format Query** and select one of the following options:
   - **Format**: Auto-creates prefixes, inserts URI abbreviations, and restructures the query for readability.
   - **Format with simplified variable names**: Auto-creates prefixes, inserts URI abbreviations, simplifies variable names by changing them to ?_var1, ?_var2, ?_var*N*, and restructures the query for readability.

   For example, the image below shows a query before it is formatted.



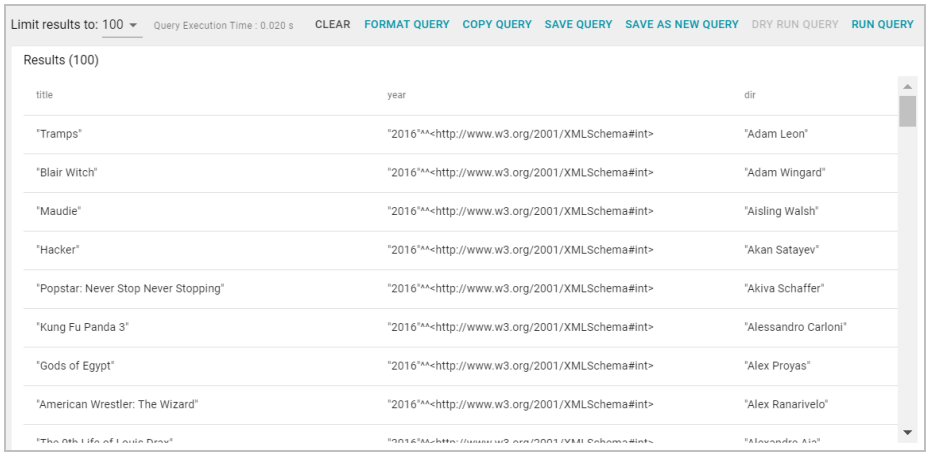
   After the query is formatted, prefixes and URI abbreviations are added. For example:

```
PREFIX Flights: <http://cambridgesemantics.com/ont/autogen/5j/Flights#>
SELECT
    ?origin
    ?destination
    ?flight

WHERE {
    ?s Flights:flights10k\_ORIGIN\_AIRPORT ?origin .
    ?s Flights:flights10k\_DESTINATION\_AIRPORT ?destination .
    ?s Flights:flights10k\_TAIL\_NUMBER ?flight .
}
```

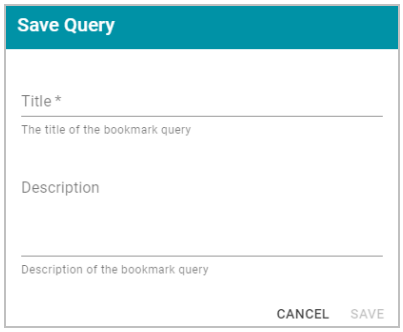
5. If the query is an INSERT or DELETE query, the **Dry Run Query** button becomes active. You can click **Dry Run Query** to do a test run of the update. In a test run, Anzo runs a version of the query where INSERT or DELETE is replaced with CONSTRUCT, and the results report the number of statements that the query affects, i.e., the number of additions or removals per graph. If the results are unexpected, you can adjust the query before clicking **Run Query** and committing the updates.
6. If necessary, change the query limit. By default, query results are limited to 500. To adjust the limit, click the **Limit results to** drop-down list below the query editor and select a value. For example:

7. To run the query, click **Run Query**. The results appear at the bottom of the screen. For example:

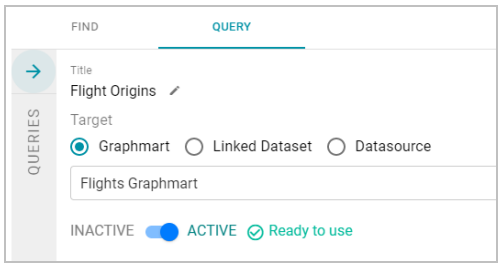


Tip You can click any value in the result list to copy that value to the clipboard.

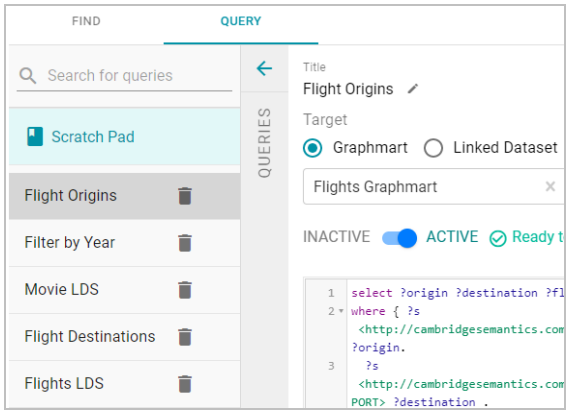
8. To save the query for later use, click **Save Query**. Anzo displays the Save Query dialog box.



9. In the Save Query dialog box, specify a name for the query in the **Title** field and an optional description in the **Description** field. Then click **Save**. The query is saved in the gray **Queries** list on the left side of the screen and is collapsed by default.



Click the arrow or anywhere on the gray tab to expand the list. For example:



Select a query to open it in the query editor. You can delete a query by clicking the trashcan icon next to the query name. If you change a query and want to save it as a new query, click **Save as New Query**.

Related Topics

Searching for Quads in the Query Builder

Analyzing Data with Hi-Res Analytics

Accessing Data from Data on Demand Endpoints

Accessing Data via the SPARQL Endpoint

Accessing Data via the HTTP Client Interface
SPARQL Query Templates and Best Practices

Accessing Data from Data on Demand Endpoints

The Anzo Data on Demand service enables users to generate Open Data Protocol (OData)-based feeds that can be used to access graphmarts programmatically via a RESTful API or from third-party business intelligence tools such as TIBCO Spotfire, Tableau, and Microsoft Power BI.

OData facilitates the creation and consumption of queryable and interoperable RESTful APIs in a simple and standard manner. The protocol enables web clients to use simple HTTP messages to publish and edit resources that are identified using URLs and are defined in a data model. OData shares some similarities with JDBC and ODBC. Like ODBC, OData is not limited to relational databases. The Anzo Data on Demand service follows the OData Version 4.0 specification, which defines the standard URL conventions, query options, and a metadata schema that describes the data model.

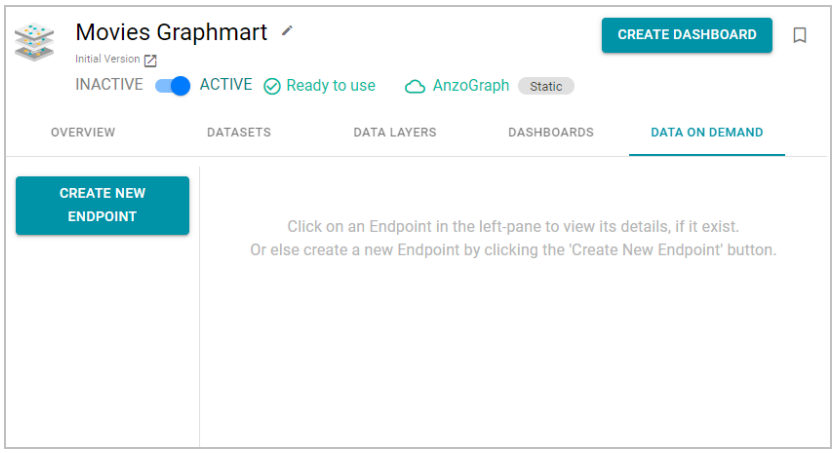
The topics in this section provide information about creating and accessing data on demand endpoints and using the Anzo ODBC and JDBC drivers.

- Creating a Data on Demand Endpoint
- Accessing an Endpoint Programmatically
- Accessing an Endpoint from an Application
- OData Reference

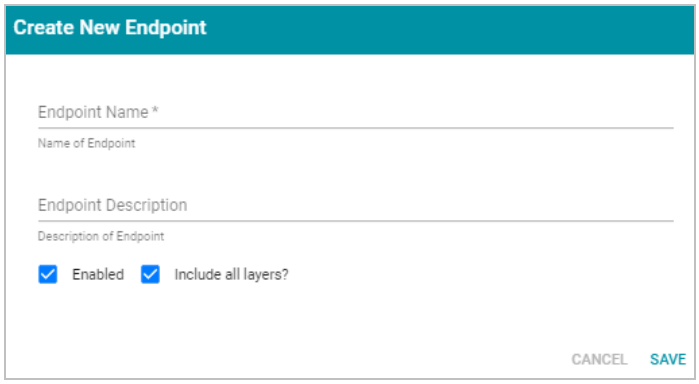
Creating a Data on Demand Endpoint

This topic provides instructions for creating a data on demand endpoint for a graphmart.

1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**.
2. On the Graphmarts screen, click the name of the graphmart for which you want to enable the data on demand service.
3. Click the **Data on Demand** tab. Anzo displays the Data on Demand screen, which lists any existing endpoints. For example:



4. On the Data on Demand screen, click the **Create New Endpoint** button. Anzo displays the Create New Endpoint screen.

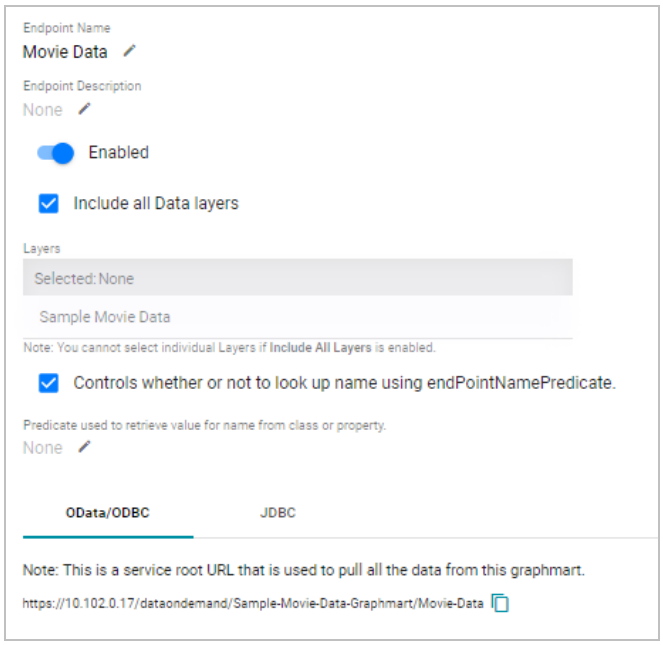


5. Type a name for the endpoint in the **Endpoint Name** field and an optional description in the **Endpoint Description** field. Make sure that the endpoint name is unique.
6. By default the **Enabled** option is selected, indicating that the endpoint will be enabled when the configuration is saved. If you want to disable the endpoint, clear the Enabled check box.

   Note If a request is sent to a disabled endpoint, Anzo displays a 503: Service Unavailable error with a message indicating that the endpoint is disabled. For example, "Unable to process request. The endpoint 'ExpandGM/TestEndPoint' is DISABLED."

7. By default the **Include all layers** option is selected, indicating that all of the data layers in the graphmart will be available from the endpoint. If you do not want to enable all layers, clear the Include all layers checkbox. After you save the new endpoint, you can edit the configuration to specify which data layers to include.
8. Click **Save**. Anzo saves the configuration and adds the endpoint to the list of endpoints on the Data on Demand screen. For example:

9. Click the endpoint name to view the configuration details. For example:



10. If you cleared the Include all Data Layers checkbox and want to include certain layers, click the Edit icon ( ) next to **Layers** and select the layers to include. Click the checkmark icon ( ) to save the change.

11. If you want to specify the predicate value to use for the class and property display names for the endpoint, such as the rdfs:label or dc:description for the entity, select the **Controls whether or not to look up name using endPointNamePredicate** option. Then specify the predicate to obtain the values from in the **Predicate used to retrieve value for name from class or property** field.

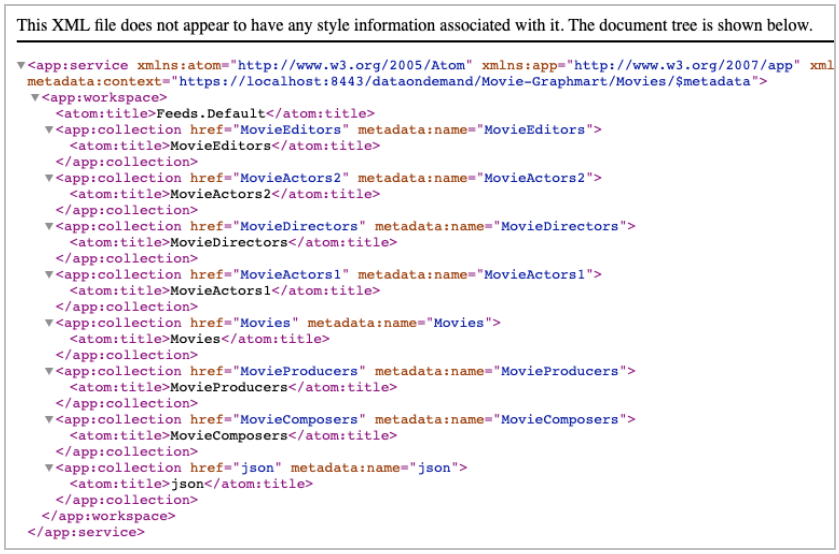
    Specify a predicate from the related data model, such as `http://www.w3.org/2000/01/rdf-schema#label` to use each entity's **Label** value or `http://purl.org/dc/elements/1.1/description` to use each entity's **Description** value.

    Note If the **Controls whether or not to look up name using endPointNamePredicate** option is disabled, Anzo displays each entity's local name. If the **Controls whether or not to look up name using endPointNamePredicate** option is enabled but the **Predicate used to retrieve value for name from class or property** field is empty, Anzo automatically uses the value in the rdfs:label (http://www.w3.org/2000/01/rdf-schema#label) predicate.

12. Once configured and enabled, this Data on Demand endpoint is ready for access via OData. At the bottom of the screen, retrieve the OData/ODBC and JDBC service URLs that you can use to access the endpoint's data

from applications.

To test whether the endpoint is active, you can copy the OData ODBC service URL and paste it into a web browser. If the endpoint is active, the browser shows an XML feed of the schema data. For example:



The Data on Demand endpoint is now available to access.

Note The endpoint is accessible only when it is **Enabled** and the associated graphmart is **Active**.

For information about accessing endpoints programmatically, see Accessing an Endpoint Programmatically. For information about accessing endpoints with third-party analytics tools, see Accessing an Endpoint from an Application. For information about the supported OData operators, output format, and query examples, see OData Reference.

Related Topics

Accessing an Endpoint Programmatically

Accessing an Endpoint from an Application

OData Reference

Accessing an Endpoint Programmatically

This topic provides guidance on accessing Data on Demand endpoints programmatically by showing some example implementations using R and Python.

- Authentication and Data Access
- Accessing an Endpoint with R (Through RStudio)
- Accessing an Endpoint with Python (Through a Linux Terminal)

Authentication and Data Access

Connections to Data on Demand endpoints must be authenticated. Users can submit their Anzo username and password when accessing data. Ultimately the data that is available to users from OData endpoints is subject to the security and composition of the graphmart as configured in Anzo.

Accessing an Endpoint with R (Through RStudio)

The following example shows how to connect to an OData endpoint from RStudio. The example uses the R programming language to access a Data on Demand endpoint and pull in data via a standard dataframe. New or existing R scripts can then be used with the data.

The first step in accessing data from RStudio is to prepare the R script that will construct the target URL and retrieve the resulting information via HTTP. The example script below accesses a pre-configured "Sample Data" endpoint. The script has sections for filtering the results as well as expanding the selection to include information from multiple classes:

```
require("httr")
require("jsonlite")
require("rstudioapi")

user   <- rstudioapi::showPrompt("Username", "Enter Anzo username", "sysadmin")
pw     <- rstudioapi::askForPassword(paste("Enter password for",user,sep=" "))

## Data on Demand endpoint
odata  <- "https://cambridgesemantics.com/dataondemand/Sample-Graphmart/Sample-Data"

## Start from Probe class
startClass  <- "Probe?"

## Filter results for Homo sapiens species
filterKw   <- "$filter="
filterVal  <- "Species eq 'Hs'"
urlify     <- URLencode(filterVal)
filterStr  <- paste(filterKw,urlify,sep="")

## Select properties of interest (FeatureID) from base class
selectKw   <- "&$select="
selectVal  <- "FeatureID"
selectStr  <- paste(selectKw,selectVal,sep="")

## Select properties of interest (symbol) from Gene class
## via corresponds_to property on base Probe class
expandKw   <- "&$expand="
expandClass  <- "corresponds_to"
expandProps  <- "symbol"
```

```
expSelStr   <- "$select="
expandStr   <- paste(expandKw,expandClass,"(",expSelStr,expandProps,")",sep="")

## Specify format
format  <- "&$format=json"

## Generate OData URL using fragments above
url <- paste(odata,startClass,filterStr,selectStr,expandStr,format,sep="")

## Access OData endpoint
resultRaw  <- GET(url, (authenticate(user,pw, type = "basic")))
resultTxt  <- content(resultRaw, "text")
resultJson  <- fromJSON(resultTxt, flatten = TRUE)

print(url)

## Read results into dataframe
resultDataFrame  <- as.data.frame(resultJson)
View(resultDataFrame)
```

Executing the above R script from RStudio results in a dataframe that represents columns from the **Probe** and **Gene** classes.

Accessing an Endpoint with Python (Through a Linux Terminal)

Many users have existing Python scripts to use with data in Anzo or a familiarity with Python that would make exploring, retrieving, and leveraging the data easier. The following example shows how to connect to an OData endpoint by executing a Python script from a Linux terminal.

The first step in accessing data using Python is to prepare the Python script that will construct the target URL and retrieve the resulting information via HTTP. The example script below accesses a pre-configured "Sample Data" endpoint. The script has sections for filtering the results as well as expanding the selection to include information from multiple classes (the same filter and class properties that were used in the R example above).

```
import requests
import getpass
from urllib.parse import urlparse

un = getpass.getpass(prompt='Username: ')
pw = getpass.getpass(prompt='Password: ')

## OData endpoint
odata = 'https://cambridgesemantics.com/dataondemand/Sample-Graphmart/Sample-Data/'
# data on demand url

## Start from Lease class
```

```
startClass = "Probe?"

## Filter results
filterKw  = "$filter="
filterVal = "Species eq 'Hs'"
urlify    = urlparse(filterVal)
filterStr = filterKw + urlify.geturl()

## Select properties of interest (start date, missed payments, lease status) from base
class
selectKw  = "&$select="
selectVal = "FeatureID"
selectStr = selectKw + selectVal

## Select properties of interest (name, social security number, credit score) from
Individual class
expandKw  = "&$expand="
expandClass = "corresponds_to"
expandProps = "symbol"
expSelStr   = "$select="
expandStr = expandKw + expandClass + "(" + expSelStr + expandProps + ")"

## Specify format
format = "&$format=text/csv"

## Generate OData URL using fragments above
url = odata + startClass + filterStr + selectStr + expandStr + format

## Access OData endpoint
r = requests.get(url, auth=(un, pw), verify=False)

print("URL")
print(url)
print("CONTENT")
print(r.content.decode('unicode_escape'))
print(type(r))
print(type(r.content))
```

In this example, the output is returned in CSV format (rather than JSON, as in the R example).

Related Topics

Creating a Data on Demand Endpoint

Accessing an Endpoint from an Application

OData Reference

Accessing an Endpoint from an Application

Since Anzo's Data on Demand service conforms to the OData standard, any tool that supports the OData V4 REST API can access a Data on Demand endpoint to leverage data in Anzo. In addition, applications that support ODBC or JDBC APIs can use the Anzo CData ODBC or JDBC drivers to interact with Data on Demand endpoints. This capability enables users to leverage the benefits of Anzo's semantic layer, data model, and data blending capabilities in their favorite analytics tools.

This topic provides information about accessing Data on Demand endpoints from third-party applications.

JDBC Driver Considerations

This section describes important items to consider when using JDBC clients for accessing Data on Demand endpoints:

Join Performance

To join results from multiple classes, Cambridge Semantics strongly recommends using OData or SPARQL. Hi-Res Analytics and SPARQL are designed to quickly return large results from multiple classes and should be strongly considered for these use cases. You can also join tables upstream in Anzo by creating data layers. For example, you can create a view that joins the data using a CONSTRUCT query. The view becomes available as an OData table. For information about view steps, see Creating a View of the Data. Joins on large data sets are well-supported with OData when best practices around paging are applied.

Because the JDBC driver generates multiple OData queries and joins the results in memory, SQL queries that include JOINs on large data sets may take a very long time to complete. When using the JDBC driver, Cambridge Semantics recommends that you query one class at a time and then use the BI tool to do analytics on the returned data. For more information, see JDBC Performance Details below.

Querying Multi-Value Properties

Some relational systems do not directly support Anzo’s RDF graph data structures. For example, sometimes the JDBC driver presents multi-value properties as arrays, which can make displaying the results difficult in typical BI tools. There are often ways to restructure the data or the query to get the data you want in a relational system and

work with standard BI tools. For guidance and advice using your specific data model, please work with Cambridge Semantics.

Working with Long Column Names

By default, the JDBC driver creates column names based on the property labels in the data model. The property labels can be too long for some clients. For example, Informatica is limited to 128 characters. When ingesting data from a tabular source, the label is a concatenation of the table and column name. Users may need to shorten the property labels to work with JDBC clients. If the label is missing, Anzo uses the localName of the IRI. For information about configuring the column names to be used for a Data on Demand endpoint, see Creating a Data on Demand Endpoint.

Authentication and Data Access

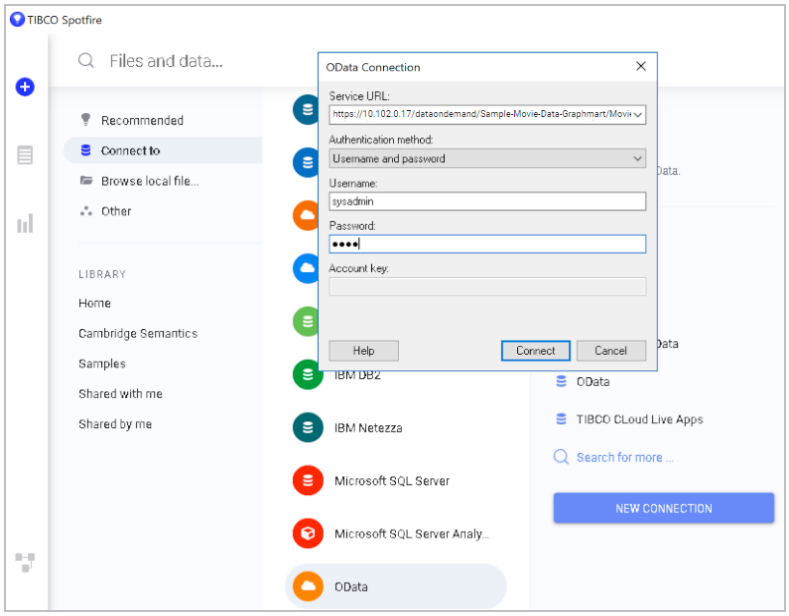
Connections to Data on Demand endpoints must be authenticated. Users can submit their Anzo username and password when accessing data. If your applications use single sign-on (SSO) authentication, you can also use SSO with Anzo. When using SSO, the client authenticates the user against the SSO provider and then passes the credentials to Anzo. All data is secured according to the user's SSO profile. For information about the supported SSO providers and instructions on configuring SSO access, see Configuring SSO Access.

Note Ultimately the data that is available to users from OData endpoints is subject to the security and composition of the graphmart as configured in Anzo.

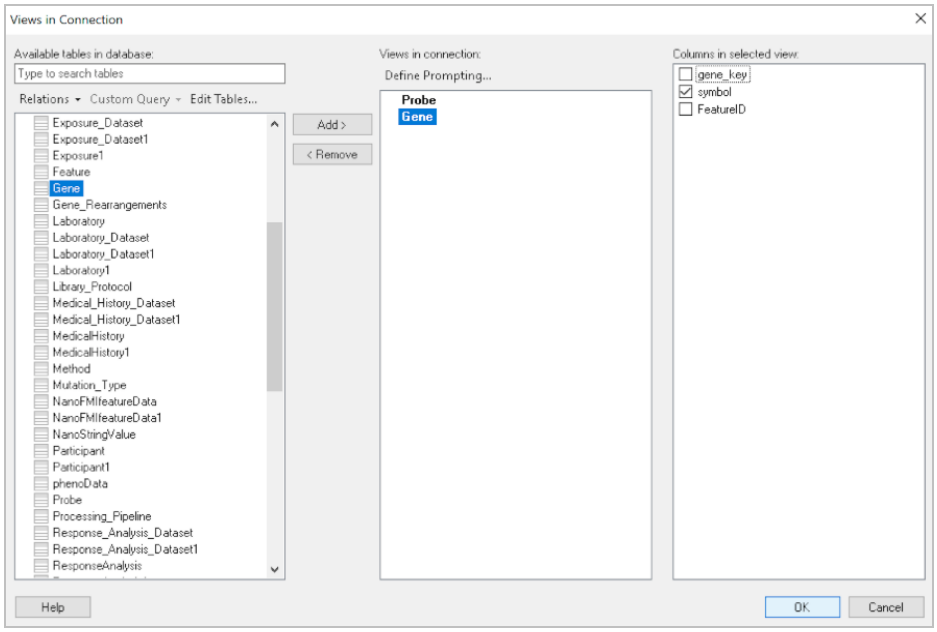
Accessing Data via the OData API

This section provides guidance on accessing a Data on Demand endpoint from an application that supports the OData REST API. It includes an example that configures an OData connection in TIBCO Spotfire. The example steps can also be applied to OData connections in other similar business intelligence tools.

The first step is to connect to the OData endpoint using the Spotfire Data sources user interface. When setting up the OData connection, the Service URL is the OData/ODBC URL from the Data on Demand endpoint configuration details in Anzo. The OData connection uses the user’s Anzo credentials for authentication.



Once the connection is established, Sportfire prompts the user to select the classes and properties to work with. In this example, the **FeatureID** property from the **Probe** class and the **symbol** property from the **Gene** class are selected:



Once the properties are chosen, the data is loaded in Spotfire and can be used to inform existing analytics and data visualizations or create new ones.

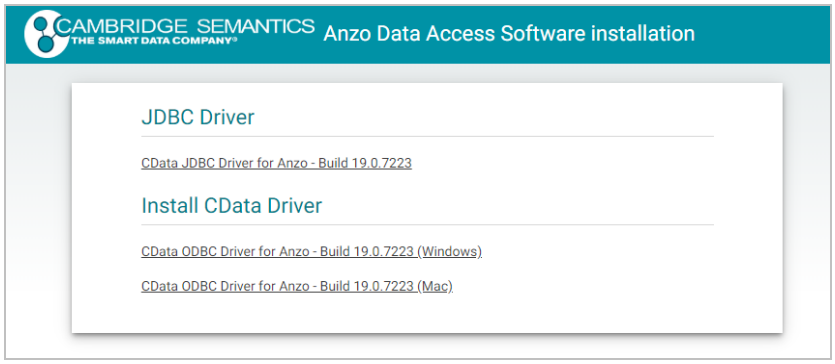
Accessing Data via the ODBC or JDBC API

This section provides guidance on accessing Data on Demand endpoints from applications that support ODBC or JDBC APIs. Your Anzo deployment includes CData ODBC and JDBC drivers to use with applications. The first step is

to retrieve the appropriate driver for your client. To download a driver, open a web browser and go to the following URL:

```
https://Anzo_server/installs/anzodataaccess
```

Where *Anzo_server* is the Anzo server DNS name or IP address. The Anzo Data Access Software Installation page provides links to download each driver. For example:



Download the appropriate driver to the client server:

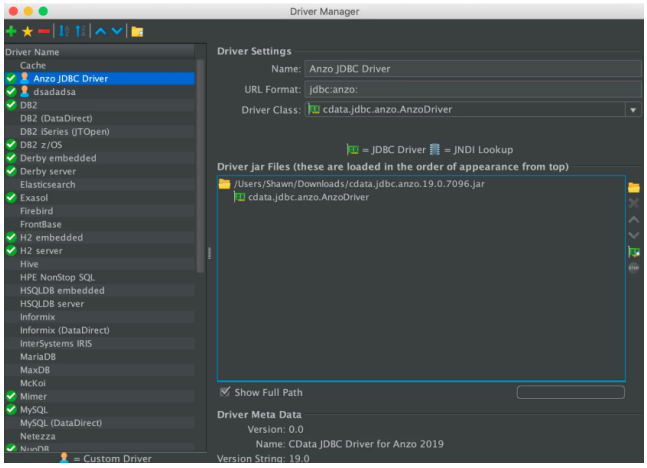
- The **CData JDBC Driver for Anzo** is the most appropriate way to connect to Anzo from most Java applications and database management tools.
- The **CData ODBC Driver for Anzo** for Windows or Mac is for use with applications and database management tools that support open database connectivity, such as Microsoft Excel or Tableau.

Configuring the Driver and Connecting to the Endpoint

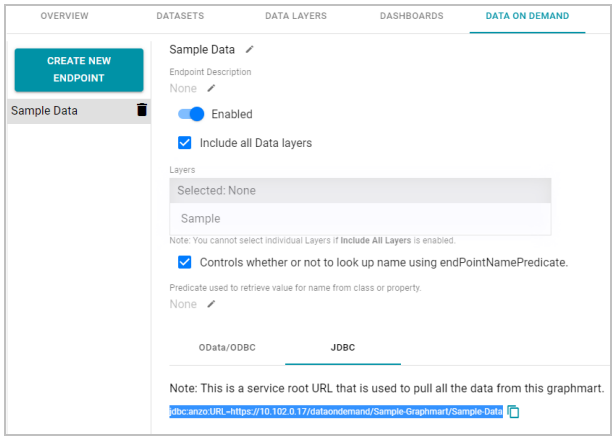
This section provides guidance configuring an ODBC or JDBC driver by showing examples of configuring DbVisualizer and Tableau to access a Data on Demand endpoint using Anzo's JDBC driver and configuring Power BI to access an endpoint using the ODBC driver.

Example JDBC Setup with DbVisualizer

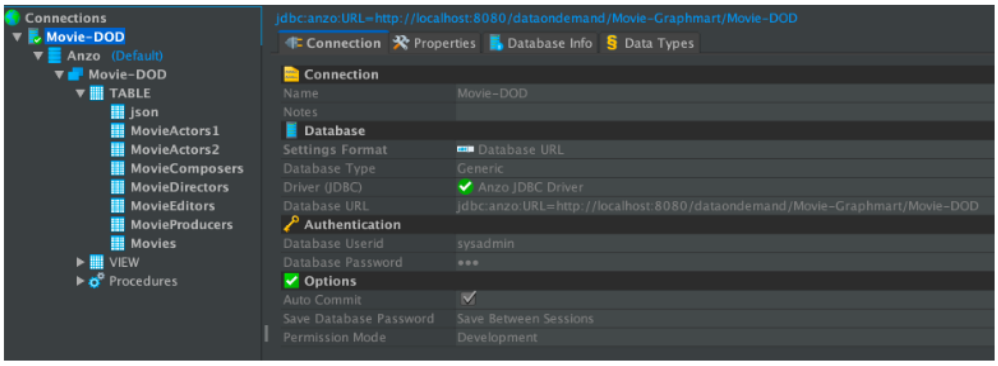
1. In DbVisualizer, go to **Tools** → **Driver Manager**.
2. In the Driver Manager, click the green plus icon to create a new driver.
3. Specify a name for the driver. For example, **Anzo JDBC Driver**.
4. In the URL Format field, specify the format **jdbc:anzo**.
5. In the **Driver File Paths** or **Driver jar Files** section of the screen, click the folder icon and then browse to and select the directory where you saved the CData JDBC Driver for Anzo cdata.jdbc.anzo.jar file that you downloaded to the server. DbVisualizer reads the jar and sets the Driver Class to **cdata.jdbc.anzo.AnzoDriver**. For example:



6. To connect to the endpoint in DbVisualizer, go to **Database → Create Database Connection**. Click **No Wizard** when prompted.
7. Specify a name for the connection in the **Name** field.
8. In the **Driver (JDBC)** field, select the Anzo JDBC driver connection.
9. In the **Database URL** field, specify the JDBC URL from the Anzo Data on Demand endpoint configuration. For example: `jdbc:anzo:URL=https://10.100.0.10/dataondemand/Sample-Graphmart/Sample-Data`

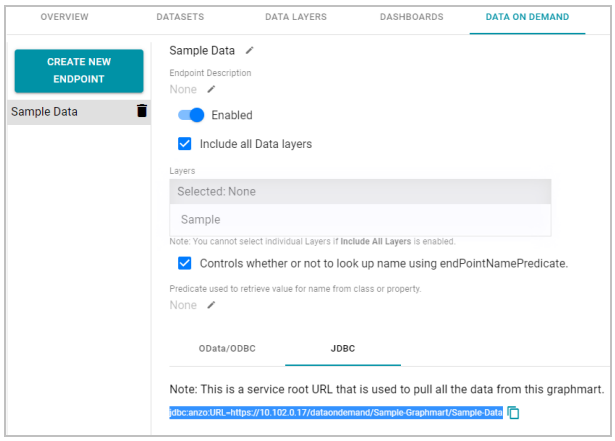


10. Under **Authentication**, enter your Anzo user ID and password. You should now be able to connect to the endpoint and view the available schemas. For example:



Example JDBC Setup with Tableau

1. After downloading the CData JDBC Driver for Anzo cdata.jdbc.anzo.jar file, place the .jar in the appropriate directory depending on your operating system:
    - **Windows**: C:\Program Files\Tableau\Drivers
    - **MacOS**: ~/Library/Tableau/Drivers
    - **Linux**: /var/opt/tableau/tableau_server/data/tabsvc/vizqlserver/Datasources/
2. Restart Tableau and then go to **Add a Connection** → **To a Server**.
3. Click **Other Databases (JDBC)**.
4. In the URL field, specify the JDBC URL from the Anzo Data on Demand endpoint configuration. For example:
   `jdbc:anzo:URL=https://10.100.0.10/dataondemand/Sample-Graphmart/Sample-Data`

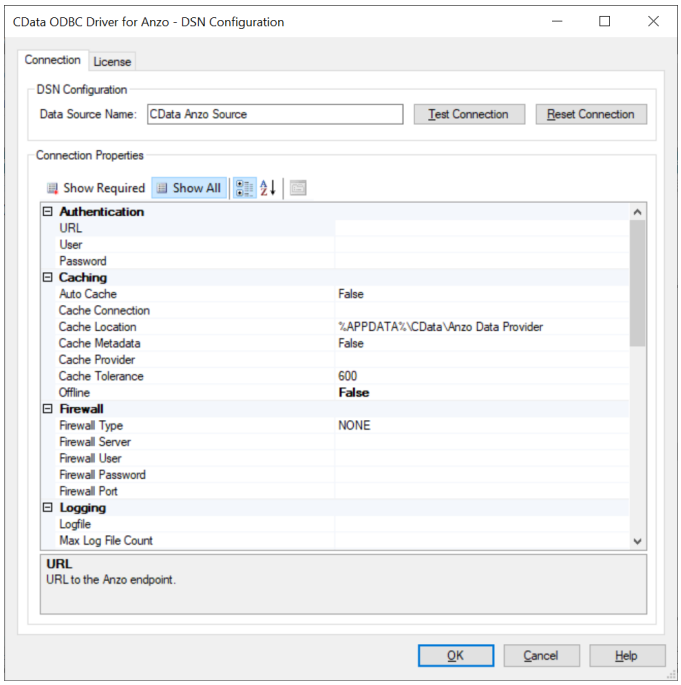


5. Enter your Anzo username and password and click **Sign In**. You should now be able to connect to the endpoint and view the available schemas.

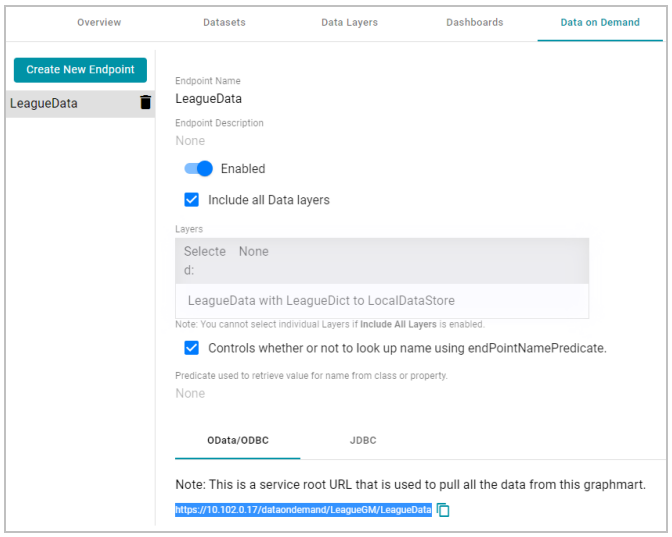
Example ODBC Setup with Microsoft Power BI

1. After downloading the Windows CData ODBC Driver for Anzo executable file, run the executable to start the installation wizard. The wizard guides you through installing the driver.

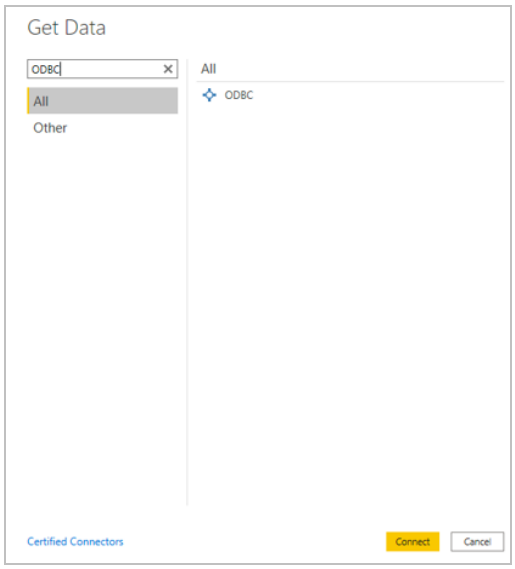
2. At the end of the installation, make sure the **Configure ODBC Data Source** checkbox is selected and click **Finish**. The wizard opens the driver's DNS Configuration screen. For example:



3. Under **Authentication** in Connection Properties, specify the **URL**, **User**, and **Password** to use for connecting to the Data on Demand endpoint. The User and Password are the Anzo username and password to use for authentication, and URL is the OData/ODBC service root URL for the endpoint. You can retrieve the URL from the Data on Demand screen for the endpoint. For example:

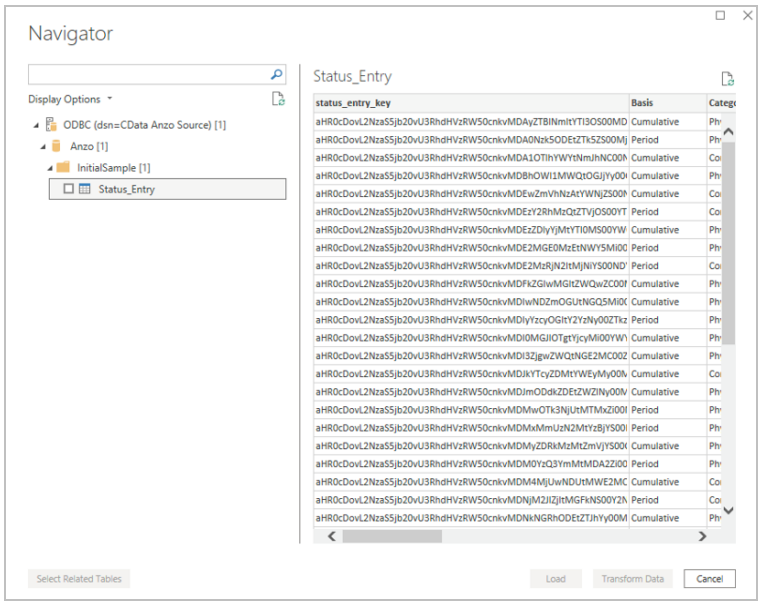


4. Click **OK** to save the configuration changes and close the dialog box.
5. Next, connect to the ODBC data source from Power BI. Open Power BI and click the **Get Data** button in the tool bar. In the Get Data dialog box, search for "ODBC." For example:



6. The search opens the wizard for creating an ODBC connection to a specified data source. Select **CData Anzo Source** from the drop-down list. You do not need to configure the advanced options.

7. Click **OK** to create the connection. Power BI opens the Navigator screen. For example:



Under Display Options, the top level container in the view represents the ODBC driver, the **Anzo** item represents the database, and the **InitialSample** item represents the schema. Each table is represented as a

table entry under the schema. In the example above there is one table. If you select a table, sample data from that table is displayed on the right side of the screen. To load table(s), select the checkbox for each table and click the **Load** button. You can also use the advanced features of Power BI to transform the data as you load it into the tool.

JDBC Driver Quick Reference

This section provides a quick reference for JDBC driver support.

- For the complete JDBC driver documentation, see CData JDBC Driver for Anzo.
- For the complete ODBC driver documentation, see CData ODBC Driver for Anzo.

SQL Compliance

The JDBC driver supports most of the standard operations for querying data. The exceptions are listed below.

- The driver does not currently support transactions.
- The driver does not support batching of SQL statements.
- The driver has support for inserting, updating, and deleting records. However, performing updates via the driver can have unexpected consequences.

For more information about SQL compliance, see the SQL Compliance section in the CData JDBC Driver documentation.

JDBC Performance Details

By default, the JDBC driver offloads to Anzo as much of the SELECT statement processing as possible and then processes the rest of the query locally in memory.

- For joins, the driver generates multiple OData queries and joins the results in memory. As a result, SQL queries that include JOINs can take up to several minutes to complete.
- For aggregates, the driver retrieves all rows necessary to process in memory.
- For predicates, the driver determines which clauses Anzo supports and sends them to Anzo to retrieve the smallest possible superset of rows that would satisfy the query. It then filters the rest of the rows client-side.
- The driver's **SupportEnhancedSQL** setting can be disabled to limit SQL execution to only what the Anzo API supports. For more information, see the Support Enhanced SQL section in the CData JDBC Driver documentation.

  Tip: To determine which query capabilities the driver can offload to the Anzo API, you can query the **sys_sqlinfo** system table. The table contains information about the functionality that is supported by the connected source. For example:

```
SELECT * FROM sys_sqlinfo WHERE name='AGGREGATE_FUNCTIONS'
or name = 'COUNT' or name = 'SUPPORTED_OPERATORS' or name = 'GROUP_BY'
```

```
or name = 'OUTER_JOINS' or name = 'OJ_CAPABILITIES' or name = 'SUBQUERIES'
or name = 'STRING_FUNCTIONS' or name = 'NUMERIC_FUNCTIONS'
or name = 'TIMEDATE_FUNCTIONS';
```

For more information, see the sys_sqlinfo section in the CData JDBC Driver documentation.

Data Caching

Due to the client-side in-memory processing of aggregates and joins, the performance of queries against extremely large data sets may suffer. If this is a common use case, consider leveraging caching in the JDBC driver. If the driver maintains a local copy of the data, it reduces the number of API calls and can increase performance for long-running queries. For more information, see the Caching Data section in the CData JDBC Driver documentation.

Supported SELECT Statement Clauses

The following list shows the supported SELECT statement clauses. For more information, see the SELECT Statement section in the CData JDBC Driver documentation.

- SELECT
- INTO
- FROM
- JOIN
- WHERE
- GROUP BY
- HAVING
- UNION
- ORDER BY
- LIMIT

Supported Aggregate Functions

The following list shows the supported aggregate functions. For more information, see the Aggregate Functions section in the CData JDBC Driver documentation.

- COUNT
- COUNT_DISTINCT
- AVG
- MIN
- MAX
- SUM

Supported Joins

The following list shows the supported JOIN types. For more information, see the JOIN Queries section in the CData JDBC Driver documentation.

- **Inner Join**: Selects only the rows from both tables that match the join condition.
- **Left Join**: Selects all of the rows in the FROM table and only matching rows in the JOIN table.

SQL Function Reference

The JDBC driver provides implementations of the following common SQL functions. For more information, see the SQL Functions section in the CData JDBC Driver documentation.

Note The driver interprets all function input as either column names or strings. Therefore, all string literals must be escaped with single quotes. For example, `SELECT DATENAME('yy',GETDATE()))`.

String Functions

- `ASCII(character_expression)`
- `CHAR(integer_expression)`
- `CHARINDEX(expressionToFind ,expressionToSearch [,start_location ])`
- `CONCAT(string_value1, string_value2 [, string_valueN])`
- `CONTAINS(expressionToSearch, expressionToFind)`
- `ENDSWITH(character_expression, character_suffix)`
- `FORMAT(value, format)`
- `FROM_UNIXTIME(time, format, issecond)`
- `INDEXOF(expressionToSearch, expressionToFind [,start_location ])`
- `ISNULL(check_expression , replacement_value)`
- `JSON_AVG(json, jsonpath)`
- `JSON_COUNT(json, jsonpath)`
- `JSON_EXTRACT(json, jsonpath)`
- `JSON_MAX(json, jsonpath)`
- `JSON_MIN(json, jsonpath)`
- `JSON_SUM(json, jsonpath)`
- `LEFT(character_expression , integer_expression)`
- `LEN(string_expression)`
- `LOWER(character_expression)`
- `LTRIM(character_expression)`
- `NCHAR(integer_expression)`
- `PATINDEX(pattern, expression)`
- `QUOTENAME(character_string [, quote_character])`

- REPLACE(string_expression, string_pattern, string_replacement)
- REPLICATE(string_expression ,integer_expression)
- REVERSE(string_expression)
- RIGHT(character_expression , integer_expression)
- RTRIM(character_expression)
- SOUNDEX(character_expression)
- SPACE(repeatcount)
- STARTSWITH(character_expression, character_prefix)
- STR(float_expression [, integer_length [, integer_decimal]])
- STUFF(character_expression , integer_start , integer_length , replaceWith_expression)
- SUBSTRING(expression,integer_start,integer_length)
- TOSTRING(string_value1)
- TRIM(character_expression)
- UNICODE(ncharacter_expression)
- UPPER(character_expression)
- XML_EXTRACT(xml, xpath [, separator])

Date Functions

- CURRENT_DATE()
- CURRENT_TIMESTAMP()
- DATEADD(datepart , integer_number , date [, dateformat])
- DATEDIFF(datepart , startdate , enddate)
- DATEFROMPARTS(integer_year, integer_month, integer_day)
- DATENAME(datepart , date)
- DATEPART(datepart, date [,integer_datefirst])
- DATETIME2FROMPARTS(integer_year, integer_month, integer_day, integer_hour, integer_minute, integer_seconds, integer_fractions, integer_precision)
- DATETIMEFROMPARTS(integer_year, integer_month, integer_day, integer_hour, integer_minute, integer_seconds, integer_milliseconds)
- EOMONTH(start_date [, integer_month_to_add])
- GETDATE()
- GETUTCDATE()
- ISDATE(date, [date_format])

- SMALLDATETIMEFROMPARTS(integer_year, integer_month, integer_day, integer_hour, integer_minute)
- SYSDATETIME()
- SYSUTCDATETIME()
- TIMEFROMPARTS(integer_hour, integer_minute, integer_seconds, integer_fractions, integer_precision)
- YEAR(date)

Math Functions

- ABS(numeric_expression)
- ACOS(float_expression)
- ASIN(float_expression)
- ATAN(float_expression)
- ATN2(float_expression1 , float_expression2)
- CEILING(numeric_expression)
- COS(float_expression)
- COT(float_expression)
- DEGREES(numeric_expression)
- EXP(float_expression)
- EXPR(expression)
- FLOOR(numeric_expression)
- LOG(float_expression [, base])
- LOG10(float_expression)
- PI( )
- POWER(float_expression , y)
- RADIANS(float_expression)
- RAND([integer_seed])
- ROUND(numeric_expression , integer_length [,function])
- SIGN(numeric_expression)
- SIN(float_expression)
- SQRT(float_expression)
- SQUARE(float_expression)
- TAN(float_expression)

Related Topics

Creating a Data on Demand Endpoint

Accessing an Endpoint Programmatically

OData Reference

OData Reference

The Anzo Data on Demand service follows the OData Version 4.0 specification, which defines the standard URL conventions and query options. This topic provides a quick reference for learning OData basics and viewing the supported string operators and output formats. It also provides some example queries.

Note The Anzo Data on Demand service does not impose limitations on the data that can be retrieved. However, some third-party applications do not support multi-value properties.

- OData URL Conventions
- Supported Query Operators
- Example OData Requests

OData URL Conventions

An OData service URL has three main parts:

1. The **Service Root URL** that Anzo provides. The service root URL is the metadata that describes all of the available feeds (tables).
2. The optional **Resource Path** that narrows the scope of the available data to the individual table (class) level, property level, or the schema.
3. The **Query Options** for analyzing the data.

For example, the following OData URL shows the service root from the Data on Demand screen in Anzo, a resource path that narrows the scope of the data to the Employees table (class), and query options that filter the result set to show data for the NA region only:

```
https://10.100.0.10/dataondemand/Northwind-Graphmart/Northwind/Employees?$filter=contains(Region, 'NA')
\_____________________________________________________/ \_______/  \_____________________________/
                      Service Root URL                   Resource Path       Query Options
```

OData requests need to be URL-encoded. Typically you can configure programs to encode requests automatically. And browsers encode URLs that are pasted into the address bar.

Supported Query Operators

OData query options are used to dynamically query data via the endpoint and control the amount and order of the data returned. The Data on Demand service supports the following OData query operators. See Example OData Requests below for example queries that employ the operators.

Operator	Description
$count	Used to count the number of matching resources in the result set.
$expand	Used to retrieve related data and include it in the results. When you query data via OData, the default response does not include related entities. The $expand option allows the related information to be embedded in the response.
$filter	Used to filter a result set. The expression specified with $filter is evaluated for each resource identified by the resource path, and only items where the expression evaluates to true are included in the response.
$format	Used to specify the output format for the results. The supported formats are text/CSV, JSON, and XML. For example: $format=json
$orderby	Used to return results in ascending (asc) or descending (desc) order. If asc or desc is not specified, solutions are returned in ascending order.
$select	Used to specify the subset of properties to include in the result set.
$skip	Used to specify the number of solutions to exclude in the results. The $top and $skip OData query options are similar to the LIMIT and OFFSET clauses in SPARQL queries.
$top	Used to limit the number of solutions that are returned.

Example OData Requests

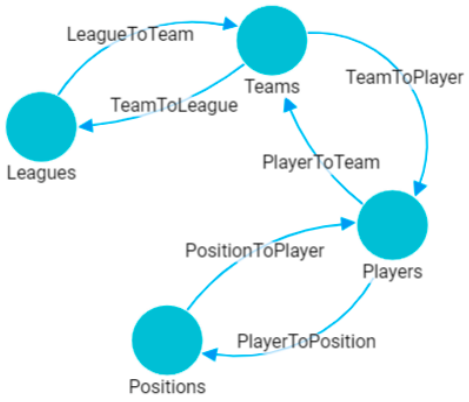
This section demonstrates the use of OData query operators by providing examples of common types of OData requests.

The examples below are run against a sample graphmart, called **LeagueGM**, that contains data about the teams and players in a small local baseball league. The Data on Demand endpoint is named **LeagueData**. The following service root URL was created by Anzo:

```
https://10.100.0.10/dataondemand/LeagueGM/LeagueData
```

For readability, the examples below abbreviate "https://10.100.0.10/dataondemand" to **dataondemand**. In addition, the examples are not URL-encoded.

The data has Leagues, Teams, Players, and Positions classes (or entities in OData). And the image below shows a graph view of the data model:



To view details about the properties and values for each class, you can click a link below to view the data for that class. The data is in JSON format.

Leagues Teams Players Positions

Counting an Entity

The request below returns the number of teams in the graphmart. Adding the resource path **Teams** to the request narrows the scope to the Teams entity (or class in Anzo).

```
dataondemand/LeagueGM/LeagueData/Teams/$count
```

Result

```
4
```

This request returns the number of players:

```
dataondemand/LeagueGM/LeagueData/Players/$count
```

Result

```
12
```

Counting a Property of an Entity

The request below counts the number of players on the Al Thomas team. The request uses the team_key to identify the team and the TeamToPlayer to identify each player.

```
dataondemand/LeagueGM/LeagueData/Teams
('aHR0cDovL2NzaS5jb20vVGVhbXMvMQ')/TeamToPlayer/$count
```

Result

```
3
```

This request counts the number of positions played by James Smith:

```
dataondemand/LeagueGM/LeagueData/Players
('aHR0cDovL2NzaS5jb20vUGxheWVycy8y')/PlayerToPosition/$count
```

Result

```
2
```

Filtering Data via Text Search

The request below filters the results to show data for the TeamName that equals "Black Sox." The request also returns results in JSON format:

```
dataondemand/LeagueGM/LeagueData/Teams?$filter=TeamName eq 'Black Sox'&$format=json
```

Result

```
{
  "@odata.context":
"https://10.100.0.10/dataondemand/LeagueGM/LeagueData/$metadata#Teams",
  "value": [
    {
      "teams_key": "aHR0cDovL2NzaS5jb20vVGVhbXMvMg",
      "TeamId": 2,
      "teamtoleague_key": [
          "aHR0cDovL2NzaS5jb20vTGVhZ3Vlcy8x"
      ],
      "TeamName": "Black Sox",
      "teamtoplayer_key": [
          "aHR0cDovL2NzaS5jb20vUGxheWVycy80",
          "aHR0cDovL2NzaS5jb20vUGxheWVycy81",
          "aHR0cDovL2NzaS5jb20vUGxheWVycy82"
      ]
    }
  ]
}
```

This request filters the data to find the players whose name contains "Ted."

```
dataondemand/LeagueGM/LeagueData/Players?$filter=contains(PlayerName,'Ted')
```

The request can also use "startswith" in place of contains to filter specifically for player names that start with "Ted."

```
dataondemand/LeagueGM/LeagueData/Players?$filter=startswith(PlayerName,'Ted')
```

Result


```
{
  "@odata.context":
"https://10.100.0.10/dataondemand/LeagueGM/LeagueData/$metadata#Players",
  "value": [
    {
      "players_key": "aHR0cDovL2NzaS5jb20vUGxheWVycy8xMA",
      "playertoposition_key": [
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzM",
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzI"
      ],
      "PlayerId": 10,
      "playertoteam_key": [
          "aHR0cDovL2NzaS5jb20vVGVhbXMvNA"
      ],
      "PlayerName": "Ted James",
      "DefensiveRating": 92.55
    },
    {
      "players_key": "aHR0cDovL2NzaS5jb20vUGxheWVycy84",
      "playertoposition_key": [
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzI",
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzEw"
      ],
      "PlayerId": 8,
      "playertoteam_key": [
          "aHR0cDovL2NzaS5jb20vVGVhbXMvMw"
      ],
      "PlayerName": "Ted Sale",
      "DefensiveRating": 77.33
    }
  ]
}
```


Selecting Properties and Ordering Results

The request below selects player names and their defensive ratings. The results are ordered by defensive rating in descending order so that the player with the highest defensive rating is listed first. The request also formats the results in text/csv.

```
dataondemand/LeagueGM/LeagueData/Players?$select=PlayerName,DefensiveRating&$orderby=Defen
siveRating desc&$format=text/csv
```

Result

```
PlayerName,DefensiveRating
James Smith,98.33
Alex Granderson,98.22
Matt Butler,95.66
Tim Hooper,93.43
Steve Jones,93.28
Ted James,92.55
Fred Wynn,88.68
Jared Bonds,86.34
Billy Roper,83.44
Mike Magazine,78.33
Ted Sale,77.33
Chris Underwood,66.22
```

Expanding the Results to Include Related Entities

The request below uses the $expand operator to retrieve data from the Players entity and include the related Positions data for each player. For this example, the request limits the number of results returned to 2 players by adding $top=2:

```
dataondemand/LeagueGM/LeagueData/Players?$expand=PlayerToPosition&$top=2
```

Result

```
{
  "@odata.context":
"https://10.100.0.10/dataondemand/LeagueGM/LeagueData/$metadata#Players",
  "value": [
    {
      "players_key": "aHR0cDovL2NzaS5jb20vUGxheWVycy8x",
      "playertoposition_key": [
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzg"
      ],
      "PlayerId": 1,
      "playertoteam_key": [
          "aHR0cDovL2NzaS5jb20vVGVhbXMvMQ"
      ],
      "PlayerName": "Steve Jones",
      "DefensiveRating": 93.28,
      "PlayerToPosition": [
          {
            "positions_key": "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzg",
            "PositionId": 8,
            "ShortName": "CF",
            "positiontoplayer_key": [
                "aHR0cDovL2NzaS5jb20vUGxheWVycy8xMg",
```

```
                "aHR0cDovL2NzaS5jb20vUGxheWVycy8x"
            ],
            "Description": "Centerfield"
          }
      ]
    },
    {
      "players_key": "aHR0cDovL2NzaS5jb20vUGxheWVycy8xMA",
      "playertoposition_key": [
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzI",
          "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzM"
      ],
      "PlayerId": 10,
      "playertoteam_key": [
          "aHR0cDovL2NzaS5jb20vVGVhbXMvNA"
      ],
      "PlayerName": "Ted James",
      "DefensiveRating": 92.55,
      "PlayerToPosition": [
          {
            "positions_key": "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzI",
            "PositionId": 2,
            "ShortName": "C",
            "positiontoplayer_key": [
                "aHR0cDovL2NzaS5jb20vUGxheWVycy84",
                "aHR0cDovL2NzaS5jb20vUGxheWVycy8xMA"
            ],
            "Description": "Catcher"
          },
          {
            "positions_key": "aHR0cDovL2NzaS5jb20vUG9zaXRpb25zLzM",
            "PositionId": 3,
            "ShortName": "1B",
            "positiontoplayer_key": [
                "aHR0cDovL2NzaS5jb20vUGxheWVycy83",
                "aHR0cDovL2NzaS5jb20vUGxheWVycy8xMA"
            ],
            "Description": "First Base"
          }
        ]
      }
   ]
}
```

Related Topics

Creating a Data on Demand Endpoint

Accessing an Endpoint Programmatically

Accessing an Endpoint from an Application

Accessing Data via the SPARQL Endpoint

Anzo offers a standard HTTP(S) SPARQL endpoint for sending SPARQL requests between client applications and Anzo. The endpoint is enabled by default. This topic provides the base endpoint URL and describes the supported HTTP methods and parameters.

Authentication

The Anzo SPARQL endpoint supports Basic Authentication. The endpoint can be configured to enable other Anzo-supported authentication methods. However, implementing alternate authentication mechanisms can have unexpected results. For more information, contact Cambridge Semantics Support.

Note Ultimately the data that is available to users from SPARQL endpoints is subject to the security and composition of the graphmart or linked data set as configured in Anzo.

HTTP Methods and Options

The Anzo SPARQL endpoint accepts HTTP GET and POST methods. GET is used to retrieve data from the endpoint, and POST is used to send data to the endpoint. Update queries must use the POST method, and read queries can be submitted using GET or POST.

Endpoint Base URL

Use the following base URL to access data in Anzo via the SPARQL endpoint. The table below describes each base URL component:

```
protocol://hostname:port/sparql/store_type/url-encoded_dataset_uri
```

Option	Description
protocol	The protocol to use for the connection: **http** for HTTP protocol or **https** for SSL protocol.
hostname	The DNS name or IP address of the Anzo server.
port	The port for the endpoint. The port that you specify depends on the protocol that you choose. By default, the HTTP port is **80** and the HTTPS port is **443**. To view the ports that are configured for your Anzo instance, see **Server Settings** in the **Administration** menu.

Option	Description
sparql	Required keyword for the SPARQL endpoint.
store_type	The type of RDF store for the data. Typically users specify **graphmart** to query data that is in a graphmart. It is also possible to query the metadata for a linked data set (LDS) in the Dataset catalog. To query an LDS that is stored in a local volume, specify **lds** as the store type.
url-encoded_ dataset_uri	The URI for the graphmart or the catalog entry for the LDS. The URI must be URL-encoded using upper case hexadecimal digits. Lower case hexadecimal digits are not supported at this time. How do I find the URI for a graphmart? How do I find the catalog entry URI for an LDS?

For example, the following base endpoint URL targets the data in a graphmart:

```
https://10.100.10.20:8443/sparql/graphmart/http%3A%2F%2Fcambridgesemantics.com%2FGraphmart%2F1ad0ee911b834097ad7f71ee0ae1c0ff
```

The example below shows a base endpoint URL that targets a Dataset catalog entry:

```
https://10.100.10.20:8443/sparql/lds/http%3A%2F%2Fopenanzo.org%2FcatEntry(%255Bhttp%253A%252F%252Fcsi.com%252FFileBasedLinkedDataSet%252F001e517db4f0eaea9f279427e4e2a828%255D%2540%255Bhttp%253A%252F%252Fopenanzo.org%252Fdatasource%252FsystemDatasource%255D)
```

HTTP Header Options

The HTTP header provides information related to the transfer of data between the requesting client and the SPARQL endpoint. The table below describes the supported HTTP header options. Both of the fields are optional.

Option	Description
Content-Type	The Content-Type specifies the type of request that is being sent by the client. Anzo supports the following Content-Type values: • **application/x-www-form-urlencoded**: Including this value specifies that the query string will be passed as the value of a "query" or "update" HTTP parameter. This is the default value. When Content-Type is not specified, the endpoint behaves as if `Content-Type: application/x-www-form-urlencoded` is specified. • **application/sparql-query**: Including this value specifies that the HTTP request body includes a SPARQL read (non-update) query. • **application/sparql-update**: Including this value specifies that the HTTP request body includes a SPARQL update query.
Accept	The Accept field specifies the response formats that are acceptable for the server to send back to the client. You can use this field to specify the output serialization format for query results in place of the format HTTP parameter. For details about the supported formats, see Format Options below.

HTTP Body Parameters

The HTTP parameters in the body of the request provide the rest of the information about the request. Certain parameters are appropriate for read-only queries, SELECT and CONSTRUCT, and others are appropriate for updates, INSERT and DELETE. The tables below describe the supported parameters for query and update requests.

Query Parameters

Parameter	Description
query	Specifies the full read-only query string to run. If you do not specify a url-encoded_dataset_uri, default-graph-uri or named-graph-uri in the request, the query string should contain the appropriate FROM clauses. To run an update query (INSERT or DELETE), use the update parameter.
default-graph-uri	Specifies a default graph URI to query. You can include this parameter multiple times in a request. When the base URL specifies a graphmart URI, you can specify a data layer URI to narrow the scope of the query to a specific data layer in the graphmart.

Parameter	Description
named-graph-uri	Specifies a named graph URI to query. You can include this parameter multiple times in a request. When the base URL specifies a graphmart URI, you can specify a data layer URI to narrow the scope of the query to a specific data layer in the graphmart.
format	Specifies the serialization format to use for the results of the query. For details about the supported formats, see Format Options below.
includeMetadataGraphs	A boolean value that specifies whether to query the metadata graphs. Only valid for queries that target a linked data set (LDS) that is stored in a local volume. The default value is **false**.
delim	Specifies a custom delimiter character to use in CSV output results. Valid only for SELECT queries where the output format is **text/csv**. This field accepts any character. When delim is not specified the default value is a **,** (comma).
dedup	A boolean value that specifies whether to deduplicate CONSTRUCT results on the client side. When dedup is not specified, the default value is **true**.
serverDedup	A boolean value that specifies whether to deduplicate CONSTRUCT results on the server side. When dedup is not specified, the default value is **true**.
hasHeader	A boolean value that specifies whether to include headers in CSV results. Valid only for SELECT queries where the output format is **text/csv**. When hasHeader is not specified, the default value is **false**.
attachResult	A boolean value that specifies whether to provide the query response as a file "attachment," i.e. the HTTP response will include the Content-Disposition of **attachment**. When attachResult is not specified, the default value is **false**. When returning results as an attachment, you can specify a file name in filename the parameter.
filename	If attachResult is true, this parameter specifies the file name to use for the attachment, excluding the file extension. If attachResult is true and filename is not specified, the default file name is **QueryResult**.

Format Options

The table below describes the options for specifying the serialization format of the results that the server sends back to the client. These format options, i.e., MIME types or file extensions, can be specified in the format parameter in the body of the request or in the Accept header.

Note When the request does not include the format parameter or Accept header, the default result format for SELECT queries is SPARQL XML (application/sparql-results+xml). For CONSTRUCT queries, the default format depends on whether the query includes GRAPH clauses. If no GRAPH clause is present, the default format for CONSTRUCT results is RDF Turtle. If GRAPH clauses are present, the default format is RDF TriG.

Format	Accepted Values	Query Type	Description
XML	**application/sparql-results+xml** **xml** **srx**	SELECT and CONSTRUCT	For SELECT queries, results are returned in SPARQL Query Results XML Format For CONSTRUCT queries, results are returned in RDF 1.1 XML format
	application/xml **xml2**	SELECT only	Returns SPARQL Query Results XML Format
	application/rdf+xml **rdf** **owl** **rdfs**	CONSTRUCT only	Returns RDF 1.1 XML
JSON	**application/json** **application/sparql-results+json** **json**	SELECT and CONSTRUCT	For SELECT queries, results are returned in SPARQL Query Results JSON Format For CONSTRUCT queries, results are returned in Anzo's native JSON RDF serialization format. See Anzo JSON RDF Serialization for details.
CSV	**text/csv** **csv**	SELECT only	Returns SPARQL Query Results CSV Format

Format	Accepted Values	Query Type	Description
TriG and Gzipped TriG	**application/x-trig** **trig** **application/x-trigz** **trigz** **gz** **trig.gz**	CONSTRUCT only	CONSTRUCT queries with a GRAPH clause return RDF 1.1 TriG by default if no format is specified
Turtle and Gzipped Turtle	**application/x-turtle** **ttl** **application/x-turtlez** **ttlz** **ttl.gz**	CONSTRUCT only	Returns RDF 1.1 Turtle CONSTRUCT queries without a GRAPH clause return Turtle by default if no format is specified
N-Triples	**text/plain** **nt**	CONSTRUCT only	Returns RDF 1.1 N-Triples
Notation3 and Gzipped Notation3	**text/rdf+n3** **n3** **text/rdf+n3z** **n3z** **n3z.gz**	CONSTRUCT only	Returns RDF Notation3
N-Quads	**text/x-nquads** **nq** **nquad** **nquads**	CONSTRUCT only	Returns RDF 1.1 N-Quads
TriX	**application/trix** **trix**	CONSTRUCT only	Returns RDF Triples in XML

Update Parameters

Parameter	Description
update	Specifies the full update string to run. If you do not specify a url-encoded_dataset_uri, using-graph-uri or using-named-graph-uri in the request, the update query should contain the appropriate USING clauses. To run a non-update query (SELECT or CONSTRUCT), use the query parameter.
using-graph-uri	Specifies a default graph URI to update. You can include this parameter multiple times in a request. When the base URL specifies a graphmart URI, you can specify a data layer URI to narrow the scope of the update to a specific data layer in the graphmart.
using-named-graph-uri	Specifies a named graph URI to update. You can include this parameter multiple times in a request. When the base URL specifies a graphmart URI, you can specify a data layer URI to narrow the scope of the update to a specific data layer in the graphmart.
includeMetadataGraphs	A boolean value that specifies whether to query the metadata graphs. Only valid for queries that target a linked data set (LDS) that is stored in a local volume. The default value is **false**.

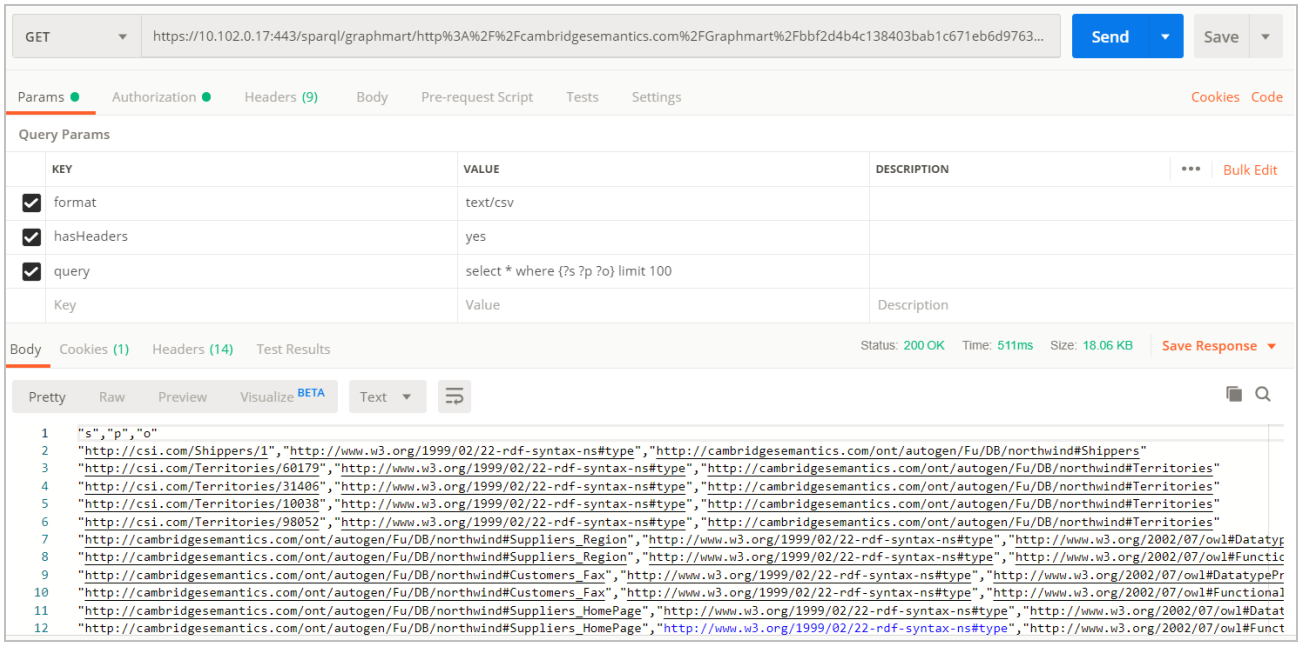
Examples

The following example uses cURL to send a request that runs a SELECT query against a graphmart. Since the request does not include an Accept header or format parameter, results will be returned in SPARQL XML format.

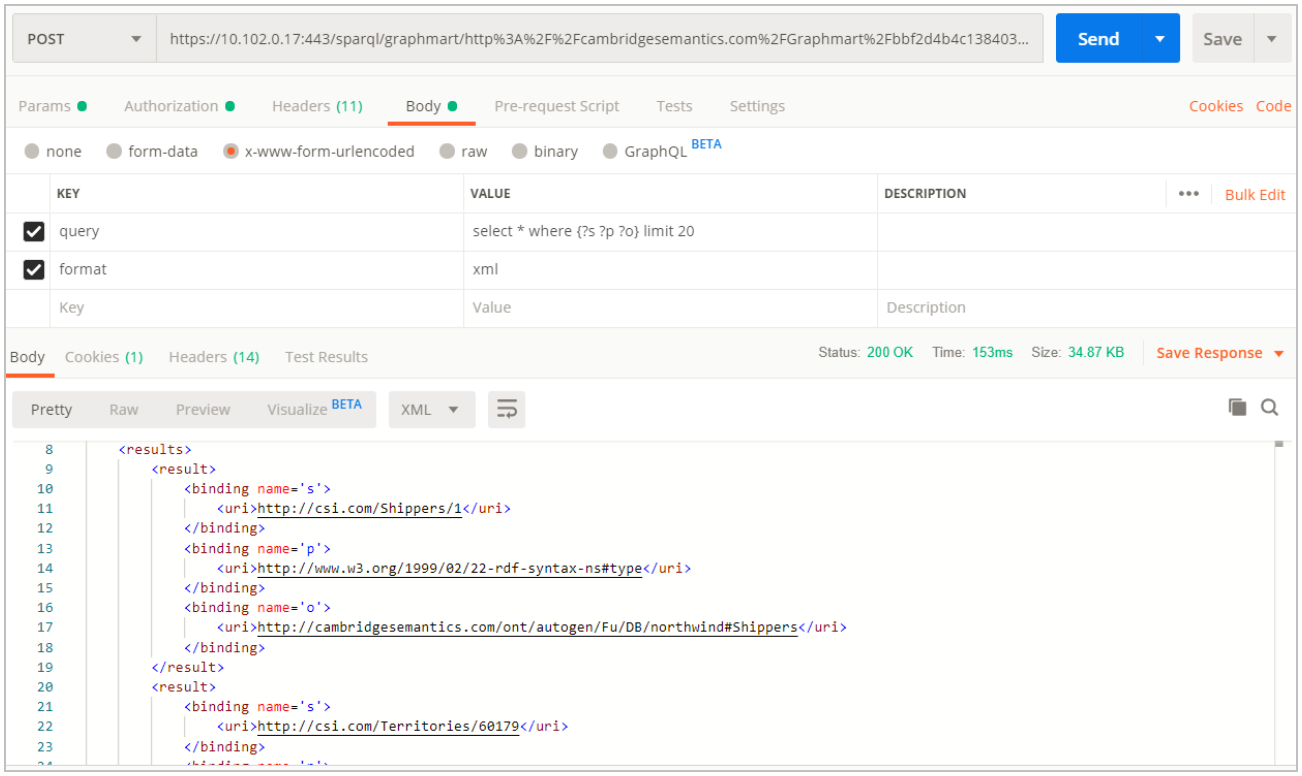
```
curl --user sysadmin:@nz0 -c cookiejar.txt -L -v -k

http://10.100.10.20/sparql/graphmart/http%3A%2F%2Fcambridgesemantics.com%2FGraphmart%2F2dc579b101654ae29eb91b0c7d046ca1
--data-urlencode "query=SELECT * WHERE{ ?s ?p ?o . } LIMIT 100"
```

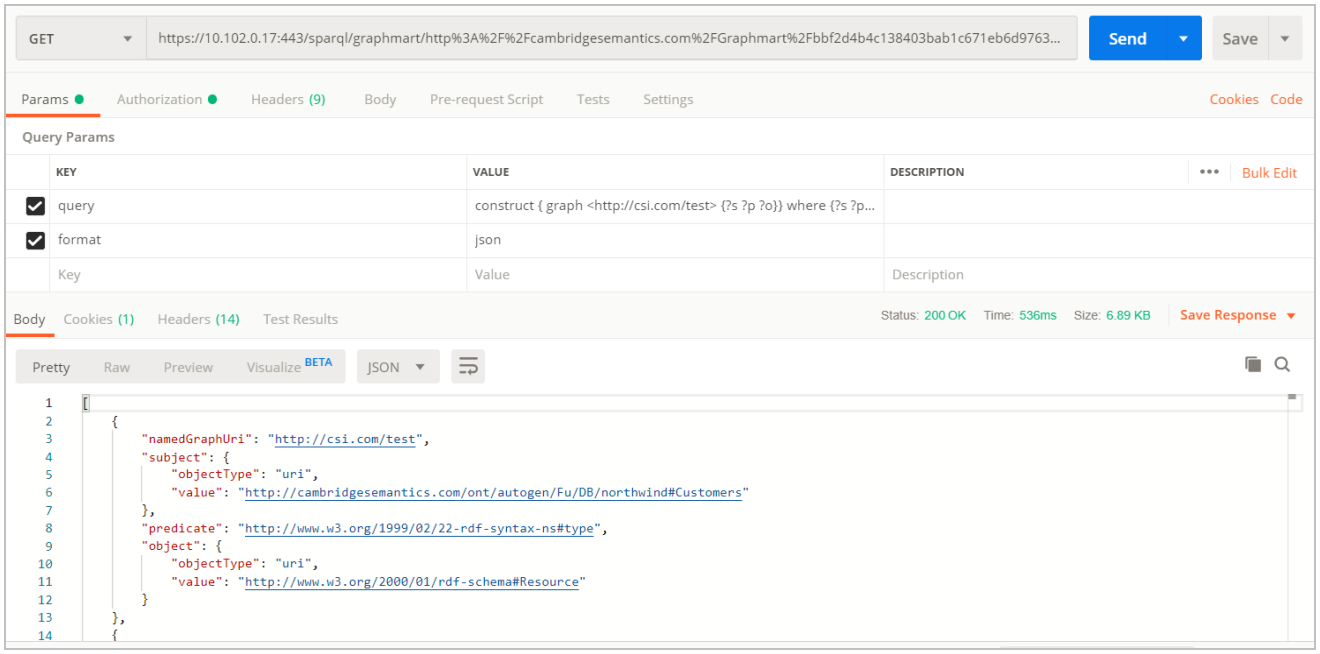
The following example sends a GET request that runs a SELECT query against a graphmart. The format parameter is included to format the results in text/csv serialization.



The example below sends a POST request that runs a SELECT query. In this example, the query is included in the body of the request and the response format is XML.



The example below sends a GET request that runs a CONSTRUCT query. The response format is set to JSON, and the results are formatted in Anzo JSON RDF Serialization.



The example below uses a Python script to send a request that runs a SPARQL query.

```
import requests
import urllib

server = 'https://company.anzo.com:'
port = 443
graphmart = 'http://cambridgesemantics.com/Graphmart/be4bd080c5654628b6fff90ca1b647d6'
url = server + str(port) + '/sparql/graphmart/' + urllib.quote_plus(graphmart)
#urllib.parse.quote_plus(graphmart) in Python 3

queryText = 'SELECT * WHERE {?instance a ?type .} LIMIT 10'
payload = {'query':queryText, 'format':'text/csv'}

r = requests.post(url, data = payload, auth = ('sysadmin','<pw>'))
print r.text
```

Related Topics

Analyzing Data with Hi-Res Analytics

Accessing Data with the Query Builder

Accessing Data from Data on Demand Endpoints

Accessing Data via the HTTP Client Interface

SPARQL Query Templates and Best Practices

Accessing Data via the HTTP Client Interface

In addition to the SPARQL HTTP(S) endpoint that enables users to send SPARQL queries to Anzo over HTTP, Anzo provides an HTTP(S) servlet that enables users to invoke Anzo client operations over HTTP. The client servlet enables external systems to interact with Anzo semantic services as well as custom services. It also enables remote servers to interact with Anzo without needing the Anzo command line interface.

HTTP Methods and Options

The Anzo client servlet accepts HTTP GET and POST methods. GET is used for operations that retrieve data, and POST is used for update operations that add or remove data. Update operations must use the POST method, and read operations can be submitted using GET or POST.

Client Servlet Base URL

Use the following URL to access Anzo services via the HTTP client servlet. The table below describes each URL component:

```
protocol://hostname:port/anzoclient/client_operation
```

For example:

```
https://10.100.10.20:8443/anzoclient/call
```

Option	Description
protocol	The protocol to use for the connection: **http** for HTTP protocol or **https** for SSL protocol.
hostname	The DNS name or IP address of the Anzo server.
port	The port for the endpoint. The port that you specify depends on the protocol that you choose. By default, the HTTP port is **80** and the HTTPS port is **443**. To view the ports that are configured for your Anzo instance, see **Server Settings** in the **Administration** menu.
anzoclient	Required keyword for the client servlet.

Option	Description
client_operation	The type of Anzo client operation to invoke. The list below provides an overview of the supported operation types. For more information about the operations, see Client Operations below. • **call**: Invokes the semantic service operation identified by the URI provided in the request (analogous to the `anzo call` CLI command) • **add**: Imports the specified statements to Anzo (analogous to the `anzo import` CLI command) • **remove**: Removes the specified statements from Anzo (analogous to the `anzo update -r` CLI command) • **get**: Gets the specified named graph from Anzo (analogous to the `anzo get` CLI command) • **find**: Finds the statements in Anzo that match the specified pattern (analogous to the `anzo find` CLI command)

Client Operations

This section provides usage information and examples for each of the Anzo client operations.

- Call
- Add
- Remove
- Get
- Find

Call

The call operation invokes a semantic service. Identify the service to call by providing the URI for the service in the request header. The call operation is supported with HTTP GET and POST methods. When including RDF data as input to the service, the request must use the POST method.

Call Header Options

Call operations support the following header parameters. Only the uri parameter is required:

- **uri**: **Required** parameter that specifies the URI of the semantic service to invoke.
- **mimeType**, **Content-Type**, or **format**: Include on of these optional parameters to specify the MIME type for the RDF serialization used in the request body as well as the response from the service. The default type is **application/json** if the header does not specify the mimeType, Content-type, or format. For more information about the supported RDF serialization types, see Format Options.

Call Body Options

If the call operation supplies data as input to the service, include the data in the request body. The data must be serialized as specified in the request header, or **application/json** if the header does not specify a serialization type.

Call Examples

The following cURL example uses a GET call to invoke a health check service.

```
curl https://10.100.10.20:8443/anzoclient/call \
  --user sysadmin:123 \
  --header 'uri: http://www.csi.com/service/genericIngestManager#healthCheck'
```

The example below uses a POST call to invoke a service operation. The call passes in a request data set that is serialized as RDF JSON.

```
curl https://10.100.10.20:8443/anzoclient/call \
  --header 'Content-Type: application/json' \
  --user sysadmin:123 \
  --header 'uri: http://someServiceURI#someOperation' \
  --data '{"subject" : {"objectType": "uri" ,"value" : "urn://test"},
           "predicate" : "urn://predicate",
           "object" : {"objectType": "uri" ,"value" : "urn://object"},
           "namedGraphUri" : "urn://ng"}'
```

The example below uses a POST call to invoke a service operation. The call passes in a request data set that is serialized as TriG.

```
curl https://10.100.10.20:8443/anzoclient/call \
  --header 'Content-Type: application/x-trig' \
  --user sysadmin:123 \
  --header 'uri: http://www.csi.com/service/genericIngestManager#healthCheck'
  --data '<urn://ng> { <urn://test> <urn://predicate> <urn://object> .}'
```

Add

The add operation adds statements to the Anzo graphstore. Add is supported with the HTTP POST method. Header options are not applicable, and the request body includes the statements to add. **The statements to add must be specified in Anzo JSON RDF serialization format**. See Anzo JSON RDF Serialization below for details.

Add Examples

The following example add operation uses cURL to issue a POST call to add a statement to the graphstore. The statement is specified in Anzo JSON RDF serialization format.

```
curl https://10.100.10.20:8443/anzoclient/add \
  --user sysadmin:123 \
  --data '{"subject" : {"objectType": "uri" ,"value" : "urn://test"},
           "predicate" : "urn://predicate",
```

```
          "object" : {"objectType": "uri" ,"value" : "urn://object"},
          "namedGraphUri" : "urn://ng"}'
```

Remove

The remove operation deletes statements from the Anzo graphstore. Remove is supported with the HTTP POST method. Header options are not applicable, and the request body specifies the statements to remove. **The statements to remove must be specified in Anzo JSON RDF serialization format**. See Anzo JSON RDF Serialization below for details.

Remove Examples

The following example remove operation uses cURL to issue a POST call to remove a statement from the graphstore. The statement is specified in Anzo JSON RDF serialization format.

```
curl https://10.100.10.20:8443/anzoclient/remove \
  --user sysadmin:123 \
  --data '{"subject" : {"objectType": "uri" ,"value" : "urn://test"},
           "predicate" : "urn://predicate",
           "object" : {"objectType": "uri" ,"value" : "urn://object"},
           "namedGraphUri" : "urn://ng"}'
```

Get

The get operation retrieves a named graph from the Anzo graphstore. The get operation is supported with HTTP GET and POST methods. Header options are not applicable. The named graph URI that contains the contents to retrieve can be included as a query parameter or as a uri parameter in the request body. The get operation also returns the metadata graph, which is equivalent to running `anzo get -m <named_graph_uri>` with the Anzo admin CLI. Graphs are returned in Anzo JSON RDF serialization format. See Anzo JSON RDF Serialization below for details.

Get Examples

The following example get operation uses cURL to retrieve the contents of a named graph.

```
curl -k -XPOST https://10.100.10.20:8443/anzoclient/get --user sysadmin:123
     --data-urlencode
     "uri=http://cambridgesemantics.com/Graphmart/9da211618a15476daa10cead2292d8e7"
```

This example uses Python with requests:

```
import requests

url = "https://10.100.10.20:8443/anzoclient/get"
data = {"uri": "http://cambridgesemantics.com/Graphmart/9da211618a15476daa10cead2292d8e7"}
username = "sysadmin"
password = "123"
```

```
r = requests.post(url, data=data, auth=(username, password), verify=False)
print (r.text
```

Find

The find operation finds the statements in the graphstore that match the pattern that is specified in the request. The find operation is supported with HTTP GET and POST methods. Header options are not applicable. The list below describes each of the supported parameters. These parameters can be included as query parameters in the URL or as parameters in the request body:

- **graph**: The named graph URI for the find pattern.
- **sub**: The subject of the find pattern.
- **pred**: The predicate of the find pattern.
- **lit**: The object of the find pattern if that object is a literal value.
- **uri**: The object URI of the find pattern if that object is a URI.
- **type**: If the object is a literal, this parameter can be used to specify the data type of the literal value.
- **lang**: If the object is a literal, this parameter can be used to specify the language of the literal value.

Results returned by the find operation are in Anzo JSON RDF serialization format. See Anzo JSON RDF Serialization below for details.

Find Examples

The following example find operation (using the GET HTTP method) finds all of the statements in the graphstore with predicate `http://w3.org/1999/02/22-rdf-syntax-ns#type` and an object URI of `http://cambridgesemantics.com/ontologies/2009/05/LinkedData#LinkedDataSet`. The parameters are specified as query parameters in the URL.

```
curl https://10.100.10.20:8443/anzoclient/find?pred=http://www.w3.org/1999/02/22-rdf-
syntax-
ns%23type&amp;uri=http://cambridgesemantics.com/ontologies/2009/05/LinkedData%23LinkedData
Set' \
  --user sysadmin:123
```

The example below finds the same statements but issues a POST call. The URL-encoded parameters are specified in the request body.

```
curl https://10.100.10.20:8443/anzoclient/find \
  --user sysadmin:123
  --data 'pred=http%3A%2F%2Fwww.w3.org%2F1999%2F02%2F22-rdf-syntax-
ns%23type&uri=http%3A%2F%2Fcambridgesemantics.com%2Fontologies%2F2009%2F05%2FLinkedData%23
LinkedDataSet'
```

Anzo JSON RDF Serialization

Anzo's JSON RDF serialization standard is straightforward but differs from the common public JSON RDF serialization standards. In Anzo JSON serialization format, a set of statements (quads) are represented as an array of JSON objects. Each JSON object (statement) is defined as a key/value pair, where the key specifies the component of the statement, i.e., the subject, predicate, object, or namedGraphUri. Depending on the component, properties such as the component's value and data type are specified in nested objects.

The following example array shows Anzo's JSON serialization. The list below the example describes the structure.

```
[
  {
    "subject" : {
        "objectType": "uri" ,
        "value" : "urn://test"
    },
    "predicate" : "urn://predicate",
    "object" : {
        "objectType": "uri" ,
        "value" : "urn://object"
    },
    "namedGraphUri" : "urn://ng"
    },
  {
    "subject" : {
        "objectType": "uri" ,
        "value" : "urn://test"
    },
    "predicate" : "urn://predicate2",
    "object" : {
        "objectType": "literal" ,
        "value" : "test literal",
        "dataType" : "http://www.w3.org/2001/XMLSchema#string"
    },
    "namedGraphUri" : "urn://ng"
  }
]
```

- **subject** is a JSON object with two properties:
  - **objectType**: The resource type of the subject value. This is either a "uri" or "bnode" (blank node).
  - **value**: The blank node value or a string literal that specifies the URI.
- **predicate** is a string literal that specifies the predicate URI.

- **object** is a JSON object with two required properties and two optional properties:
  - **objectType**: Required property that specifies whether the object is a "uri," "literal," or "bnode."
  - **value**: Required property that specifies the string representation of the object value.
  - **dataType**: Optional property for use if the objectType is "literal." This property describes the data type of the literal value. It is a string literal of the XSD data type URI. For example: "http://www.w3.org/2001/XMLSchema#string"
  - **language**:Optional property for use if the objectType is "literal." This property describes the language of the literal value.
- **namedGraphUri** is a string literal that specifies the named graph URI.

Related Topics

Analyzing Data with Hi-Res Analytics

Accessing Data with the Query Builder

Accessing Data from Data on Demand Endpoints

Accessing Data via the SPARQL Endpoint

Anzo Admin CLI

SPARQL Query Templates and Best Practices

SPARQL Query Templates and Best Practices

To provide guidance on developing performant SPARQL queries and avoiding unexpected results, this topic offers SPARQL best practices and query templates that you can use as a starting point for writing SPARQL queries in Anzo, such as in data layer steps, dashboard query lenses, and the Query Builder.

- SPARQL Query Templates
- SPARQL Best Practices

SPARQL Query Templates

This section provides templates that you can use as a starting point for writing SPARQL queries.

- Template for Basic Data Selection
- Template for Graph Traversal Data Selection
- Template for Data Aggregation
- Template for Applying a Filter to Selected Data
- Template for Creating or Deriving New Variables

Template for Basic Data Selection

The most fundamental use case for writing SPARQL queries is to select data from properties from a collection of instances. The following template and example query illustrate how to access a class in a model and return the

properties on that class using their URIs.

Abstracted Query Template – Replace the bold text to modify the query

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

# select the variables that are populated in the WHERE clause
SELECT ?var1 ?var2
WHERE {
  ?instanceOfClass a uriRoot:ClassName ;
      uriRoot:varName1 ?var1 ;
      # use a prefix to abbreviate a property URI as shown above
      # or use the full URI as shown below
      <http://example.com/rootOfUris#varName2> ?var2 .
}
```

Example Implemented Query – Get Sample ID and Anatomical Location for each Sample

```
PREFIX bm: <http://identifiers.csi.com/pharmakg/def/biomarker#>

SELECT ?sampleId ?anatomicalLocation
WHERE {
  ?sample a bm:Sample ;
      bm:sampleId ?sampleId ;
      <http://identifiers.csi.com/pharmakg/def/biomarker#fmi_anatomicalLocation>
?anatomicalLocation .
}
```

Template for Graph Traversal Data Selection

The graph model enables the flexibility to combine data from different classes. The following template illustrates how to traverse between classes in the data model and access data from properties on multiple classes.

Abstracted Query Template – Replace the bold text to modify the query

```
PREFIX uriRoot: <http://example.com/rootOfUris#>
# select the variables that are populated in the WHERE clause
SELECT ?var1 ?var2 ?varFromOtherClass
WHERE {
  ?instanceOfClass a uriRoot:ClassName ;
      uriRoot:varName1 ?var1 ;
      # use a prefix to abbreviate a property URI as shown above
      # or use the full URI as shown below
      <http://example.com/rootOfUris#varName2> ?var2 ;
      # getting data from other classes requires traversing per the model
      uriRoot:pointerToOtherClass ?instanceOfOtherClass .

  ?instanceOfOtherClass a uriRoot:OtherClassName ;
```

```
      uriRoot:varName3 ?varFromOtherClass .
}
```

Template for Data Aggregation

Grouping data selections around a central property yields a more complete representation or summary of the data available. The following template illustrates how to use one property to act as a pivot point for collecting all the data from another property.

Abstracted Query Template – Replace the bold text to modify the query

```
PREFIX pref: <http://example.com/rootOfUris#>

SELECT
# data can be aggregated to yield counts, concatenations of data, etc.
  ?instanceId GROUP_CONCAT(DISTINCT(?instanceDetail) as ?instanceDetails)
WHERE {
  # apply selection/filtering logic to narrow the aggregation
  # or get summaries of total data by applying only simple restrictions
  ?instance a pref:Class ;
      pref:instanceId ?instanceId ;
      pref:instanceDetail ?instanceDetail .
}
GROUP BY ?instanceId
# all non-aggregated variables must be grouped in GROUP BY
```

Template for Applying a Filter to Selected Data

Filtering the results for a query gives the ability to focus on specific aspects of the data. The following template illustrates how to restrict the total selected result set by including a filter on a variable.

Abstracted Query Template – Replace the bold text to modify the query

```
PREFIX pref1: <http://example.com/rootOfUris1#>
PREFIX pref2: <http://example.com/rootOfUris2#>

SELECT ?varFromClass1 ?varFromClass2 ?varFromClass3 ?filteredVar
WHERE {
  ?instance1 a pref1:Class1 ;
      pref1:varName1 ?varFromClass1 ;
      # the path on the model points from Class1 to Class2
      pref1:pointerToClass2 ?instance2 .

  ?instance2 a pref1:Class2 ;
      pref1:varName2 ?varFromClass2 .

  # models with different prefixes can still be joined
```

```
  ?instance3 a pref2:Class3 ;
      # the path on the model points from Class3 to Class2
      pref2:pointerToClass2 ?instance2 ;
      pref2:filteredVarName ?filteredVar .

  # filters use comparisons to scope the selected data
  # they can use existence checks or other boolean expressions as well
  FILTER(?filteredVar = 'COMPAREDDATA')
}
```

Tip For optimal query performance, replace FILTER clauses when possible. See Replace FILTER with VALUES or Triple Patterns when Possible below for more information.

Template for Creating or Deriving New Variables

Storing intermediate or derived data within a query enables a single query to answer more complex questions. The following template illustrates how to bind a derived value to a variable. That variable is then available for selection or further manipulation.

Abstracted Query Template – Replace the bold text to modify the query

```
PREFIX pref1: <http://example.com/rootOfUris1#>
PREFIX pref2: <http://example.com/rootOfUris2#>
PREFIX pref3: <http://example.com/rootOfUris3#>

SELECT ?var1 ?filterVar ?var2AndVar3
WHERE {
  ?instance1 a pref1:Class1 ;
      pref1:varName1 ?var1 .

  ?filterInstance a pref2:MedicalHistory ;
      pref2:filterVarName ?filterVar ;
      # multiple traversals between classes may be necessary to link appropriate data
      pref2:pointerToIntermediateClass ?intermediateInstance .

  ?intermediateInstance a pref2:IntermediateClass ;
      pref2:pointerToClass1 ?instance1 .

  ?instance2 a pref3:Class2 ;
      # forwards traversals tend to be more performant
      # it is still possible to identify a latter class and do a backwards traversal
      pref3:pointerToClass1 ?instance1 ;
      pref3:varName2 ?var2 .

  ?instance3 a pref3:Class3 ;
      pref3:pointerToClass2 ?instance2 ;
```

```
      pref3:varName3 ?var3 .

  # filters can be executed on various data types
  FILTER(?filterVar < "filterData"^^xsd:filterDataType)

  # binding allows population of new/derived variables
  BIND(CONCAT(?var2, "--", ?var3) as ?var2AndVar3)
}
```

SPARQL Best Practices

To ensure that your SPARQL queries perform well and do not overtax Anzo, Cambridge Semantics recommends that you follow these guidelines when writing and testing your queries:

- Limit Results when Developing and Testing Queries
- Replace FILTER with VALUES or Triple Patterns when Possible
- Beware of Cross-Product Joins
- Use Subqueries when Querying Large Amounts of Data

Limit Results when Developing and Testing Queries

The easiest way to reduce query execution time in some cases is to apply a LIMIT statement to limit the result set to a specific number of solutions. Limiting the number of results improves performance for cases where query results are calculated and returned in a streaming fashion. Limiting results is particularly useful when results need to be ordered so that the first group of results are the only ones of interest.

Example Solution – Get Sample ID and the Binding Density for the top 10 most dense Samples

```
PREFIX bm: <http://identifiers.csi.com/pharmakg/def/biomarker#>

SELECT ?sampleId ?bindingDensity
WHERE {
  ?sample a bm:Sample ;
      bm:sampleId ?sampleId ;
      bm:bindingDensity ?bindingDensity .
}
ORDER BY DESC(?bindingDensity)
LIMIT 10
```

Replace FILTER with VALUES or Triple Patterns when Possible

While a FILTER clause is useful for narrowing down selected data per a set of requirements, only use FILTER when the logic does not lend to other operations. In many cases, replacing FILTER with a VALUES clause or a well-organized set of triple patterns increases query performance. When processing a FILTER statement, all non-filtered

data must be retrieved before the FILTER can be applied. Using a VALUES clause or triple pattern, however, reduces the amount of data that is retrieved and processed after the retrieval.

Example – Inappropriate use of FILTER for value-driven SELECT

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?var1 ?var2
WHERE {
  ?instanceOfClass a uriRoot:ClassName ;
      uriRoot:varName1 ?var1 ;
      uriRoot:varName2 ?var2 ;
      uriRoot:filteredVar ?filteredVar .
  FILTER(?filteredVar = 'COMPAREDDATA1' || ?filteredVar = 'COMPAREDDATA2' || ?filteredVar
= 'COMPAREDDATA3')
  # filteredVar is first retrieved, then run through several comparisons
}
```

Solution – VALUES used to select data of certain values

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?var1 ?var2
WHERE {
  ?instanceOfClass a uriRoot:ClassName ;
      uriRoot:varName1 ?var1 ;
      uriRoot:varName2 ?var2 ;
      uriRoot:filteredVar ?valueVar .

  VALUES (?valueVar) {
      ('COMPAREDDATA1')
      ('COMPAREDDATA2')
      ('COMPAREDDATA3')
  }
  # selection is performed once for each entry in the VALUES clause,
  # retrieving no more data than necessary
}
```

Example – Inappropriate use of FILTER for value-driven SELECT

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?var1 ?filteredVar
WHERE {
  ?instanceOfClass a uriRoot:ClassName ;
      uriRoot:varName1 ?var1 ;
      uriRoot:varName2 ?var2 ;
```

```
      uriRoot:filteredVar ?filteredVar .
  FILTER(?filteredVar = 'COMPAREDDATA1')
  # filteredVar is first retrieved, then compared
}
```

Solution – Triple literal used to select data of a certain value

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?var1 ?filteredVar
WHERE {
  ?instanceOfClass a uriRoot:ClassName ;
      uriRoot:varName1 ?var1 ;
      uriRoot:filteredVar 'COMPAREDDATA' .
      # data is only retrieved if filteredVar matches desired compared data upon initial
retrieval
}
```

Beware of Cross-Product Joins

When trying to gather data from multiple classes at once, it is possible to accidentally create a cross-product join, a selection that combines the selected data in a hyper-linear way rather than simply assembling the data and returning an unprocessed set.

Example – Accidental cross-product query

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?var1 ?var2
WHERE {
  ?instanceOfClass1 a uriRoot:ClassName1 ;
      uriRoot:varName1 ?var1 .
  ?instanceOfClass2 a uriRoot:ClassName2 ;
      uriRoot:varName2 ?var2 .
}
```

In the above example, the goal may have been to retrieve IDs from all instances of ClassName1 and all instances of ClassName2, for example, all of the Participants and all of the Subjects. However, the result of the query would be every combination of Participant and Subject. If there are 10 Participants and 5 Subjects, there would be 50 results rather than 15. In large data sets, this severely affects performance and puts the system under unnecessary strain.

There are two straightforward ways to separate or parameterize data to write a more performant query.

Solution 1 – Use UNION to replace the cross-product

```
PREFIX uriRoot: <http://example.com/rootOfUris#>
```

```
SELECT ?commonVar
WHERE {
  {
    ?instanceOfClass1 a uriRoot:ClassName1 ;
        uriRoot:varName1 ?var1 .
    BIND(?var1 as ?commonVar)
  }
  UNION

{
    ?instanceOfClass2 a uriRoot:ClassName2 ;
        uriRoot:varName2 ?var2 .
    BIND(?var2 as ?commonVar)
  }
  # this creates an ephemeral graph that is a union of two graphs
  # in each of the two graphs, the desired data is saved under the same name
}
```

Solution 2 – Use VALUES to replace the cross-product

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?commonVar
WHERE {
  ?instanceOfClass a ?classURI ;
      ?propertyURI ?commonVar .

  VALUES (?classURI ?propertyURI) {
      (uriRoot:ClassName1 uriRoot:varName1)
      (uriRoot:ClassName2 uriRoot:varName2)
  }
}
```

Use Subqueries when Querying Large Amounts of Data

When analyzing data, there may be a need to aggregate data and then perform a selection or derivation on the resulting aggregate. In this case, it is advisable to use one or more subselects or subqueries, where a SELECT query is included inside the WHERE clause and the remainder of the WHERE clause operates on the results of that SELECT as though that data were immediately available in the graph.

Example Solution – Aggregate a variable and then process the aggregation

```
PREFIX uriRoot: <http://example.com/rootOfUris#>

SELECT ?var1 ?var2Aggregation
WHERE {
```

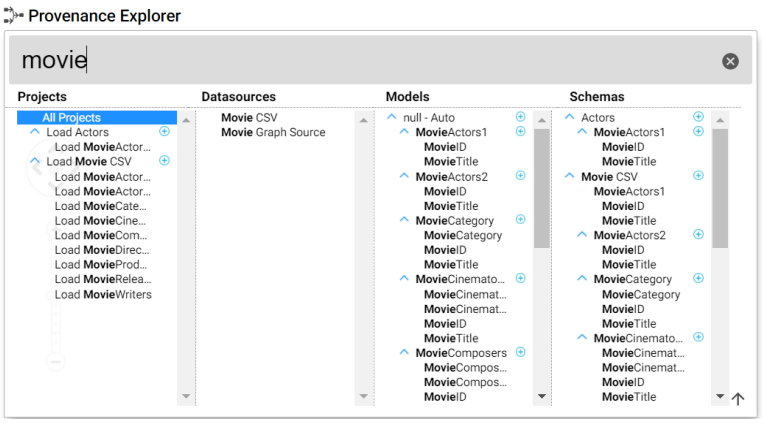
```
  {
    SELECT ?var1 (GROUP_CONCAT(?var2) as ?var2Aggregation)
    WHERE {
        ?instanceOfClass1 a uriRoot:ClassName1 ;
            uriRoot:varName1 ?var1 .
        ?instanceOfClass2 a uriRoot:ClassName2 ;
            uriRoot:varName2 ?var2 .
    }
    GROUP BY ?var1
  }
  # var1 and var2Aggregation are now available for the usual processing
  # while var2 is no longer available as it only existed within the subselect

  FILTER(regex(?var2Aggregation, 'DESIREDVAR2VAL'))
  # FILTER is used for illustrative purposes, but any processing would work
}
```

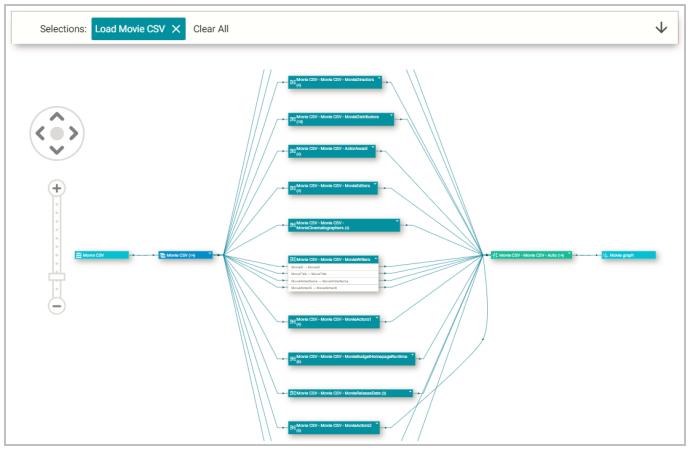
Exploring Data Provenance

The Anzo provenance explorer enables you to trace the lineage of your structured data. You can search for data entities and view associated projects, data sources, models, and schemas. This topic provides information about using the provenance explorer.

1. To open the provenance explorer, log in to the Anzo console and click **Provenance**.
2. Click in the gray **Search** box and type a value to search for. Anzo populates the table in the search drop-down box with any projects, data sources, models, and schemas that include the search value. For example, searching for "movie" displays the pipelines, sources, classes, properties, and schemas that include "movie" and its relationships. Click the plus icon (⊕) to view any related elements.

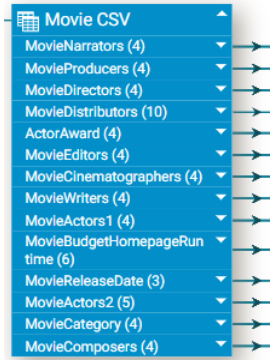


3. To view the provenance for any of the pipeline, data source, model, or schema objects, click the object to highlight it, and then click outside of the search drop-down. Anzo displays the provenance graph for the item that you selected. For example, the image below shows the provenance for the sample Movie data set:

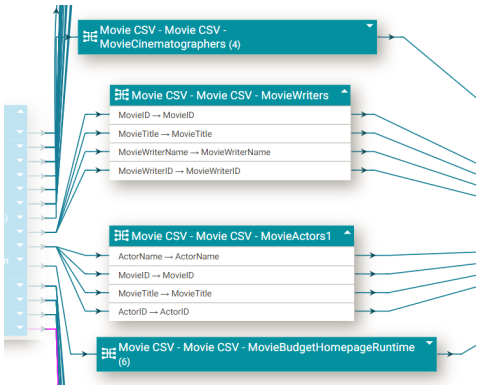


Viewing the provenance graph from left to right, the first object shows the data source, **Movie CSV** in the example:

The second object shows the data that came from the data source, 14 Movie CSV files as shown in the example. To expand the rectangle to view the files and additional details such as the schema information or list of properties from each file, click the triangle icon on the top right of the rectangle:

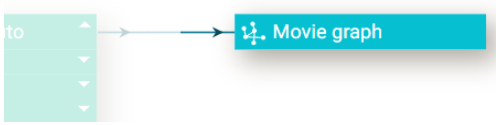


The third group of objects in the graph show the mappings that Anzo generated from the source schemas:



The fourth object shows the model or ontology that Anzo generated:

And the final object shows the graph data source where Anzo generated the load files after ingestion.



4. To view the provenance graph for a different object, click in the search box and then choose an item from the drop-down. Click out of the search box to view the graph.

   The provenance explorer includes navigation tools that you can use to zoom in or out of the graph or move the graph on the screen.

   To view different regions of the graph, click the > characters to move the graph vertically or horizontally. You can also click and drag the graph on the screen. Click the plus and minus icons to zoom in and out. You can click and drag the individual tables to rearrange them in the graph or collapse and expand tables or columns using the triangle icons to the right of the table or column name.

Versioning and Migrating Entities and Metadata

The topics in this section provide information about managing backup versions of entities and migrating entities by exporting and importing versions.

Creating and Restoring Versions

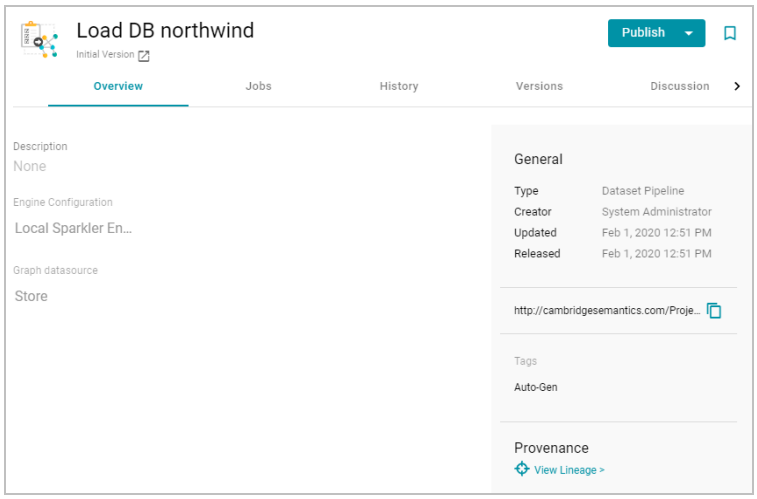
Anzo provides a versioning feature that enables users to quickly back up and restore versions of the entities that make up a solution. Before making changes to data sources, schemas, mappings, pipelines, or data models, users can take a snapshot of the current version of that artifact. When a backup is created, Anzo automatically creates a version of each entity that is related to that artifact. For example, backing up a version of a pipeline automatically backs up the same version of any related data models, mappings, schemas, and so on. In addition, Anzo backs up the metadata graphs for all of the entities. The metadata graphs store information such as the creator and creation date and the permissions or access control list (ACL) information. Changed artifacts can be reverted at any time to any of the saved versions. If an artifact is restored to a previous version, Anzo automatically saves a version of the current state of the artifact and its related entities and metadata.

This topic provides instructions for backing up and restoring versions of artifacts.

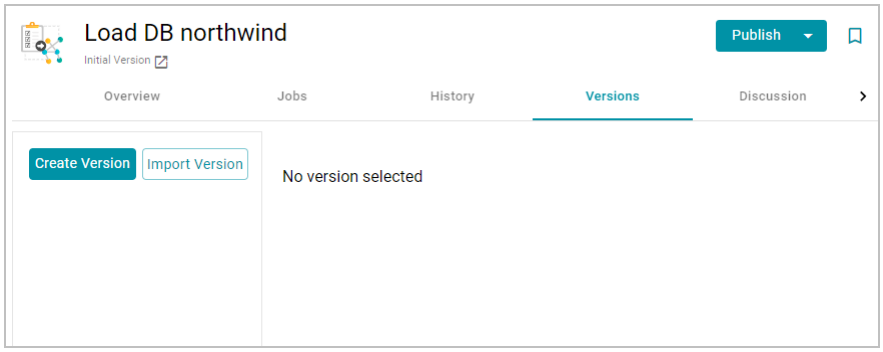
Creating a Backup Version

Follow the instructions below to save a snapshot of a data source, schema, mapping, pipeline, or data model.

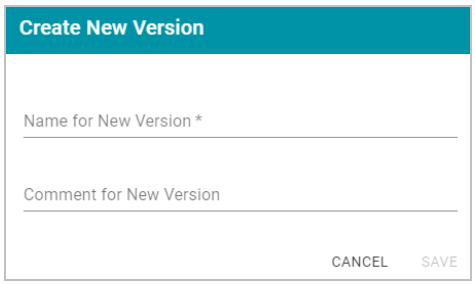
1. In the Anzo console, go to the detail view or overview for the component that you want to back up. For data models, add the model to the working set and then open it in the model editor. For example, the image below opens a pipeline:



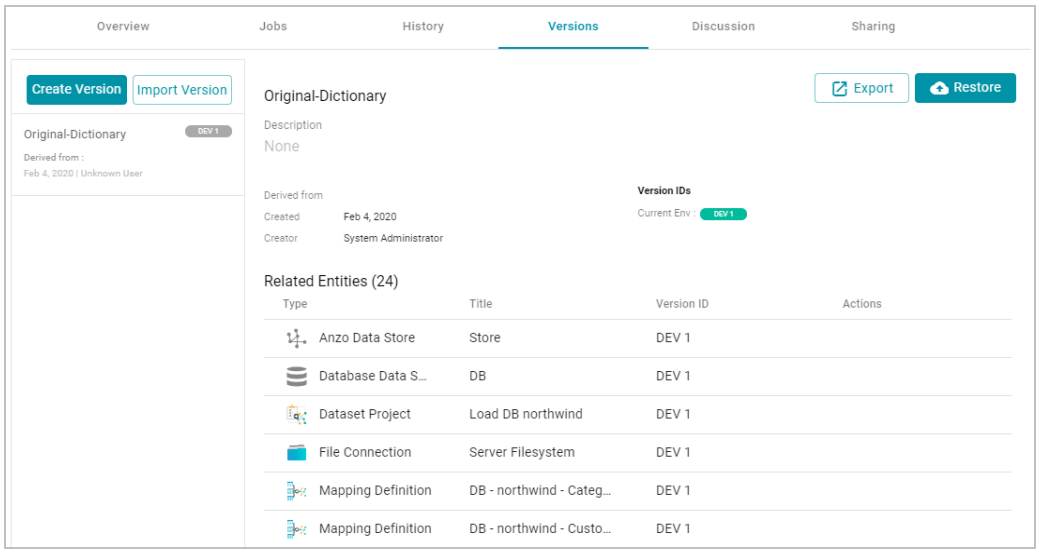
2. Click the **Versions** tab. Anzo displays the Versions screen. For example:



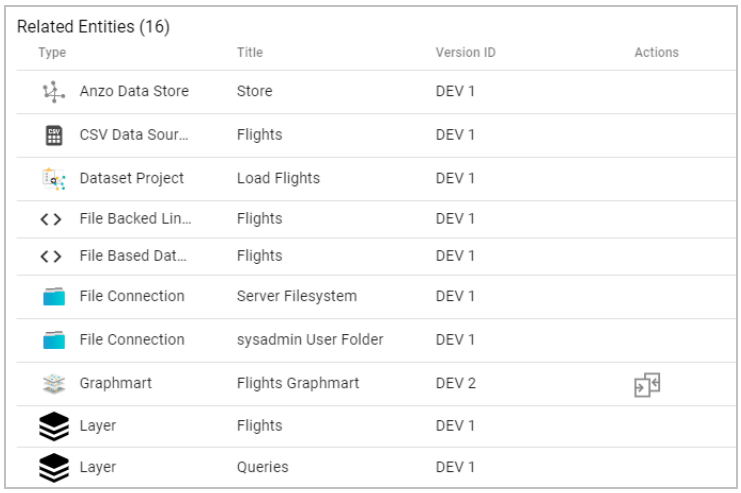
3. Click **Create Version**. Anzo displays the Create New Version screen.



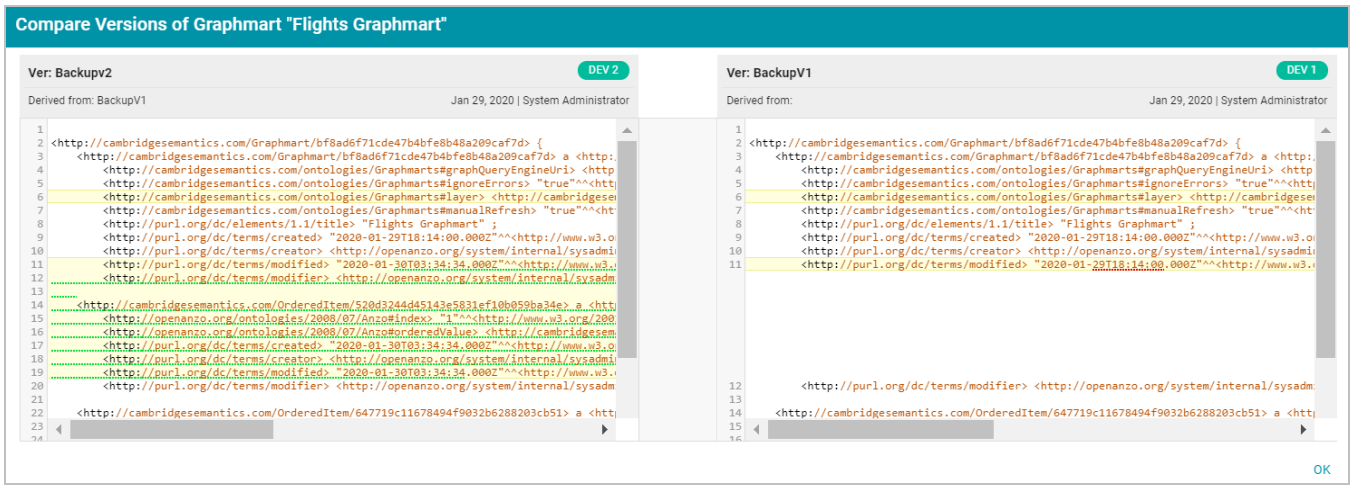
4. In the **Name for New Version** field, type a name for the backup version. Then type details about the version in the optional **Comment for New Version** field.
5. Click **Save**. Anzo takes a snapshot of the component as well as its related entities and adds the version to the list on the left side of the screen. Depending on the size and number of related entities, the backup operation can take a few minutes to complete. For example:



6. If necessary, select the new version in the list to view details on the right side of the screen. The screen displays details such as the version creator and created date and lists each of the related entities that were also backed up. In the list of related entities, the Actions column displays a compare icon next to each entity that has changed since the previous version. For example, in the image below, the compare icon in the Graphmart row indicates that this version of the graphmart includes changes that were not in the previous version:



Clicking the icon in the Actions column opens the Compare Versions dialog box, which shows a side-by-side comparison of the TriG files for the two versions. For example:

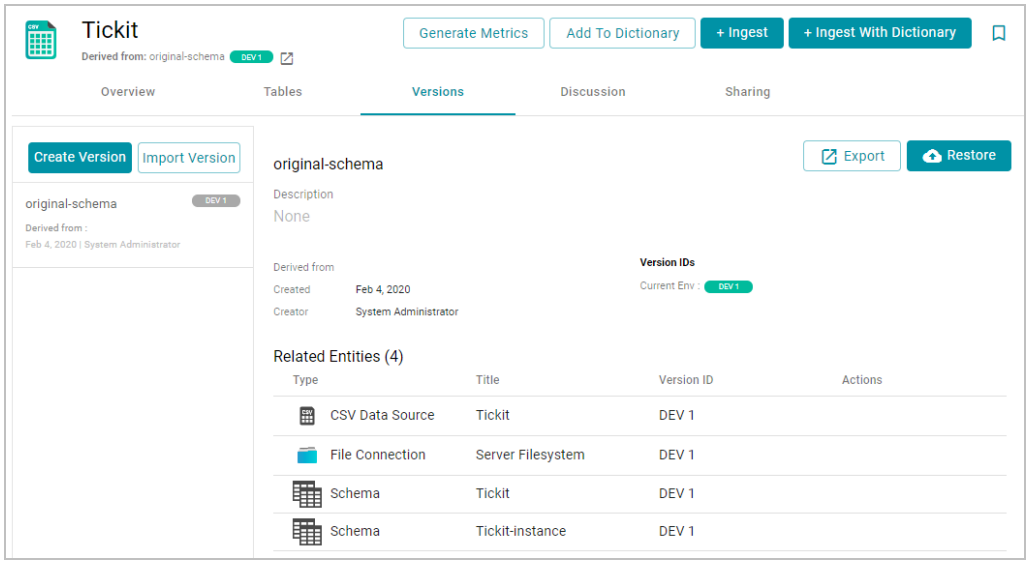


Users can now make changes to the current version of the backed up components, and the new changes can be reverted to a backup version at any time.

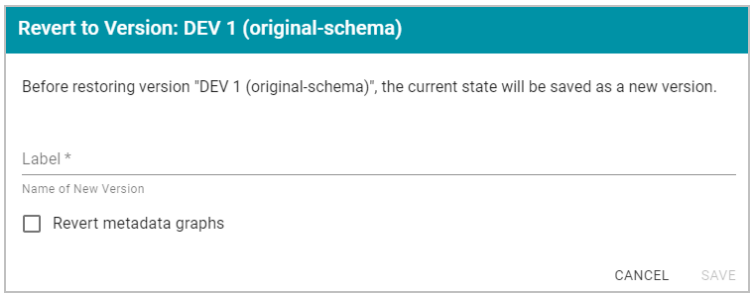
Restoring a Backup Version

Follow the instructions below to restore a component and its related entities to a previous backup version

1. In the Anzo console, go to Versions tab for the component that you want to restore. For data models, add the model to the working set, open it in the model editor, and then click the Versions tab.
2. On the Versions screen, select the backup version that you want to restore. For example:



3. Click the **Restore** button to restore the component to the version that you selected. Since Anzo automatically creates a snapshot of the current version before you restore a component, Anzo displays the Revert to Version dialog box so that you can specify a label for the new version.



4. In the Restore to Version dialog box, type a name for the new version in the **Label** field.
5. Specify whether you want to revert to this version's metadata graphs for this component and its related entities:
    - If you want the restored version to use the metadata, such as access control list information and last created date, that was saved at the time of the backup, select the **Revert metadata graphs** checkbox. Anzo will revert the metadata to the saved version.
    - If changes were made to the metadata for the current version of the component and you want to preserve those changes, such as if the permissions were modified to further restrict or allow access, leave the **Revert metadata graphs** checkbox blank. Anzo will preserve the current metadata graphs instead of reverting the metadata to the saved version.
6. Click **Save**. Anzo saves the current version and restores the current files to the backup version. The new version is added to the list of available backups.

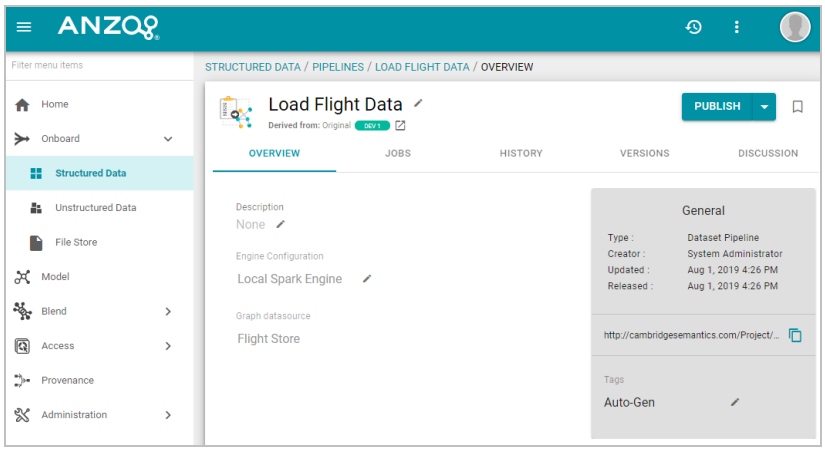
Exporting Versions

This topic provides instructions for exporting artifacts, such as data source definitions, pipelines, mappings, and graphmarts, and their related entities. Users can export the current version of an entity or any backup version. Follow the appropriate instructions below to export artifacts:

- Export the Current Version of an Artifact
- Export a Backup Version of an Artifact
- Exported ZIP File Contents

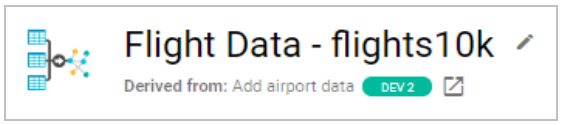
Export the Current Version of an Artifact

1. Log in to the Anzo console and go to the detail view or overview for the component that you want to export. For example, the image below shows the overview for a pipeline:

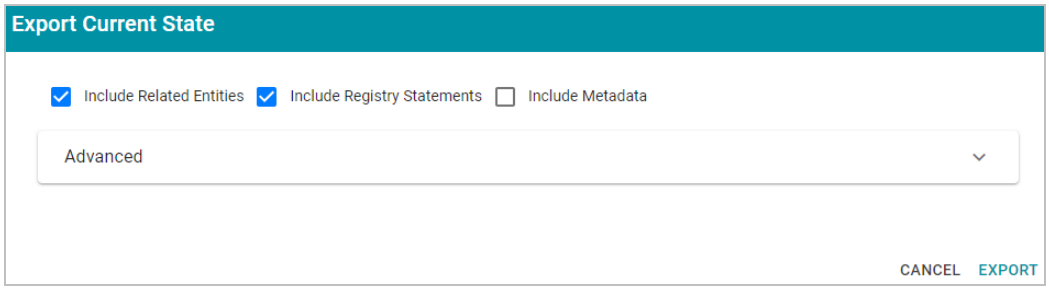


2. Click the Export icon ( ) under the component name. For example:

The image below shows an example of the Export icon for an entity that has a backup version. Clicking exports the current version of the entity, not the backup version that is listed.



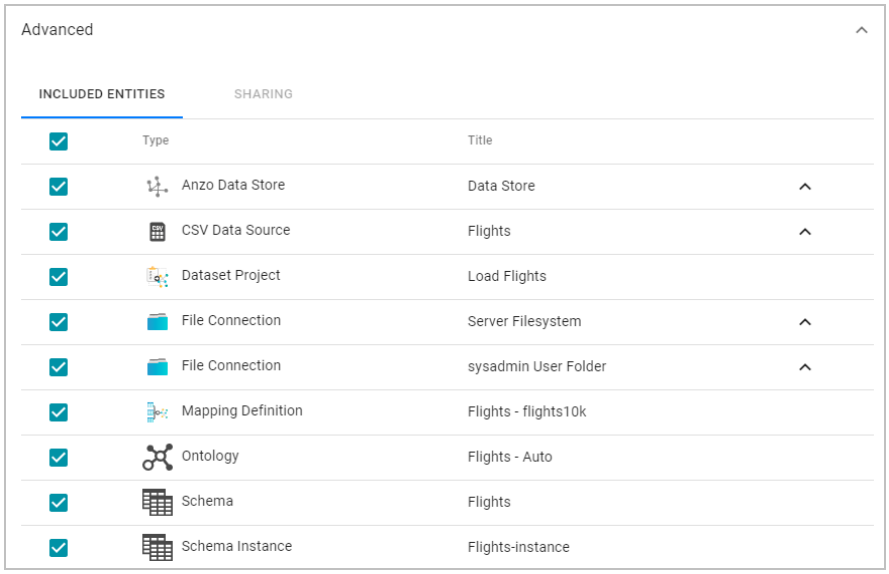
Anzo displays the Export Current State dialog box. For example:



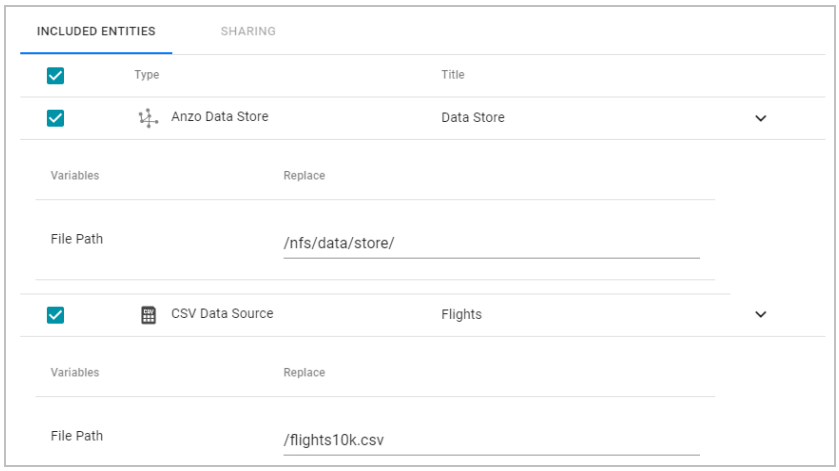
3. On the Export dialog box, configure the export options as needed. The list below describes each option.
   - **Include Related Entities**: Indicates whether to export the artifact's related entities. Since most artifacts have dependencies with other components, Cambridge Semantics recommends that you enable **Include Related Entities** (selected by default) and export all related entities.
   - **Include Registry Statements**: Indicates whether to export the registry statements for the artifact and its related entities, such as the named graph information for the data source, schema, model, and instance data.
   - **Include Metadata**: Indicates whether to export the metadata graph for the artifact and its related entities, such as the access control list (ACL) information and last modified date. If you exclude the

metadata, the artifacts in this export will follow the ACL configuration on the destination server when they are imported. Select Include Metadata if you want to migrate the existing ACLs to the destination server. Enabling this setting also gives you the option to change the ACL configuration for the exported entities. To change the ACL configuration, expand the **Advanced** option and click the **Sharing** tab.

- **Advanced**: If you want to change permissions or replace the values for certain properties in the exported version of an entity, such as the user name and password for a database data source, the base folder location for a file connection, or the file path for an Anzo data store, expand the **Advanced** option to view the Included Entities list. For example:

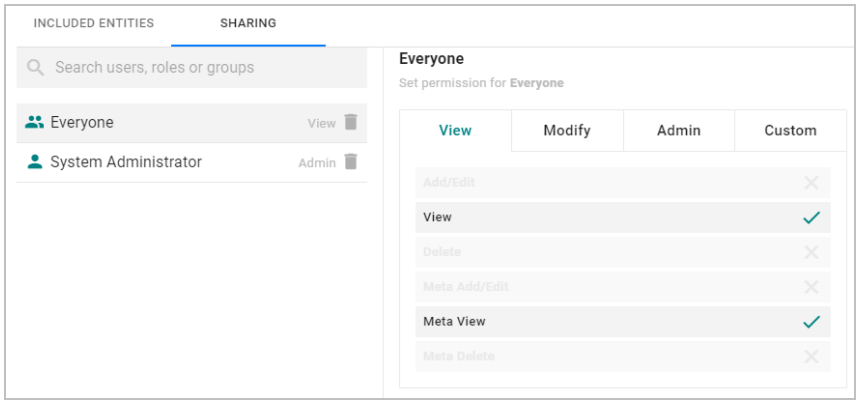


The entities with replaceable values are expandable. Click the ^ character to the right of an entity name to expand the options and view the editable properties. For example:



Replace any of the existing values with the new values that you want to define for the exported version of the entity. For information about configuring properties so that their values are replaceable on export, see Making Values Replaceable on Export.

If you specified **Include Metadata** and want modify ACL settings for the exported entities, click the **Sharing** tab and edit or add permissions for users, roles, and groups.

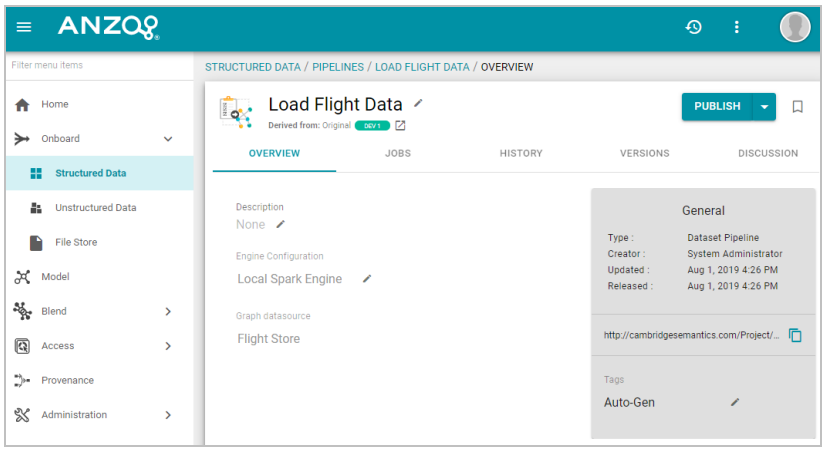


4. Click **Export** to export the artifacts. Anzo packages the files into a .zip file and downloads it to your computer. You do not need to extract the files in order to import the artifacts to another Anzo server. See Exported ZIP File Contents below for a description of the files that are included in the .zip file.

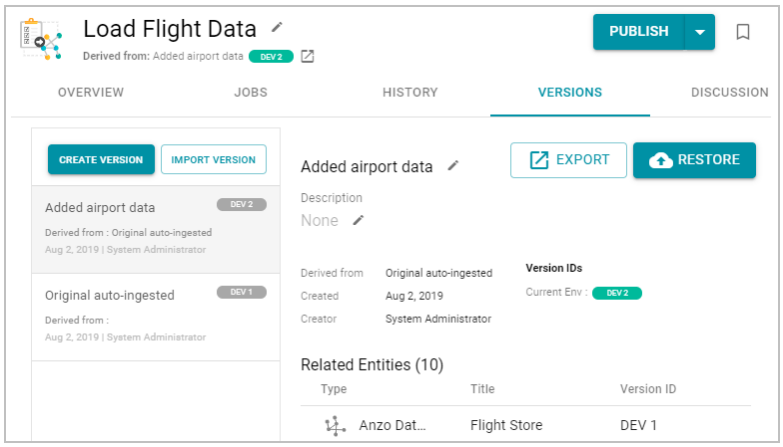
Export a Backup Version of an Artifact

For instructions on creating a backup version, see Creating and Restoring Versions.

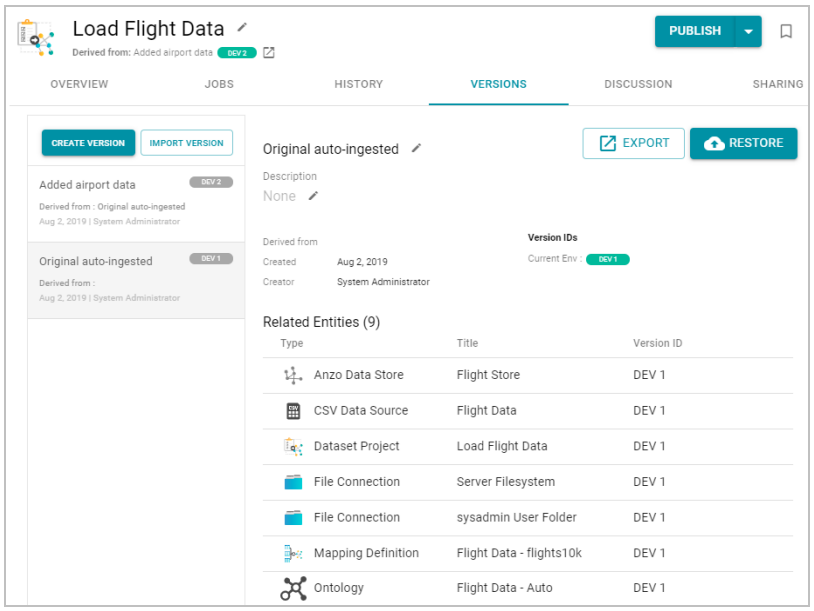
1. Log in to the Anzo console and go to the detail view or overview for the component that you want to export. For example, the image below shows the overview for a pipeline:



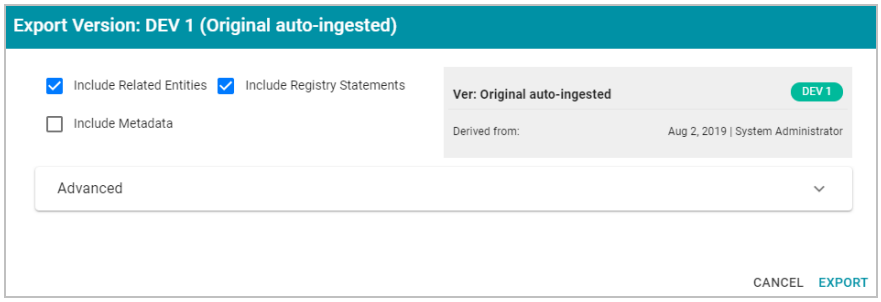
2. Click the **Versions** tab. Anzo displays the Versions screen, which lists the backups that exist for the component. For example:



3. On the left side of the screen, select the version that you want to export. For example:

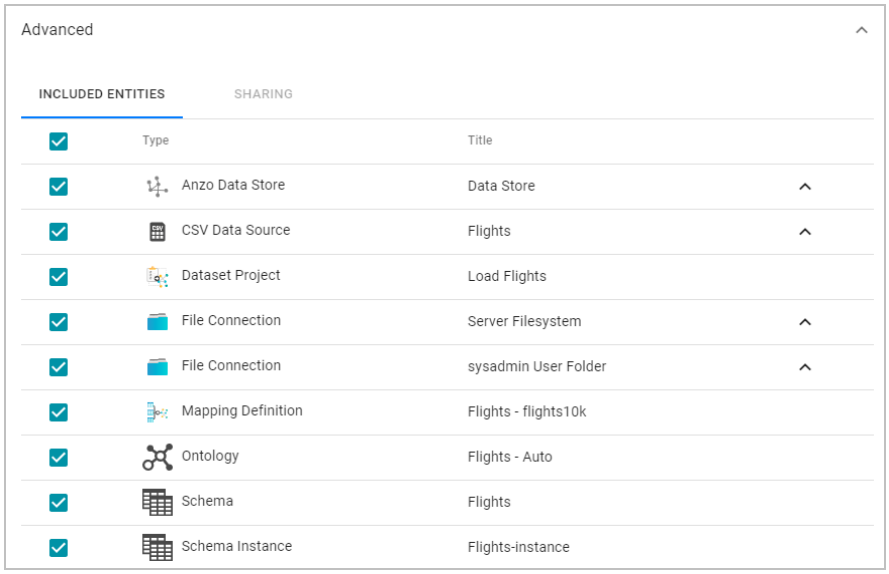


4. Click the **Export** button. Anzo opens the Export Version dialog box. For example:

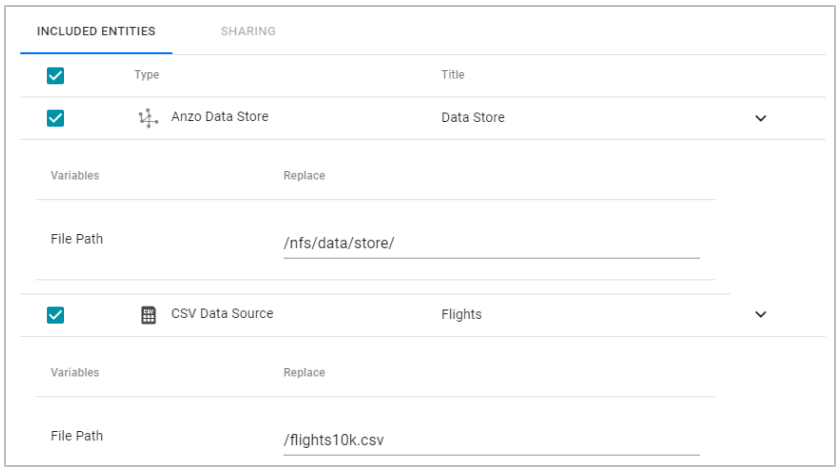


5. On the Export dialog box, configure the export options as needed. The list below describes each option.
   - **Include Related Entities**: Indicates whether to export the artifact's related entities. Since most artifacts have dependencies with other components, Cambridge Semantics recommends that you enable **Include Related Entities** (selected by default) and export all related entities.

- **Include Registry Statements**: Indicates whether to export the registry statements for the artifact and its related entities, such as the named graph information for the data source, schema, model, and instance data.
- **Include Metadata**: Indicates whether to export the metadata graph for the artifact and its related entities, such as the access control list (ACL) information and last modified date. If you exclude the metadata, the artifacts in this export will follow the ACL configuration on the destination server when they are imported. Select Include Metadata if you want to migrate the existing ACLs to the destination server. Enabling this setting also gives you the option to change the ACL configuration for the exported entities. To change the ACL configuration, expand the **Advanced** option and click the **Sharing** tab.
- **Advanced**: If you want to change permissions or replace the values for certain properties in the exported version of an entity, such as the user name and password for a database data source, the base folder location for a file connection, or the file path for an Anzo data store, expand the **Advanced** option to view the Included Entities list. For example:

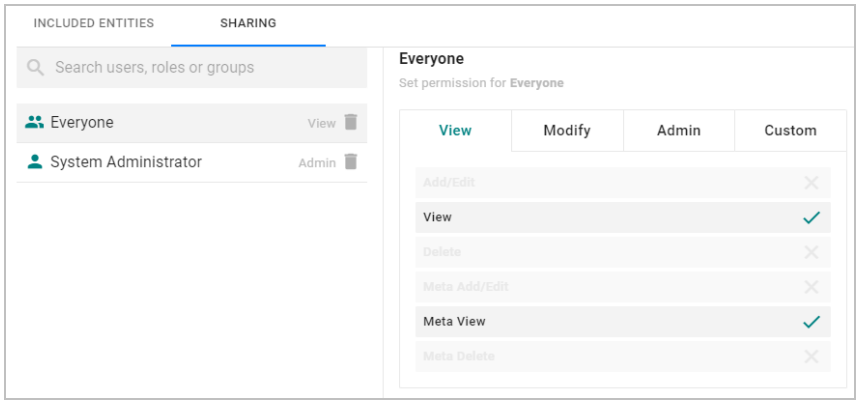


The entities with replaceable values are expandable. Click the ^ character to the right of an entity name to expand the options and view the editable properties. For example:



Replace any of the existing values with the new values that you want to define for the exported version of the entity. For information about configuring properties so that their values are replaceable on export, see Making Values Replaceable on Export.

If you specified **Include Metadata** and want modify ACL settings for the exported entities, click the **Sharing** tab and edit or add permissions for users, roles, and groups.



6. Click **Export** to export the artifacts. Anzo packages the files into a .zip file and downloads it to your computer. You do not need to extract the files in order to import the artifacts to another Anzo server. See Exported ZIP File Contents below for a description of the files that are included in the .zip file.

Exported ZIP File Contents

Depending on the options configured for the export, the .zip file contains one or more of the following files:

- ***artifact_name*_graph.trig** contains the model, data source, schema, and mapping definitions.
- ***artifact_name*_metadata.trig** contains metadata statements such as the access control list and last modified date for the exported entities.
- ***artifact_name*_registry.trig** contains registry statements such as the named graph information for the data source, schema, model, and instance data.

- ***artifact_name*_version.trig** contains statements about the backup version that the entities were exported from.

Making Values Replaceable on Export

When exporting artifacts, Anzo enables users to replace the existing values for properties like the user name and password for database data sources, the base folder location for file connections, and the file path for graph data sources. This topic provides instructions for configuring additional properties so that their values can be modified in the exported version of an entity.

To configure a property so that its value is replaceable on export, add the following statement to the **http://cambridgesemantics.com/annotations/replaceStatements** graph:

```
class_URI http://cambridgesemantics.com/ontologies/2018/06/Export#replaceStatement
property_URI
```

Where *class_URI* is the URI for the class that defines the property whose value should be replaceable. And *property_URI* is the URI of the property.

Note The specified property must be a Datatype property that contains a literal value.

For reference, the following TriG file is used to define the default properties with replaceable values:

```
@prefix ds:  <http://cambridgesemantics.com/ontologies/DataSources#> .
@prefix exp: <http://cambridgesemantics.com/ontologies/2018/06/Export#> .
@prefix ann: <http://cambridgesemantics.com/annotations/> .

#Mode:ADD

ann:replaceStatements {
  ds:PathConnection exp:replaceStatement ds:filePath .
  ds:FileConnection exp:replaceStatement ds:fileConnectionBaseFolder .
  ds:DbDataSource exp:replaceStatement ds:dbUser , ds:dbDatabase, ds:dbPassword .
}
```

Related Topics

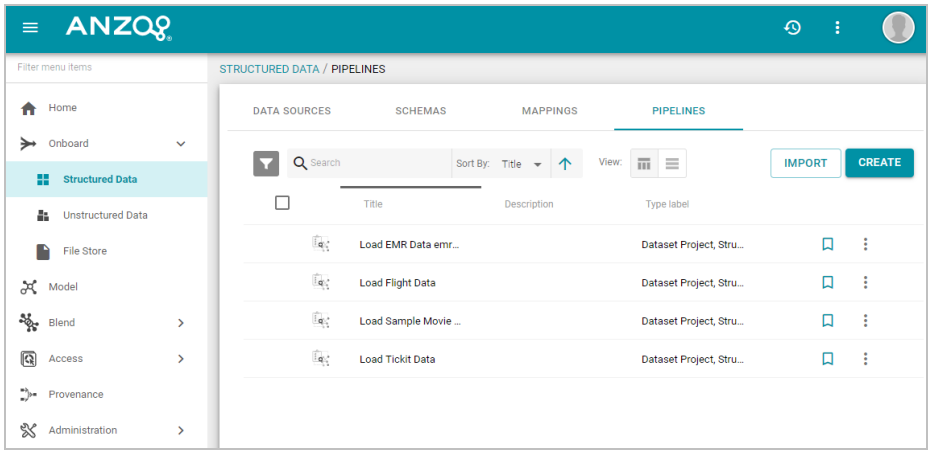
Exporting Versions

Importing Exported Versions

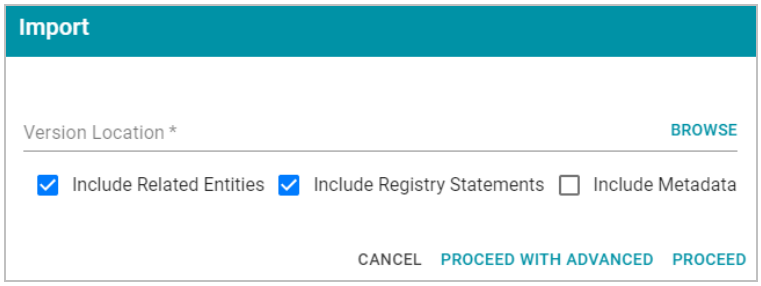
This topic provides instructions for importing the exported versions of entities, such as data source definitions, pipelines, mappings, and their related entities. For instructions on exporting entities, see Exporting Versions.

Note If you want to import a model that was created outside of Anzo or was downloaded from Anzo as described in Downloading a Model, see Uploading a Model to Anzo for instructions on uploading the model. If you want to import a version of a model that was exported from Anzo as described in Exporting Versions, follow the instructions in this topic.

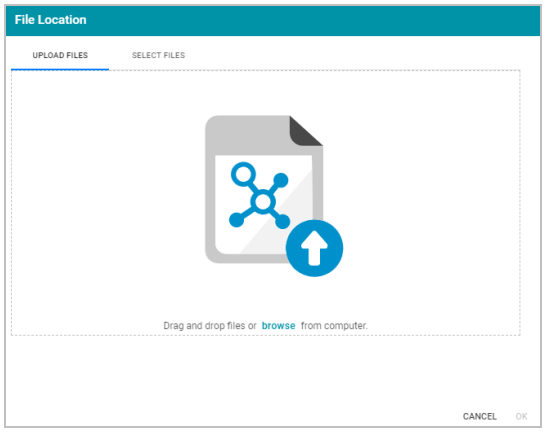
1. In the Anzo console, go to the resource selection screen for the artifact that you want to import. For example, the image below shows the Pipelines screen:



2. Click the **Import** button on the top of the screen. Anzo opens the Import dialog box.

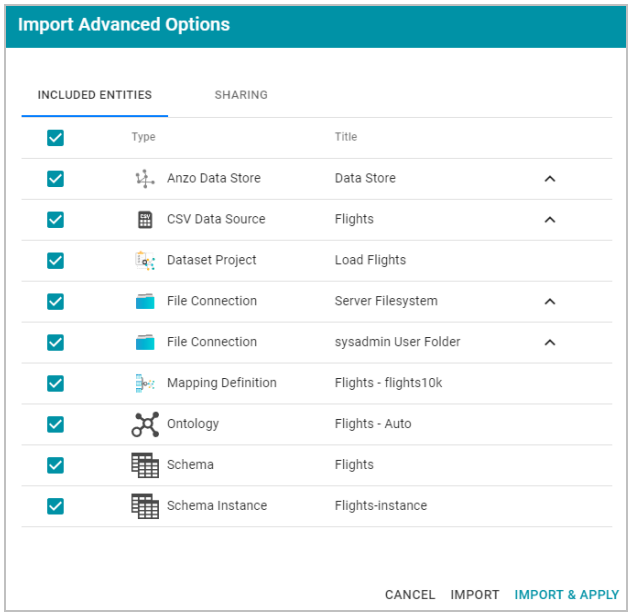


3. On the Import screen, click the **Version Location** field to open the File Location dialog box and find the file to import.



Click the **Upload Files** tab to browse for the file on your computer, or click the **Select Files** tab to browse for the file on a file store. Find and select the .zip file and then click **OK**.

4. Enable or disable the **Include Related Entities**, **Include Registry Statements**, and **Include Metadata** options as needed, depending on the data that the import file contains.
    - **Include Related Entities**: Indicates whether to import the artifact's related entities. Since most artifacts have dependencies with other components, and these dependencies were likely exported, Cambridge Semantics recommends that you enable **Include Related Entities** (selected by default).
    - **Include Registry Statements**: Indicates whether to import the registry statements for the artifact and its related entities, such as the named graph information for the data source, schema, model, and instance data.
    - **Include Metadata**: Indicates whether to import the metadata graph for the artifact and its related entities, such as the access control list (ACL) information and last modified date. If you select **Include Metadata**, you have the option to edit the ACL configuration before importing the artifact.
5. Choose one of the following options to proceed with the import:
    - If you want to import the files as alternate versions of entities and not as the current, active version, and you do not want to replace any property values or change permissions on the imported entities, click **Proceed**. Anzo imports the data and the imported files become available as versions on the relevant Version screens for the imported entities.
    - If you want to import these files as the current active version, and/or you want to change certain property values or modify the ACL configuration, click **Proceed With Advanced**. Anzo opens the Import Advanced Options dialog box. For example:



Click the ^ character to the right of an entity name to expand the options and view the editable properties. Replace any of the existing values with the new values that you want to define for the imported version of the entity. If you specified **Include Metadata** and want modify ACL settings for the import, click the **Sharing** tab and edit or add permissions for users, roles, and groups.

When you are ready to import the entities, choose one of the following options:

- If you want to import the files as alternate versions and not as the current, active version, click **Import**. Anzo imports the files and the entities become available as versions on the relevant Version screens.
- If you want to import the files so that they become the current, active versions of the entities, click **Import & Apply**. Anzo creates a backup version of the existing active versions and then imports the files as the current, active version.

Related Topics

Exporting Versions

Making Values Replaceable on Export

Graph Data Storage Architecture

This topic describes the way onboarded graph data is shared between and stored in the Anzo and AnzoGraph graph stores.

The onboarding process generates different types of graph data artifacts. Storage of the artifacts differs based on the type of data that is being stored and the purpose of the data. The list below describes the artifacts and storage methods:

- The metadata, such as data models, data source configuration details, catalog entries, registries, mappings and access control definitions, are stored in Anzo's embedded graph store. The Anzo graph store is a trans-action-oriented store that is built for processing many updates to small amounts of data. Data is persisted to disk in a journal, also known as a volume. For more information, see Journal or Volume.
- The instance data and copies of the data models are written to a File-Based Linked Data Set (FLDS) on the shared file store. Each FLDS is represented as a data set in Anzo's metadata catalog.
- When a data set from the catalog is added to a graphmart and the graphmart is activated, Anzo loads the data from the FLDS into the AnzoGraph graph store. AnzoGraph is an in-memory graph OLAP store that is built for processing complex analytics on large amounts of data. Once the instance data is in memory, the rest of the graphmart's data layer steps are executed by AnzoGraph (known as the ELT process). Each data layer becomes a graph in AnzoGraph, and each layer graph includes the instance data created by that layer as well as the related data models.
- Anzo system ontologies and metadata remain in Anzo's graph store, the system data source, and are not loaded to AnzoGraph unless the system data is added to a graphmart and the graphmart is activated.

As an example, an Anzo instance has two active graphmarts. Each graphmart has two data layers, one for loading data sets into memory and another for creating views and running ELT queries. When the following query is run against AnzoGraph to return a list of all distinct graphs, the results show that there are five graphs:

```
SELECT DISTINCT ?graph
WHERE {
  GRAPH ?graph {
    ?s ?p ?o
  }
}
```

```
graph
-------------------------------------------------------------------------------
http://cambridgesemantics.com/Layer/546fb89ac6d245f8bea2777a52077bc9
http://cambridgesemantics.com/Layer/1162fb0d0b724a18b4133c10d69f16b7
http://cambridgesemantics.com/Layer/12c7eedddff9449ab4b133373b56e65c
http://cambridgesemantics.com/Layer/b69bb3295ba3434e846b1ed372039416
```

```
http://cambridgesemantics.com/GqeDatasource/guid_10492203b5aa4a54f217ababb3dc6dee
5 rows
```

The first four graphs are the data layers for the two graphmarts. The graph URIs match the data layer URIs in Anzo.

How do I find the graph URI for a layer in a graphmart?

Note AnzoGraph does not have a "graphmart" construct, and graphmart URIs do not exist in the database. Though a graphmart acts as a container for data layers and its metadata can be queried in Anzo's embedded graph store, it does not include instance data that is needed by AnzoGraph.

The last graph in the results above is the AnzoGraph data source graph. This graph contains one triple that records a timestamp for the last time the data source was updated. If Anzo loses the connection to AnzoGraph, it checks this timestamp when it reconnects. The last updated time is used to determine whether the Anzo and AnzoGraph graph stores are in sync or if the graphmarts need to be reloaded to AnzoGraph.

Typically organizations manage all data with Anzo, i.e., data is onboarded to Anzo through pipelines or it is dynamically blended into data layers from remote endpoints. Users then use Anzo to load data to AnzoGraph for analytics. When data is loaded to AnzoGraph through Anzo, Anzo automatically manages the reloading of data if AnzoGraph is restarted. Though users can load data and create named graphs directly in AnzoGraph, AnzoGraph is not configured by default to persist the data in memory to disk. Graphs that do not originate in Anzo must be reloaded manually any time AnzoGraph is restarted. If you want to work with named graphs directly in AnzoGraph, consider configuring AnzoGraph to save data to disk. For more information, see Enabling AnzoGraph Persistence.

Related Topics

Onboarding Structured Data

Onboarding Unstructured Data

Administration Guide

The Administration Guide provides guidance for Anzo administrators. The topics in this section provide information about managing the initial set up and administration of Anzo components.

Anzo Server Administration

The topics in this section provide information about managing the Anzo server configuration.

Starting and Stopping Anzo

If Anzo is run via a systemd service, as described in Configure an Anzo Service Manager, use systemctl to start and stop Anzo. To start Anzo, run the following command:

```
systemctl start service_name
```

For example: `systemctl start anzo-server`

To stop Anzo, run the following command:

```
systemctl stop service_name
```

For example: `systemctl stop anzo-server`

To start Anzo using the AnzoServer utility, run the following command:

```
/install_path/Anzo/Server/AnzoServer start
```

To stop Anzo, run the following command:

```
/install_path/Anzo/Server/AnzoServer stop
```

You can also start and stop Anzo from the symbolic links if they were created for your installation. For example, `/etc/init.d/AnzoServer start` or `/etc/init.d/AnzoServer stop`.

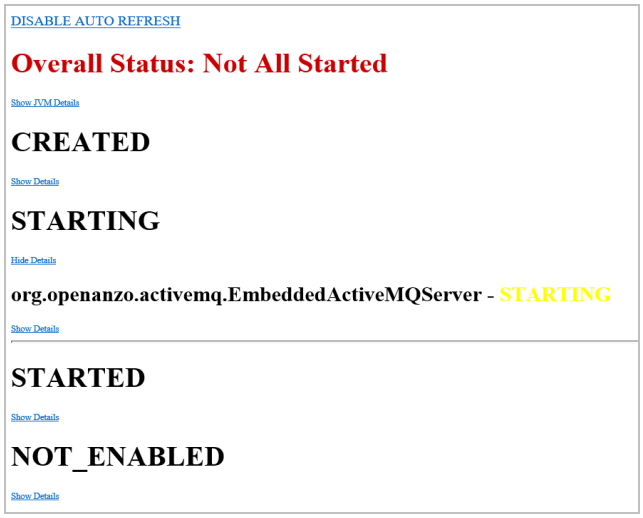
Monitoring Startup Status

It can take a few minutes for Anzo to complete the startup process. You can monitor the status by viewing the Anzo Status page. To see the Status page, go to the following URL in your browser:

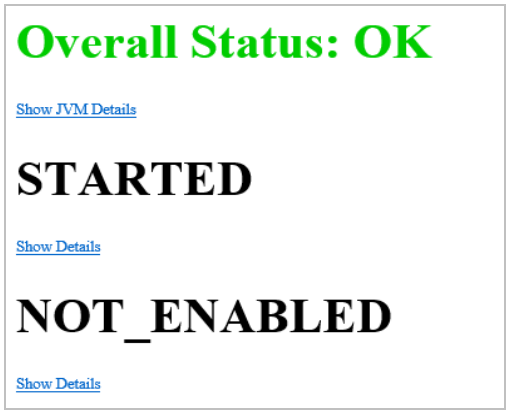
```
http://server_name_or_IP_address:8945/status
```

Where server_name_or_IP_address is the name or IP address of the server that hosts Anzo.

For example, the following image shows the Status page message displayed while Anzo is starting:



The image below shows the Status page message when Anzo startup is complete:



Changing Anzo Server Settings

The topics in this section provide instructions for changing server settings.

Important After changing any of the server configuration settings, restart Anzo to complete the configuration.

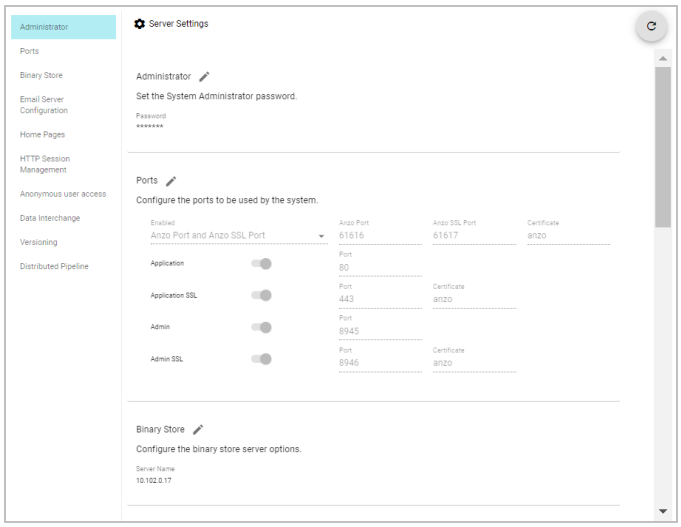
- Changing the Administrator Password
- Changing the Anzo Ports
- Changing the Binary Store Host
- Configuring an Email Server
- Changing the Application Home Page Path
- Changing the HTTP Session Timeout
- Enabling Anonymous User Access
- Security Implications of Anonymous Access

- Changing URI Prefixes and SPARQL Options
- Changing the Variable for Environment Version IDs
- Changing the Distributed Pipeline Configuration

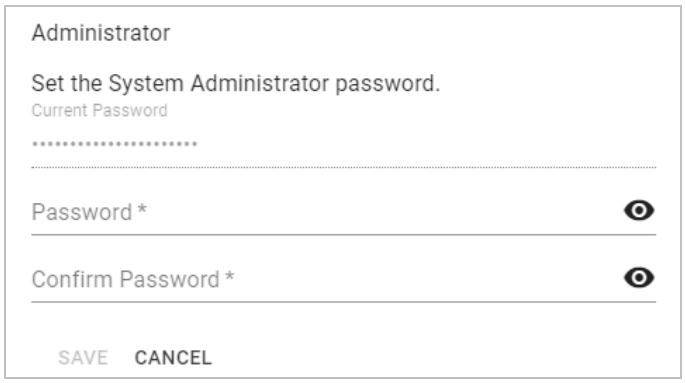
Changing the Administrator Password

This topic provides instructions for changing the Anzo administrator password.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the Server Settings screen, click the **Edit** icon (✎) next to **Administrator**. Anzo displays the Administrator settings.

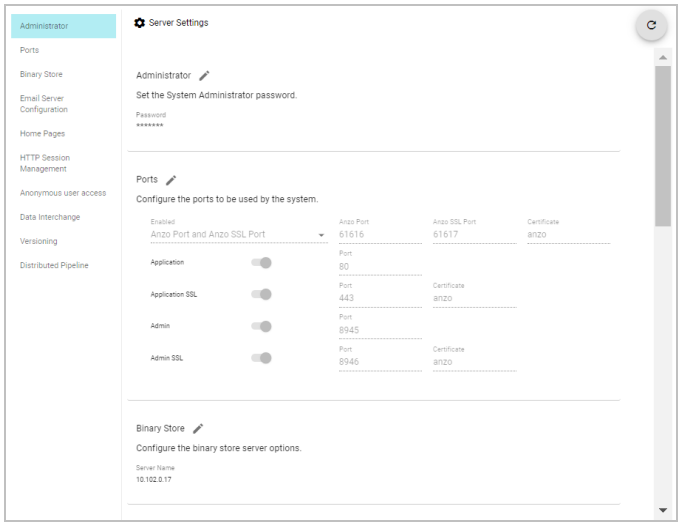


3. Type the new password in the **Password** and **Confirm Password** fields. Then click **Save**.
4. Restart Anzo to complete the new password configuration. You can restart the server by clicking the **Restart** button (↻) at the top of the Server Settings screen.

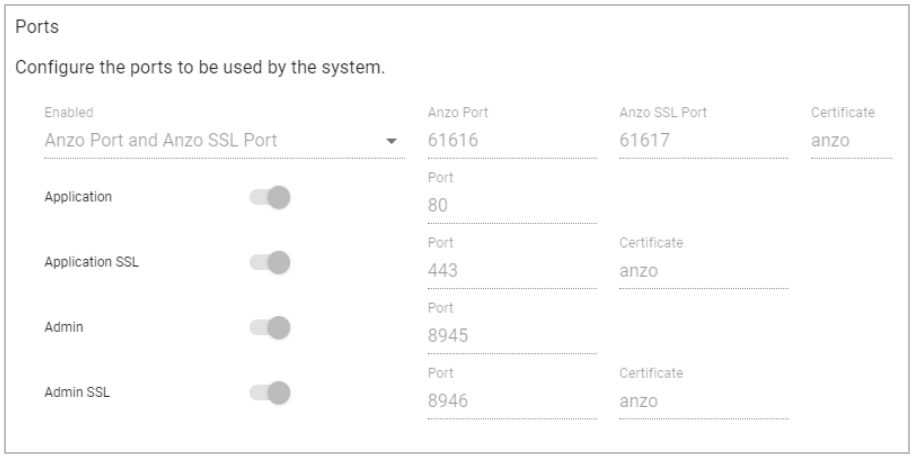
Changing the Anzo Ports

This topic provides instructions for changing or disabling Anzo server ports.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Ports** to navigate to the Ports section in the main screen. Then click the Edit icon (✎) next to **Ports**. The settings become editable. For example:



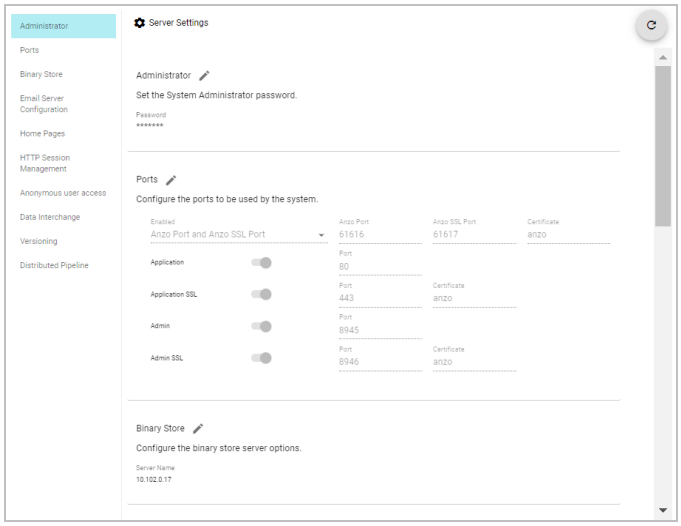
3. Change the values in the Port fields to designate alternate ports. If you want to disable a port, move the **Enable** slider to the left for that port.
    - **Anzo Port** and **Anzo SSL Port**: Anzo server ports.
    - **Application** and **Application SSL**: Anzo client application ports.
    - **Admin** and **Admin SSL**: Anzo administration client ports.

    For information about managing the certificates to use for the SSL ports, see Using a Signed Certificate for Anzo.
4. Click **Save** to save the port changes.
5. Restart Anzo to complete the port configuration. You can restart the server by clicking the **Restart** button (↻) at the top of the Server Settings screen.

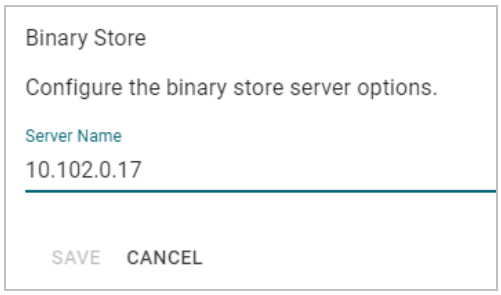
Changing the Binary Store Host

This topic provides instructions for changing the server host location for the binary (blob) store. This setting defaults to the host name for the Anzo server.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Binary Store** to navigate to the Binary Store section in the main screen. Then click the Edit icon (✎) next to Binary Store. The setting becomes editable. For example:

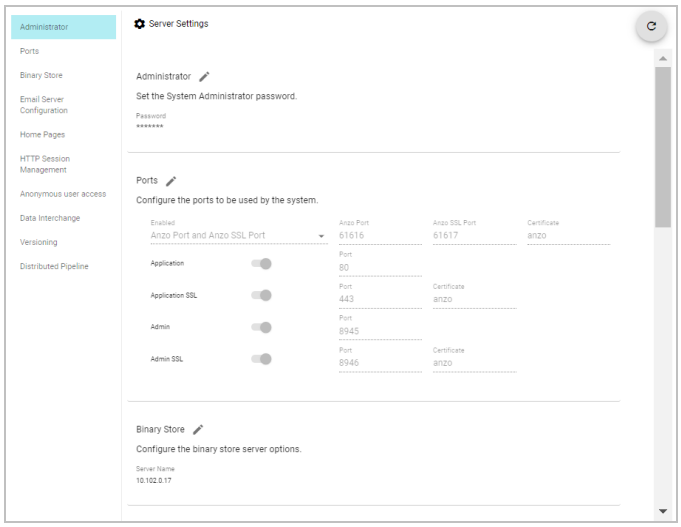


3. To specify a different host for the binary store, type the new host name in the **Server Name** field, and then click **Save**.
4. Restart Anzo to complete the binary store configuration. You can restart the server by clicking the **Restart** button (↻) at the top of the Server Settings screen.

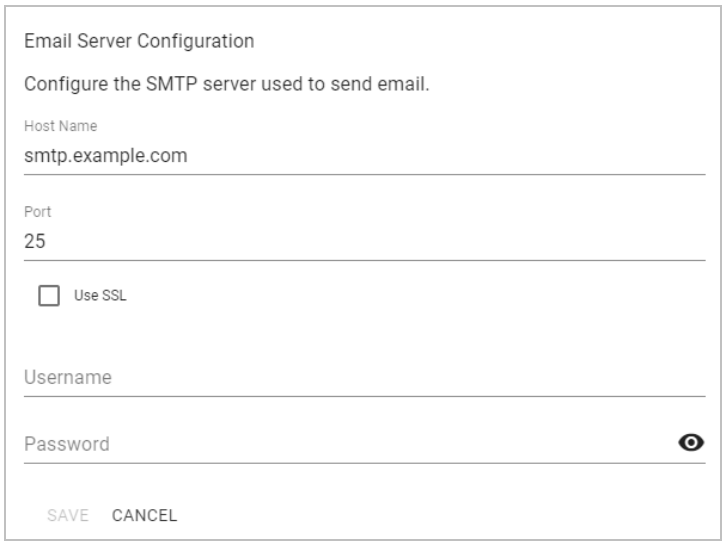
Configuring an Email Server

This topic provides instructions for configuring the SMTP server for sending email.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Email Server Configuration** to navigate to the Email Server Configuration section in the main screen. Then click the Edit icon (✎) next to Email Server Configuration. The settings become editable. For example:

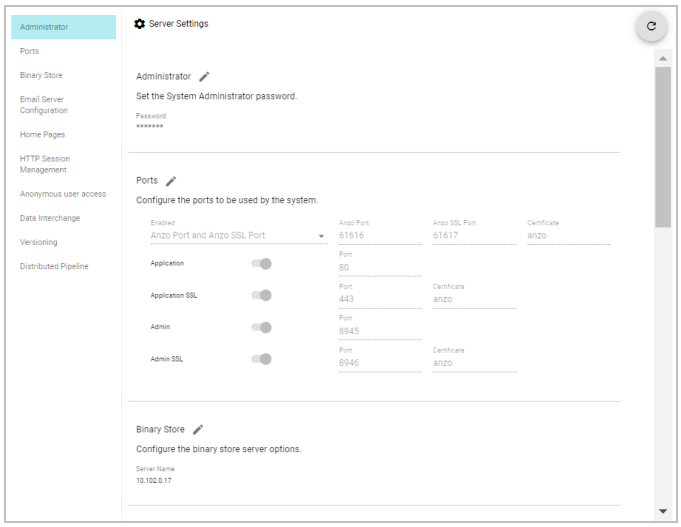


3. Replace the **Host Name** and **Port** values with the values for your server.
4. If the email server is configured for SSL authentication, select the **Use SSL** checkbox to enable SSL authentication. Then specify the **Username** and **Password** to use for authentication.
5. Click **Save** to save the configuration changes.
6. Restart Anzo to complete the email server configuration. You can restart the server by clicking the **Restart** button (↻) at the top of the Server Settings screen.

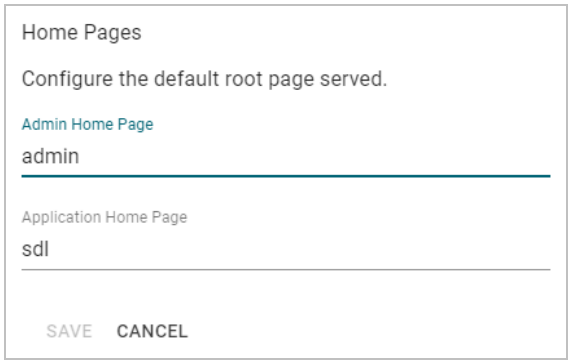
Changing the Application Home Page Path

This topic provides instructions for changing the path in the URL for accessing the Anzo application home pages.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Home Pages** to navigate to the Home Pages section in the main screen. Then click the Edit icon () next to Home Pages. The settings become editable. For example:

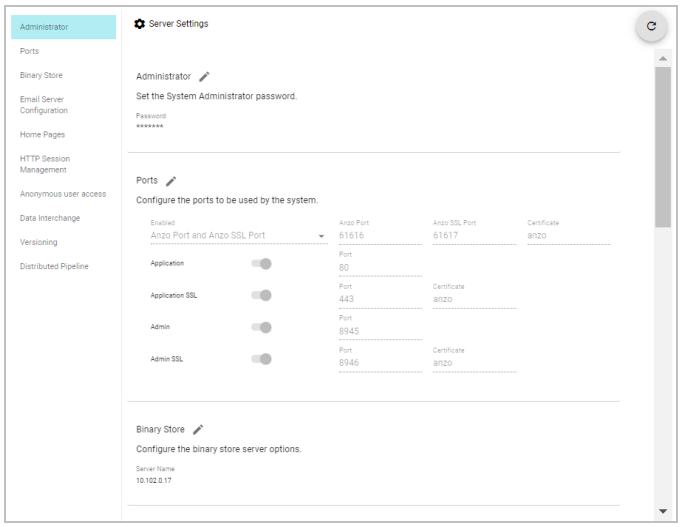


3. To change the path for the admin application home page, type the new value in the **Admin Home Page** field.
4. To change the path for the Anzo application home page, type the new value in the **Application Home Page** field.
5. Click **Save** to save the changes.
6. Restart Anzo to complete the home page configuration. You can restart the server by clicking the **Restart** button ( ) at the top of the Server Settings screen.

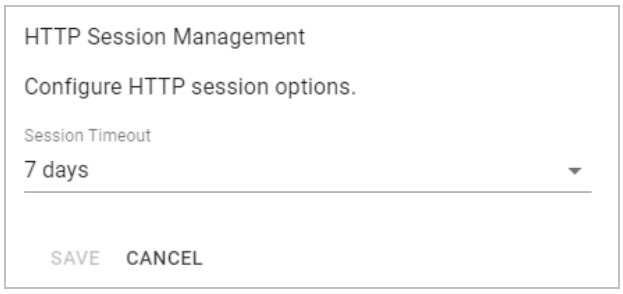
Changing the HTTP Session Timeout

This topic provides instructions for increasing or decreasing the HTTP session timeout value.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **HTTP Session Management** to navigate to the HTTP Session Management section in the main screen. Then click the Edit icon (✎) next to HTTP Session Management. The setting becomes editable. For example:



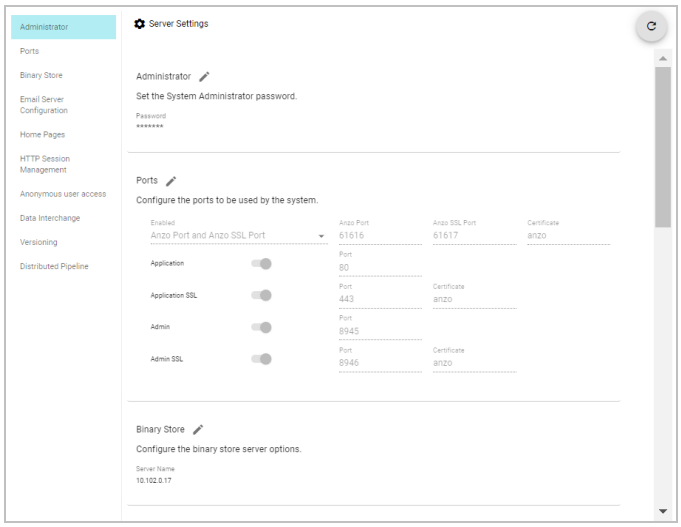
3. To choose an alternate value, click the **Session Timeout** drop-down list and select a value. Then click **Save**.
4. Restart Anzo to complete the session timeout configuration. You can restart the server by clicking the **Restart** button (↻) at the top of the Server Settings screen.

Enabling Anonymous User Access

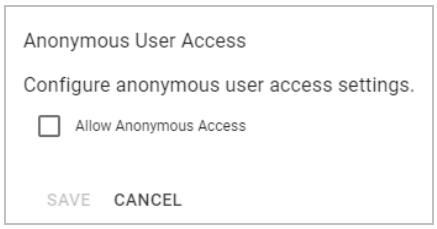
Important: Consider the security implications before you enable anonymous access. See Security Implications of Anonymous Access for information.

This topic provides instructions on enabling anonymous access and restricting anonymous users from seeing certain data.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Anonymous User Access** to navigate to the Anonymous User Access section in the main screen. Then click the Edit icon (✎) next to Anonymous User Access. The setting becomes editable. For example:



3. To enable anonymous access, slide select the **Allow Anonymous Access** checkbox, and then click **Save**.
4. Restart Anzo to complete the access configuration. You can restart the server by clicking the **Restart** button ( ) at the top of the Server Settings screen.

Data that is viewable by the **Everyone** role becomes visible to anonymous users. You might need to change the access control for existing data, such as by granting read access to the **Authenticated Users** role instead of the Everyone role. For information about changing the access to specific analytics, see Managing Dashboard and Lens Security.

Security Implications of Anonymous Access

This topic describes the security implications of using anonymous access. For instructions on enabling anonymous access and restricting anonymous users from seeing certain data, see Enabling Anonymous User Access.

- Anonymous User Permissions
- Anonymous User Limitations
- Important Considerations

Anonymous User Permissions

When anonymous access is enabled:

- The server allows users to connect without a username or password. A user can connect to the server without having an account in the directory server.
- Anonymous users are considered members of the Everyone role. Anonymous users can read data in the data lake that is tagged as readable by Everyone.

Anonymous User Limitations

Anonymous users cannot:

- Add, delete, or modify data. Anonymous users cannot write of delete data even if the Everyone role has write or delete access.
- Change access control lists on data. Anonymous users cannot change the access controls for any data on the server even if the Everyone role has write or delete access to the access control list.

Important Considerations

This section lists important ideas to consider before enabling anonymous access.

Consider Existing Access Control

Users might have been assigning access control without anticipating that users could have anonymous access. Before enabling anonymous access, consider that data that is viewable by the **Everyone** role becomes visible to anonymous users. You might need to change the access control for existing data, such as by granting read access to the **Authenticated Users** role instead of the Everyone role. For information about changing the access to specific analytics, see Managing Dashboard and Lens Security.

Consider Server Network Protections

Consider that anyone who can reach the server via the network will be able to use it as an anonymous user. Evaluate firewalls and other network protection mechanisms to limit access to the data lake server as desired. For example, you might want to allow anonymous access to anyone inside your organization's internal network but disable access to the server from the public internet.

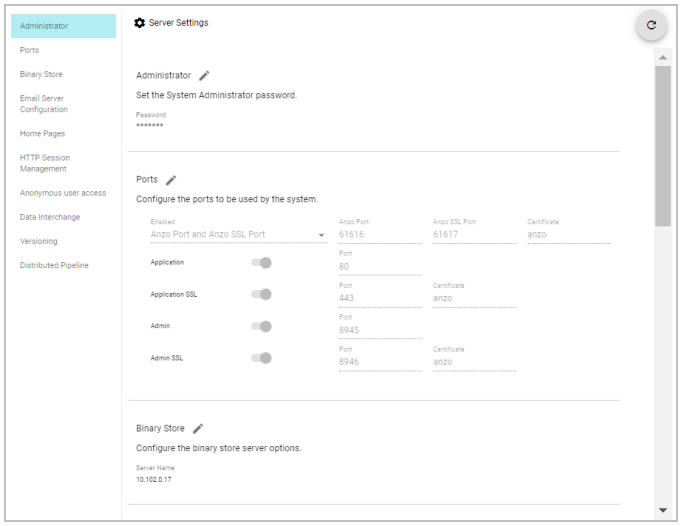
Anonymous Access Can Be Useful

Allowing anonymous access makes it easy to share data and views of data with others. For example, it means that you can share your Hi-Res Analytics with people who do not have a user account. It also lets you embed read-only interactive Hi-Res Analytic views inside other websites.

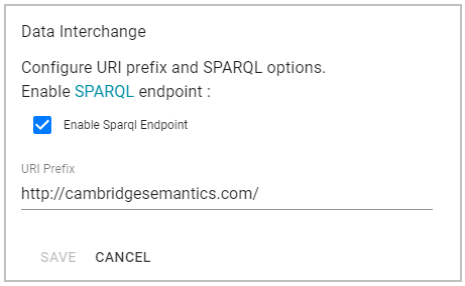
Changing URI Prefixes and SPARQL Options

This topic provides instructions for enabling or disabling the Anzo SPARQL endpoint and specifying the prefix to use when generating data identifiers.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Data Interchange** to navigate to the Data Interchange section in the main screen. Then click the Edit icon (✎) next to Data Interchange. The settings become editable. For example:

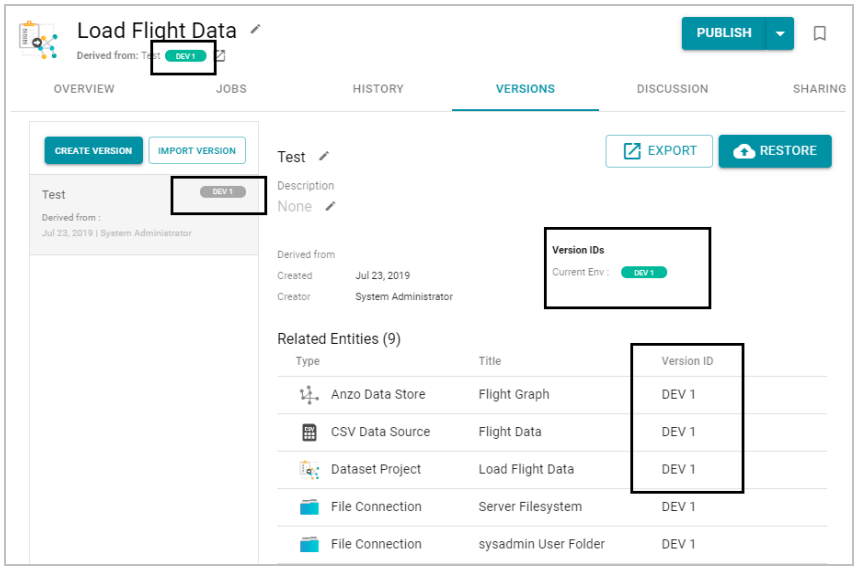


3. If you want to enable or disable the Anzo SPARQL endpoint, select or clear the **Enable SPARQL Endpoint** checkbox.
4. If you want to change the URI prefix that Anzo generates for URIs, type the new value in the **URI Prefix** field. The URI Prefix is mostly used for consistency in internal data, but it is also used by default for data model URI prefixes when the model does not define the URI template to use. When changing the URI Prefix, make sure that the value is a valid prefix. See Relative IRIs in the SPARQL Query Language specification for more information.
5. Click **Save** to save the changes.
6. Restart Anzo to complete the data interchange configuration. You can restart the server by clicking the **Restart** button (↻) at the top of the Server Settings screen.

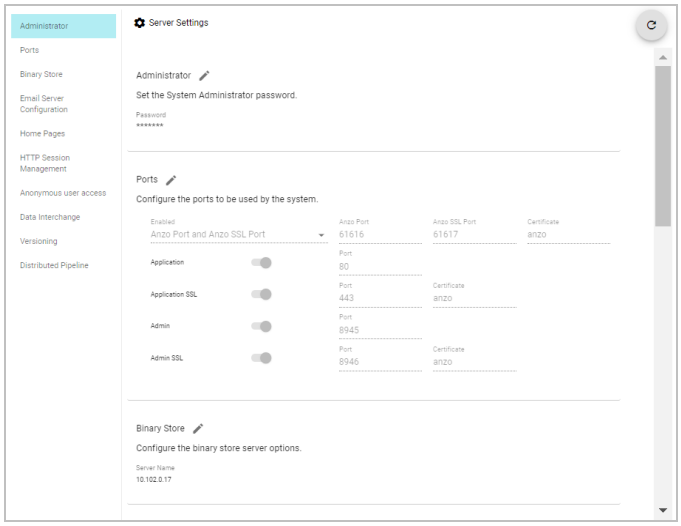
Changing the Variable for Environment Version IDs

This topic provides instructions for changing the variable value for the Version Environment tag that Anzo adds to archived versions of entities. The image below shows an example of the version tags that are controlled by the

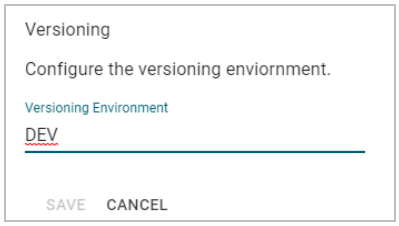
Version Environment setting. The black rectangles highlight the areas where the environment version variable value is displayed:



1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Versioning** to navigate to the Versioning section in the main screen. Then click the Edit icon (✎) next to Versioning. The setting becomes editable. For example:



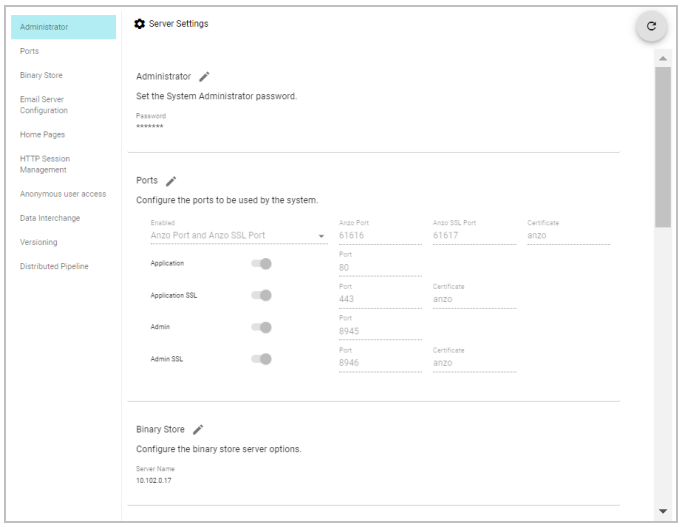
3. To change the value for version tags, modify the value in the **Versioning Environment** field.

4. Click **Save** to save the change.
5. Restart Anzo to complete the configuration. You can restart the server by clicking the **Restart** button ( ) at the top of the Server Settings screen.

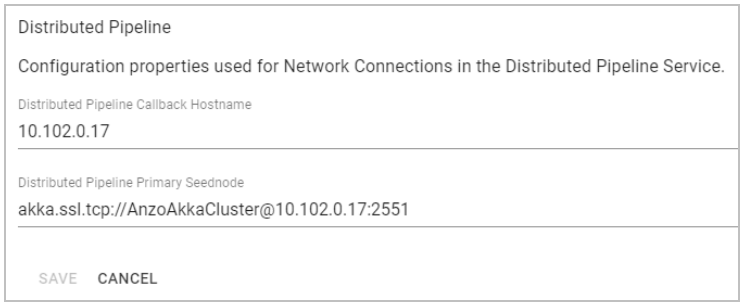
Changing the Distributed Pipeline Configuration

This topic provides instructions for changing the Anzo Distributed Unstructured configuration.

1. In the Anzo console, expand the **Administration** menu and click **Server Settings**. Anzo displays the Server Settings screen.



2. On the left side of the screen, click **Distributed Pipeline** to navigate to the Distributed Pipeline section in the main screen. Then click the Edit icon ( ) next to Distributed Pipeline. The settings become editable. For example:



3. Modify the settings as needed:
   - **Distributed Pipeline Callback Hostname**: The IP address for the Anzo Unstructured leader instance.

     Important: The value must be a routable IP address. If the leader instance is installed on the Anzo host server, specify the IP address of the server; do not use 127.0.0.1 or localhost.
   - **Distributed Pipeline Primary Seednode**: The IP address and port for the leader instance. By default the leader port is **2551**.

4. Click **Save** to save the changes.
5. Restart Anzo to complete the configuration. You can restart the server by clicking the **Restart** button ( ) at the top of the Server Settings screen.

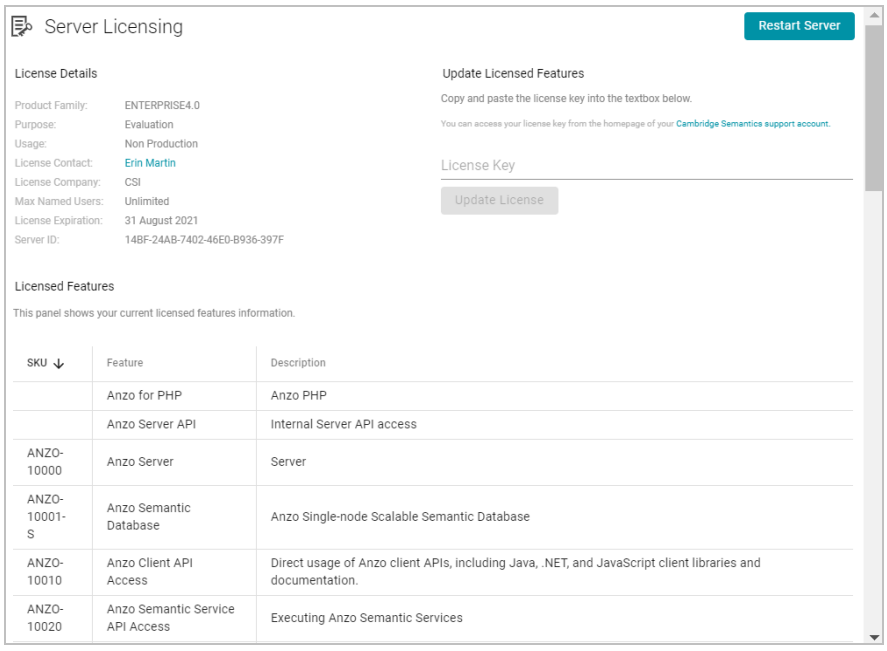
Updating the Server License

This topic provides important information about licenses and user accounts as well as instructions for updating a license key.

Updating the License Key

Follow the instructions below to update the Anzo server license key.

1. In the Anzo application, expand the **Administration** menu and click **Licensing**. Anzo displays the Server Licensing screen. For example:



2. Paste the new license key into the **License Key** field, and then click the **Update License** button. The new key is applied to the server. You can view the Licensed Features section to see the features for your license.

   Note You might have to refresh the browser before you can see the license changes.

Licensing and User Account Best Practices

When Anzo is initially installed, a server ID is generated based on a number of system properties, including the user account that runs the installation script. The Anzo server license is tied to that server ID. If Anzo is re-installed (for instance, during an upgrade) by a different user account, a new server ID is generated and the existing license

becomes invalid for the current installation. Whenever you upgrade or re-install Anzo, it is important to use the same user account that was used for the initial installation.

Restoring the Server ID if Anzo is Updated by the Wrong User

If Anzo is updated by a different user, the best way to resolve the issue is to revert the server ID to its original value by rolling back the update:

- If it was a new installation that used the wrong user account, uninstall Anzo. Then change to the correct user and run the installation script again.
- If your backup is a snapshot of the previous application disk, restore the disk. Then change to the correct user and update the installation.
- If it was an upgrade that used the wrong user account, restore Anzo from the backup that was saved before the upgrade:

  If your backup is a copy of the Anzo system journal, follow these steps:

  a. Uninstall Anzo.
  b. Change to the correct user account.
  c. Reinstall the previous version of Anzo using the original installation script.
  d. After the installation, replace the **anzo.jnl** file in the `install_path/Server/data/journal` directory with the backup version of the file.

     At this point, Anzo is restored to the previous version and has the server ID that is associated with the license.

  e. Now Anzo can be re-upgraded to the later release.

  If your backup is a copy of the entire Anzo installation directory, follow these steps:

  a. Uninstall Anzo.
  b. Change to the correct user account.
  c. Move the copy of the previous Anzo installation directory to the original location on the file system.

     At this point, Anzo is restored to the previous version and has the server ID that is associated with the license.

  d. Now Anzo can be re-upgraded to the later release.

Important Cambridge Semantics strongly recommends that you do NOT change the user running Anzo. If it is absolutely necessary, the license can be changed so that it is associated with the new server ID, and Anzo can be restarted once the license is updated. However, using a new server ID resets (or regenerates from non-customer-specific templates) all previously configured OSGI properties to their default values. Changing the Anzo user should only be attempted if there is a complete record of all of the customized OSGI properties and their values as well as a thorough change log so that the configuration can be restored if necessary.

Related Topics

Upgrading Anzo

Using a Signed Certificate for Anzo

This topic includes instructions for generating an SSL certificate using the OpenSSL utility, creating a signing request, and uploading the signed certificate to Anzo.

- Generating an SSL Certificate and Signing Request
- Uploading a Signed Certificate to Anzo

Generating an SSL Certificate and Signing Request

1. If necessary, install OpenSSL.
2. Run the following command to use OpenSSL to generate an RSA, 2048-bit public/private key pair called anzo-key.pem:

```
openssl genrsa -out anzo-key.pem 2048
```

```
Generating RSA private key, 2048 bit long modulus
.....................................................+++
.........+++
```

3. Run the following command to generate the Certificate Signing Request (CSR). This command creates a CSR called anzo-csr.csr for the public key in anzo-key.pem.

```
openssl req -new -key anzo-key.pem -out anzo-csr.csr
```

4. OpenSSL prompts you to enter information to incorporate into the CSR. Provide the required information. For example:

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
If you enter '.', the field will be left blank.
```

```
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:MA
Locality Name (eg, city) []:Boston
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Cambridge Semantics
Organizational Unit Name (eg, section) []:Solutions
Common Name (e.g. server FQDN or YOUR name) []:sample.cambridgesemantics.com
Email Address []:webmaster@cambridgesemantics.com
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

5. Send the resulting CSR to a certificate authority for signing.

Uploading a Signed Certificate to Anzo

1. When you receive the signed certificate from the certificate authority, rename the certificate to **anzo-crt.crt**.
2. Create a PKCS12 key:
   a. Run the following command to concatenate the signed certificate and key file into an anzo.pem file:

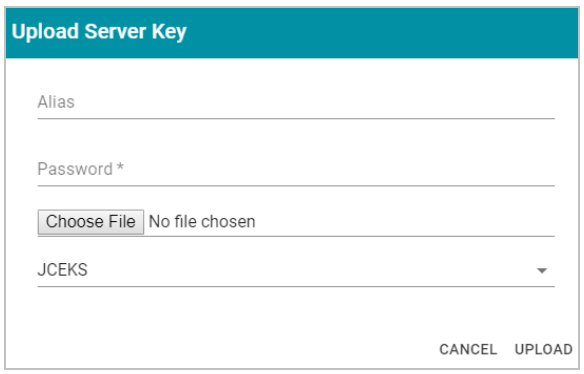
   ```
   cat anzo-key.pem anzo-crt.crt > anzo.pem
   ```

   b. Run the following command to use OpenSSL to convert the resulting anzo.pem file to PKCS12, choose a name for the certificate, and set an export password:

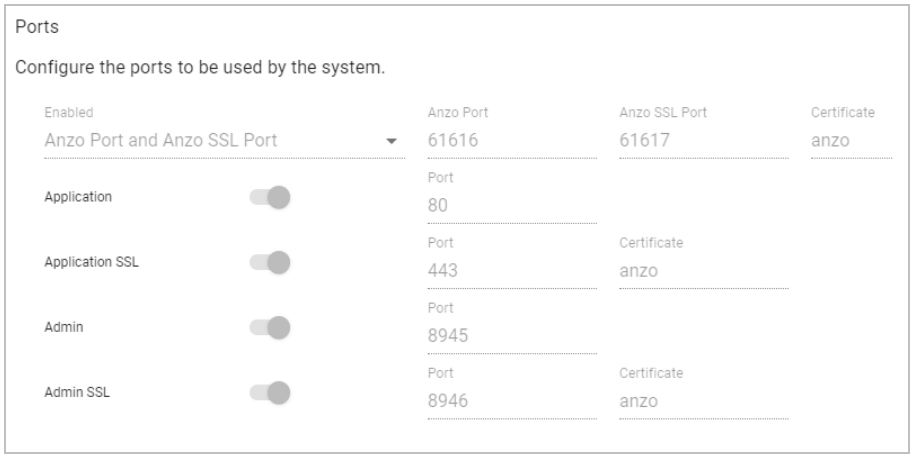
   ```
   openssl pkcs12 -export -in anzo.pem -out anzo.pkcs12 -name "alias"
   ```

   ```
   Enter Export Password:
   Verifying - Enter Export Password:
   ```

3. Copy the anzo.pkcs12 certificate to a server or location where you can select it using the Anzo console.
4. Log in to the Anzo console, expand the **Administration** menu item, and click **Server Certificates**.
5. On the Server Certificates screen, click **Upload Server Key**. Anzo displays the Upload Server Key dialog box.



6. Follow these steps to upload the key:
    a. In the **Alias** field, type that name that you chose when you created the PKCS12 certificate.
    b. In the **Password** field, type the export password that you set when you created the PKCS12 certificate.
    c. Click the **Choose File** button and select the **anzo.pkcs12** file.
    d. Click the **Keystore type** field and select **PKCS12** from the drop-down list.
    e. Click **Upload** to upload the certificate.
7. To apply the new certificate to the SSL ports for the Anzo server:
    a. In the Anzo console, click **Server Settings** under the Administration menu.
    b. On the left side of the screen, click **Ports** to navigate to the Ports section in the main screen. Then click the Edit icon (✎) next to **Ports**. The settings become editable. For example:

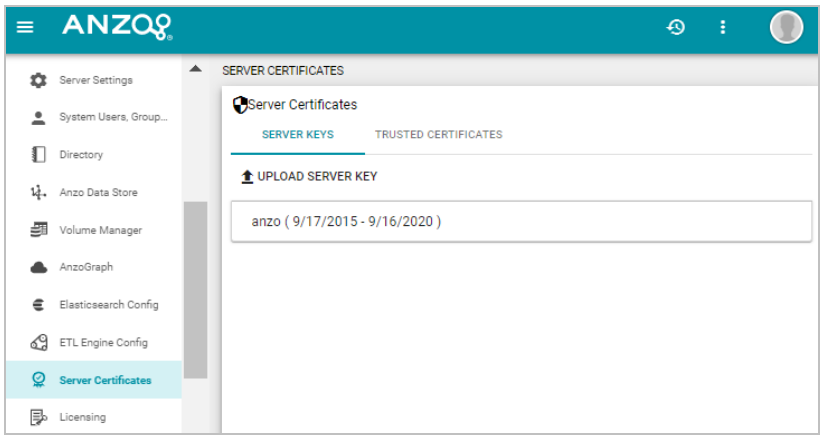


    c. Select the new certificate from the **Certificate** drop-down lists. Then click **Save**.
8. Restart Anzo to complete the certificate configuration. You can restart the server by clicking the **Restart** button () at the top of the Server Settings screen.

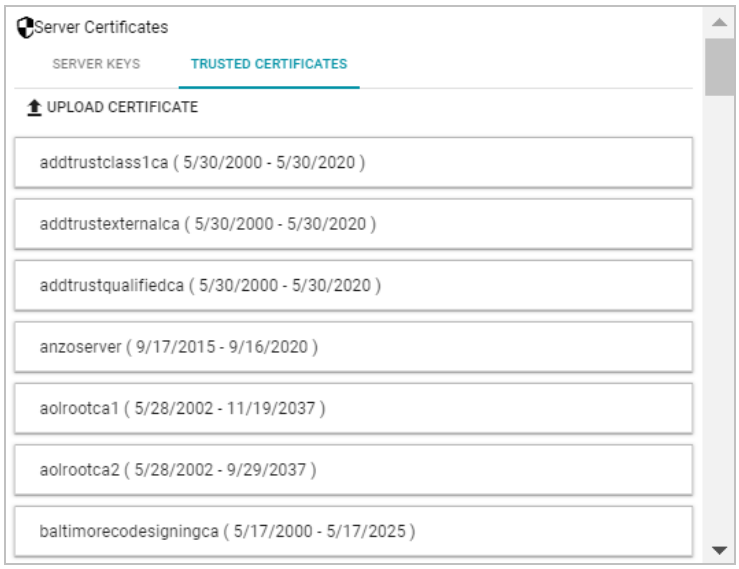
Uploading a Certificate to the Anzo Trust Store

This topic provides instructions for adding a trusted certificate to the Anzo trust store.

1. In the Anzo console, expand the **Administration** menu and click **Server Certificates**. Anzo displays the Server Certificates screen. For example:



2. On the Server Certificates screen, click the **Trusted Certificates** tab. Anzo displays the list of existing certificates. For example:



3. To upload a new trusted certificate, click **Upload Certificate**. Browse to the certificate file, and double-click the file to upload it to Anzo.

Managing Volumes

The topics in this section provide information about creating new volumes (database instances) and mounting existing volumes.

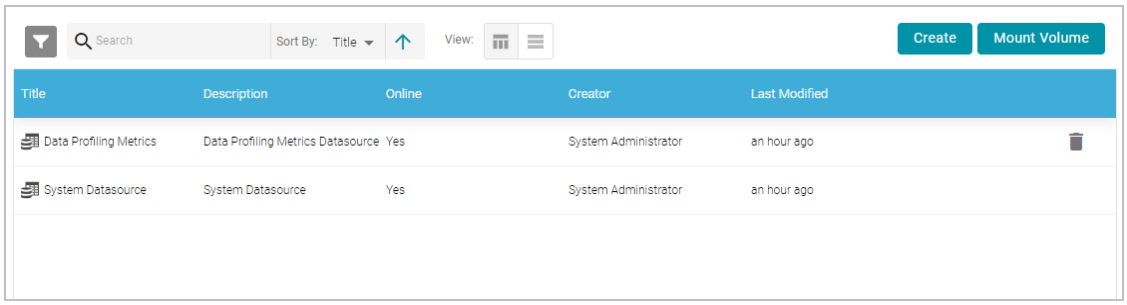
- Creating a New Volume
- Mounting an Existing Volume

Creating a New Volume

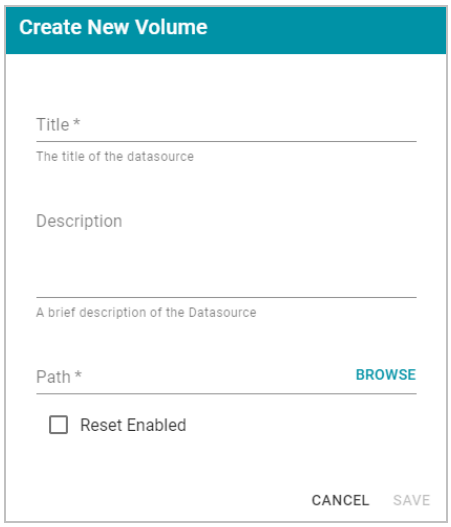
This topic provides instructions for creating new volumes or database instances.

Note The number of volumes that you can create depends on your software license. For more information, contact Cambridge Semantics Support.

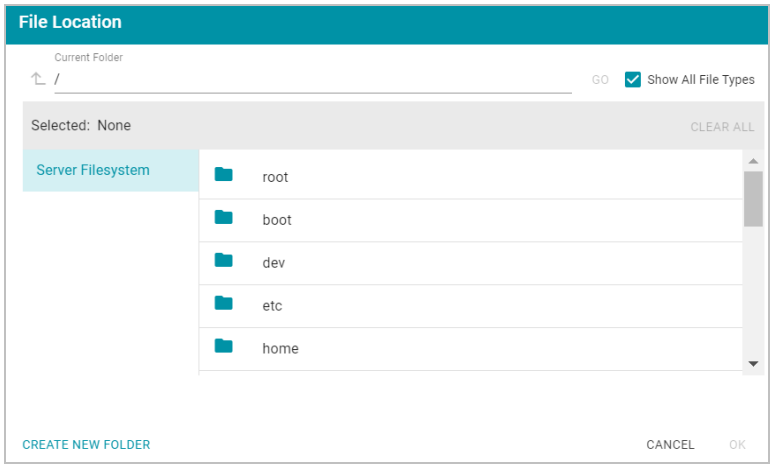
1. In the Anzo console, expand the **Administration** menu and click **Volume Manager**. Anzo displays the Volume Manager screen, which lists any existing volumes. For example:



2. At the top of the screen, click **Create**. Anzo displays the Create New Volume screen.



3. In the **Title** field, type a name for the new volume, and type an optional description in the **Description** field.
4. Click in the **Path** field. Anzo opens the File Location dialog box. For example:



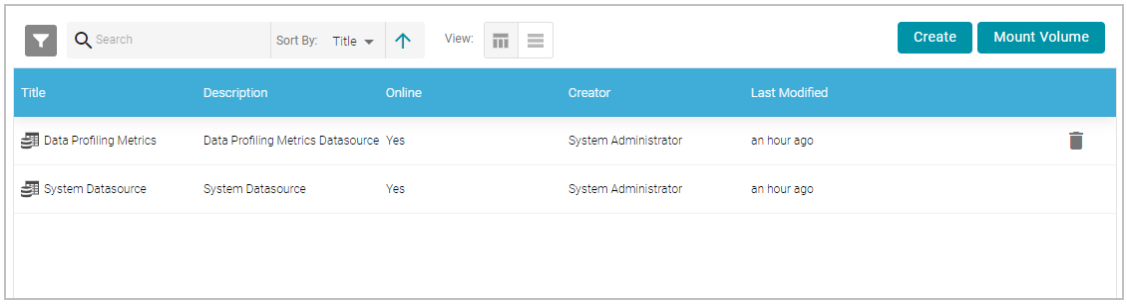
5. On the left side of the screen, select the file store or storage location where you want to create this volume. On the right side of the screen, select the directory where you want Anzo to save the volume. Then click **OK**. For instructions on creating a new file store, see Connecting to a File Store.
6. On the Create New Volume screen, specify whether to enable resets. When reset is enabled, the option to reset the entire contents of the volume becomes available. To enable resets for this volume, select the **Reset Enabled** checkbox. To disable resets, leave the checkbox clear.
7. Click **Save** to create the new volume in the location that you specified.

Mounting an Existing Volume

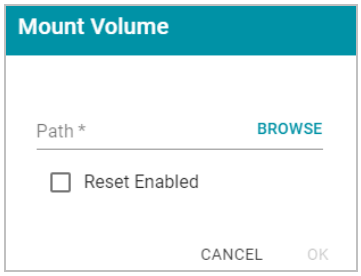
This topic provides instructions for mounting an existing volume to Anzo.

Note: The number of volumes that you can mount depends on your software license. For more information, contact Cambridge Semantics Support.

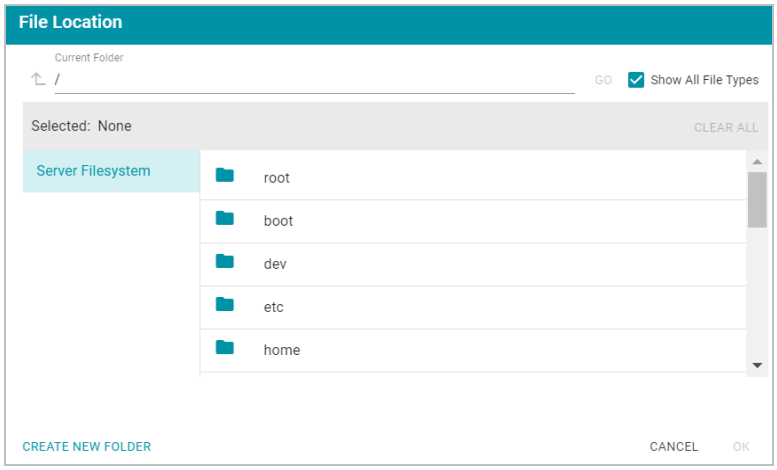
1. In the Anzo console, expand the **Administration** menu and click **Volume Manager**. Anzo displays the Volume Manager screen, which lists any existing volumes. For example:



2. At the top of the screen, click **Mount Volume**. Anzo displays the Mount Volume screen.



3. Click in the **Path** field. Anzo opens the File Location dialog box. For example:

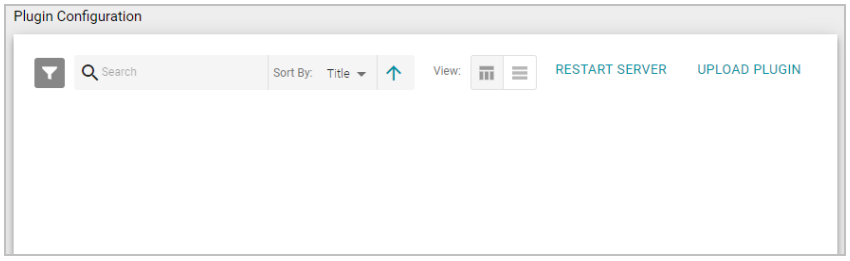


4. On the left side of the screen, select the file store that hosts the volume (.jnl file) that you want to mount. On the right side of the screen, navigate to the .jnl file and select it. Then click **OK**. Anzo mounts the new volume.

Uploading a Plugin

When connecting to a relational database to import data, you may need to upload a JDBC driver to Anzo. You may also need to import custom bundles or bundles received from Cambridge Semantics. This topic provides instructions for uploading executable .jar files from your computer to Anzo.

1. In the Anzo console, expand the **Administration** menu and click **Plugin Configuration**. Anzo displays the Plugin Configuration screen. For example:



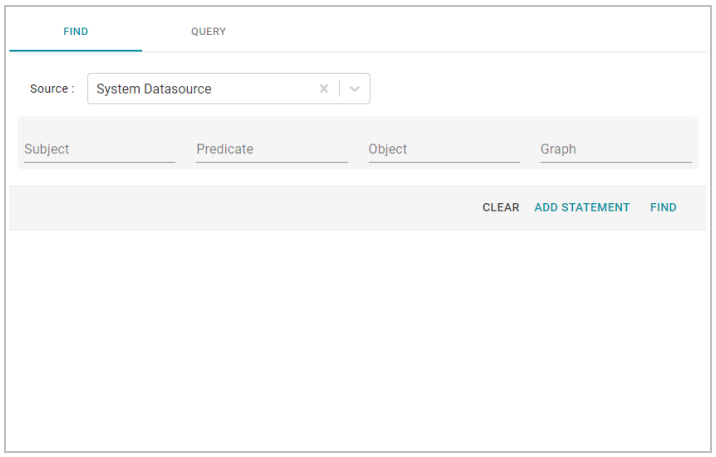
2. In the top right corner of the Plugin Configuration screen, click **Upload Plugin**. The application opens the file browser on your computer.
3. In the file browser, navigate to the .jar file to upload, and then double-click the file to upload it. Anzo uploads the file and displays a "Completed" message. You do not need to restart the Anzo server to apply the new executable.

Routing Hi-Res Analytics to a Custom URL

If you have a custom skin or personality for the Hi-Res Analytics application, and you want those customizations to be loaded automatically when users access the application, you can configure the Anzo application to re-route users to the preferred URL. Follow the instructions below to change the entry points to the Hi-Res application in the Anzo

application. The instructions use the Find feature in the Query Builder to find and modify the object of the Hi-Res Analytics routing property.

1. In the Anzo application, expand the **Access** menu and click **Query Builder**.
2. In the Query Builder, click the **Find** tab. The Find screen is displayed with the **System Datasource** selected as the Source.



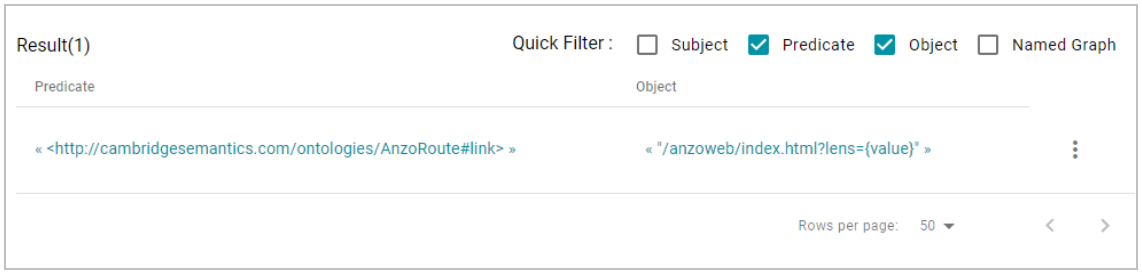
3. In the **Subject** field, specify the following URI:

```
http://cambridgesemantics.com/Routes/sdi/hi-res-analytics-urn
```

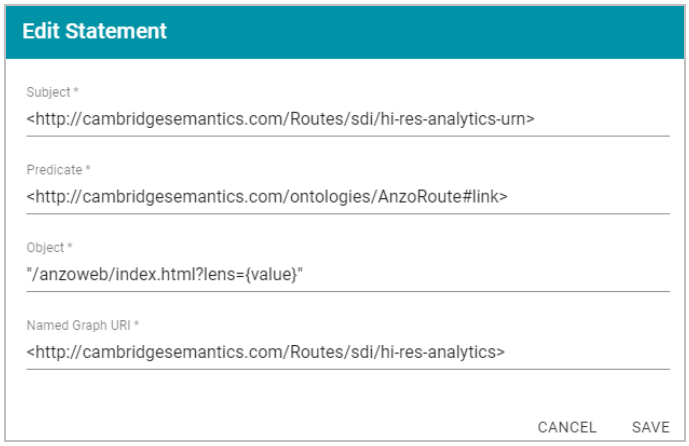
4. In the **Predicate** field, specify this URI:

```
http://cambridgesemantics.com/ontologies/AnzoRoute#link
```

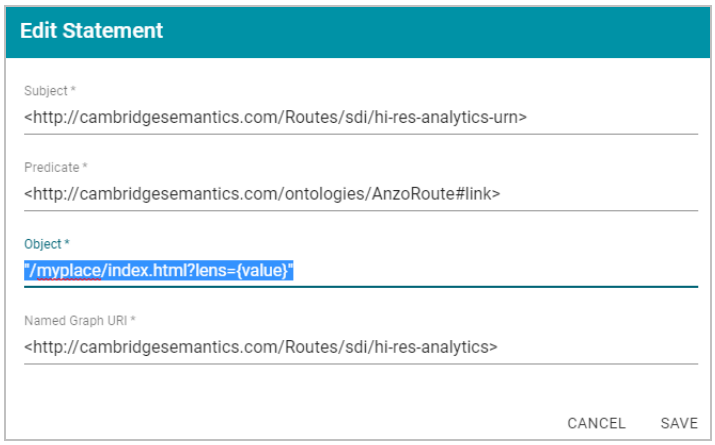
5. Click **Find** to display the quads with the specified subject and predicate. You can clear the **Subject** and **Named Graph** Quick Filter checkboxes to make the results easier to read. For example:



6. Click the menu icon (⋮) for the quad and select **Edit**. Anzo opens the Edit Statement dialog box.



7. In the Edit Statement dialog box, replace the **Object** value (**"/anzoweb/index.html?lens={value}"**) with the URL that you want to route users to. For example: **"/myplace/index.html?lens={value}"**.



8. Click **Save** to apply the change and return to the Find screen.

The Anzo application is now configured to route users to the custom URL if they open the Hi-Res Analytics application from the Home page, open a dashboard from the Hi-Res Analytics screen, or click **Create Dashboard** from a Graphmart screen.

Related Topics

Analyzing Data with Hi-Res Analytics

Connection Administration

The topics in this section provide information about managing connections to the Anzo server.

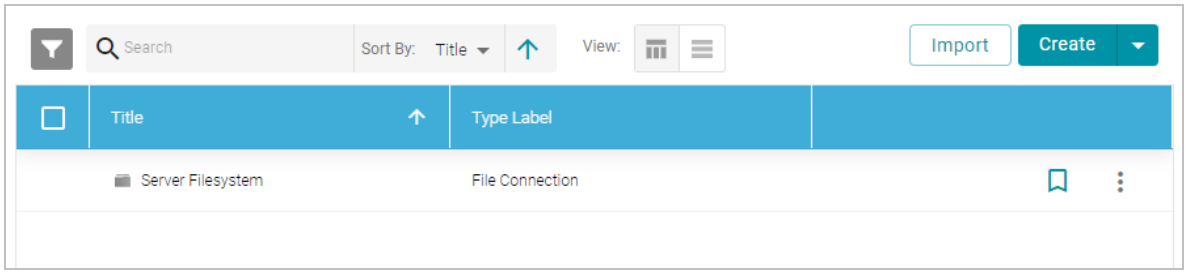
- Connecting to a File Store
- Creating an Anzo Data Store
- Connecting to AnzoGraph
- Connecting to Elasticsearch
- Connecting to an ETL Engine
- Managing Cloud Locations

Connecting to a File Store

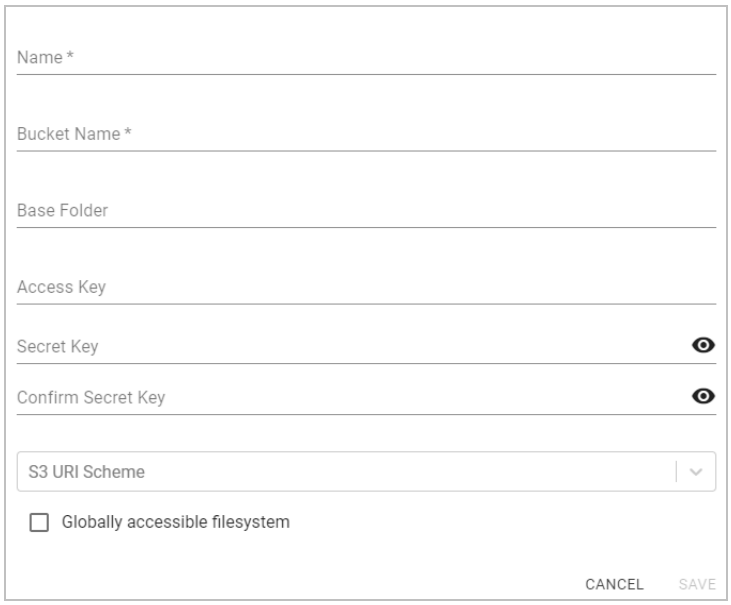
This topic provides instructions for defining a file system location for storing the files that Anzo ingests or creates. Anzo supports reading from and writing to local or mounted file systems (such as NFS), Hadoop Distributed File Systems (HDFS), File Transfer Protocol (FTP or FTPS) systems, Google Cloud Platform (GCP) storage, and Amazon Simple Cloud Storage Service (S3).

Tip The Anzo server file system location is configured and accessible by default. If you store files on a storage system that is mounted directly onto the Anzo, AnzoGraph, Elasticsearch, and Anzo Unstructured servers, you are not required to configure that location.

1. In the Anzo console, expand the **Administration** menu and click **File Store**. Anzo displays the File Store screen, which lists existing file store connections. For example:



2. Click the **Create** button and select the type of file connection that you want to create. For the local disk or mounted NFS, choose **File**. Anzo displays the create connection screen. For example:



3. On the connection screen, provide the file system details. The settings that display depend on the type of file connection that you chose. The list below describes the settings for each file connection type.

File

- **Name**: The name to use to describe this file connection within Anzo.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

HDFS

- **Name**: The name to use to describe this file connection within Anzo.
- **Nameservice IP or Name**: The IP address or host name for the storage system.
- **Port**: The port to access the server on.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **HDFS Configuration Path**: Enter the full path to the configuration files.
- **Keytab Path**: The full path to the keytab file.
- **Username**: The user name for the account used to access the server.
- **Password** and **Confirm Password**: The password for the account used to access the server.

- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

Important If you use Kerberos Authentication with HDFS, you must also configure your AnzoGraph cluster to authenticate with Kerberos. For instructions, see Configuring AnzoGraph for Kerberos Authentication.

FTP and FTPS

- **Name**: The name to use to describe this file connection within Anzo.
- **Server IP or Name**: The IP address or host name for the storage system.
- **Port**: The port to access the server on.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Username**: The user name for the account used to access the server.
- **Password** and **Confirm Password**: The password for the account used to access the server.
- **Keystore Path**: For FTPS connections, the full path to the keystore file.
- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

GCP

- **Name**: The name to use to describe this file connection within Anzo.
- **Bucket Name**: The name of the bucket to store files in.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Account Email**: The email address for the account used to access the storage.
- **Key File Location**: The full path to the keystore password file.
- **Globally accessible filesystem**: Select this option if this file store is accessible by all of the servers in an AnzoGraph cluster. If only the AnzoGraph leader server can access this system, leave this option blank.

S3

Important When using Amazon S3 for file storage, make sure that AWS server-side encryption is disabled. Anzo cannot read and write files on S3 if the location uses server-side encryption.

- **Name**: The name to use to describe this file connection within Anzo.
- **Bucket Name**: The name of the bucket to store files in.
- **Base Folder**: The base or root folder on the file system where you want Anzo to either read or write files. Each time Anzo generates new files it creates a new subdirectory under this base location.
- **Access Key**: The Access Key ID to use for accessing the S3 location.
- **Secret Key** and **Confirm Secret Key**: The Secret Key ID for the Access Key.
- **S3 URI Scheme**: Specifies whether the URI scheme is S3, S3 Native, or S3A.
- **Globally accessible filesystem**: **Required**. Enable this option for S3 file stores.

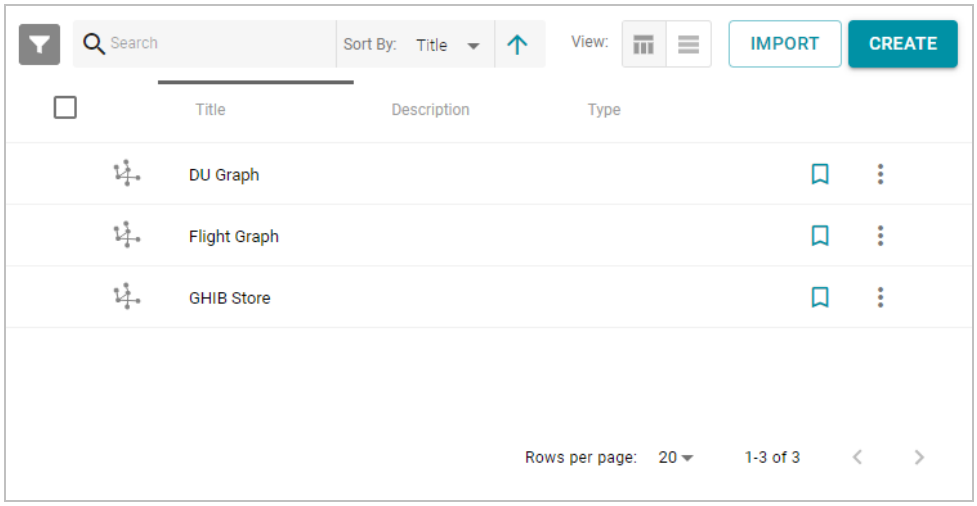
4. Click **Save** to save the configuration. The file store connection that you specified becomes available as a choice when you create graph data stores or select source files to onboard.

Creating an Anzo Data Store

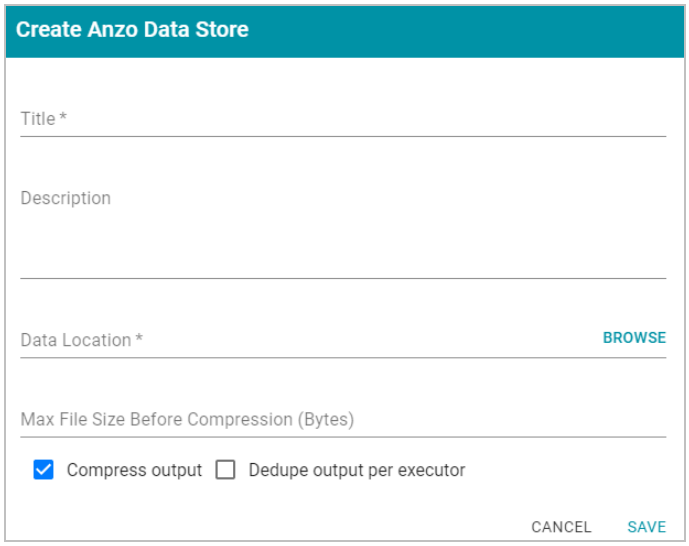
An Anzo data store, also known as a graph data source, is a designated directory on the shared file system where Anzo saves the AnzoGraph load files that are generated during the ETL process. You can create one Anzo data store and configure all pipelines to write to that store (each ETL run automatically creates a new sub-directory under the data store directory) or you can create multiple Anzo data stores and use a different one for each data set. This topic provides instructions for creating a data store.

For information about setting up a file store connection, see Connecting to a File Store.

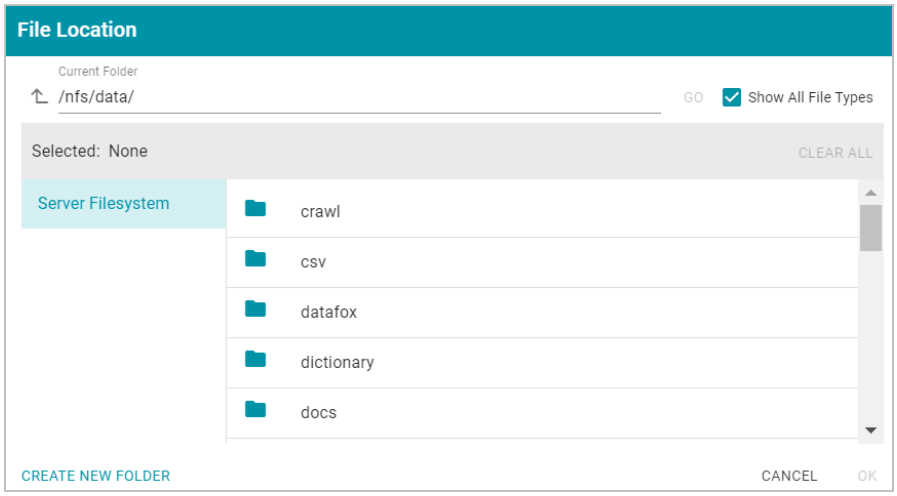
1. In the Anzo console, expand the **Administration** menu and click **Anzo Data Store**. Anzo displays the Anzo Data Store screen, which lists any existing data stores. For example:



2. On the Anzo Data Store screen, click the **Create** button. Anzo opens the Create Anzo Data Store screen.



3. Type a **Title** and optional **Description** for the data store.
4. Click in the **Data Location** field. Anzo opens the File Location dialog box.



5. On the left side of the screen, select the storage location where you want to create this data store. On the right side of the screen, navigate to the base directory where you want Anzo to save the data files for this data store. Select a directory, and then click **OK**. Each time ETL runs for this store, Anzo creates a new subdirectory under the base location that you specify.

   Note Ideally, the Data Location is a directory that the Anzo, AnzoGraph, and any Anzo Unstructured and Elasticsearch servers have access to, such as a mounted file system or cloud storage location. If you want Anzo to generate files for this data store in one location and load the files into AnzoGraph from another location, specify the file generation location in this field, and then specify the AnzoGraph load location in the **Alternate Data Location** field that is displayed on the details screen after you save the data store.

6. If necessary, you can modify the maximum limit for the size of the files that are created by pipelines that write to this data store by specifying the size (in bytes) in the **Max File Size Before Compression (Bytes)** field. The value applies to files before they are compressed. The Spark ETL engine partitions files on output, and the default maximum file size is 100 MB (uncompressed). The Sparkler ETL engine partitions files on input, and the default maximum file size is 128 MB (uncompressed). Since Sparkler files are partitioned on input, the resulting output FLDS files can be significantly larger than 128 MB since the source is converted to Turtle (TTL) format after it is partitioned.

   Note Cambridge Semantics recommends that you do not set this value unless instructed to do so by Cambridge Semantics Support.

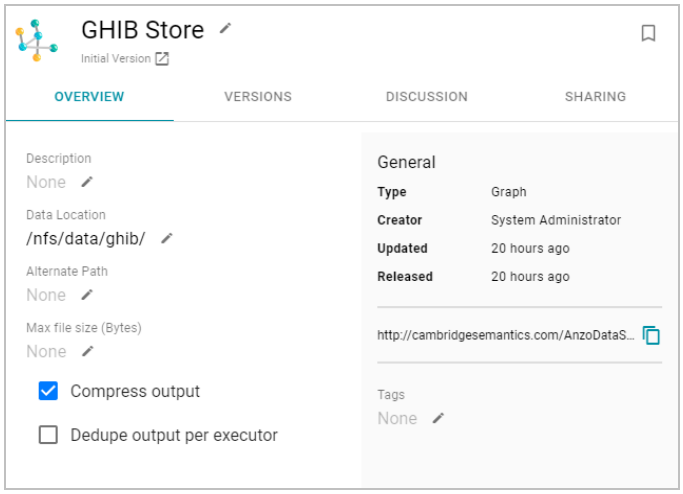
7. Specify whether to compress the generated load files. By default, the **Compress output** checkbox is selected, indicating that Anzo generates .ttl.gz files when writing to this graph data source. If you clear the checkbox, Anzo generates uncompressed .ttl files. To preserve disk space and reduce read times when loading data into memory, Cambridge Semantics recommends that you accept the default configuration and compress load files.
8. The Spark ETL engine does not remove duplicates by default when running pipelines. If the source contains a significant number of duplicate entities, you have two options for deduplicating the data:
   - **Deduplicate the data during the ETL process**: To deduplicate the data while running the jobs that will generate this graph source, select the **Dedupe output per executor** option. Enabling the dedupe option limits the number of duplicates to one duplicate per executor node. For example, if the Spark configuration has 10 executor nodes, the resulting data set can contain a maximum of 10 duplicate entities. Note that deduplication is based on primary keys and URI templates. **If the source does not employ templating, do not enable the dedupe option**.

     Important Enabling this option substantially increases the time it takes to run the jobs for this graph source.

   - **Deduplicate the data after loading it to AnzoGraph**: AnzoGraph deduplicates data during a "vacuum" process that runs automatically after data is loaded into memory. If you leave the Dedupe output per executor option disabled, duplicates will be removed by AnzoGraph.

     Note Deduplicating data with AnzoGraph streamlines the ETL process but can increase load time and temporary memory usage in AnzoGraph during the load.

9. Click **Save** to create the data store. Anzo saves the configuration and displays the details view. For example:



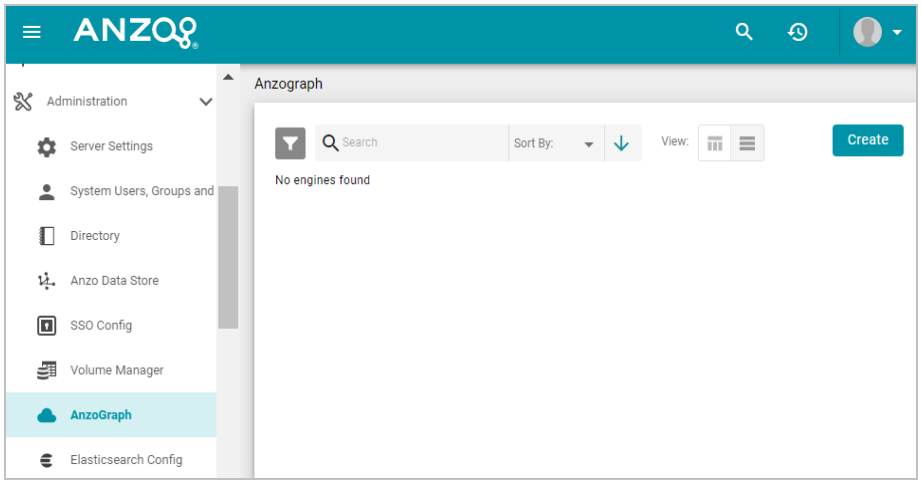
   You can click the Edit icon ( ) to modify any of the options. Click the check mark icon ( ) to save changes to an option, or click the X icon ( ) to clear the value for an option.

10. If you plan to load files into AnzoGraph from a location that is different than the **Data Location** that you specified, edit the **Alternate Data Location** field and select the location for AnzoGraph load files.

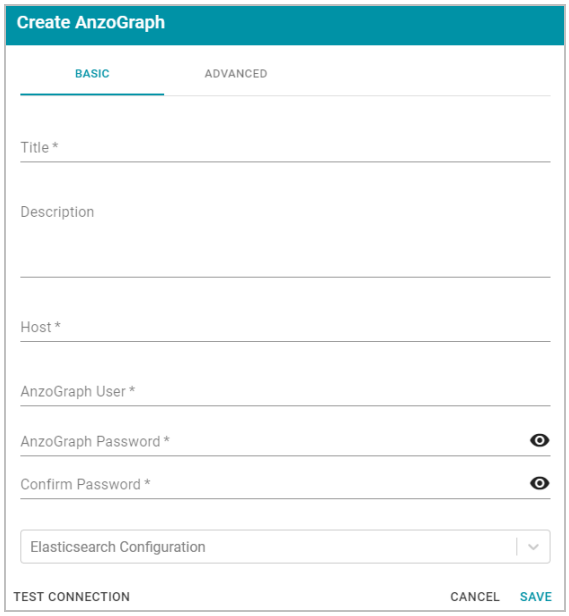
Connecting to AnzoGraph

This topic provides instructions for configuring the connection to AnzoGraph in the Anzo console. For information about managing AnzoGraph servers, see AnzoGraph Server Administration.

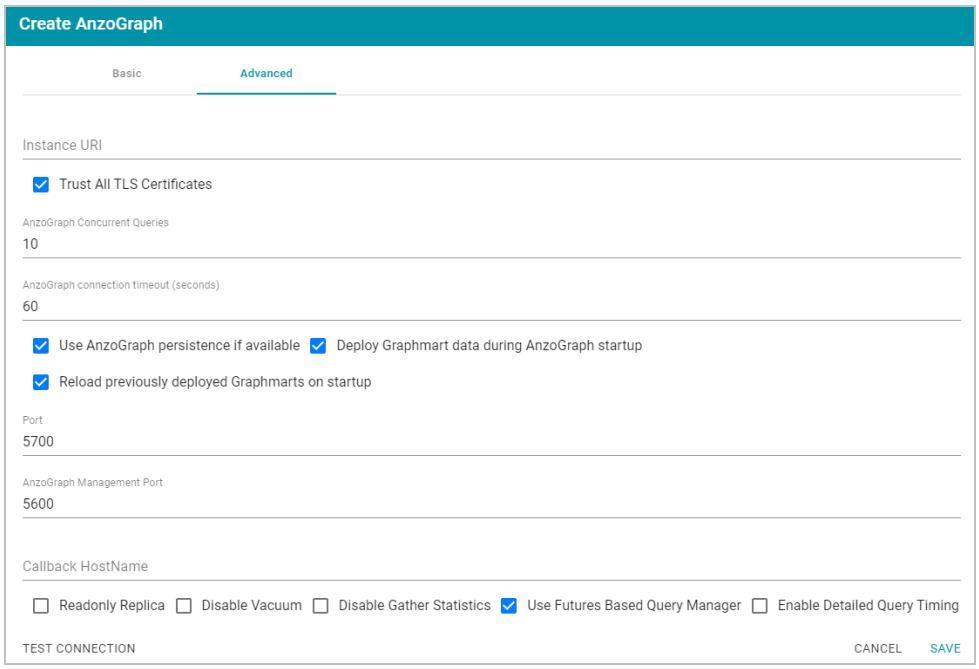
1. In the Anzo console, expand the **Administration** menu and click **AnzoGraph**.
2. On the AnzoGraph screen, click the **Create** button in the top right corner of the screen. Anzo opens the AnzoGraph connection overview screen, which lists any existing connections.



3. Click **Create** to add a connection. Anzo displays the Create AnzoGraph dialog box.



4. On the Basic tab, type a name for the engine in the **Title** field.
5. In the optional **Description** field, type a description for the graph query engine. If you leave this field blank, Anzo creates a description when you save the configuration.
6. In the **Host** field, type the AnzoGraph server host name or IP address. If you have a cluster, type the name or IP address of the leader server.
7. In the **AnzoGraph User** field, type the username that was created when AnzoGraph was installed.
8. Type the password for the AnzoGraph user in the **AnzoGraph Password** and **Confirm Password** fields.
9. If this AnzoGraph instance will host data from unstructured pipelines, click the **Elasticsearch Configuration** drop-down list and select the Elasticsearch instance to associate with this AnzoGraph connection. For information about configuring an Elasticsearch connection, see Connecting to Elasticsearch.
10. Click **Test Connection** to check if Anzo can connect to AnzoGraph. If the connection fails, make sure that AnzoGraph is running and that you typed the correct username and password.
11. **Optional**: Click the **Advanced** tab and configure any of the optional advanced settings. The table below describes each setting.



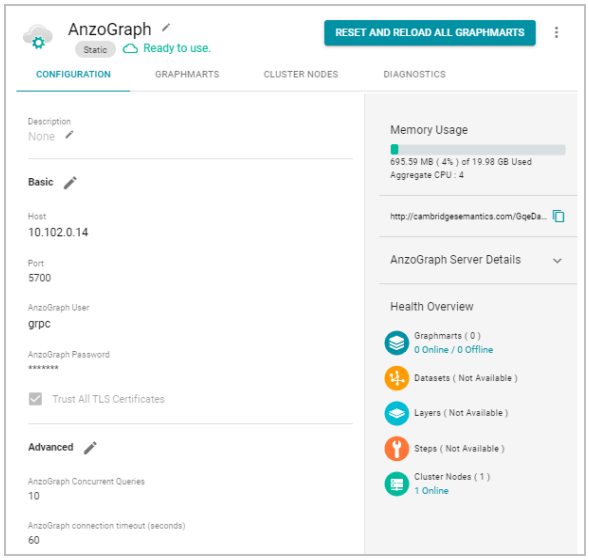
Setting	Description
Instance URI	Anzo automatically assigns an instance URI to this instance. If you want to choose a custom URI, type the URI in this field.
Trust All TLS Certificates	Indicates whether Anzo should trust the AnzoGraph certificates for this connection. Cambridge Semantics recommends that you accept the default value of enabled.
AnzoGraph Concurrent Queries	The number of queries that Anzo sends to AnzoGraph concurrently. The default value is 10 queries. Cambridge Semantics recommends that you accept the default value or set the number of queries to a value between 10 - 20.
AnzoGraph connection timeout (seconds)	This setting controls how often (in seconds) Anzo checks the status of the connection to AnzoGraph. Every *N* seconds (where *N* is the value for this setting), Anzo tests the connection. If the test fails, Anzo re-tests the connection every 15 seconds for 2 minutes to rule out a brief network glitch. If the connection continues to fail after 2 minutes, AnzoGraph's status changes to "Offline." If the connection is re-established within the 2-minute window, Anzo determines whether the connection came back automatically or whether AnzoGraph was restarted.

Setting	Description
Use AnzoGraph persistence if available	This setting indicates whether Anzo should useAnzoGraph's persist data to disk option if it is enabled in AnzoGraph. This setting is enabled by default but persistence is disabled in AnzoGraph by default. For more information, see Enabling AnzoGraph Persistence.
Deploy Graphmart data during AnzoGraph startup	This option is selected by default and means that Anzo redeploys your graphmarts to AnzoGraph if they were loaded at the time AnzoGraph was shut down. If you deselect this option, Anzo assumes that the data already exists in AnzoGraph and does not attempt to reload graphmarts when AnzoGraph restarts. The graphmarts remain "online" in Anzo but are not reloaded to AnzoGraph. Typically users accept the default value and let Anzo redeploy graphmarts to AnzoGraph automatically.
Reload previously deployed Graphmarts on startup	This option is selected by default and means that Anzo automatically activates any graphmarts in Anzo that were online at the time Anzo was shut down. If you deselect this option, Anzo does not automatically reload the graphmart structures into Anzo and they remain "offline" when Anzo is restarted. Typically users accept the default value and let Anzo redeploy graphmarts to Anzo automatically.
Port	The port to use for communication between AnzoGraph and Anzo. The default value is **5700**, the Anzo protocol (gRPC) port for secure communication. Do not change the value unless instructed by Cambridge Semantics Support.
AnzoGraph Management Port	The SSL system management port for AnzoGraph. The default value is **5600**. Do not change the value unless instructed by Cambridge Semantics Support.
Callback Hostname	The Callback Hostname is the Anzo server to use when AnzoGraph makes service callbacks. If you have multiple Anzo servers and one or more of them are not routable by the AnzoGraph server, the Callback Hostname is the Anzo host that AnzoGraph can target when making service calls.
Readonly Replica	This option is for use if you have multiple Anzo servers, and only one of those servers loads graphmarts to AnzoGraph. When Is Replica is selected, Anzo treats this AnzoGraph as a read-only source so that this Anzo server can view the data in AnzoGraph but cannot change it.

Setting	Description
Disable Vacuum	This option controls whether Anzo initiates an AnzoGraph vacuum process after each data load. The vacuum process improves data organization in memory, deduplicates data, and reclaims memory after data is deleted. Completing a vacuum after update operations is extremely important for maintaining overall query performance and memory allocation accuracy. Do not disable vacuum by selecting the Disable Vacuum checkbox unless you are instructed to do so by Cambridge Semantics Support.
Disable Gather Statistics	This option controls whether Anzo initiates AnzoGraph's internal statistics gathering queries immediately after loading data. When this option is enabled (the checkbox is clear), the AnzoGraph statistics queries are run immediately after a graphmart is loaded. It increases graphmart load time but reduces execution time for the first analytic queries, such as when a Hi-Res Analytic dashboard is created. When this option is disabled (the checkbox is selected), AnzoGraph automatically performs statistics gathering when the first queries are run, increasing the execution time for the first queries. **Note** Cambridge Semantics recommends that you leave Gather Statistics enabled so that AnzoGraph gathers statistics at the end of a load rather than during query execution. Since loads take longer than queries, adding more time to the load is less noticeable than waiting for statistics to be generated during initial query execution.
Use Futures Based Query Manager	This option controls whether Anzo provides a view of the queries that are in the queue waiting to be run. The queued queries are displayed in the System Query Audit log. **Note** Enabling or disabling this option after saving the initial configuration requires a restart of Anzo.

Setting	Description
Enable Detailed Query Timing	When the Futures Based Query Manager is enabled, this option controls whether Anzo obtains detailed timing statistics for every AnzoGraph query. If **Enable Detailed Query Timing** is enabled, Anzo sends additional statistics gathering queries to AnzoGraph for each user query. The extra query timing details, such as query compilation time, compilation statistics, and a query summary, are displayed in the System Query Audit log. **Important** Enabling detailed query timing increases the AnzoGraph workload and may decrease overall query performance.

12. Click **Save** to save the configuration. Anzo connects to AnzoGraph and returns to the Connection Overview screen. Click the new AnzoGraph connection to view the connection status as well as memory usage details, overall data statistics, and graphmart details. For example:



To change configuration details, click the **Configuration** tab and adjust values as needed. For instructions on loading data to AnzoGraph, see Creating a Graphmart.

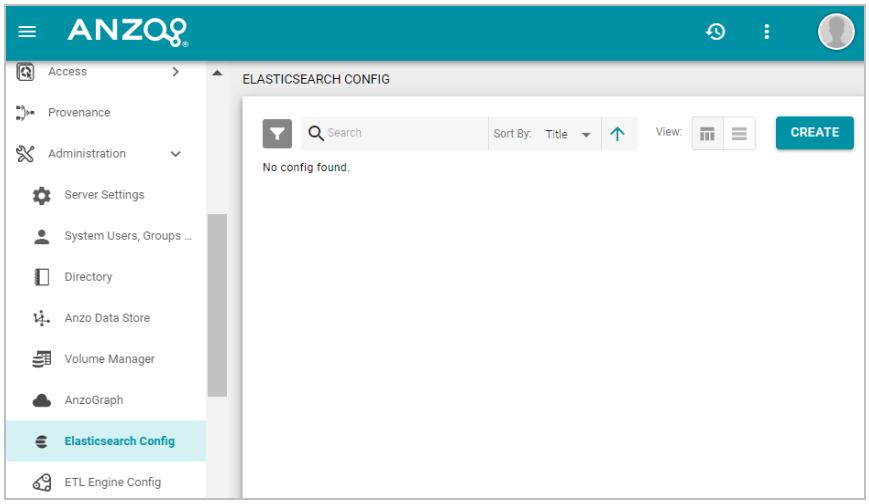
Related Topics

AnzoGraph Server Administration

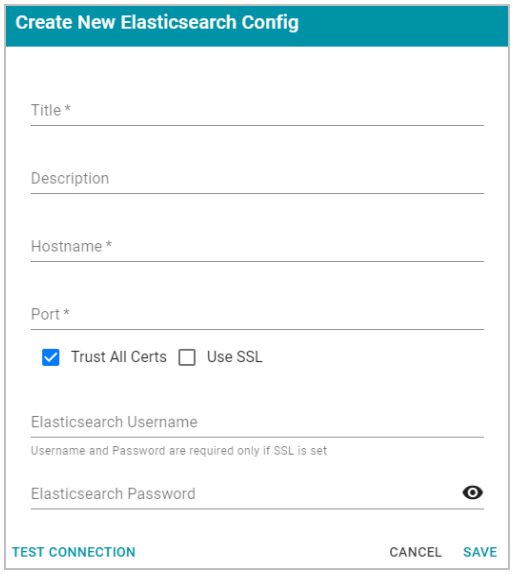
Connecting to Elasticsearch

This topic provides instructions for configuring a connection to an Elasticsearch instance in the Anzo console. For information about installing Elasticsearch, see Installing and Configuring Elasticsearch.

1. In the Anzo console, expand the **Administration** menu and click **Elasticsearch Config**. Anzo displays the Elasticsearch Config screen, which lists any existing Elasticsearch connections. For example:



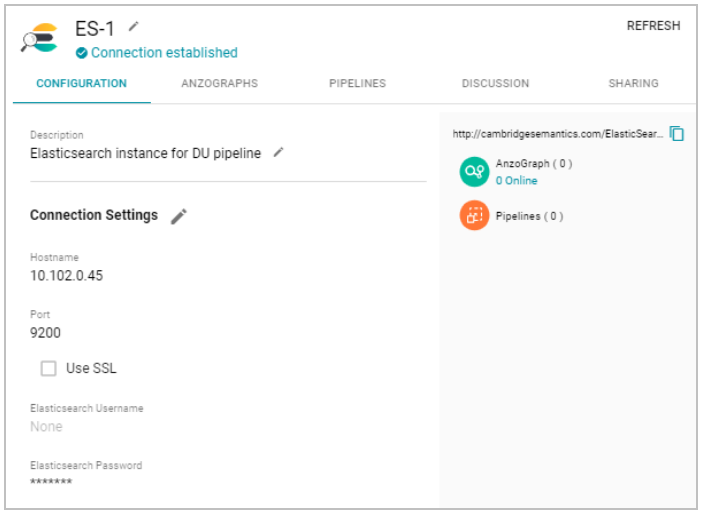
2. On the Elasticsearch Config screen, click the **Create** button. Anzo opens the Create New Elasticsearch Config dialog box.



3. On the Create New Elasticsearch Config screen, provide the following details about the Elasticsearch instance:
   - **Title**: Type a name for this Elasticsearch connection.
   - **Description**: Optional description for this connection.
   - **Hostname**: Specify the IP address or hostname of the Elasticsearch server.
   - **Port**: Specify the port to use for the Elasticsearch connection. The default Elasticsearch port is **9200**.
   - **Trust All Certs**: Indicates whether Anzo should trust the Elasticsearch certificates for this connection. Cambridge Semantics recommends that you accept the default value of enabled.

- **Use SSL**: If this Elasticsearch instance is configured for SSL authentication, select the **Use SSL** check-box.
- **Elasticsearch Username**: If Use SSL is specified, type the user name to use to connect to Elastic-search.
- **Elasticsearch Password**: If Use SSL is specified, type the password for the user name that you spe-cified.

4. Click **Test Connection** to check if Anzo can connect to Elasticsearch. If the connection fails, make sure that Elasticsearch is running and that you entered the correct connection details.
5. Anzo displays a Connection Successful dialog box. Click **OK** to close the dialog, and then click **Save** to save the new connection. Anzo saves the connection and displays the Configuration overview screen. For example:



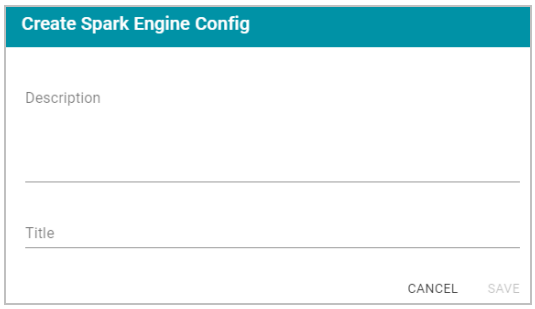
You can adjust configuration details as needed. For instructions on creating an unstructured pipeline, see Onboarding Unstructured Data.

Connecting to an ETL Engine

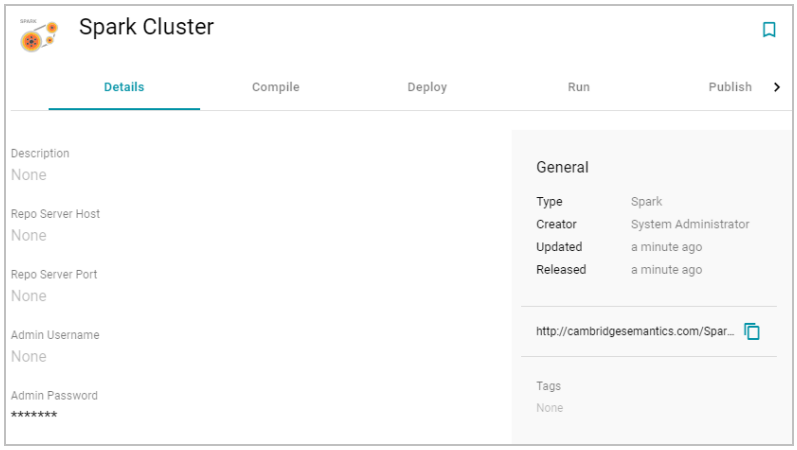
The default Anzo installation includes a pre-configured local Spark ETL engine and Sparkler ETL compiler. Sparkler is Cambridge Semantics' SPARQL-driven ETL compiler that supports the ingestion of wide CSV files with a large number of columns and increases performance over the Spark Scala-based compiler in many cases. The topics in this section provide instructions for changing the configuration of the local engines or connecting to an alternate Spark ETL engine or Sparkler compiler.

Note Before you can configure Informatica or Pentaho ETL engines, you must import the appropriate Cambridge Semantics bundle. Contact Cambridge Semantics Support to obtain the bundles.

1. In the Anzo console, expand the **Administration** menu and click **ETL Engine Config**.
2. On the ETL Engine Config screen, click the **Create** button and select the type of ETL engine that you want to configure. Anzo displays the Create ETL Engine Config screen. For example:



3. On the Create ETL Engine Config screen, type a **Title** and optional **Description** for the engine. Then click **Save**. Anzo displays the Details view for the new engine. For example:



4. The options that appear depend on the type of ETL engine you chose. Enter the required details for the engine. Hover the pointer over an options and click the Edit icon ( ) to modify any of the options. Click the check mark icon ( ) to save changes to an option, or click the X icon ( ) to clear the value for an option.

Spark

- **Repo Server Host**: Leave this field blank.
- **Repo Server Port**: Leave this field blank.
- **Admin Username**: Not currently used.
- **Admin Password**: Not currently used.
- **Compile** tab: The Compile settings control where Anzo saves the compiled Scala .jar files for the Spark job.
    - **Remote Server**: The host name or IP address of the server where the compilation will be performed.
    - **Target Folder**: The path and directory on the server where Anzo can stage temporary artifacts created during the compilation and upload process. The location must be a valid path on the Anzo server that the user running the ETL job has access to.

- **Deploy** tab: The Deploy step is performed after the job is compiled locally and before the job is submitted to Spark. The Deploy settings control how and where the job's .jar files will be copied from the Anzo server to a file system that Spark can access.
    - **Deployment Working Dir**: The directory that the Anzo server should use when executing the deploy commands.
    - **Deploy Command**: The command line script that the deploy step should run.
- **Run** tab:
    - **Job Runner Endpoint**: The HTTP endpoint used to reach the Livy server. For example, when using the local Anzo Spark engine, the endpoint is localhost:8998.
    - **SDI Jobs Dir**: The file system location where the Spark engine will look for the compiled .jar files.
    - **SDI Dependencies Dir**: The file system location where the Spark engine will look for the dependency .jar files, **sdi-full-deps.jar** and **sdi-deps.jar**. These files can be copied to the Spark master node from the `installation_path/Anzo/Server/data/sdiScripts/Spark_version/compile/dependencies-lib` directory on the Anzo server.
    - **Additional Jars**: For relational database sources, the file system location for the JDBC driver .jar file or files that are used to connect to the databases. For multiple jar files, specify a comma-separated list. Do not include a space after the commas.
    - **Execute Locally**: Select this option for local Spark engines on the Anzo server. Make sure this option is not selected when using a remote Spark server.
    - **Do Callback**: Select this option when you want Anzo to create a new data set in the Dataset catalog and generate load files for the graph source.
    - **Run with Yarn**: Employs the Spark YARN cluster manager when running ETL jobs.
    - **Callback URL**: When **Do Callback** is selected, enter one of the following URLs:

      ```
      http://Anzo_server_hostname_or_IP:Anzo_application_HTTP_port/anzoclient/call
      ```

      ```
      https://Anzo_server_hostname_or_IP:Anzo_application_HTTPS_port/anzoclient/call
      ```

- **Publish** tab: This tab controls the action of the **Publish All** button when running jobs.
- **Sharing** tab: Enables you to share or restrict access to this ETL engine.

Sparkler

- **Description**: Description of this engine.
- **Run** tab:
    - **Remote Server Name**: The host name or IP address of the server where the compilation will be performed.

- **Job Runner Endpoint**: The HTTP endpoint used to reach the Livy server. For example, when using the local Anzo Sparkler engine, the endpoint is localhost:8998.
- **Target Folder Name**: The path and directory on the host where temporary artifacts can be created during the compilation and upload process. The location must be a valid path on the server that the user running the ETL job has access to.
- **Sparkler Home**: The path and directory where the Sparkler compiler is installed on the host server.
- **SDI Dependencies Dir**: The file system location where the Sparkler engine will look for the dependency .jar files, **sdi-full-deps.jar** and **sdi-deps.jar**. These files can be copied to the Sparkler host server from the `install_path/Anzo/Server/data/sdiScripts/Spark_version/compile/dependencies-lib` directory on the Anzo server.
- **Additional Jars**: For relational database sources, the file system location for the JDBC driver .jar file or files that are used to connect to the databases. For multiple jar files, specify a comma-separated list. Do not include a space after the commas.
- **Execute Locally**: Select this option for local Sparkler engines on the Anzo server. Make sure this option is not selected when using a remote Sparkler server.
- **Do Callback**: Select this option when you want Anzo to create a new data set in the Dataset catalog and generate load files for the graph source.
- **Run with Yarn**: Employs the Spark YARN cluster manager when running ETL jobs.
- **Callback URL**: When **Do Callback** is selected, enter one of the following URLs:

```
http://Anzo_server_hostname_or_IP:Anzo_application_HTTP_
port/anzoclient/call
```

```
https://Anzo_server_hostname_or_IP:Anzo_application_HTTPS_
port/anzoclient/call
```

- **Advanced** tab: The options on this tab enable users with advanced Spark expertise to customize the values that are passed to Spark.
  - **Enable CSV Error Reporting**: Controls whether detailed CSV errors are displayed in the Anzo user interface.
  - **Run As User**: Specifies the user to impersonate when starting the Livy session.
  - **Max Graph Output File Size Default (Bytes)**: The maximum number of bytes to limit graph output files to.
  - **Max Input File Partition Size (Bytes)**: The maximum number of bytes to pack into a partition when reading files. Maps to the `spark.files.maxPartitionBytes` Spark configuration setting.

- **Spark Job Driver Cores**: The number of cores to use for the driver process. Maps to the `spark.driver.cores` Spark configuration setting.
- **Spark Job Driver Memory**: The amount of memory to use for the driver process. Maps to the `spark.driver.memory` Spark configuration setting.
- **Number of Executors Per Spark Job**: The number of executors to request per Spark job. Maps to the `spark.executor.instances` Spark configuration setting.
- **Spark Job Cores Per Executor**: The number of cores to use on each executor. Maps to the `spark.executor.cores` Spark configuration setting.
- **Spark Job Memory Per Executor**: The amount of memory to use per executor process. Maps to the `spark.executor.memory` Spark configuration setting.
- **Off Heap Size (Bytes)**: The amount of memory in bytes that can be used for off-heap allocation. Maps to the `spark.memory.offHeap.size` Spark configuration setting.
- **Job Dependencies (Maven Package Coordinate)**: The comma-separated list of Maven jar coordinates to include on the driver and executor classpaths. Maps to the `spark.jars.packages` Spark configuration setting.
- **Maven Package Excludes**: To avoid dependency conflicts, this is the comma-separated list of **groupId:artifactId** to exclude while resolving the dependencies listed in spark.jars.packages. Maps to the `spark.jars.excludes` Spark configuration setting.
- **Maven Repositories**: A comma-separated list of additional remote repositories to search for the maven coordinates from the Job Dependencies setting. Maps to the `spark.jars.repositories` Spark configuration setting.
- **Spark Job Deploy Mode (Livy Config has Precedence)**: The deploy mode of the Spark driver program. If this value is set in the Livy configuration, the Livy value takes precedence. Maps to the `spark.submit.deployMode` Spark configuration setting.

- **Publish** tab: This tab controls the action of the **Publish All** button when running jobs.
- **Sharing** tab: Enables you to share or restrict access to this Sparkler engine.

Note By default, the Sparkler Livy RSC client is configured to use a range of 10 internal ports for running jobs. Cambridge Semantics recommends that you increase the range of ports by 100. To do so, modify the **livy.rsc.launcher.port.range** value in the **livy-client.conf** file. If you use the embedded Anzo Sparkler compiler, the file is in the `install_path/Server/spark/csi-livy-spark/conf` directory. Set **livy.rsc.launcher.port.range = 10000~10110** and restart the Livy server.

When the configuration is complete, Anzo provides this ETL engine as a choice to select when ingesting data and configuring pipelines.

Managing Cloud Locations

This page is in progress.

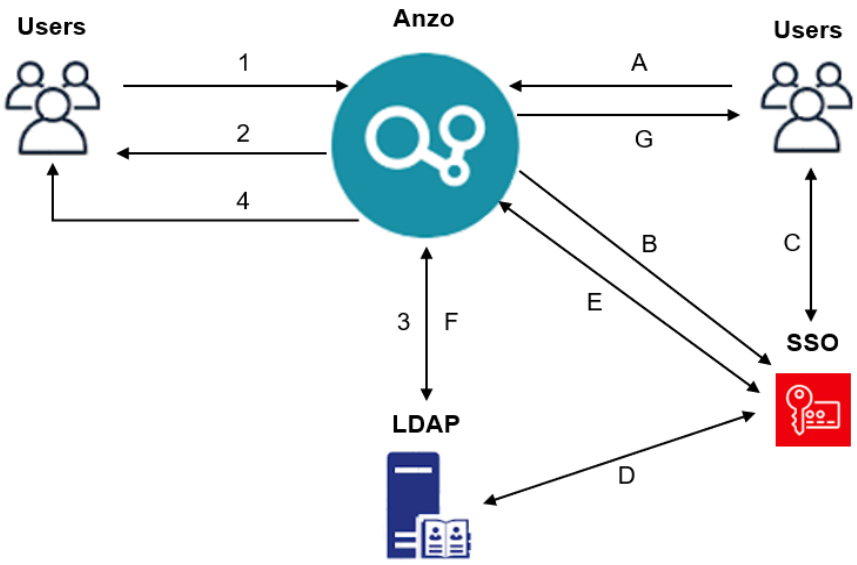
User Administration

Anzo offers granular access control as well as role-based security. The topics in this section provide information about the predefined Anzo roles and permissions and include instructions for connecting to an external directory server, creating new roles and users in Anzo, adding users to roles, and changing role permissions.

Leveraging an LDAP Server

This topic provides an overview of the ways Anzo can leverage a central directory server (LDAP).

There are two methods by which Anzo can leverage your LDAP server: Direct Authorization and Single Sign-On (SSO). The diagram below shows the procedures that are followed for both methods. The left side of the diagram (the numbered steps) shows the direct authorization method. The right side of the diagram (the lettered steps) shows the SSO method. The table below the diagram describes the processes for each method.



Direct Authorization	Single Sign-On
1. A new (unknown) user navigates to the Anzo application. 2. Anzo redirects the user to a login form. The user supplies credentials and submits the form. 3. Anzo queries the LDAP for the user and group membership. 4. Anzo redirects the user to the application with the appropriate roles applied.	A. A new (unknown) user navigates to the Anzo application. B. Anzo redirects the user to the SSO provider. The SSO provider controls authentication validation. C. Depending on the policy, the SSO provider presents a login in screen for the user to complete and submit. D. As needed, the SSO provider validates the credentials with the LDAP server. E. The SSO provider authenticates the Anzo session with a callback. F. Anzo fetches group information from the LDAP server. **Note** For SSO-configured systems, Anzo currently requires direct access to the LDAP directory (and a bind user) to look up groups. G. Anzo redirects the user to the application with the appropriate roles applied.

For information about connecting to a central LDAP server, see Connecting to a Directory Server. And for information about setting up an SSO provider, see Configuring SSO Access.

Related Topics

Connecting to a Directory Server

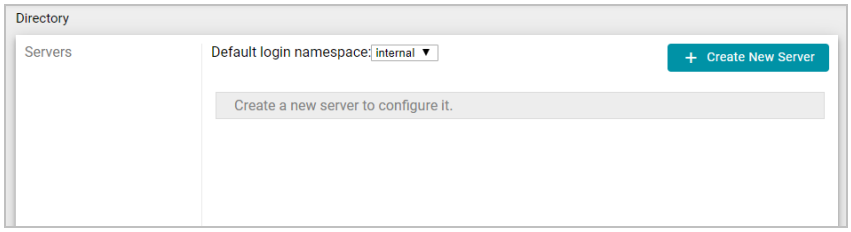
Configuring SSO Access

Connecting to a Directory Server

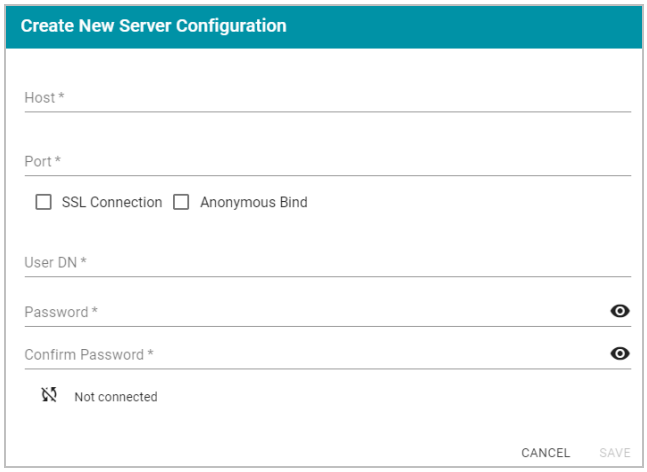
This section provides instructions for connecting to an external LDAP or Active Directory server. When you connect Anzo to a directory server, Anzo inherits the user account information and groups from that server. Once the user and group information becomes available in Anzo, you must add the users or groups to Anzo roles to enable and control access to Anzo features.

To connect to a directory server and map user and role data:

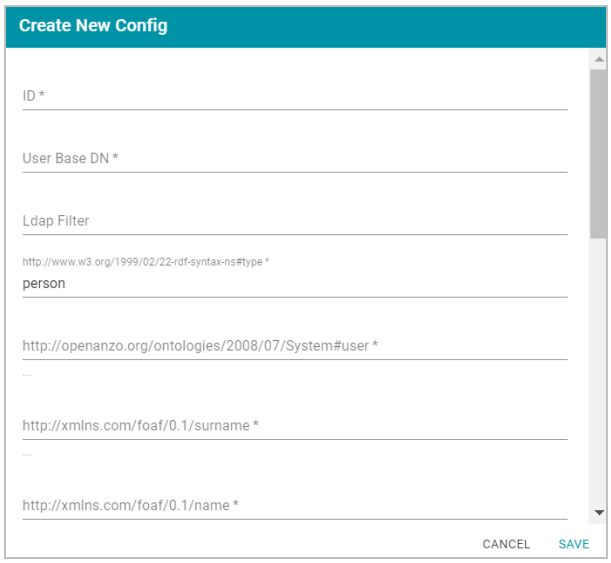
1. In the Anzo console, expand the **Administration** menu and click **Directory**. Anzo displays the Directory screen.



2. On the Directory screen, click the **Create New Server** button. Anzo displays the Create New Server Configuration screen.



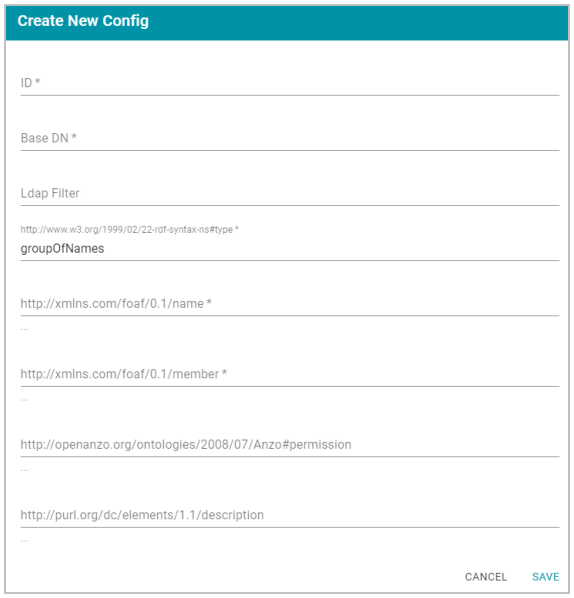
3. Enter the connection details for the LDAP or Active Directory server:
    - **Host**: The host name or IP address for the server.
    - **Port**: The port to use to connect to the directory server.
    - **SSL Connection**: Indicates whether the directory server uses an SSL connection. Select the SSL Connection checkbox to enable the SSL connection. Make sure that you load the directory server's certificate to the Anzo server by following the instructions in Uploading a Signed Certificate to Anzo.
    - **Anonymous Bind**: This option indicates whether you want Anzo to connect to the directory server anonymously. To avoid Anzo login problems, make sure the directory server allows anonymous binding and searches when bound anonymously. Select the Anonymous Bind checkbox to enable anonymous binding.
    - **User DN**: The full distinguished name of the account that Anzo will bind against to perform searches on the directory server.
    - **Password** and **Confirm Password**: The password for the User DN.
4. Click **Save** to save the server configuration and return to the Directory screen.
5. On the Directory screen, click the **User Configs** tab. To configure the mapping attributes for users, click the **Create New User Config** button. Anzo displays the Create New Config dialog box.



6. Provide the following details to map user information:
    - **ID**: Defines the unique name for this user configuration. Anzo uses this value as a namespace for user-names in case you connect to multiple directories with conflicting names.
    - **User Base DN**: The LDAP distinguished name that contains all of the Anzo system users.
    - **LDAP Filter**: An optional LDAP filter to apply when searching for users (usually left blank).
    - Attribute mapping information:
        - **http://www.w3.org/1999/02/22-rdf-syntax-ns#type**: The LDAP class of the type of accounts that should be logged on. Typically **person**.
        - **http://openanzo.org/ontologies/2008/07/System#user**: The attribute that contains user login information. Typically **uid**.
        - **http://xmlns.com/foaf/0.1/name**: The LDAP attribute that contains users' full names. Typically **cn**.
        - **http://xmlns.com/foaf/0.1/title**: The LDAP attribute that contains users' job titles. Typically **title**.
        - **hhttp://xmlns.com/foaf/0.1/surname**: The LDAP attribute that contains users' surnames. Typically **sn**.
        - **http://xmlns.com/foaf/0.1/phone**: The LDAP attribute that contains user phone numbers. Typically **telephoneNumber**.
        - **http://xmlns.com/foaf/0.1/mbox**: The LDAP attribute that contains users' email addresses. Typically **mail**.
        - **http://openanzo.org/ontologies/2008/07/Anzo#location**: The LDAP attribute that contains user location information.
        - **http://xmlns.com/foaf/0.1/img**:The LDAP attribute that contains images for users.
        - **http://xmlns.com/foaf/0.1/givenname**: The LDAP attribute that contains users' given (first) names. Typically **givenName**.

- **http://purl.org/dc/elements/1.1/description**: The LDAP attribute that contains user descriptions. Typically **description**.
- **http://openanzo.org/ontologies/2008/07/Anzo#companyDepartment**: The LDAP attribute that contains user department information. Typically **department**.

7. Click **Save** to save the user configuration and return to the Directory screen.
8. On the Directory screen, click the **Role Configs** tab. To configure the mapping information for roles, click the **Create New Role Config** button. Anzo displays the Create New Config dialog box.



9. Provide the following details to map role information:
   - **ID**: Defines the unique name for this role configuration.
   - **Base DN**: The LDAP distinguished name that contains all of the system roles.
   - **LDAP Filter**: An optional LDAP filter to apply when searching for roles (usually left blank).
   - Attribute mapping information for the role:
     - **http://www.w3.org/1999/02/22-rdf-syntax-ns#type**: The group object class of the type of roles. Typically **groupofnames**.
     - **http://xmlns.com/foaf/0.1/name**: The LDAP attribute that contains the names of the roles.
     - **http://xmlns.com/foaf/0.1/member**: The LDAP attribute that contains common member attributes. Typically **member** or **uniqueMember**.
     - **http://openanzo.org/ontologies/2008/07/Anzo#permission**: The LDAP attribute that contains the permissions for the roles.
     - **http://purl.org/dc/elements/1.1/description**: The LDAP attribute that contains role descriptions.
10. Click **Save** to save the role configuration and return to the Directory screen.

11. At the top of the Directory screen, click the **Default login namespace** drop-down and choose the user configuration that you would like to designate as the default namespace to look in if the user does not qualify their username with an @ suffix. **Internal** indicates internal Anzo users who are added via the Anzo console. The other option is the ID that you specified when configured the new user.
12. Restart Anzo to complete the configuration.

The user and group information from the server becomes available in Anzo. To enable and control access to Anzo features, add the users or groups to Anzo roles. See Managing Roles for instructions.

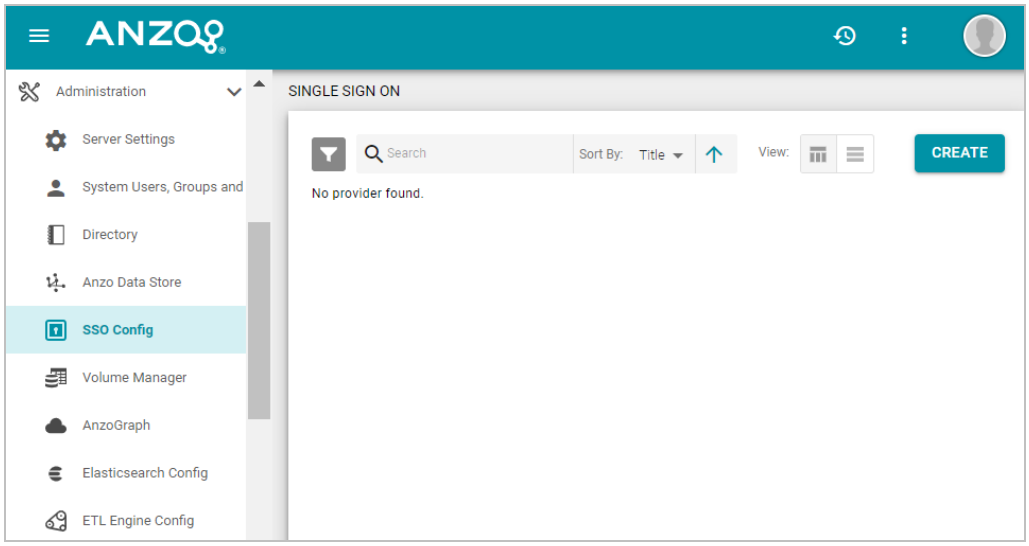
Configuring SSO Access

This topic provides instructions for configuring Anzo to enable single sign-on (SSO) access using one of the following SSO providers:

- Direct and Indirect Basic
- Direct and Indirect Kerberos
- Facebook
- JSON Web Tokens (JWT) Header and Parameter
- OpenID Connect (OIDC)
- Security Assertion Markup Language (SAML)
- Google OpenID Connect (OIDC)

Follow the instructions below to configure a provider.

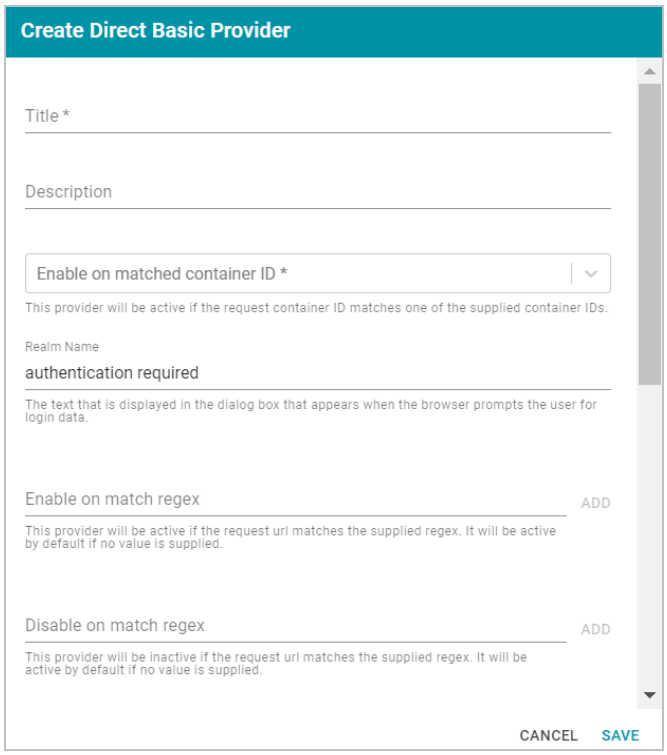
1. In the Anzo console, expand the **Administration** menu and click **SSO Config**. Anzo displays the Single Sign On screen, which lists any existing SSO providers. For example:



2. To add a provider, click the **Create** button and select the type of provider to configure. Anzo opens the Create dialog box for that provider. Complete the required fields and supply any of the relevant optional information. The list below provides details about the properties for each provider.

Direct Basic Provider

This section describes the settings that are available on the Create Direct Basic Provider screen:

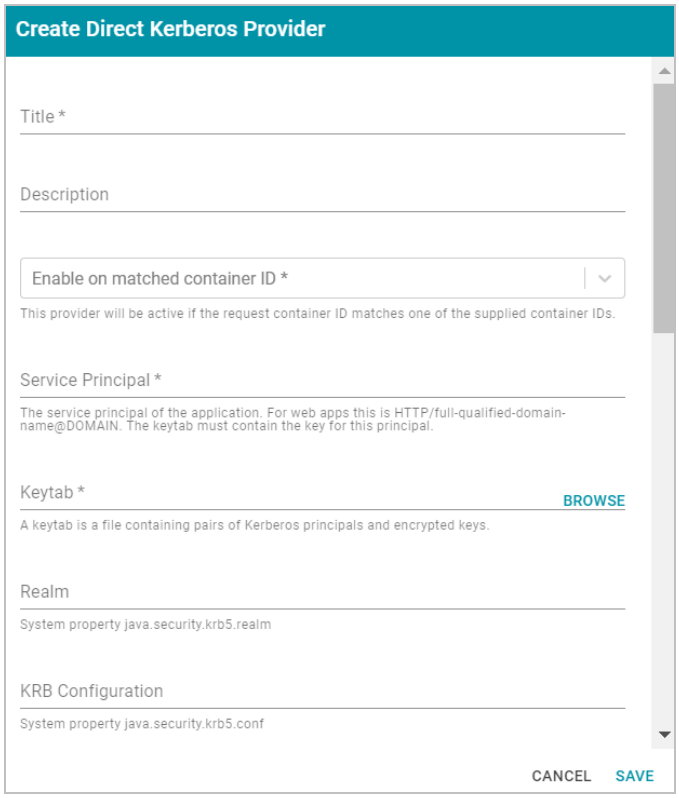


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Realm Name**: Optional field that specifies the name of the security realm.
- **Enable on match regex**: Optional field that defines regular expression rules for matching request URLs to enable. To add a rule, type an expression in the field and click **Add**. This provider will be active if the request URL matches any of the supplied expressions. If Enable on match regex is blank, the provider will be active by default.

- **Disable on match regex**: Optional field that defines regular expression rules for matching request URLs to disable. To add a rule, type an expression in the field and click **Add**. This provider will be inactive if the request URL matches any of the supplied expressions. If Disable on match regex is blank, the provider will be active by default.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://openanzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.

Direct Kerberos Provider

This section describes the settings that are available on the Create Direct Kerberos Provider screen:



- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Service Principal**: **Required** field that specifies the service and DNS name for the application. For authentication through the web browser, specify the service principal value in the following format:

  ```
  HTTP/fully_qualified_domain_name@domain
  ```

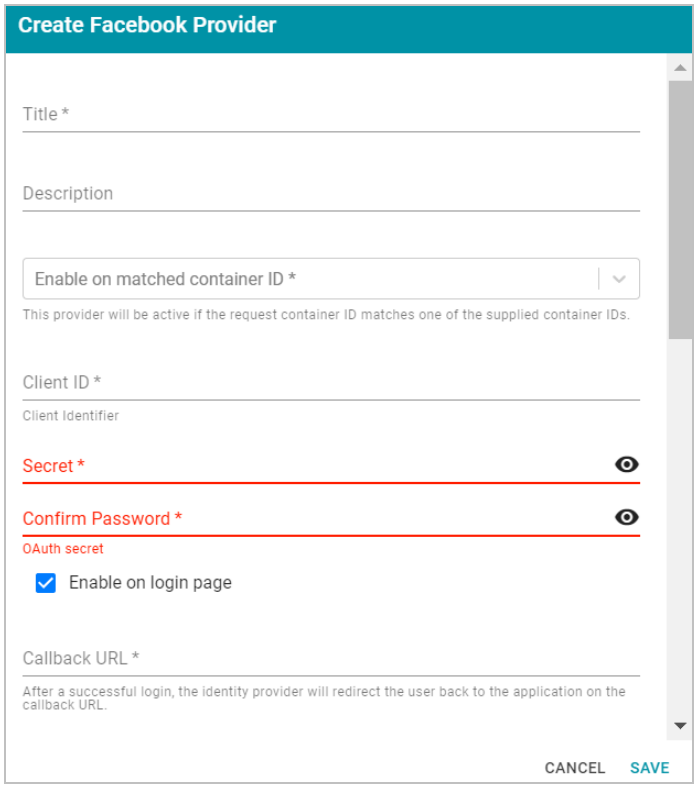
  For example, `HTTP/server.example.com@example.com`.

  Note: The keytab file must contain the key for this principal.
- **Keytab**: **Required** field that specifies the .keytab file that lists the Kerberos principals and encrypted keys. Click the **Keytab** field to open the File Location dialog box and select the keytab file.
- **Realm**: Optional field that specifies the Kerberos realm that the service principal maps to.

- **KRB Configuration**: Optional field that specifies the path and file name for the krb5.conf file on the Kerberos instance. The default location is `/etc/krb5.conf`.
- **KDC**: Optional field that specifies the domain name for the Key Distribution Center.
- **Debug mode**: Optional field that specifies whether Kerberos debug logging is enabled for your provider.
- **Enable on match regex**: Optional field that defines regular expression rules for matching request URLs to enable. To add a rule, type an expression in the field and click **Add**. This provider will be active if the request URL matches any of the supplied expressions. If Enable on match regex is blank, the provider will be active by default.
- **Disable on match regex**: Optional field that defines regular expression rules for matching request URLs to disable. To add a rule, type an expression in the field and click **Add**. This provider will be inactive if the request URL matches any of the supplied expressions. If Disable on match regex is blank, the provider will be active by default.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.

Facebook Provider

This section describes the settings that are available on the Create Facebook Provider screen:

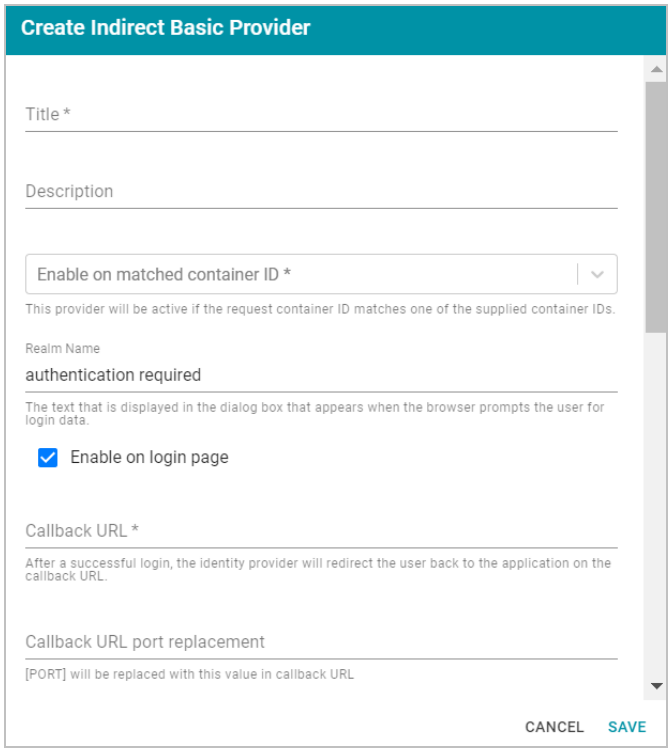


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Client ID**: **Required** field that specifies the unique App ID for the client application.
- **Secret**: **Required** field that specifies the App Secret for the specified Client ID.
- **Confirm Password**: **Required** field that confirms the specified Secret.
- **Enable on login page**: Optional field that specifies whether to enable a link for this provider on the Anzo login screen.
- **Callback URL**: **Required** field that specifies the URL for the provider to use to redirect users back to the Anzo application after a successful login.
- **Callback URL port replacement**: Optional field that specifies the port to use for the Callback URL.
- **User Identifier**: Optional field that specifies the SSO provider attribute, such as email or username, to use for looking up users in the directory server.

- **Logout of IDP**: Optional field that specifies whether logging out of Anzo also logs out of the identity provider session. When this option is enabled, users are prompted to log out of the SSO session when they log out of the Anzo application.
- **Logout URL Suffix**: When Logout of IDP is enabled, the Logout URL Suffix is used to access the logout URL for the SSO provider. The [urlAfterLogout] placeholder is replaced with the SSO provider server URL.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.
- **Icon**: Optional property that specifies an SSO icon to use on the Anzo login screen. To select an image file, click the **Icon** field and select **Add File**.

Indirect Basic Provider

This section describes the settings that are available on the Create Indirect Basic Provider screen:

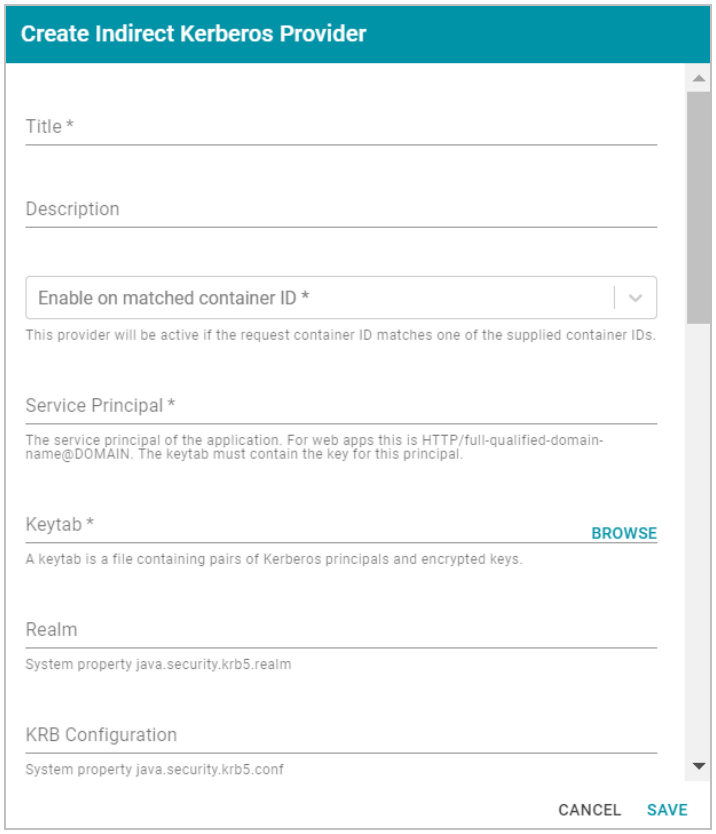


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Realm Name**: Optional field that specifies the name of the security realm.
- **Enable on login page**: Optional field that specifies whether to enable a link for this provider on the Anzo login screen.
- **Callback URL**: **Required** field that specifies the URL for the provider to use to redirect users back to the Anzo application after a successful login.
- **Callback URL port replacement**: Optional field that specifies the port to use for the Callback URL.
- **User Identifier**: Optional field that specifies the SSO provider attribute, such as email or username, to use for looking up users in the directory server.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.

- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.
- **Icon**: Optional property that specifies an SSO icon to use on the Anzo login screen. To select an image file, click the **Icon** field and select **Add File**.

Indirect Kerberos Provider

This section describes the settings that are available on the Create Indirect Kerberos Provider screen:



- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Service Principal**: **Required** field that specifies the service and DNS name for the application. For authentication through the web browser, specify the service principal value in the following format:

  ```
  HTTP/fully_qualified_domain_name@domain
  ```

  For example, `HTTP/server.example.com@example.com`.

  Note: The keytab file must contain the key for this principal.

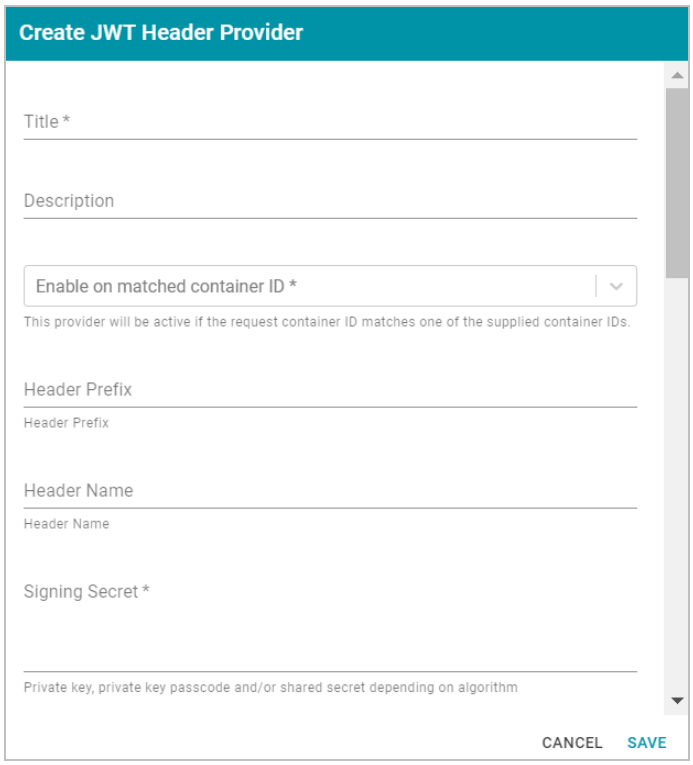
- **Keytab**: **Required** field that specifies the .keytab file that lists the Kerberos principals and encrypted keys. Click the **Keytab** field to open the File Location dialog box and select the keytab file.
- **Realm**: Optional field that specifies the Kerberos realm that the service principal maps to.

- **KRB Configuration**: Optional field that specifies the path and file name for the krb5.conf file on the Kerberos instance. The default location is `/etc/krb5.conf`.
- **KDC**: Optional field that specifies the domain name for the Key Distribution Center.
- **Debug mode**: Optional field that specifies whether Kerberos debug logging is enabled for your provider.
- **Enable on login page**: Optional field that specifies whether to enable a link for this provider on the Anzo login screen.
- **Callback URL**: **Required** field that specifies the URL for the provider to use to redirect users back to the Anzo application after a successful login.
- **Callback URL port replacement**: Optional field that specifies the port to use for the Callback URL.
- **User Identifier**: Optional field that specifies the SSO provider attribute, such as email or username, to use for looking up users in the directory server.
- **Logout of IDP**: Optional field that specifies whether logging out of Anzo also logs out of the identity provider session. When this option is enabled, users are prompted to log out of the SSO session when they log out of the Anzo application.
- **Logout URL Suffix**: When Logout of IDP is enabled, the Logout URL Suffix is used to access the logout URL for the SSO provider. The [urlAfterLogout] placeholder is replaced with the SSO provider server URL.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.

- **Icon**: Optional property that specifies an SSO icon to use on the Anzo login screen. To select an image file, click the **Icon** field and select **Add File**.

JWT Header Provider

This section describes the settings that are available on the Create JWT Header Provider screen:

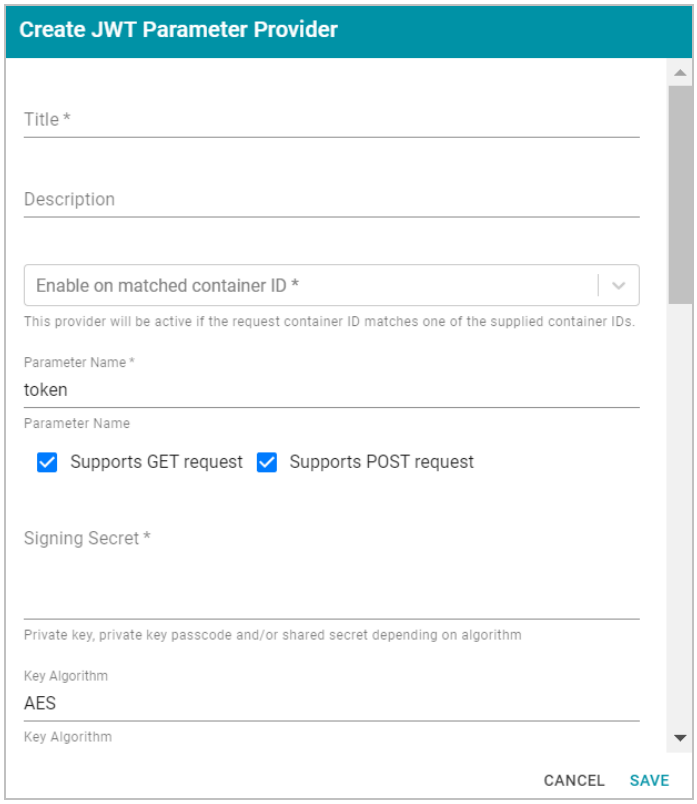


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Header Prefix**: Optional field that specifies the header prefix if one is used.
- **Header Name**: Optional field that specifies the header name.
- **Signing Secret**: **Required** field that specifies the secret the token is signed with.
- **Key Algorithm**: Optional field that specifies the signing algorithm that is used.
- **Encryption Method**: Optional field that specifies the encryption method used for encrypted tokens.
- **Encryption Secret**: Optional field that specifies the secret used for encrypted tokens.

- **Enable on match regex**: Optional field that defines regular expression rules for matching request URLs to enable. To add a rule, type an expression in the field and click **Add**. This provider will be active if the request URL matches any of the supplied expressions. If Enable on match regex is blank, the provider will be active by default.
- **Disable on match regex**: Optional field that defines regular expression rules for matching request URLs to disable. To add a rule, type an expression in the field and click **Add**. This provider will be inactive if the request URL matches any of the supplied expressions. If Disable on match regex is blank, the provider will be active by default.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.

JWT Parameter Provider

This section describes the settings that are available on the Create JWT Parameter Provider screen:

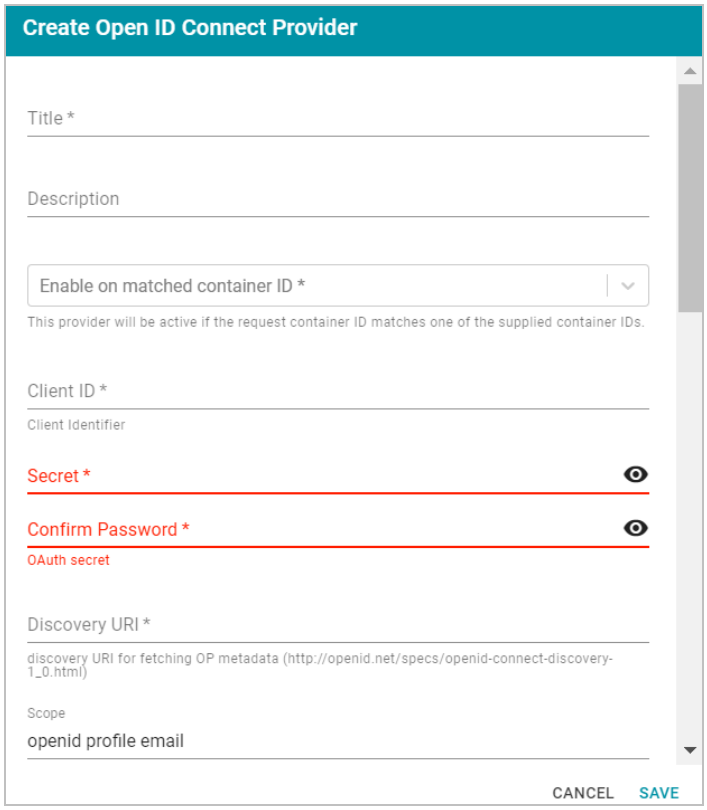


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Parameter Name**: **Required** field that specifies the header parameter name.
- **Supports GET request**: Optional field that indicates whether GET requests are supported using the token.
- **Supports POST request**: Optional field that indicates whether POST requests are supported using the token.
- **Signing Secret**: **Required** field that specifies the secret the token is signed with.
- **Key Algorithm**: Optional field that specifies the signing algorithm that is used.
- **Encryption Algorithm**:
- **Encryption Method**: Optional field that specifies the encryption method used for encrypted tokens.
- **Encryption Secret**: Optional field that specifies the secret used for encrypted tokens.

- **Enable on match regex**: Optional field that defines regular expression rules for matching request URLs to enable. To add a rule, type an expression in the field and click **Add**. This provider will be active if the request URL matches any of the supplied expressions. If Enable on match regex is blank, the provider will be active by default.
- **Disable on match regex**: Optional field that defines regular expression rules for matching request URLs to disable. To add a rule, type an expression in the field and click **Add**. This provider will be inactive if the request URL matches any of the supplied expressions. If Disable on match regex is blank, the provider will be active by default.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.

Open ID Connect Provider

This section describes the settings that are available on the Create Open ID Connect Provider screen:

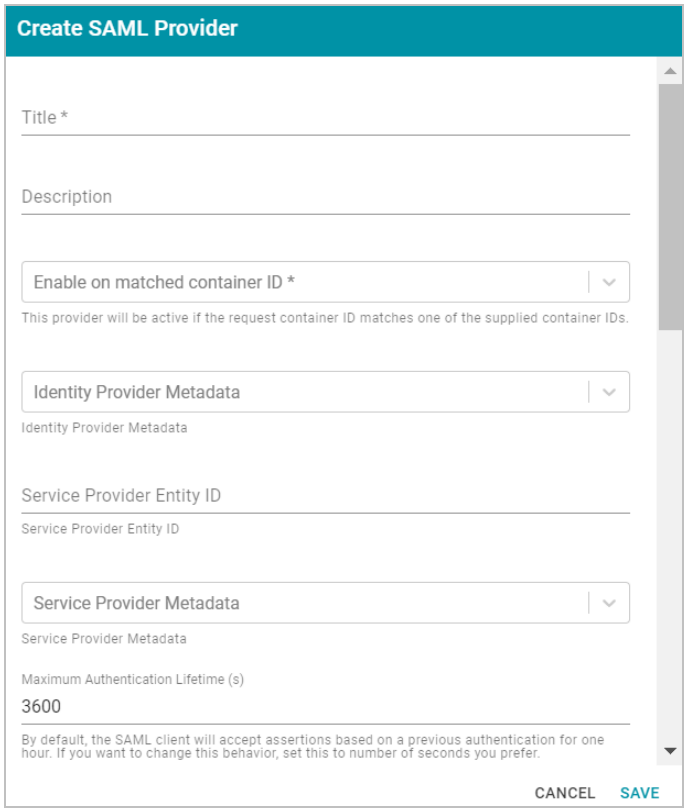


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Client ID**: **Required** field that specifies client ID or consumer key value from the provider application.
- **Secret**: **Required** field that specifies the client secret from the provider application.
- **Confirm Secret**: **Required** field to confirm the specified Secret.
- **Discovery URI**: **Required** field that specifies the discovery URI to use for fetching OP Metadata.
- **Scope**: Optional field that specifies the scope to send to the authorization endpoint with the request.
- **Preferred JWS Algorithm**: Optional field that lists the preferred signing algorithm.
- **Enable on login page**: Optional field that specifies whether to enable a link for this provider on the Anzo login screen.
- **Callback URL**: **Required** field that specifies the URL for the provider to use to redirect users back to the Anzo application after a successful login.

- **Callback URL port replacement**: Optional field that specifies the port to use for the Callback URL.
- **User Identifier**: Optional field that specifies the SSO provider attribute, such as email or username, to use for looking up users in the directory server.
- **Logout of IDP**: Optional field that specifies whether logging out of Anzo also logs out of the identity provider session. When this option is enabled, users are prompted to log out of the SSO session when they log out of the Anzo application.
- **Logout URL Suffix**: When Logout of IDP is enabled, the Logout URL Suffix is used to access the logout URL for the SSO provider. The [urlAfterLogout] placeholder is replaced with the SSO provider server URL.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.
- **Icon**: Optional property that specifies an SSO icon to use on the Anzo login screen. To select an image file, click the **Icon** field and select **Add File**.

SAML Provider

This section describes the settings that are available on the Create SAML Provider screen:

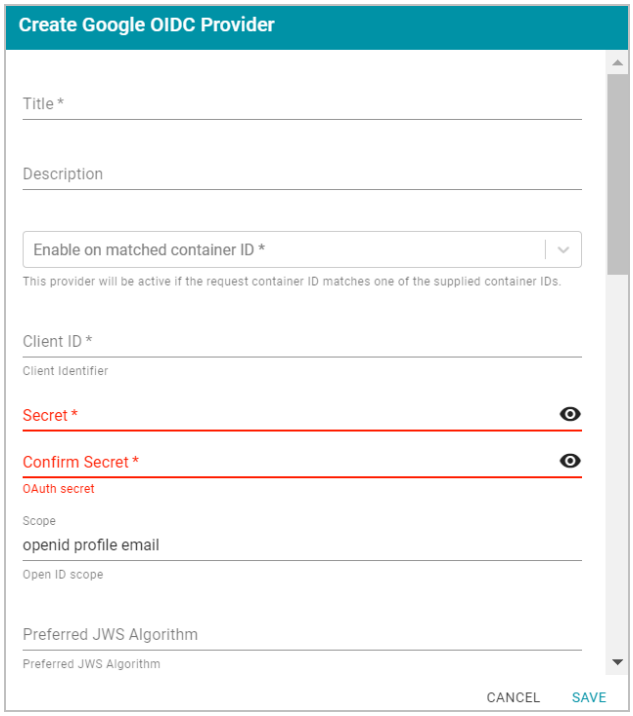


- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Identity Provider Metadata**: **Required** field that specifies the identity provider metadata .xml file. To add the file, click the **Identity Provider Metadata** field, click **Add File**, and select the file.
- **Service Provider Entity ID**:
- **Service Provider Metadata**: Optional field that specifies the server provider metadata .xml file. To add the file, click the **Server Provider Metadata** field, click **Add File**, and select the file.
- **Maximum Authentication Lifetime (s)**:
- **Enable on login page**: Optional field that specifies whether to enable a link for this provider on the Anzo login screen.
- **Callback URL**: **Required** field that specifies the URL for the provider to use to redirect users back to the Anzo application after a successful login.

- **Callback URL port replacement**: Optional field that specifies the port to use for the Callback URL.
- **User Identifier**: Optional field that specifies the SSO provider attribute, such as email or username, to use for looking up users in the directory server.
- **Logout of IDP**: Optional field that specifies whether logging out of Anzo also logs out of the identity provider session. When this option is enabled, users are prompted to log out of the SSO session when they log out of the Anzo application.
- **Logout URL Suffix**: When Logout of IDP is enabled, the Logout URL Suffix is used to access the logout URL for the SSO provider. The [urlAfterLogout] placeholder is replaced with the SSO provider server URL.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.
- **Icon**: Optional property that specifies an SSO icon to use on the Anzo login screen. To select an image file, click the **Icon** field and select **Add File**.

Google OIDC Provider

This section describes the settings that are available on the Create Google OIDC Provider screen:



- **Title**: **Required** field that specifies the name for this provider configuration.
- **Description**: Optional field that provides a description for this provider configuration.
- **Enable on matched container ID**: **Required** field that lists the container ID(s) to match. This provider will be active if the request container ID matches one of the container IDs specified in this property. Click the field and select a container ID from the drop-down list. To specify multiple IDs, click the field again and select another value. To remove a container from the list, click the X on the right of the container name.
- **Client ID**: **Required** field that specifies client ID or consumer key value from the provider application.
- **Secret**: **Required** field that specifies the client secret from the provider application.
- **Confirm Secret**: **Required** field to confirm the specified Secret.
- **Scope**: Optional field that specifies the scope to send to the authorization endpoint with the request.
- **Preferred JWS Algorithm**: Optional field that lists the preferred signing algorithm.
- **Enable on login page**: Optional field that specifies whether to enable a link for this provider on the Anzo login screen.
- **Callback URL**: **Required** field that specifies the URL for the provider to use to redirect users back to the Anzo application after a successful login.
- **Callback URL port replacement**: Optional field that specifies the port to use for the Callback URL.
- **User Identifier**: Optional field that specifies the SSO provider attribute, such as email or username, to use for looking up users in the directory server.

- **Logout of IDP**: Optional field that specifies whether logging out of Anzo also logs out of the identity provider session. When this option is enabled, users are prompted to log out of the SSO session when they log out of the Anzo application.
- **Logout URL Suffix**: When Logout of IDP is enabled, the Logout URL Suffix is used to access the logout URL for the SSO provider. The [urlAfterLogout] placeholder is replaced with the SSO provider server URL.
- **Email Template regex**: If email was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between email addresses stored by the SSO provider and email addresses returned by the directory server.
- **Email Template Replacement**: Optional field that specifies a replacement email template to use if there are variations found by Email Template regex.
- **User Template regex**: If user was specified as the User Identifier, you can use this optional field to include a regular expression to use for identifying variations between user names stored by the SSO provider and user names returned by the directory server.
- **User Template Replacement**: Optional field that specifies a replacement user template to use if there are variations found by User Template regex.
- **Use username directly**:
- **Skip CSRF check**: Optional property that specifies whether to perform a cross-site request forgery (CSRF) check.
- **LDAP domain**: Optional field that specifies the LDAP domain to use for user lookup.
- **LDAP email property**: Optional field that specifies the LDAP email property to use to find the associated user. For example, `http://open-anzo.org/ontologies/2008/07/Anzo#ldapEmailInfo`.
- **Icon**: Optional property that specifies an SSO icon to use on the Anzo login screen. To select an image file, click the **Icon** field and select **Add File**.

3. Click **Save** to save the provider configuration.

Default Anzo Roles and Permissions

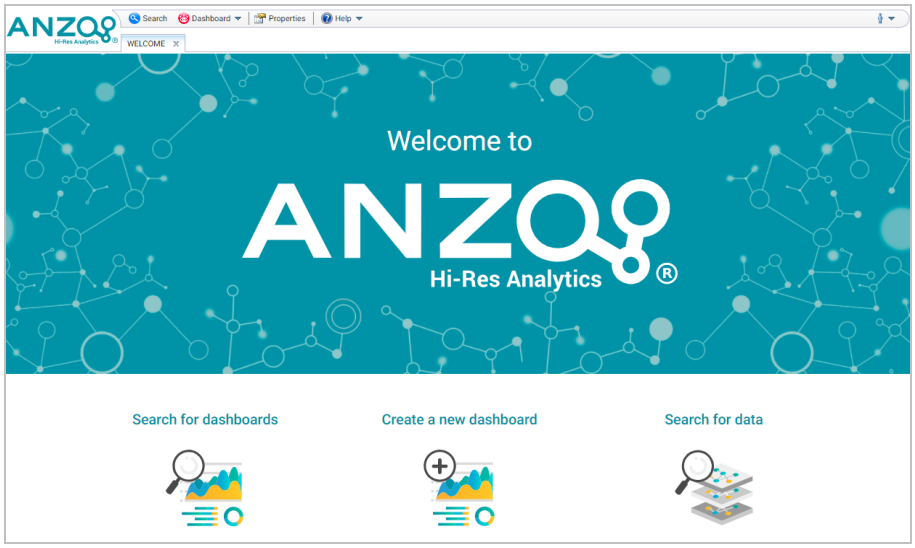
This topic describes the roles that are predefined in Anzo and lists the permissions that are assigned to each role by default. The predefined roles can be removed and modified as desired. For instructions on changing roles, see Managing Roles.

- Base Permissions
- Anzo Administrator
- Data Analyst
- Data Citizen

- Data Curator
- Data Governor
- Data Scientist

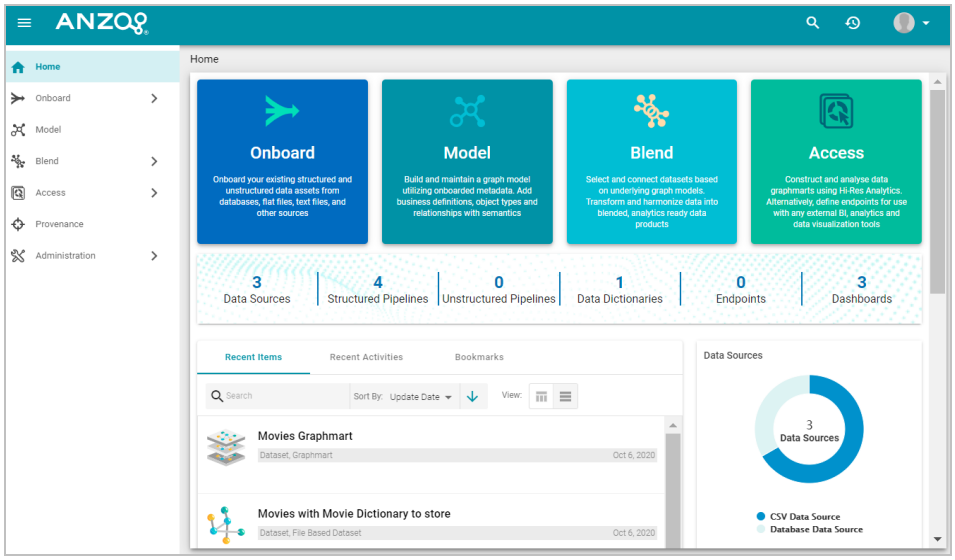
Base Permissions

There is a set of base permissions that are applied to all user accounts by default. User accounts that do not have any assigned roles cannot access the Anzo application but can access the Hi-Res Analytics application where they can browse for and create dashboards. They can also view data that is shared from Data on Demand endpoints. The image below shows an example of the view an authenticated user has in the Hi-Res Analytics application.



Anzo Administrator

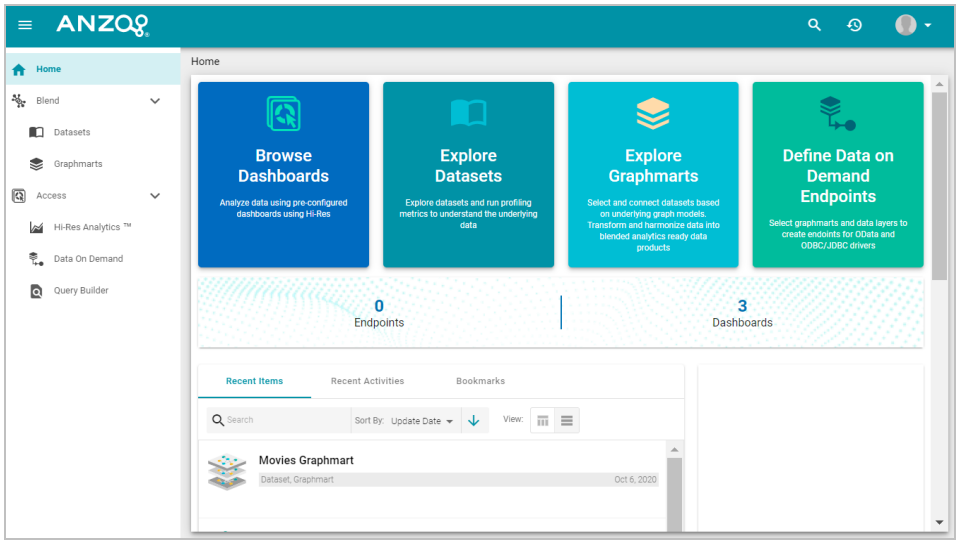
By default the Anzo Administrator role has access to all menus and features in the Anzo application as well as the Administration application. The image below shows an example of the view a user with the Anzo Administrator role has in the Anzo application.



Note Having full access to all features does not mean the Anzo Administrator has full access to all of the data in the system. Anzo Administrators must still be granted edit access to specific entities.

Data Analyst

By default the Data Analyst role has access to the Blend menu, Access menu, and Activity Log in the Anzo application. The image below shows an example of the view a user with the Data Analyst role has in the Anzo application.

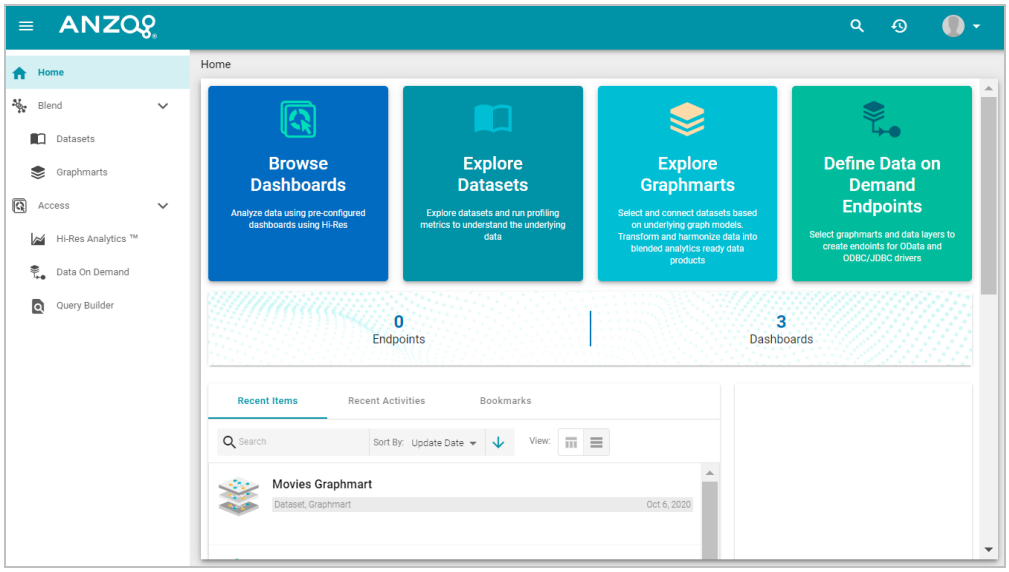


Members of the Data Analyst role can:

- View the Dataset catalog
- View and create graphmarts
- View and create Hi-Res Analytics dashboards
- View the Activity Log
- Access data with the Query Builder
- Create and access Data on Demand endpoints

Data Citizen

By default the Data Citizen role has access to the Blend menu, Access menu, and Activity Log in the Anzo application. The image below shows an example of the view a user with the Data Citizen role has in the Anzo application.

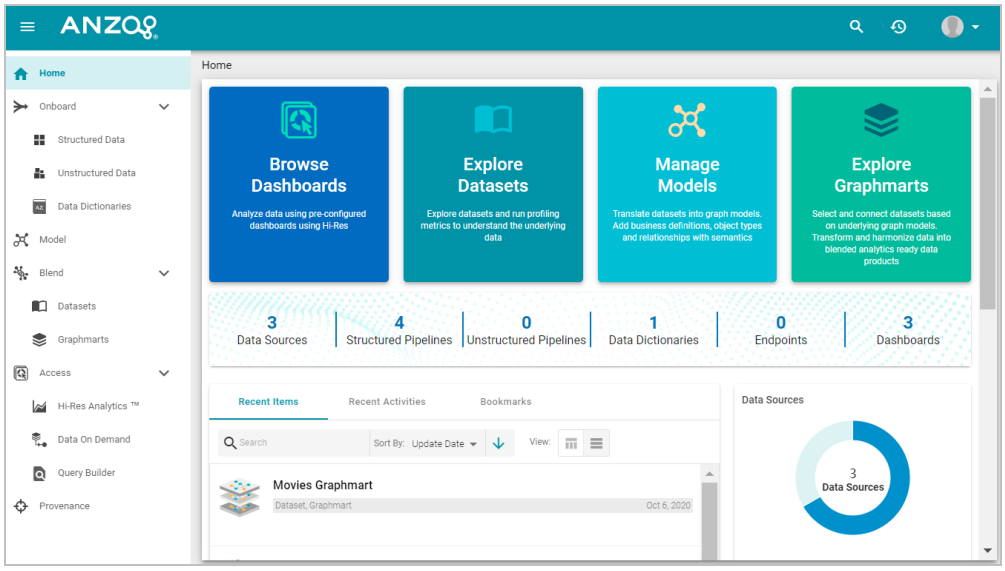


Members of the Data Citizen role can:

- View the Dataset catalog
- View graphmarts
- View and create Hi-Res Analytics dashboards
- View the Activity Log
- Access data with the Query Builder
- Create and access Data on Demand endpoints

Data Curator

By default the Data Curator role has access to the Onboard menu, Model manager, Blend menu, Access menu, Provenance, and Activity Log in the Anzo application. The image below shows an example of the view a user with the Data Curator role has in the Anzo application.

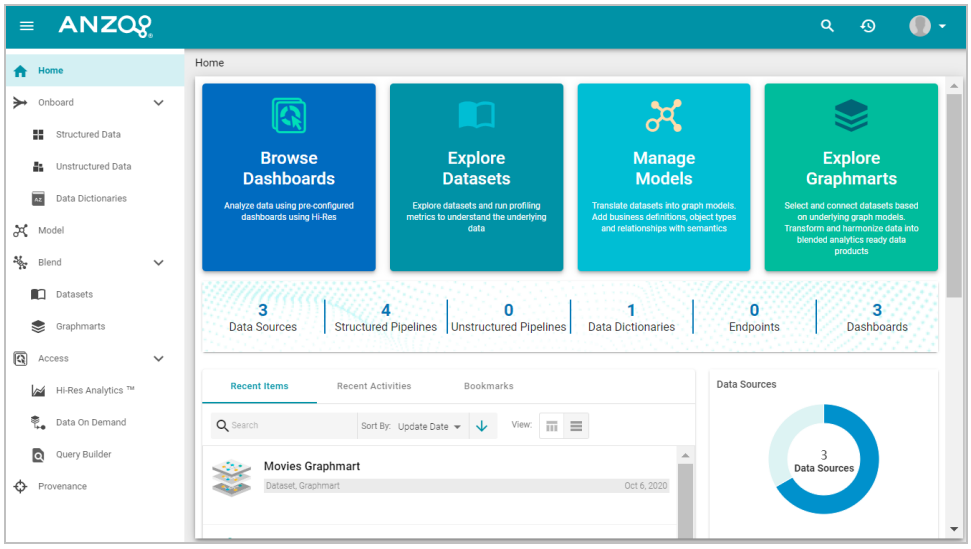


Members of the Data Curator role can:

- Connect to data sources and onboard structured and unstructured data
- View and create data models, mappings, and pipelines
- View and create metadata dictionaries
- View the Dataset catalog
- View and create graphmarts
- View and create Hi-Res Analytics dashboards
- Manage the Query Blacklist Editor in the Hi-Res Analytics application
- View the Activity Log
- Access data with the Query Builder
- Create and access Data on Demand endpoints
- View data provenance

Data Governor

By default the Data Governor role has access to the Onboard menu, Model manager, Blend menu, Access menu, Provenance, and Activity Log in the Anzo application. The image below shows an example of the view a user with the Data Governor role has in the Anzo application.



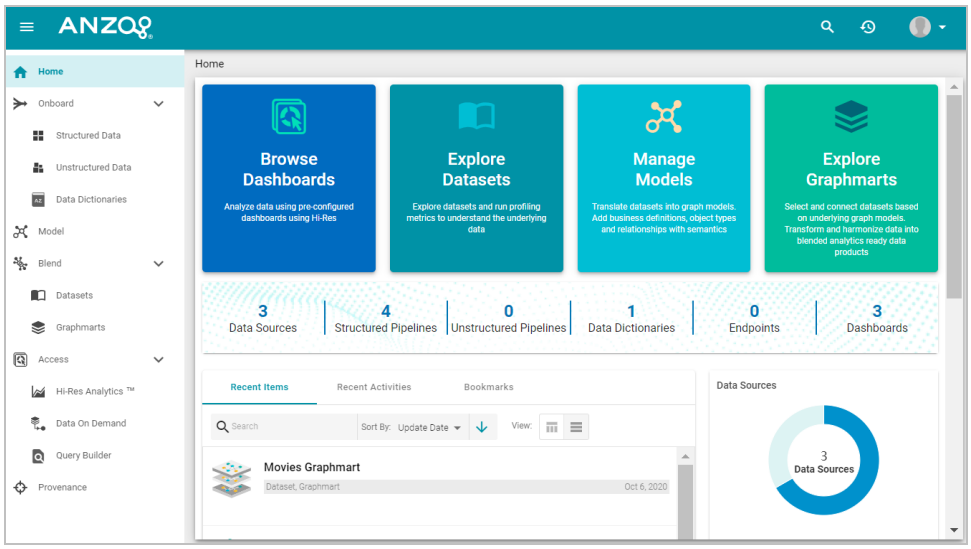
Members of the Data Governor role can:

- Connect to data sources and onboard structured and unstructured data
- View and create data models, mappings, and pipelines
- View and create metadata dictionaries
- View the Dataset catalog
- View and create graphmarts

- View and create Hi-Res Analytics dashboards
- Manage the Query Blacklist Editor in the Hi-Res Analytics application
- View the Activity Log
- Access data with the Query Builder
- Create and access Data on Demand endpoints
- View data provenance

Data Scientist

By default the Data Scientist role has access to the Onboard menu, Model manager, Blend menu, Access menu, Provenance, and Activity Log in the Anzo application. The image below shows an example of the view a user with the Data Scientist role has in the Anzo application.



Members of the Data Scientist role can:

- Connect to data sources and onboard structured and unstructured data
- View and create data models, mappings, and pipelines
- View and create metadata dictionaries
- View the Dataset catalog
- View and create graphmarts
- View and create Hi-Res Analytics dashboards
- View the Activity Log
- Access data with the Query Builder
- Create and access Data on Demand endpoints
- View data provenance

To review the specific permissions for each role, select **Roles** in the **User Management** menu in the Admin application. Click a role to open the Edit dialog box and review the permissions. For more information about the permissions, see Permissions Reference.

Related Topics

Managing Roles

Permissions Reference

Connecting to a Directory Server

Configuring SSO Access

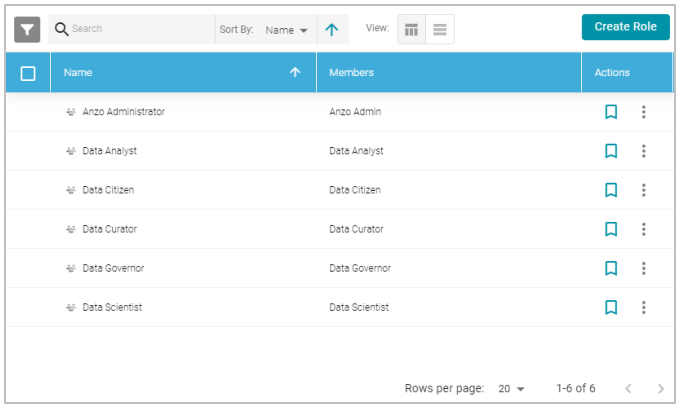
Managing Roles

This topic provides instructions for creating and managing roles. For information about the predefined Anzo roles, see Default Anzo Roles and Permissions.

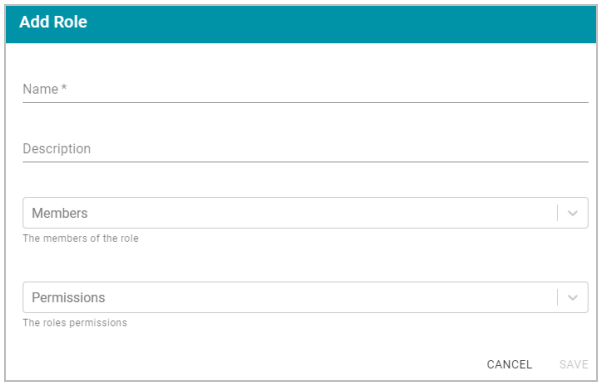
- Creating a New Role
- Adding Users or Groups to a Role
- Configuring Role Permissions

Creating a New Role

1. In the Anzo application, expand the **Administration** menu and click **System Users, Groups, and Roles**. Anzo displays the Roles screen, which lists the existing roles. For example:



2. On the Roles screen, click the **Create Role** button. Anzo displays the Add Role dialog box.

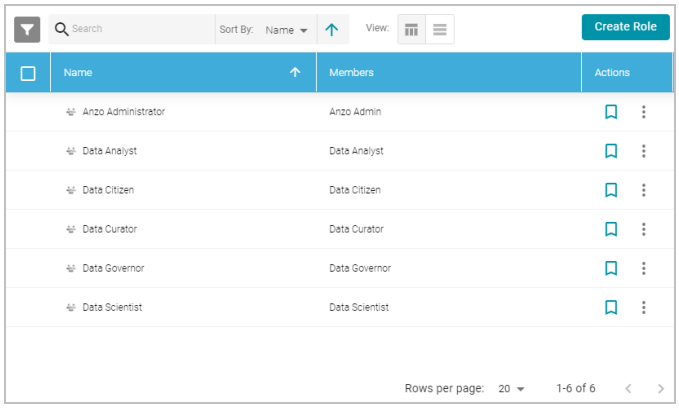


3. Complete the required fields and enter any optional group details:
   - **Name**: The name for the new role.
   - **Description**: An optional description of the role.
   - **Members**: The users or groups who are members of the role. Click the **Members** field to select a member. Click the field again to select additional members.
   - **Permissions**: The list of Anzo features that this role has permission to access. Click the **Permissions** field and select a permission to add it to the list. Click the field again to select additional permissions. For details about each of the permissions, see the Permissions Reference.
4. Click **Save** to add the role to the system. Anzo adds the new role to the list of roles on the Roles screen.

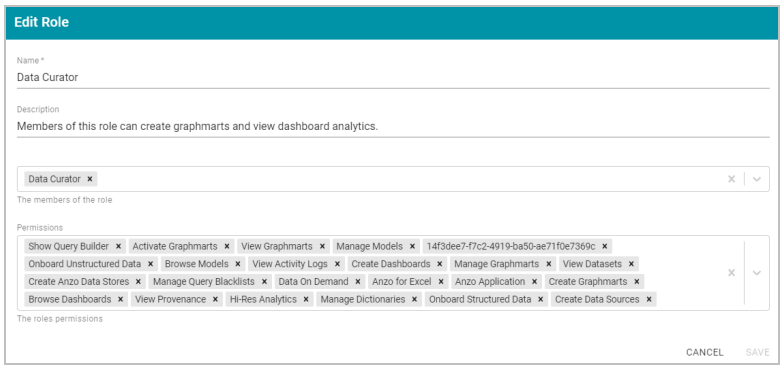
Adding Users or Groups to a Role

Follow the instructions below to add users and/or directory groups to a role.

1. In the Anzo application, expand the **Administration** menu and click **System Users, Groups, and Roles**. Anzo displays the Roles screen, which lists the existing roles. For example:



2. Click the name of the role that you want to add users or groups to. Anzo opens the Edit Role dialog box. For example:



3. Click the **Members** drop-down list to display the list of all available users and groups. You can also search for a user or group name by typing in the Members field. Click a name to add that user or group to the role. Click the field again to select additional members.

   Note If you do not see users or groups that you expect to see, it is possible for Anzo to be out of sync with the directory server if groups have been altered or users have not logged into Anzo in a long time and their account has changed in the directory server. The **Users** and **Groups** screens in the User Management menu have **Sync Directory** buttons that you can click to synchronize with the directory server and update the data in Anzo if necessary.

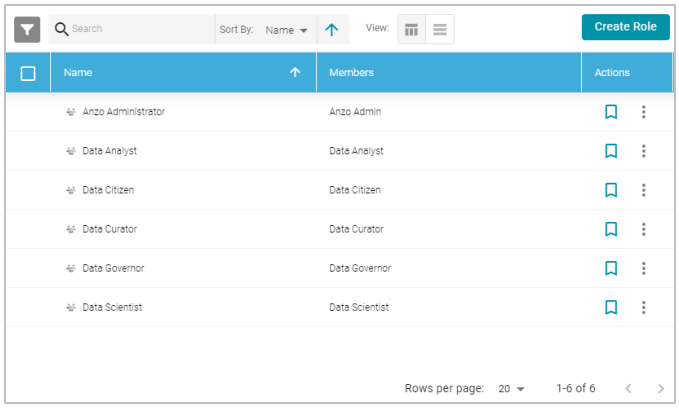
   To remove a member from the role, click the X to the right of the name.

4. When you have finished adding members, click **Save** to save the changes to the role.

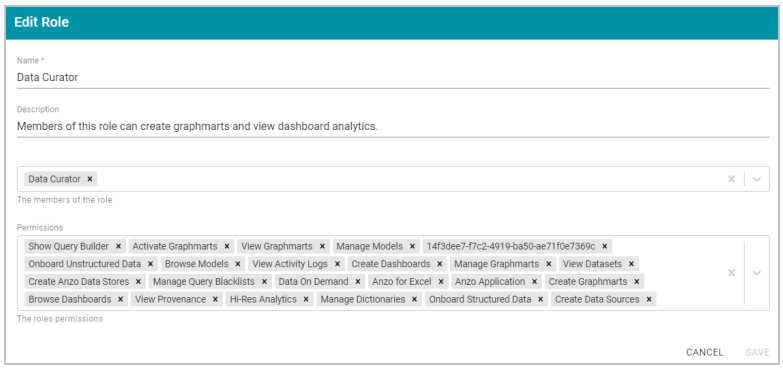
Configuring Role Permissions

Follow the instructions below to add or remove permissions from a role. For details about each of the permissions, see the Permissions Reference.

1. In the Anzo application, expand the **Administration** menu and click **System Users, Groups, and Roles**. Anzo displays the Roles screen, which lists the existing roles. For example:



2. Click the name of the role for which you want to configure permissions. Anzo opens the Edit Role dialog box. For example:



3. The **Permissions** field lists all of the permissions that are applied to the role. To remove a permission, click the X to the right of the permission name. To add a permission click the field to open the Permissions drop-down list. Click a name to add that permission to the role. Click the field again to select additional permissions.
4. When you have finished changing permissions, click **Save** to save the changes to the role.

Related Topics

Permissions Reference

Managing Users and Groups

Default Anzo Roles and Permissions

Connecting to a Directory Server

Configuring SSO Access

Managing Users and Groups

Users and groups are typically managed in an external directory server that is connected to Anzo. The groups from the directory server are added to Anzo roles, and access to Anzo applications and features is controlled by the roles. For instructions on connecting to a directory server, see Connecting to a Directory Server. For information about roles, see Managing Roles.

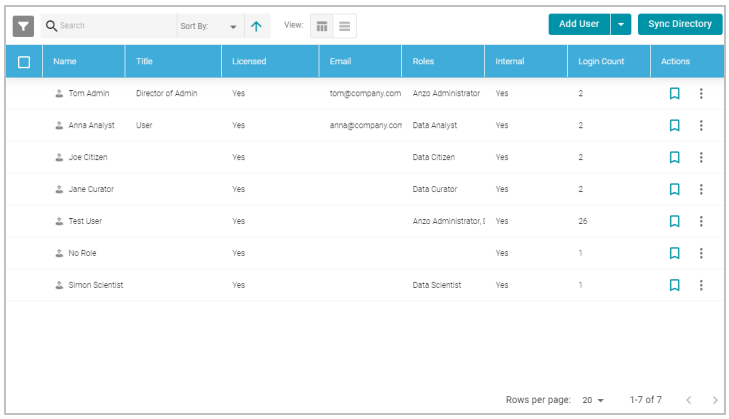
This topic provides information about creating and modifying local Anzo user accounts that are stored in Anzo's internal LDAP server.

- Adding an Internal Anzo User

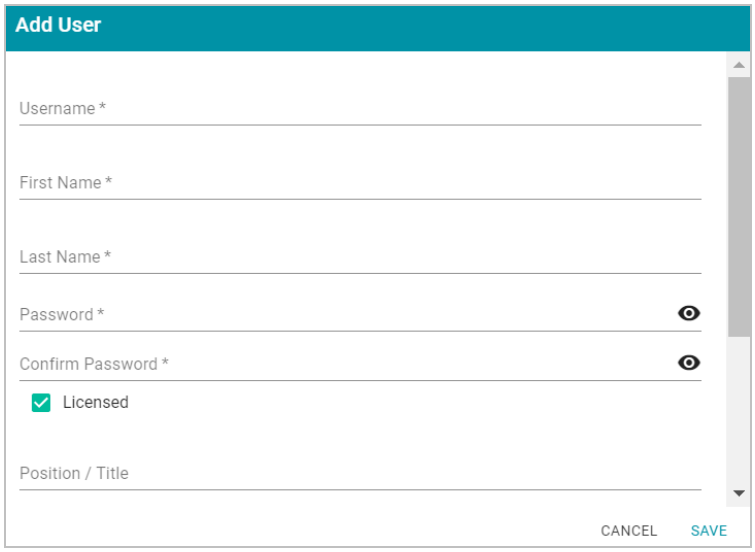
Adding an Internal Anzo User

Follow the instructions below to create a new internal Anzo user account.

1. In the Anzo application, expand the **Administration** menu and click **System Users, Groups, and Roles**. Then click the **Users** tab. Anzo displays the Users screen, which lists the existing users. For example:



2. On the Users screen, click the **Add User** button and select **Add User**. Anzo opens the Add User dialog box.



3. Complete the required fields and enter any optional user details:
    - **Username**: The user name that the user will use to log in to Anzo.
    - **First Name**: The user's first name.
    - **Last Name**: The user's last name.
    - **Password** and **Confirm Password**: Type a password for the user.
    - **Licensed**: Select the **Licensed** checkbox to enable this user to log in to the Anzo applications. If you want to add this user to the system but do not want to give him or her access to Anzo applications at this time, clear the Licensed checkbox.
    - **Position/Title**: The user's job title or position.
    - **Email**: The user's email address.
    - **Phone**: The user's phone number.
    - **Roles**: The role or roles that the user is a member of. The role defines the user's permissions. Click the **Roles** field and select a role from the drop-down list. Click the field again to select additional roles. For

more information about roles, see Managing Roles. For a description of the default Anzo roles, see Default Anzo Roles and Permissions.

4. Click **Save** to add the user to the system.

Permissions Reference

This topic provides reference information for the permissions that are configured in Anzo. The table below lists each of the permissions and describes the pages and menus that are enabled for members of a role where that permission is applied.

Note The permissions described below give access to pages and menus in the Anzo application. Whether members of the role have view or edit access to certain data sets, dashboards, graphmarts, etc. depends on the permissions that are granted at the entity level.

Permission	Description
Activate Graphmarts	Gives permission to view and activate existing graphmarts and import backup versions of graphmarts into Anzo. Does not give permission to create new graphmarts or delete graphmarts. Must also have the **Anzo Application** permission to be able to activate graphmarts.
Administer System Setup	Gives permission to access the menus and options in the Administration application that are related to system setup, such as creating file store connections, Anzo data stores, ETL engine connections, and user and role management. Does not give permission to view or modify **Server Settings**.
Anzo Admin	Grants access to the Administration menu.
Anzo Application	Grants access to the main Anzo application.
Anzo CLI	Gives permission to use the administration command line interface.
Anzo for Excel	Gives permission to open, edit, and create mappings using the Anzo for Office Excel plugin.
Browse Dashboards	Gives permission to view the list of existing dashboards, open the dashboards, and create new dashboards. Must also have the **Anzo Application** permission to view the Hi-Res Analytics screen in the Anzo application.

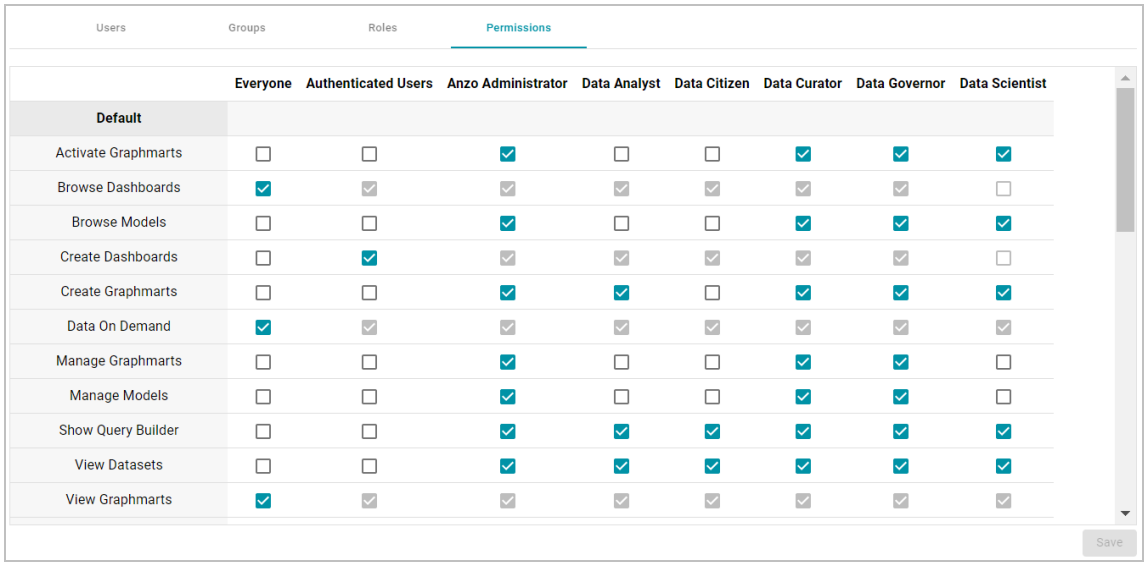
Permission	Description
Browse Models	Gives permission to view existing data models. Applying this permission exposes the **Models** menu in the Anzo application. Must also have the **Anzo Application** permission to access the Model manager.
Create Anzo Data Stores	Gives permission to create Anzo Data Stores. Must also have the **Anzo Admin** permission to access the Administration application as well as **Administer System Setup** so that the **Anzo Data Store** option is available under the **Connections** menu in the Administration application.
Create Dashboards	Gives permission to create dashboards. Applying this permission exposes the **Create Dashboard** button on the Graphmarts screens. Must also have the **Anzo Application** permission to create dashboards from the Graphmarts screens.
Create Data Sources	Gives permission to connect to new data sources. Must also have the **Anzo Application** and **Onboard Structured Data** permissions to access the Data Sources screen and connect to data sources.
Create Graphmarts	Gives permission to create new graphmarts. Applying this permission exposes the **Create** button on the Graphmarts screen. Must also have the **Anzo Application** permission to create graphmarts.
Data on Demand	Gives permission to create Data on Demand endpoints. Applying this permission enables the **Create New Endpoint** button on the Data on Demand tab for graphmarts. Must also have the **Anzo Application** permission to access the application and create endpoints.
Generate Metrics	Gives permission to generate metrics on data sources. Applying this permission exposes the **Generate Metrics** button on the Data Source screens. Must also have the **Anzo Application** permission to access the application.
Hi-Res Analytics	Grants access to the Hi-Res Analytics application.
Manage AnzoGraph	Gives permission to view, configure, and create AnzoGraph connections. Must also have the **Anzo Admin** permission to access the Administration application as well as **Administer System Setup** so that the **AnzoGraph** option is available in the **Connections** menu in the Administration application.

Permission	Description
Manage Certificates	Gives permission to manage server certificates. Must also have the **Anzo Admin** permission to access the Administration application as well as **Administer System Setup** so that the **Server Certificates** option is available in the Administration menu.
Manage Dictionaries	Gives permission to view, edit, and create metadata dictionaries. Applying this permission exposes the **Metadata Hub** option in the Onboard menu. Must also have the **Anzo Application** and **Onboard Structured Data** permissions to expose the Onboard menu.
Manage ETL Engines	Gives permission to configure and connect to ETL engines. Must also have the **Anzo Admin** permission to access the Administration application.
Manage File Stores	Gives permission to view and create file store connections. Must also have the **Anzo Admin** permission to access the Administration application.
Manage Graphmarts	Gives permission to view, configure, and create graphmarts. Must also have the **Anzo Application** permission to access the Graphmarts screens.
Manage Models	Gives permission to view, edit, and create data models. Applying this permission exposes the **Models** menu in the Anzo application. Must also have the **Anzo Application** permission to access the Model manager.
Manage Query Blacklists	Gives permission to manage the Query Blacklist in the System Query Audit Log. Must also have the **Anzo Admin** permission to access the Administration application as well as **Administer System Setup** so that the **Monitoring & Diagnostics** menu is available in the Administration application.
Manage Semantic Services	Gives permission to view and manage semantic services. Must also have the **Anzo Admin** permission to access the Administration application as well as **Administer System Setup** so that the **Monitoring & Diagnostics** menu is available in the Administration application.
Manage Users, Groups, and Roles	Gives permission to create and manage users, directory groups, and roles. Must also have the **Anzo Admin** permission to access the Administration application as well as **Administer System Setup** so that the **User Management** menu is available.

Permission	Description
Onboard Structured Data	Gives permission to connect to structured data sources and onboard data. Applying this permission exposes the **Onboard** > **Structured Data** menu. Must also have the **Anzo Application** permission.
Onboard Unstructured Data	Gives permission to create pipelines to onboard unstructured data. Applying this permission exposes the **Onboard** > **Unstructured Data** menu. Must also have the **Anzo Application** permission.
Show Query Builder	Gives permission to find data and run SPARQL queries using the Query Builder. Applying this permission exposes the **Query Builder** option in the **Access** menu. Must also have the **Anzo Application** permission.
View Activity Logs	Gives permission to view the Activity Log. Applying this permission exposes the Activity Log icon in the top menu bar of the Anzo application. Must also have the **Anzo Application** permission.
View Datasets	Gives permission to view the metadata catalog. Applying this permission exposes the **Datasets** option in the **Blend** menu in the Anzo application. Must also have the **Anzo Application** permission.
View Graphmarts	Gives permission to view the list of existing graphmarts. Must also have the **Anzo Application** permission to view the Graphmarts screen in the Anzo application.
View Log Files	Gives permission to view logs and configure logging in the **Monitoring & Diagnostics** menu. Must also have the **Anzo Admin** permission to access the Administration application.
View Provenance	Gives permission to view provenance. Applying this permission exposes the **Provenance** option in the navigation menu. Must also have the **Anzo Application** permission.

Permissions Overview Screen

To view an overview of the configured permissions for all Anzo roles, you can view the Permissions screen by selecting **System Users, Groups, and Roles** in the **Administration** menu and clicking the **Permissions** tab. The screen displays a table. The heading row lists each role, and the first column lists each permission. The permissions are grouped into categories, such as Application or Onboarding. For example:



The rows for each role column include checkboxes that control permissions. You can select or clear checkboxes to enable or disable permissions.

Related Topics

Managing Roles

Default Anzo Roles and Permissions

Connecting to a Directory Server

Configuring SSO Access

Managing Default Access Policies

Default Access Policies are the security policies that are applied by default to each of the components in a registry. These are the base permissions that get assigned to an entity when it is created--**if that component does not automatically inherit its permissions from another component**. Any component-level permissions that are added or changed from the Sharing or Security tabs in the Anzo application supersede the default access policy.

Note Default access policies only apply to components that have no other access control logic added at creation time. To facilitate common workflows, when a component is created using the Anzo user interface the application applies logic so that all of the components in the workflow inherit the same permissions. For example, when the Ingest workflow is used to onboard data from a data source, the resulting model, pipeline, and mapping components inherit permissions from the data source by default. And when a pipeline is published, the resulting file-based linked data set inherits permissions from the pipeline. Because of permission inheritance, default access policies are rarely applied to new components and typically are not changed.

This topic describes the default access policies for each registry and provides instructions for changing access policies.

Default Access Policy Permissions Reference

Default access policies use the same predefined permission sets and mechanism for assigning permissions as other components in the Anzo application, such as graphmart and data source sharing. There are three predefined permission sets that include a combination of six permissions that can be assigned to the creator of the component and other Anzo users, groups, or roles.

The table below lists the predefined permission sets and describes the privileges that are granted for each permission that is part of the predefined set:

Set	Applied Permission	Allows a user to:
View	**View**	• See the new component in the Anzo application. • Copy the component URI in the Anzo application, if applicable. • Create and import versions of the component.
	Meta View	• The Meta View permission relates only to a component's permissions. If a user has this permission, they can see the permission configuration (e.g., the Sharing tab for graphmarts), but they cannot modify, add, or remove permissions.
Modify	In addition to the **View** and **Meta View** permissions described above, the **Modify** set includes the **Add/Edit** and **Delete** permissions described below.	
	Add/Edit	• Change a component, such as to rename it or edit its description. • Add artifacts to a component, such as adding a schema to a data source or a data layer to a graphmart. • Create new components of that registry type.
	Delete	• Remove artifacts from the component, such as deleting data layers from a graphmart or schemas from a data source. • Delete the component.

Set	Applied Permission	Allows a user to:
Admin	In addition to the **View**, **Meta View**, **Add/Edit**, and **Delete** permissions described above, the **Admin** set includes the **Meta Add/Edit** and **Meta Delete** permissions described below.	
	Meta Add/Edit	• The Meta Add/Edit permission relates only to a component's permissions. If a user has this permission, they can see the permission configuration (e.g., the Sharing tab for graphmarts), and they can change the configuration by adding permissions to a user, group, or role. They cannot remove permissions from any user, group, or role.
	Meta Delete	• The Meta Delete permission relates only to a component's permissions. If a user has this permission, they can see the permission configuration (e.g., the Sharing tab for graphmarts), and they can change the configuration by removing permissions from a user, group, or role.

Default Access Policy Registry Reference

There is a default access policy for each of the Anzo registries. A registry is a system-level collection of application and system component configurations of the same type, such as graphmarts, data sources, ontologies, roles, etc. Like data, registries are stored and managed as RDF named graphs according to ontologies. Each registry is a Linked Data Set.

Important Never modify any of the Anzo registries. Changing or removing a registry can irreparably damage your Anzo server.

To see and manage the default access policies, expand the **Administration** menu and click **Default Access Policies**. The sections below provide details about each of the registries for which you can configure default access policies:

- Data Sources Registry
- Global Linked Data Config
- LinkedDataSets Collection Catalog
- Elastic Search Configuration Registry
- Ontology Registry
- Role Registry
- Graphmarts Registry

- Datasets Registry
- Favorites Registry
- Comments Registry
- Linked Data Set Registry
- Persisted Queries Registry

Data Sources Registry

Item	Description
Data Sources Registry and Policy	This registry is the collection of all data source configurations. This access policy is applied by default when a data source is created.
Default Access Configuration	Creator: Admin Everyone: View
Default Access Details	The **Creator** of a data source is assigned the Admin permission set for that data source. The **Everyone** role is assigned the View permission set.

Global Linked Data Config

Item	Description
Global Linked Data Config Registry and Policy	This registry is a catch-all for system settings that are related to URIs, including URI prefixes and the base URI (such as csi.com for class instance URIs), as well as other specifics about entities that are created in Anzo. This registry also supplies the default ACL configuration for components if nothing else applies.
Default Access Configuration	Creator: Modify
Default Access Details	The **Creator** of a component that follows this policy is assigned theModify permission set.

LinkedDataSets Collection Catalog

Item	Description
LinkedDataSets Collection Catalog Registry and Policy	This is a legacy registry that stores Linked Dataset Collection configuration data. Linked dataset collections were the precursor to graphmarts and have been deprecated in most organizations. If you do use linked dataset collections, this access policy is applied by default when a new collection is created.
Default Access Configuration	Creator: Admin Everyone: View
Default Access Details	The **Creator** of a linked dataset collection is assigned the Admin permission set for that collection. The **Everyone** role is assigned the View permission set.

Elastic Search Configuration Registry

Item	Description
Elastic Search Configuration Registry and Policy	This registry is the collection of all Elasticsearch connection configurations. This access policy is applied by default when an Elasticsearch connection is created.
Default Access Configuration	Creator: Admin Everyone: View
Default Access Details	The **Creator** of an Elasticsearch connection is assigned the Admin permission set for that configuration. The **Everyone** role is assigned the View permission set.

Ontology Registry

Item	Description
Ontology Registry and Policy	This registry is the collection of all ontology configurations. This access policy is applied by default if a model is created and no other access control logic is applied.

Item	Description
Default Access Configuration	Creator: Admin Everyone: View
Default Access Details	The **Creator** of a data model is assigned the Admin permission set for that model. The **Everyone** role is assigned the View permission set.

Role Registry

Item	Description
Role Registry and Policy	This registry is the collection of all role configurations. This access policy is applied by default when a role is created.
Default Access Configuration	System Administrator: Admin Everyone: View
Default Access Details	The **System Administrator** is assigned the Admin permission set and is the only user who can create roles. The **Everyone** role is assigned the View permission set.

Graphmarts Registry

Item	Description
Graphmarts Registry and Policy	This registry stores all graphmart configurations. This access policy is applied by default if a graphmart is created and no other access control logic is applied.
Default Access Configuration	Creator: Admin Everyone: View
Default Access Details	The **Creator** of a graphmart is assigned the Admin permission set for that graphmart. The **Everyone** role is assigned the View permission set.

Datasets Registry

Item	Description
Datasets Registry and Policy	This legacy registry stores configurations for non-ontology-backed data sets. This registry is not used; there is nothing that can be created in the user interface that affects this registry.
Default Access Configuration	N/A
Default Access Details	N/A

Favorites Registry

Item	Description
Favorites Registry and Policy	This registry stores configurations for components that have been tagged as favorites. The access policy applies by default when a component is marked as a favorite.
Default Access Configuration	Creator: Admin
Default Access Details	The user who tags a component as a favorite is assigned the Admin permission set. Favorites are unique to each user, and a user cannot see another user's favorites.

Comments Registry

Item	Description
Comments Registry and Policy	This registry stores configurations for comments that have been added to components. The access policy applies by default to new comments.
Default Access Configuration	Creator: Admin Everyone: View

Item	Description
Default Access Details	The **Creator** of a comment is assigned the Admin permission set for that comment. The **Everyone** role is assigned the View permission set, which means they can see the comment but not change or remove it.

Linked Data Set Registry

Item	Description
Linked Data Set Registry and Policy	This registry stores configuration information for all linked data sets, notably the file-based linked data sets that are listed on the Datasets screen in the Anzo application and are available to add to graphmarts.
Default Access Configuration	N/A
Default Access Details	File-based linked data sets (FLDS) in the Linked Data Set Registry inherit their permissions from the components that make up the FLDS.

Persisted Queries Registry

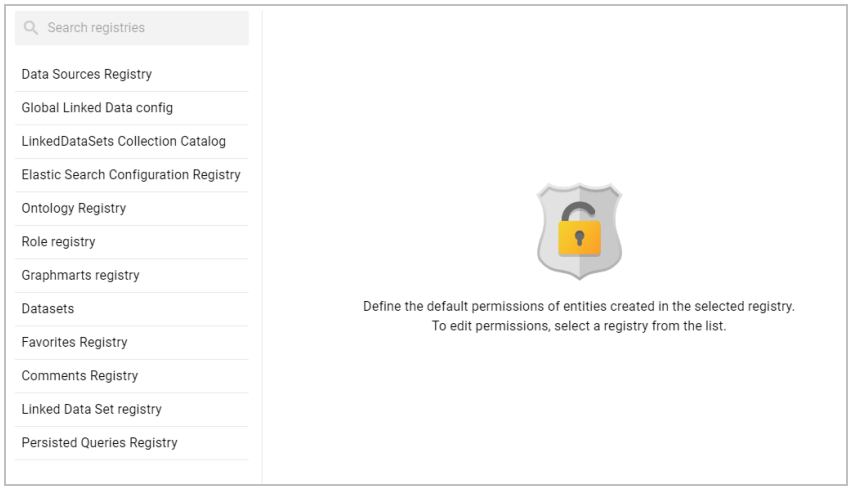
Item	Description
Persisted Queries Registry and Policy	This registry stores configuration information for all of the saved queries in the Query Builder. This access policy is applied by default when a new query is saved.
Default Access Configuration	Creator: Admin
Default Access Details	The user who saves a query is assigned the Admin permission set. By default, saved queries are unique to each creator, and other users cannot see another user's queries.

Configuring Default Access Policy Permissions

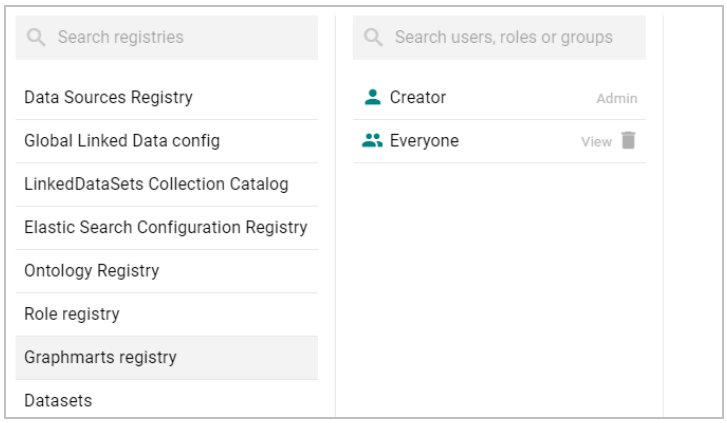
Follow the instructions below to change the default access policy for a registry.

Important Changing default access control policies does not change permissions on any existing components. The changes affect only new components that are created after the change and if no other permission inheritance logic applies. Because of permission inheritance, default access policies are rarely applied to new components and typically are not changed.

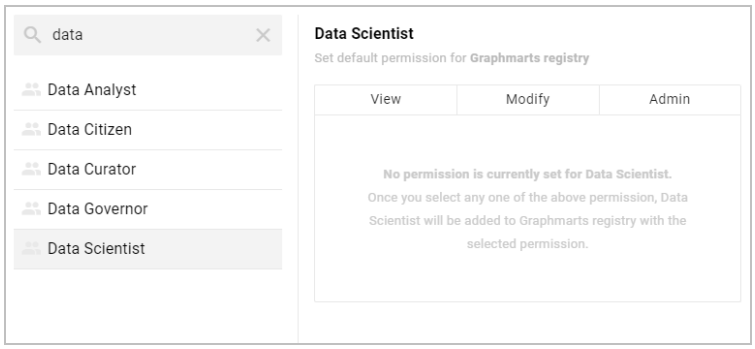
1. In the Anzo application, expand the **Administration** menu and click **Default Access Policies**. The Default Access Policies screen is displayed.



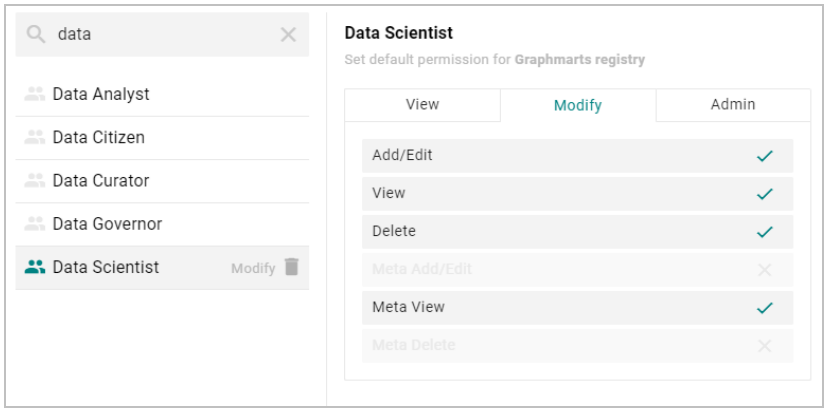
2. On the left side of the screen, select the access policy that you want to configure. The current configuration for that policy is shown on the right side of the screen. For example, the image below shows the Graphmarts Registry. The graphmart Creator has **Admin** permissions, and the Everyone role has **View** permissions.



3. To change a configured user, group, or role, select a name in the list to view the permissions on the right side of the screen. To add a user, role, or group to configure, type a term in the **Search** field. Then select a name in the result list to view the permissions details. For example, the image below shows the search results for additional roles and selects the Data Scientist role:



4. On the right side of the screen, click the tab for the predefined permission set that you want to assign to the selected user, group, or role. For information about the permission sets, see Default Access Policy Permissions Reference above. For example, the image below assigns the **Modify** permission set to the Data Scientist role.



   To clear permissions for a user, role, or group, click the trashcan icon ( ) next to the user, role, or group name.

5. To configure additional users, groups, or roles, select the name and then repeat the step above to apply a permission set. Changes to access control policies are automatically saved.

Related Topics

User Administration

Monitoring and Diagnostics

The topics in this section provide information about monitoring events and managing Anzo and AnzoGraph diagnostic files.

- Managing Anzo Server Logging
- Retrieving AnzoGraph Diagnostic Files
- Viewing AnzoGraph Memory and Performance Statistics

Managing Anzo Server Logging

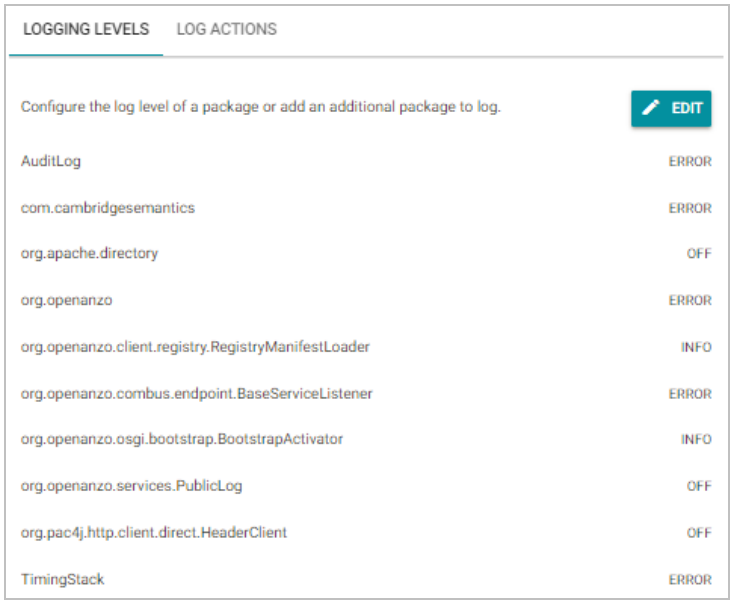
When investigating issues, Cambridge Semantics Support can provide guidance on generating any necessary diagnostic information for troubleshooting. The topics in this section provide general instructions for reviewing log files, adding or removing logs for components, and changing the level or type of information that is logged.

- Reviewing Anzo Log Files
- Adding Logs for New Components

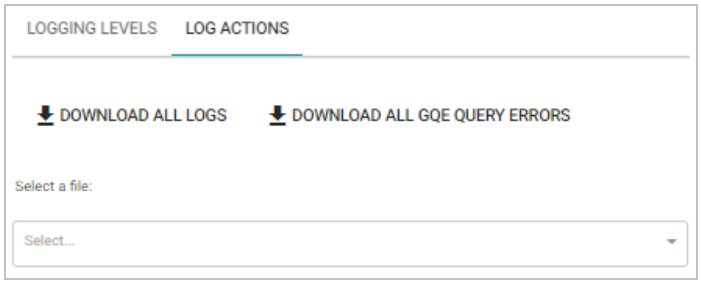
Reviewing Anzo Log Files

You can review the Anzo log files to get more detailed information about errors or obtain more granular information about server operations. This topic provides information about viewing Anzo log files.

1. In the Anzo console, expand the **Administration** menu and click **Logging**. The Logging screen displays the existing log packages and logging levels that are configured for the server.



2. On the Logging screen, click the **Log Actions** tab:

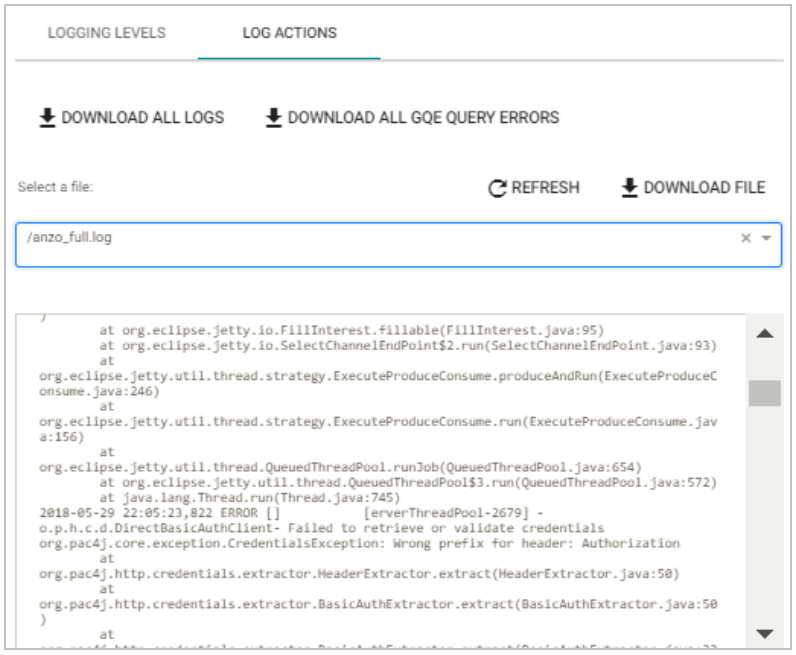


3. If you want to download a .zip file that contains all Anzo log files, click the **Download All Logs** link. If you want to download just the query error logs for AnzoGraph, click **Download All GQE Query Errors**.

   Note: For information about generating additional AnzoGraph diagnostic files for Cambridge Semantics Support, see Generating Diagnostic Files with the System Manager.

4. To view the contents of a log file, click the **Select** drop-down field and select a log from the drop-down list. Major issues are logged in files with the suffix "_error," and other server operational information is logged in files with the suffix "_info."

   When you select a file, Anzo displays the contents of the file onscreen and enables the Refresh and Download File buttons. For example:



   Click **Refresh** to re-load the display with the latest version of the file in `Anzo/Server/logs` on the server. Or click **Download File** to download a copy of the file to your machine.

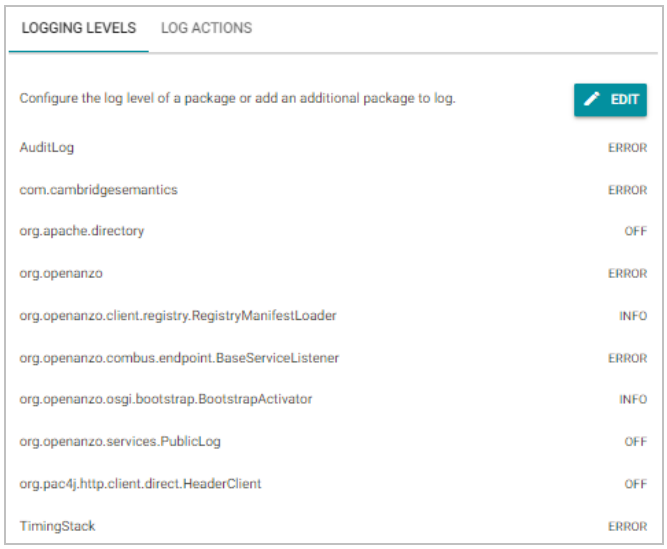
Related Topics

Adding Logs for New Components

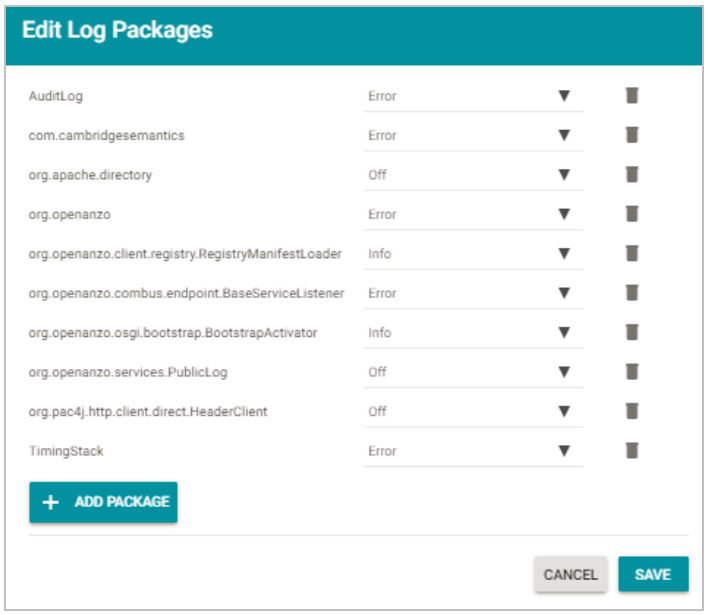
Adding Logs for New Components

This topic provides instructions for configuring the Anzo server to produce log files for additional Anzo components.

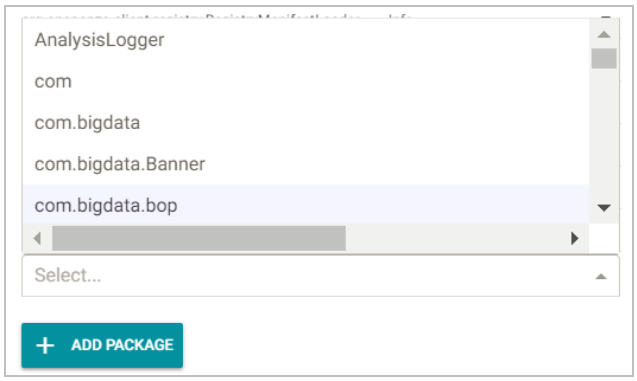
1. In the Anzo console, expand the **Administration** menu and click **Logging**. Anzo displays the Logging screen, which lists the existing log packages and logging levels that are configured for the server.



2. On the Logging screen, click the **Edit** button. Anzo displays the Edit Log Packages dialog box.



3. Click **Add Package** to open the package drop-down list. You can scroll through the list or start typing a keyword to search for a package.



4. Click a package to select it. Anzo adds the package to the list of packages on the Log Actions screen.
5. Click the drop-down list next to the new package to set the logging level. The log level that you choose specifies the type and amount of information that Anzo logs for the component:
    - **Off**: Turns logging off for the component.
    - **Trace**: Logs finer-grained error information than Debug.
    - **Debug**: Logs fine-grained error messages that are intended to help debug an application.
    - **Info**: Logs course-grained information that highlights the progress of the application.
    - **Warn**: Logs information about potentially problematic situations.
    - **Error**: Logs errors that usually allow the application to continue running.
    - **Fatal**: Logs severe errors that prevent the application from running.
6. Click **Save** to submit the new logging configuration.

Related Topics

Reviewing Anzo Log Files

Retrieving AnzoGraph Diagnostic Files

When Cambridge Semantics Support requests AnzoGraph diagnostic files for troubleshooting an issue, you can quickly retrieve the files from the Diagnostics tab on the AnzoGraph Administration page in the Anzo console. This topic provides information about the AnzoGraph diagnostics and instructions for retrieving the files.

Introduction to AnzoGraph Diagnostic Files

There are two types of AnzoGraph diagnostic files:

- **XRay**: XRays are generated on-demand. If you encounter an error and the database remains running, you generate an XRay to produce the diagnostic files.
- **Crash Dump**: If you encounter an error that crashes the database, AnzoGraph automatically generates a crash dump that contains diagnostic information about the crash.

Xrays and crash dumps are valuable tools that enable Cambridge Semantics to diagnose and fix issues without access or any other visibility into a customer's data or database system. They can also be used to report on overall and detailed system performance, resulting in improved query performance for future releases of AnzoGraph.

Xrays and crash dumps harvest the diagnostic data that is stored in AnzoGraph's system tables. They include information such as:

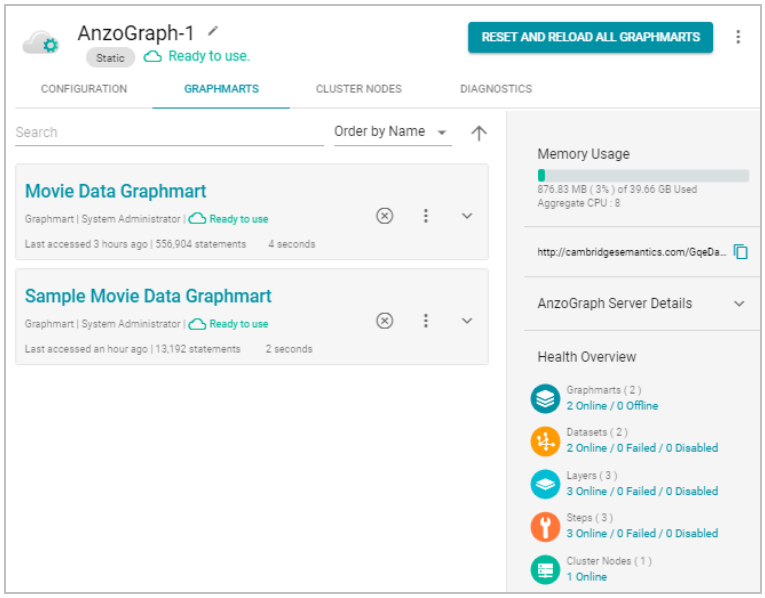
- A low level, de-identified log of the requests that were sent to the database.
- Statistics like query operation step execution times, number of rows processed, and amount of memory used.
- Detailed but de-identified trace information for errors that were encountered.
- Configuration information such as the number of nodes in the cluster and AnzoGraph system settings values.

Xrays and crash dumps are designed to be anonymous and can be safely shared with Cambridge Semantics Support. They do NOT capture user information or any of the data that is loaded into memory by a user, nor do they expose details that could be used to reveal the nature of the data being queried.

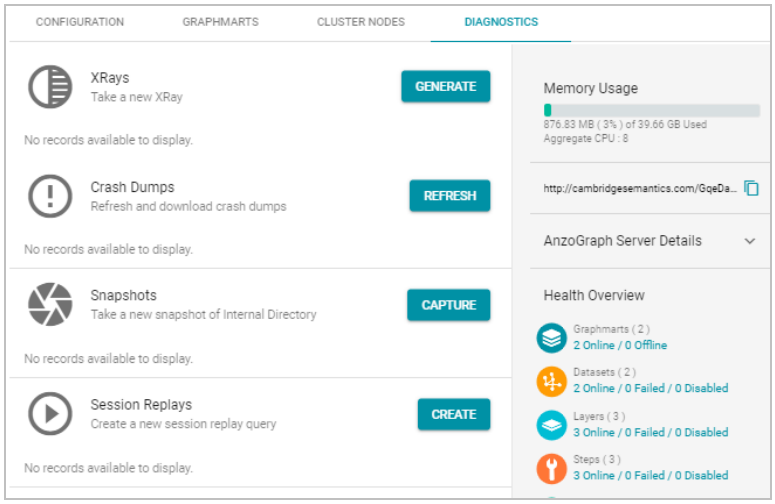
Retrieving the Files

Follow the instructions below to download an xray or crash dump to send to Cambridge Semantics Support.

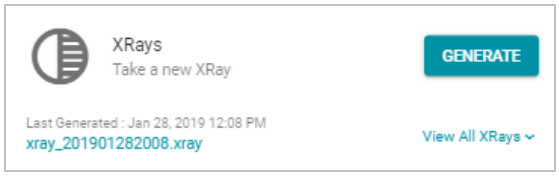
1. In the Anzo console, expand the **Administration** menu and select **AnzoGraph**. Anzo displays the list of the configured AnzoGraph instances.
2. Click the name of the AnzoGraph instance for which you want to download an xray or crash dump. Anzo displays the graphmart details for the instance. For example:



3. Click the **Diagnostics** tab. Anzo displays the available options. For example:



4. If you want to retrieve an xray, click the **Generate** button. Anzo creates the xray and produces a tarball with a .xray extension. For example:



   Click the xray file name to download the tarball to you computer for sending to Cambridge Semantics.

   Note: The files in the tarball are compressed. Do not compress the .xray file before sending it to Cambridge Semantics.

5. If you want to retrieve a crash dump, click the **Refresh** button to refresh the list of available crash dump .xray files. Click the file name that you want to download. Anzo downloads the file to your computer.

Viewing AnzoGraph Memory and Performance Statistics

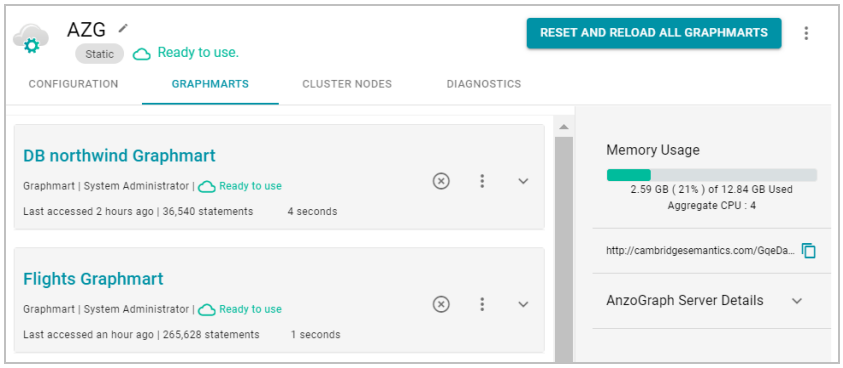
This topic provides information about viewing AnzoGraph's memory usage, query performance statistics, and network bandwidth.

- Viewing Current Memory Usage
- Reviewing Query Performance Statistics
- Evaluating Network Performance on Clusters

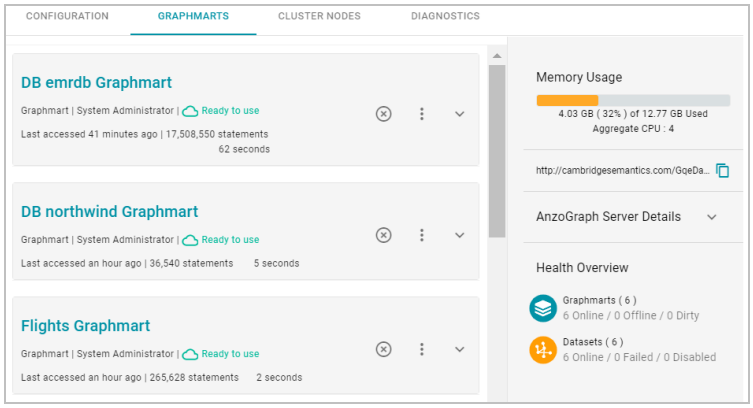
Viewing Current Memory Usage

The AnzoGraph instance details screens in the Anzo user interface provide a view of AnzoGraph's current memory usage.

1. In the Anzo console, expand the **Administration** menu and select **AnzoGraph**. Anzo displays the AnzoGraph screen, which lists the connected AnzoGraph instances.
2. Click the name of the instance that you want to evaluate. Anzo displays the graphmart screen for that instance. The memory usage details are displayed in the top right corner on all of the tabs. For example, the test instance below shows that 21% of the available memory is in use:



Ideally, the data at rest should use only 25%-30% of the available memory because query execution and intermediate result storage can temporarily consume a very large amount of RAM, especially when multiple users run queries concurrently. When memory usage increases so that the data uses more than 25% - 30% of the available memory, the status bar changes color to orange as a warning . For example:



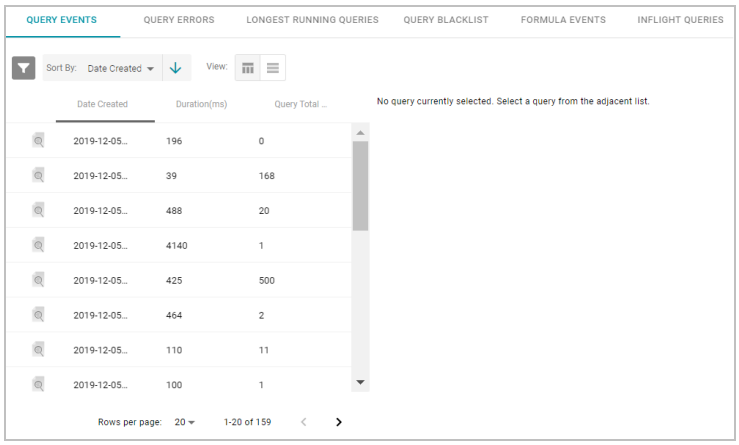
If memory usage for the data at rest remains above 50%, Cambridge Semantics recommends that you increase the amount of RAM available. For more information about memory usage, see Memory Sizing Guidelines.

Reviewing Query Performance Statistics

The System Query Audit log in the Anzo user interface provides details about all system events. Users can filter the query audit log to view query execution times for AnzoGraph queries.

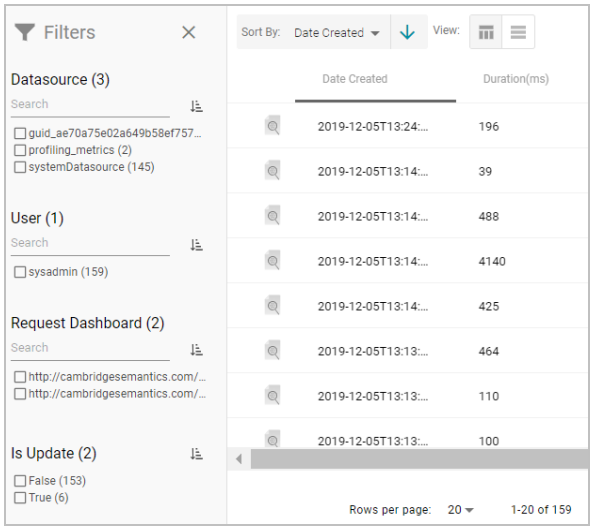
Viewing AnzoGraph Query Statistics

1. In the Anzo console, expand the **Administration** menu and select **System Query Audit**. Anzo displays the Query Events log. For example:

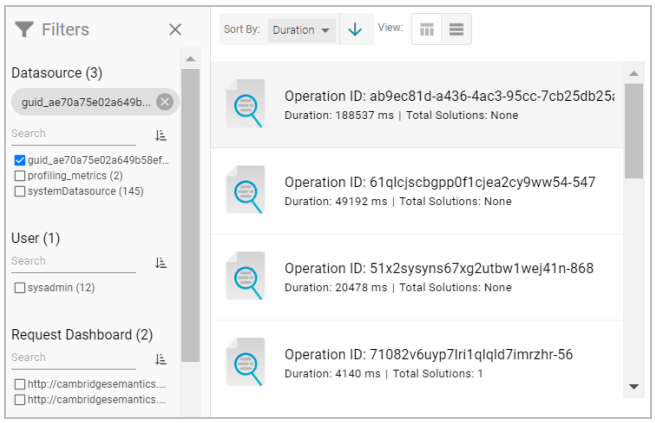


By default, the log shows an overview of all query events for all data sources. The table lists the date created, duration in milliseconds, and total number of solutions returned for each query event. You can select an event in the table to view details about that event, such as the target data source and query text, on the right side of the screen.

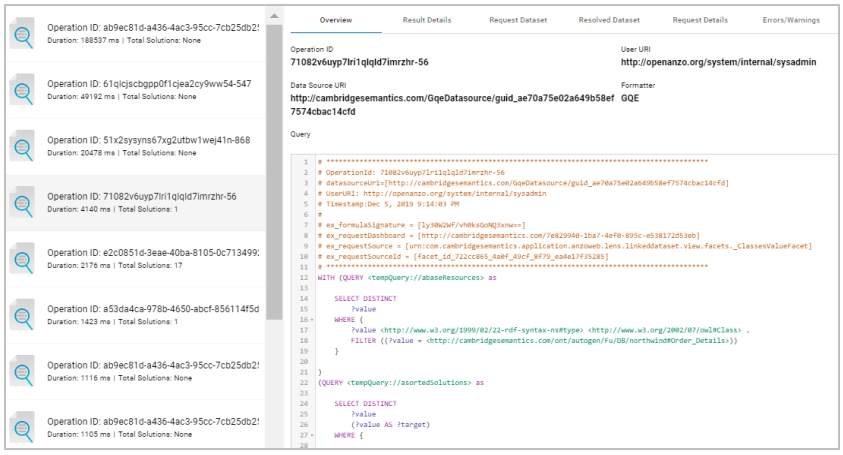
2. To filter the events to display only AnzoGraph queries, open the Filters panel by clicking the filter icon ( ) in the top left corner of the screen. For example:



3. In the Filters panel under **Datasource**, select the checkbox for the AnzoGraph data source. Typically the name starts with **guid_**. The table of events is filtered to display AnzoGraph events. At the top of the screen, you can choose between a table view ( ) or list view ( ), and you can sort by date, duration, or total solutions. For example, the image below shows a list view of AnzoGraph query events sorted by duration:



4. Select any query in the list to view the event overview on the right side of the screen. For example:



To view more details about the query event, click the additional tabs to the right of the Overview tab.

Evaluating Network Performance on Clusters

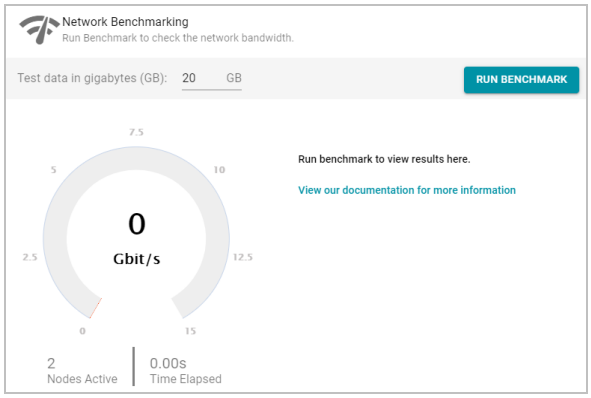
The AnzoGraph Diagnostics screen provides a network benchmark that you can run to evaluate the network bandwidth of a cluster.

Note: Network performance is not applicable for single servers. The benchmark described below is not available for single-server AnzoGraph deployments.

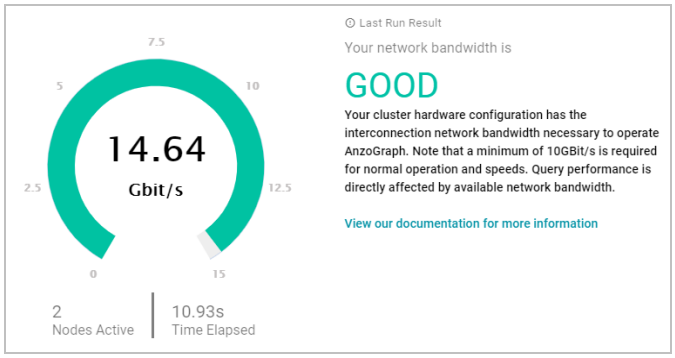
Running the Network Benchmark

1. In the Anzo console, expand the **Administration** menu and select **AnzoGraph**. Anzo displays the AnzoGraph screen, which lists the connected AnzoGraph instances.
2. Click the name of the cluster that you want to evaluate. Anzo displays the graphmart details screen for the cluster.

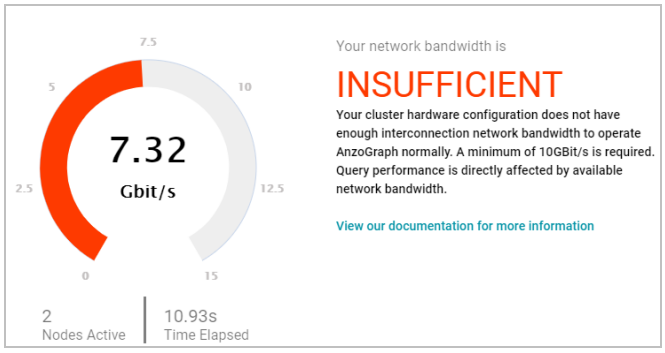
3. Click the **Diagnostics** tab and find the Network Benchmarking option at the bottom the screen. For example:



4. By default, the benchmark is set to distribute 20 GB of data per node over the network. Each node in the cluster sends 20 GB to every other node. You can specify a different size if necessary. Note that increasing the value also increases the time to run the benchmark.
5. To run the test, click the **Run Benchmark** button. Anzo runs the benchmark and displays the results. For example:



If the bandwidth is less than 10 Gbit/s, Anzo displays an "Insufficient" result. For example:



When the results are insufficient, Cambridge Semantics recommends that you increase the network bandwidth. You can continue to use the cluster with the expectation of slower performance for network-bound operations.

Related Topics

Retrieving AnzoGraph Diagnostic Files

AnzoGraph Server Administration

AnzoGraph Server Administration

The topics in this section provide reference information and instructions for performing administrative tasks on an AnzoGraph server. Some tasks, such as modifying server configuration settings, cannot be done via the Anzo console. Other tasks, such as starting and stopping AnzoGraph using the system manager, are documented as alternate methods of managing AnzoGraph if the Anzo console is unavailable or you prefer to use the AnzoGraph command line interface.

- Starting and Stopping AnzoGraph
- Configuring AnzoGraph for Kerberos Authentication
- Using the AnzoGraph Command Line Interface
- Changing AnzoGraph Configuration Settings
- AnzoGraph System Settings Reference
- Generating Diagnostic Files with the System Manager

Starting and Stopping AnzoGraph

This topic provides instructions for starting and stopping AnzoGraph.

Note The system management daemon, **azgmgrd**, should remain running at all times. When you restart the database, do not stop and start the daemon. There are two circumstances that require you to restart azgmgrd:

1. When Upgrading AnzoGraph.
2. When making changes to the `install_path/config/ip_addrs.conf` file if you add or remove servers from an AnzoGraph cluster.

Follow the appropriate instructions below, depending on the current state of AnzoGraph and your use case:

- Stop the Database and Leave the System Management Daemon Running
- Start the Database (the System Management Daemon is Running)
- Stop the Database and the System Management Daemon
- Start the System Management Daemon and the Database
- Reinitializing the Database

Stop the Database and Leave the System Management Daemon Running

To stop the database, run one of the following commands from the **leader server**:

- If services are set up as described in Installing AnzoGraph on a Cluster, run the following command:

```
sudo systemctl stop anzograph
```

- If services are not set up, stop the database with the following system manager command:

```
/install_path/bin/azgctl -stop
```

Important Make sure that you are logged in as the Anzo service user any time you start and stop AnzoGraph using the system manager commands.

If queries are running, the system manager waits the number of seconds in stop_timeout (the default value is 30 seconds) for any outstanding queries to complete and then stops the database.

Start the Database (the System Management Daemon is Running)

To start the database, run one of the following commands from the **leader server**:

- If services are set up as described in Installing AnzoGraph on a Cluster, run the following command:

```
sudo systemctl start anzograph
```

- If services are not set up, start the database with the following system manager command:

```
/install_path/bin/azgctl -start
```

Important Make sure that you are logged in as the Anzo service user any time you start and stop AnzoGraph using the system manager commands.

Stop the Database and the System Management Daemon

To stop the database and system management daemon, run the appropriate commands from the **leader server**:

- If services are set up as described in Installing AnzoGraph on a Cluster, run the following commands on the leader server to stop the database and daemon on all servers in the cluster:

```
sudo systemctl stop anzograph
```

```
sudo systemctl stop azgmgrd
```

- If services are not set up, run the following commands on the leader server to stop the database and daemon on all servers in the cluster:

```
/install_path/bin/azgctl -stop
```

```
/install_path/bin/azgctl -stopdaemon
```

Important Make sure that you are logged in as the Anzo service user any time you start and stop AnzoGraph using the system manager commands.

Start the System Management Daemon and the Database

To start the system management daemon, run one of the following commands. On clusters, run the command on **each server in the cluster**:

- If services are set up as described in Installing AnzoGraph on a Cluster, run the following command on all servers in the cluster:

```
sudo systemctl start azgmgrd
```

- If services are not set up, run the following command on all servers in the cluster:

```
/install_path/bin/azgmgrd
```

Important Make sure that you are logged in as the Anzo service user any time you start and stop AnzoGraph using the system manager commands.

To start the database after the system management daemon is running, run one of the following commands on the **leader node**:

- If services are set up as described in Installing AnzoGraph on a Cluster, run the following command:

```
sudo systemctl start anzograph
```

- If services are not set up, start the database with the following system manager command:

```
/install_path/bin/azgctl -start
```

Reinitializing the Database

If you need to reinitialize the database to remove the generated code and any persisted data, run the following command. The system management daemon (azgmgrd) should be running.

```
/install_path/bin/azgctl -start -init
```

Configuring AnzoGraph for Kerberos Authentication

If you plan to load data to AnzoGraph from an HDFS file store that uses Kerberos authentication, follow the steps below to configure AnzoGraph for Kerberos authentication.

1. In order to be able to generate an authentication token for requesting encrypted ticket-granting tickets (TGT) from the key distribution center (KDC), each AnzoGraph host server must include the Kerberos workstation package, **krb5-workstation**. On each server in the cluster, run the following command to install the package:

```
sudo yum install -y krb5-workstation
```

2. In order to establish a connection to the KDC, AnzoGraph must have a copy of the KDC's **krb5.conf** file. Place a copy of krb5.conf in the **/etc** directory on each AnzoGraph host server.
3. In addition to krb5.conf, each AnzoGraph server needs a copy of the **.keytab** file from the principal node. The keytab file and principal name are used to generate an authentication token.

   Note To find the location of the .keytab file and the principal name, you can look up the `dfs.web.authentication.kerberos.keytab` and `dfs.web.authentication.kerberos.principal` values in **hdfs-site.xml** on the HDFS master node.

   Copy the .keytab file to any location on each AnzoGraph host server, and then run the following command to generate the authentication token:

   ```
   kinit -p principal_name -k -t path/keytab_file
   ```

   Where *principal_name* is the Kerberos principal name and *path/keytab_file* is the location and name of the .keytab file.

Related Topics

Connecting to a File Store

Using the AnzoGraph Command Line Interface

You can use the **azgi** command line interface (CLI) in the `/install_path/bin` directory to issue commands directly to the database.

Important The azgi CLI works on the SPARQL HTTP port and is enabled only when HTTP access is enabled. HTTP access is controlled by the enable_sparql_protocol setting. If you disabled HTTP access and want to enable it so that you can use the command line, see Changing AnzoGraph Configuration Settings for instructions.

This section describes the available azgi commands. To view the list of options from the command line, run `azgi -help`.

AZGI Usage

```
azgi [-f filename] [-c "command"] [-csv] [-json] [-xml] [-h hostname] [-p port]
     [-timer] [-silent] [-max_colwid width] [-wide] [-set parameter=value] [-v]
```

Option	Description
-f **filename**	Runs the specified SPARQL query file. For example, the following command runs the query or queries in the query.rq file: `azgi -f /home/user/query.rq`

-c "**command**"	Runs the command in quotation marks. For example, this command runs a query:
	`azgi -c "select distinct ?eventname from <tickit> where {?event <eventname> ?eventname} limit 100"`
	You can include multiple -c options to run multiple commands. For example, this command runs two queries:
	`azgi -c "select * from <tickit> where {?s ?p ?o} limit 100" -c "select distinct ?likes from <tickit> where {?person <like> ?likes}"`

-csv	Returns results in CSV format. For example:
	`azgi -csv -c "select * from <tickit> where {<person1> ?p ?o} limit 5"`
	`p,o`
	`http://www.w3.org/1999/02/22-rdf-syntax-ns#type,person`
	`like,classical`
	`like,musicals`
	`like,sports`
	`like,theatre`

-json	Returns results in JSON format. For example:
	`azgi -json -c "select distinct ?name from <tickit> where {?event <eventname> ?name} limit 3"`
	`{"head":{"vars":["name"]},"results":{"bindings":[{"name":{ "datatype":"http://www.w3.org/2001/XMLSchema#string","type":"typed-literal", "value":"La Rondine"}},{"name": {"datatype":"http://www.w3.org/2001/XMLSchema#string", "type":"typed-literal","value":"Il Trovatore"}},{"name":{ "datatype":"http://www.w3.org/2001/XMLSchema#string", "type":"typed-literal","value":"Ring Cycle"}}]}}`

Option	Description
-xml	Returns results in XML format.
-h **hostname**	Connects to a remote AnzoGraph server. For example, the following statement runs a query against AnzoGraph installed on host 10.104.55.27: `azgi -h 10.104.55.27 -c "select * from <tickit> where {?s ?p ?o} limit 100"`
-p **port**	Used to connect to AnzoGraph on a non-default port or when using the azgi CLI from a remote server.
-timer	Reports query execution time in milliseconds.
-silent	Suppresses the query output.
-max_colwid **width**	Overrides the default maximum column width of 50 characters for tabular query results. For example, for a data set with long graph names, you can expand column width to view the entire name: `azgi -max_colwid 10000 -c "select ?g where {graph ?g {?s ?p ?o}} limit 100"` **Note**: Using the -wide option described below is equivalent to "max_colwid 60000."
-wide	Increases the column width for tabular query results from the default 50 characters to 60,000 characters. Equivalent to "max_colwid 60000."
-set **parameter=value**	Sets or changes parameter values in query files. For example this command runs the query in the query_summary.rq file with the $query parameter set to 2: `azgi -set query=2 -f query_summary.rq`

Option	Description

-v	Displays verbose output such as client connection details. For example:
	`azgi -v -c "select distinct ?p from <tickit> where {<person1> ?p ?o}"`
	`do_connect: connecting to host=localhost port=7070`
	`POST /sparql HTTP/1.1`
	`Host: Anon`
	`Accept: application/xml, text/html`
	`User-Agent: azgi`
	`Connection: keep-alive`
	`Content-Length: 26`
	`Content-Type: application/sparql-query`
	`...`

Changing AnzoGraph Configuration Settings

The default AnzoGraph system configuration is optimized for most AnzoGraph installations. If Cambridge Semantics Support recommends that you change the configuration, you can edit the configuration file, ***install_path*/config/settings.conf**, to modify or add settings. Each time you start the database, AnzoGraph reads this file and stores the configuration in memory. **On a cluster, change settings.conf on the leader server only**. See the AnzoGraph System Settings Reference for information about the units of measurement for the settings as well as any special instructions.

- To modify an existing setting value, uncomment the line and edit the value portion of the line `setting_name=`**`value`**.
- To add settings to settings.conf, add the setting and new value in the format below. Type each setting and value pair on a new line.

  ```
  setting_name=value
  ```

  Note AnzoGraph applies settings from the top to the bottom of the file. If the same setting appears more than once, AnzoGraph applies the value for the last instance of the setting. The last instance overrides any previous instances.

- To revert AnzoGraph to a previous configuration from a backup file, rename the existing settings.conf file and then change the name of the desired backup file to **settings.conf**.

Important After you change settings.conf, you must restart AnzoGraph for the settings to take effect. See Starting and Stopping AnzoGraph for instructions.

Relocating AnzoGraph Directories

Follow the instructions in this section to designate alternate locations for certain directories included in the AnzoGraph installation. You have the option to relocate the **persistence** directory where the system saves the data in memory to the file system, the **internal** directory where the system saves database-related files such as logs and generated code, and the **spill** directory where the system saves any temporary query files that spill to disk.

You can change the settings described in this section at any time. Once you restart the database, AnzoGraph starts saving any new files in the directory locations that you specify. **The system does not relocate any existing directories or files, however. You can move the existing files manually if needed.**

1. Stop the database.
2. **On the leader node**, open the AnzoGraph settings file, **settings.conf**, in a text editor. The file is in the `install_path/config` directory.
3. Uncomment the lines for any of the following settings in settings.conf. Then edit the value portion of *setting=value* to specify the desired directory.
    - internal_directory: The directory where you want AnzoGraph to save internal database-related files such as generated code, logs, and query plans.
    - persistence_directory: The directory where you want AnzoGraph to save data when writing data to disk.
    - spill_directory: The directory where you want the AnzoGraph to save any temporary query files that spill to disk.

      Important AnzoGraph uses O_DIRECT to read the spill files into the database. If you relocate the spill directory, make sure to place it on an ext4 file system that supports O_DIRECT.
4. Save and close settings.conf.
5. Restart the database to apply the configuration change.

Enabling AnzoGraph Persistence

By default, Anzo manages the data in AnzoGraph by automatically reloading graphmart data into memory any time you restart AnzoGraph. Users have the option, however, to enable AnzoGraph's save data to disk option where AnzoGraph saves the data in memory to disk after every transaction. Each time AnzoGraph is restarted, the persisted data is automatically loaded back into memory.

This topic provides instructions for enabling AnzoGraph persistence.

Important Considerations

Before enabling persistence, consider the following important notes:

- In general, each AnzoGraph server needs access to about twice as much disk space as RAM on the server. By default, AnzoGraph saves data to the `install_path/persistence` directory on the local file system. You can also configure AnzoGraph to save data to a mounted file system. For more information, see Relocating

AnzoGraph Directories.

- Persisted data is unique to each AnzoGraph version and cannot be re-used after an upgrade. If you upgrade AnzoGraph and persistence is enabled, the database will not start until it is reinitialized to remove the persisted data. See Reinitializing the Database for instructions.
- When persistence is enabled, transactional workloads that perform many concurrent write operations may experience a performance degradation due to the overhead of writing the data from each transaction to disk.

Enabling Persistence

Follow the steps below to enable the AnzoGraph save to disk option.

1. Stop the database.
2. **On the leader node**, open the AnzoGraph settings file, **settings.conf**, in a text editor. The file is in the `install_path/config` directory.
3. In settings.conf, find the following line in the file:

   ```
   enable_persistence=false
   ```

4. Change the enable_persistence value to **true**:

   ```
   enable_persistence=true
   ```

5. Save and close settings.conf.
6. Restart the database to apply the configuration change.

After each transaction, AnzoGraph saves the data in memory to disk in the location specified in the persistence_directory setting.

Note To avoid unnecessary reloads, make sure that the AnzoGraph connection in Anzo is configured to enable the **Use AnzoGraph persistence if available** option. See Connecting to AnzoGraph for more information.

Enabling Root User Access

Since AnzoGraph offers features such as user-defined extensions, it is not secure software certified and Cambridge Semantics recommends that you do not install or run AnzoGraph as the root user. To aid in ensuring that AnzoGraph is not run using root privileges, AnzoGraph is configured by default to disallow system startup by the root user. If you want to enable root user access, follow the instructions below.

1. Stop the database.
2. **On the leader node**, open the AnzoGraph settings file, **settings.conf**, in a text editor. The file is in the `install_path/config` directory.

3. In settings.conf, uncomment the `enable_root_user=false` line and change the value to true:

```
enable_root_user=true
```

4. Save and close settings.conf.
5. Restart the database to apply the configuration change.

Ignoring Missing Graphs

By default, AnzoGraph returns a "No such graph or view" error and aborts the query if a query references a graph that does not exist. You can configure AnzoGraph to conform to the SPARQL specification and return an empty result instead of an error, however, if a query references a missing graph. Follow the instructions below to

1. Stop the database.
2. **On the leader node**, open the AnzoGraph settings file, **settings.conf**, in a text editor. The file is in the `install_path/config` directory.
3. In settings.conf, uncomment the `enable_unbound_variables=false` line and change the value to true:

```
enable_unbound_variables=true
```

4. Save and close settings.conf.
5. Restart the database to apply the configuration change.

Note In addition to allowing queries that reference non-existent graphs to succeed, setting enable_unbound_variables to true also configures AnzoGraph to ignore unbound variables elsewhere in queries. For example, by default (when enable_unbound_variables=false), if a query includes a variable in the SELECT list that is not referenced in a WHERE clause pattern, AnzoGraph aborts the query and returns a "Named variable not in contained WHERE clause" error. When enable_unbound_variables=true, AnzoGraph does not warn the user about unbound variables. Instead, the results are empty for the unbound variable. For example:

```
SELECT ?unbound ?person ?name
FROM <http://cambridgesemantics.com/people>
WHERE {?person <http://cambridgesemantics.com/people#firstname> ?name}
LIMIT 5
```

```
 unbound | person      | name
---------+-------------+---------
         | person35632 | Ross
         | person20216 | Quin
         | person35859 | Kellie
         | person2551  | Maris
         | person24963 | Madonna
5 rows
```

AnzoGraph System Settings Reference

This section provides reference information for each of the AnzoGraph system configuration settings. For instructions on changing settings, see Changing AnzoGraph Configuration Settings.

Setting	Description	Default Value (Type)
anzo_protocol_port	The Anzo protocol (gRPC) port for secure communication between AnzoGraph and Anzo.	**5700** (int) **The Anzo protocol port is not configurable at this time**
enable_persistence	Whether to enable AnzoGraph's save data to disk option. For more information, see Enabling AnzoGraph Persistence.	**false** (boolean)
enable_root_user	Whether to allow a user running with root privileges to start AnzoGraph. For more information, see Enabling Root User Access.	**false** (boolean)
enable_sparql_protocol	Whether to enable the HTTP SPARQL protocol service. The sparql_protocol_port setting controls the port to use. **Note**: Enabling the SPARQL HTTP protocol opens the standard SPARQL-compliant HTTP endpoint. Unlike the Anzo protocol endpoint, the SPARQL HTTP endpoint is not secured.	**false** (boolean)
enable_ssl_protocol	Whether to enable the HTTPS SPARQL protocol service. The ssl_protocol_port setting controls the port to use. **Note**: Enabling the SPARQL HTTPS protocol opens the standard SPARQL-compliant HTTPS endpoint. Unlike the Anzo protocol endpoint, the SPARQL HTTPS endpoint is encrypted but not authenticated.	**false** (boolean)
internal_directory	The directory where AnzoGraph should save internal database-related files such as generated code, logs, and query plans. For more information, see Relocating AnzoGraph Directories.	Not set (char) The default directory for internal data is ***install_path*/internal**.

Setting	Description	Default Value (Type)
persistence_directory	The directory where AnzoGraph should save data when it is persisted to disk. For more information, see Relocating AnzoGraph Directories.	Not set (char) The default directory for persisted data is ***install_path*/persistence**.
sparql_protocol_port	SPARQL service HTTP port to use if enable_sparql_protocol is **true**.	**7070** (int)
spill_directory	The directory where AnzoGraph should save temporary query files that spill to disk. For more information, see Relocating AnzoGraph Directories. **Important**: AnzoGraph uses O_DIRECT to read the spill files into the database. If you relocate the spill directory, make sure to place it on an ext4 file system that supports O_DIRECT.	Not set (char) The default directory for spill files is ***install_path*/spill**.
ssl_protocol_port	SPARQL service HTTPS port to use if enable_ssl_protocol is **true**.	**8256** (int)
stop_timeout	The number of seconds to wait for queries to finish before stopping the database.	**30** (int)

Related Topics

Changing AnzoGraph Configuration Settings

Generating Diagnostic Files with the System Manager

When Cambridge Semantics Support requests AnzoGraph diagnostic files for troubleshooting an issue, you can use the AnzoGraph system manager to generate the required system information. If you encounter an error and the database remains running, you run an XRAY command to produce the diagnostic files. If you encounter an error that crashes the database, you run a CRASHFETCH command to produce a "crashdump" that includes the diagnostic files. This section provides instructions for generating the diagnostic files using the AnzoGraph system manager. For instructions on retrieving diagnostic files from the Anzo console, see Retrieving AnzoGraph Diagnostic Files.

- Generating an X-ray on a Running Database
- Generating a Crashdump after a Crash

Generating an X-ray on a Running Database

If you encounter an error and the database remains running, run the following command to take an x-ray from the command line on the AnzoGraph leader server. This command creates a tarball that includes the necessary diagnostic files:

```
/install_path/bin/azgctl -xray /path/name.xray
```

- **path**: The location on the server where you want to save the tarball.
- **name**: The name for the tarball. The name must be unique; AnzoGraph will not overwrite existing files.
- **.xray**: All x-ray files must be named with the .xray extension.

For example, this command runs an x-ray on the leader server:

```
/opt/anzograph/bin/azgctl -xray /tmp/query_error.xray
```

Generating a Crashdump after a Crash

If you encounter an issue that stops the database, AnzoGraph automatically generates diagnostic files for Support. Follow the instructions below to retrieve the files after a crash.

Note: The database does not need to be running to collect the crashdump.

1. Run the following command on the leader server to view a list of the available crash diagnostics.

   ```
   /install_path/bin/azgctl -crashlist
   ```

   The results show a list of available crash dumps by timestamp. For example:

   ```
   Crash ID               Time
   ----------------------------------
   520460982     2017-06-28 20:30:35
   520457655     2017-06-28 20:28:25
   ```

2. Run the following command to retrieve the appropriate crash files. This command creates a tarball that includes the necessary files:

   ```
   ./gqe/bin/azgctl -crashfetch crash_id /path/name.xray
   ```

   - **crash_id**: The ID for the crash that you want to retrieve, as shown in the crashlist from the previous step. To automatically retrieve the latest crash files, omit the crash_id.
   - **path**: The location on the server where you want to save the tarball.
   - **name**: The name for the tarball. The name must be unique; AnzoGraph will not overwrite existing files.
   - **.xray**: All crashdumps files must be named with the .xray extension.

   For example, this command runs a crashfetch to capture the diagnostics with the ID 520457655:

```
/opt/anzograph/bin/azgctl -crashfetch 520457655 /tmp/query_crash.xray
```

This command captures the most recent crash diagnostic files:

```
/opt/anzograph/bin/azgctl -crashfetch /tmp/query_crash.xray
```

Tip You can run the following command to remove all crash dumps from the server.

```
/install_path/bin/azgctl -crashtoss
```

Anzo Admin CLI

The Anzo command line interface (CLI) utility, called **anzo**, is an advanced administration tool for managing Anzo. It is primarily used for migrations and deployments. The topics in this section provide information about the CLI.

Note To script user interface operations or control Anzo with the CLI, please contact Cambridge Semantics.

Setting up the Admin CLI

Important The anzo CLI is an advanced administration tool for managing Anzo. It is primarily used for migrations and deployments. To script user interface operations or control Anzo with the CLI, please contact Cambridge Semantics.

This topic provides instructions for configuring the admin command line interface, **anzo**, and viewing the help menu. The anzo client is in the *install_path*/Client directory.

Adding the CLI to the Anzo Service User PATH

Follow the instructions below to configure the PATH environment variable to include the Client directory so that you call the anzo CLI from anywhere.

1. If necessary, run the following command to become the Anzo service user:

   ```
   sudo su - anzo_user_name
   ```

   For example:

   ```
   sudo su - anzo
   ```

2. Open **~/.bash_profile** in a text editor.
3. Change the PATH to the following value:

   ```
   PATH=$PATH:$HOME/.local/bin:$HOME/bin:Anzo_install_path/Client
   ```

   For example:

```
PATH=$PATH:$HOME/.local/bin:$HOME/bin:/opt/Anzo/Client
```

4. Save and close the file, and then run the following command:

```
source ~/.bash_profile
```

5. Type **anzo** to verify that you can access the CLI. For example:

```
[anzo@anzo-server ~]$ anzo
Anzo Command Line Client.
Copyright (c) 2017 - 2019 Cambridge Semantics Inc and others.
All rights reserved.
Version: 4.4.0.r201910171220
Type anzo help for usage
```

Configuring the CLI

Follow the instructions below to configure a settings file that specifies the default Anzo CLI configuration values for parameters such as host, port, user, and password. Specifying these details in the settings file eliminates the need to include those options in subsequent commands.

To create and populate the settings file, **settings.trig**, in your home directory, run the following command:

```
anzo setup options
```

Where *options* include the following choices:

```
-beep , --beep                    beep when command is completed
-ds , --datasource <datasource>  URI of the datasource to query, if other than primary
datasource.
                                  Option not available for dataset queries.
-h , --host <hostname>            anzo server hostname
-http , --http                    Use http connection to server.
-p , --port <int>                 anzo server port
-pause , --pause-exit             Wait for a user key entry before an abnormal exit.
-ssl , --use-ssl                  Use SSL for connection.
-t , --timeout <timeout>          override the default 30 second timeout for operations
-timer , --timer                  Print out the total operation time
-trace , --show-trace             Show stack trace for errors.
-trust , --trust-all              Trust all certificates including invalid ones
-u , --user <string>              username to connect with
-w , --password <string>          user's password
-x , --exclude-prefixes          Do not use prefixes defined in user settings to expand
options,
                                  arguments, or to write RDF.
-z , --settings <file>            override the default settings file location
```

For example:

```
anzo setup -h localhost -p 61616 -u sysadmin -w @nz0
```

Anzo creates the settings.trig file in the `~/user/.anzo` directory. You can edit the file as needed. The installation also includes a sample settings file, **settings_example.trig**, in the `Client` directory. You can view the sample file for reference. For example:

```
### standard prefixes
@prefix foaf      : <http://xmlns.com/foaf/0.1/> .
@prefix rdfs      : <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dc        : <http://purl.org/dc/elements/1.1/> .
@prefix xsd       : <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf       : <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
#### anzo prefixes:
@prefix cli : <http://openanzo.org/cli/> .
@prefix system : <http://openanzo.org/ontologies/2008/07/System#> .
@prefix anzo : <http://openanzo.org/ontologies/2008/07/Anzo#> .
@prefix ld : <http://cambridgesemantics.com/ontologies/2009/05/LinkedData#> .
@prefix anzowt : <http://cambridgesemantics.com/ontologies/2009/05/AnzoWebToolkit#> .
@prefix reg : <http://cambridgesemantics.com/registries/> .
@prefix ontserv : <http://cambridgesemantics.com/semanticServices/OntologyService#> .
@prefix ldserv : <http://cambridgesemantics.com/semanticServices/LinkedData#> .
cli:config {
  cli:config
#       system:user "" ;
#       system:password "" ;
  system:timeout  "0";
  system:useSsl "false";
  system:port "61616";
  system:keystoreFile "${ANZO_CLI_HOME}/../Common/ssl/client.ks";
  system:keystoreType "JCEKS";
  system:keystorePassword "p@ssw0rd";
  system:truststoreFile "${ANZO_CLI_HOME}/../Common/ssl/client.ts";
  system:truststoreType "JCEKS";
  system:truststorePassword "p@ssw0rd";
  .
}
```

Viewing the CLI Help Menu

The CLI help menu lists all of the available subcommands. To view the subcommands, run **anzo help**.

```
usage: anzo <subcommand> [options] [args]
Anzo Command Line Client.
Type 'anzo help <subcommand>' for help with a specific subcommand.
Available subcommands:
```

```
acls               Ensure the graphs in a dataset inherit their ACLs from the dataset
analyze            Provides several flavors of analysis for Anzo request/response logs
call               Calls an anzo semantic service and prints the service response to the
console
collapse           Collapse all URI arguments to prefixed URIs (CURIEs) using user defined
prefixes
collapseGraph      In specified graph(s), collapse object properties with only one literal
value
                   into a datatype property
convert            Converts between the various RDF file formats
count              Counts the statements in an RDF file
create             Creates named graphs in the repository from the provided RDF
csv                Export instances of an ontology class with all of their property values
deploy             Import, export, or delete a linked data set and related components
deregister        Deregister given resource from appropriate registries based on rdf:type of
resource
expand             Expands all prefixed URI (CURIE) arguments to expanded URIs using
                   user defined prefix map
find               Retrieves statements from the server via simple pattern find
gen                Generates code for the ontologies as supplied by the input RDF or
arguments
get                Retrieves named graphs from the server
graph2lds          Creates a Linked Data Set from the statements in a graph(s)
import             Imports statements into the repository, creating graphs in the
repository as needed
inspectOntology    Inspects a dataset for an ontology
link               Link an excel workbook using a layout
load               Loads file based linked datasets
loadXML            Imports xml as statements into a graph in the repository as needed
ls                List resources from appropriate registries based on type of resource
play               Play back a sequence of recorded requests
query              Executes a SPARQL query against the repository or a local RDF file
rdfformats         Show available rdf formats
register           Register given resource to appropriate registries based on rdf:type of
resource.
                  Supported types:[
                        http://cambridgesemantics.com/ontologies/2009/05/LinkedData#LinkedDataSet
                        http://cambridgesemantics.com/ontologies/2009/05/LinkedData#LinkedDataCollection

http://cambridgesemantics.com/ontologies/2009/05/LinkedData#LinkedDataCollectionInstance
                        http://www.w3.org/2002/07/owl#Ontology
                        http://openanzo.org/ontologies/2008/07/SemanticService#SemanticService
                        http://cambridgesemantics.com/ontologies/2009/05/Spreadsheets#LinkedWorkbook
                        http://cambridgesemantics.com/ontologies/2009/05/AnzoWebToolkit#Component
                        http://cambridgesemantics.com/ontologies/Graphmarts#Graphmart
                        http://cambridgesemantics.com/ontologies/Graphmarts#Step
```

```
                        http://cambridgesemantics.com/ontologies/Graphmarts#Layer
                        http://cambridgesemantics.com/ontologies/Graphmarts#View
                        ]
remove              Removes named graphs from the repository
replace             Replaces named graphs in the repository with the provided RDF
reset               Resets the repository, replacing all contents of repository with rdf
provided
retrieve            Retrieves content from the binary store and saves it in a local file
setup               Set up settings.trig file
sortedConvert       Converts between the various RDF file formats
store               Stores a local file in the Anzo server's binary store
union               Unions RDF from the arguments and optionally from STDIN as well
update              Updates existing graphs in the repository
uploadBundle        Upload bundle to server
uploadCertificate Upload trusted certificate to server
watch               Listens for changes to a graph and prints them out
xray               Export system tables into trig file


URI arguments to commands may either be fully qualified URIs ("http://...") or prefixed
URIs ("dc:title").
The prefix mapping is defined in the users settings file.
User settings are loaded from a user's "~/.anzo/settings.trig" file.
See documentation for details.
```

To view the help for a specific subcommand, run **anzo help *command_name***. For example, the following command displays help for the find command:

```
[user@anzo Client]# ./anzo help find
usage: anzo find [options] [NAMED-GRAPH-URI...]
Retrieves statements from the server via simple pattern find.
-beep , --beep                       beep when command is completed
-ds , --datasource <datasource>      URI of the datasource to query, if other than primary
datasource.
                                     Option not available for dataset queries.
-f , --output-file <file>            write the find results to a file
-h , --host <hostname>               anzo server hostname
-http , --http                       Use http connection to server.
-lang , --literal-language <string> The literal language
-lit , --literal-object <string>     The literal object of find pattern
-n , --count                         Outputs only the total number of matching statements
-o , --output-format <rdf-Format>    Override the default RDF format associated with the
RDF output(s)
-p , --port <int>                    anzo server port
-pause , --pause-exit                Wait for a user key entry before an abnormal exit.
-pred , --predicate <URI>            The predicate of find pattern
```

```
-pretty , --pretty-print            PrettyPrint output (currently only json)
-ssl , --use-ssl                    Use SSL for connection.
-sub , --subject <subject>          The subject of find pattern
-t , --timeout <timeout>            override the default 30 second timeout for operations
-timer , --timer                    Print out the total operation time
-trace , --show-trace               Show stack trace for errors.
-trust , --trust-all                Trust all certificates including invalid ones
-type , --literal-datatype <URI>    The literal datatype
-u , --user <string>                username to connect with
-uri , --uri-object <URI>           The uri object of find pattern
-w , --password <string>            user's password
-x , --exclude-prefixes             Do not use prefixes defined in user settings to expand
options,
                                    arguments, or to write RDF.
-z , --settings <file>              override the default settings file location

'help rdfformats' for list of available RDF formats.
Filename arguments default to the file format matching their filename extension.
STDIN and STDOUT default to 'trig'.
```

Querying Graphmart Data

Important The anzo CLI is an advanced administration tool for managing Anzo. It is primarily used for migrations and deployments. To script user interface operations or control Anzo with the CLI, please contact Cambridge Semantics.

This topic provides information about using the anzo CLI to query graphmart data in AnzoGraph.

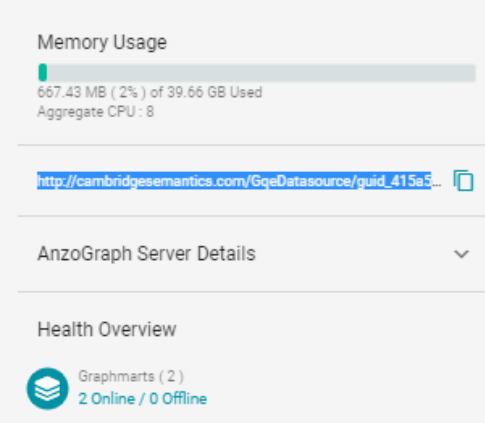
Use the **query** subcommand to access the data in graphmarts that are loaded in AnzoGraph:

```
anzo query "query_text" -ds AZG_URI -dataset graphmart_URI
```

If you saved the query in a file, run the following command to run the query in the file:

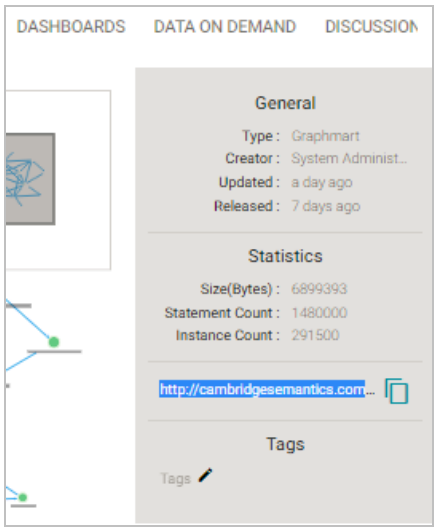
```
anzo query -f filename.rq -ds AZG_URI -dataset graphmart_URI
```

Where *filename*.rq is the path to and name of the query file and *AZG_URI* is the Datasource URI shown on the **Administration** > **AnzoGraph** screen in the Anzo console. For example:



And *graphmart_URI* is the URI for graphmart. To view the URI for a graphmart:

1. Click **Graphmarts** in the Anzo console.
2. On the Graphmarts screen, click the graphmart that you want to query.
3. On the details screen for the graphmart, you can view the graphmart URI in the statistics section. For example:



4. Click the clipboard icon ( ) to copy the graphmart URI to your clipboard.

Examples

The example below queries a data set to list its classes:

```
anzo query "SELECT DISTINCT ?p WHERE { ?s ?p ?o.} LIMIT 100"
-ds http://cambridgesemantics.com/GqeDatasource/guid_
b833b32453694342c7bbc22422035e07
-dataset http://cambridgesemantics.com/Graphmart/f4bc354ebe9540329eef561f66e42454
```

This example runs a query in a file:

```
anzo query -f /home/user/queries/classes.rq
-ds http://cambridgesemantics.com/GqeDatasource/guid_
b833b32453694342c7bbc22422035e07
-dataset http://cambridgesemantics.com/Graphmart/f4bc354ebe9540329eef561f66e42454
```

Accessing a Graph's Metadata

Important The anzo CLI is an advanced administration tool for managing Anzo. It is primarily used for migrations and deployments. To script user interface operations or control Anzo with the CLI, please contact Cambridge Semantics.

Each graph has a metadata graph associated with it. The metadata graph includes details such as ACL information, the last modified date, and which user created and modified the graph. To include the metadata graph when you retrieve graph details, use the **get** subcommand with the **-m** option:

```
anzo get -m URI
```

The **-m** option indicates that you want to see the metadata graph for the specified URI. For example, the following command retrieves the metadata graph for a graphmart:

```
anzo get -m
http://cambridgesemantics.com/Graphmart/89baf53cc5644600961778c88bd3d7fd
```

In addition to showing the graphmart details for the <http://cambridgesemantics.com/Graphmart/89baf53cc5644600961778c88bd3d7fd> graph, the results include the additional metadata for the graph:

```
...
<http://openanzo.org/metadataGraphs
(http%3A%2F%2Fcambridgesemantics.com%2FGraphmart%2F89baf53cc5644600961778c88bd3d7fd)>
{
  <http://cambridgesemantics.com/Graphmart/89baf53cc5644600961778c88bd3d7fd> a
anzo:NamedGraph ;
    anzo:createdBy <http://openanzo.org/system/internal/sysadmin> ;
    anzo:lastModifiedByUser <http://openanzo.org/system/internal/sysadmin> ;
    anzo:created "2020-03-24T17:25:48.004Z"^^xsd:dateTime ;
    anzo:datasource datasource:systemDatasource ;
...
}
```

Specifying an Output Format

Important The anzo CLI is an advanced administration tool for managing Anzo. It is primarily used for migrations and deployments. To script user interface operations or control Anzo with the CLI, please contact Cambridge Semantics.

The anzo CLI enables you to request results in the following formats: TriG (default), RDF, RDFS, XML, NT, N3, TTL, TriX, and JSON. To change the format for results, you use the -o option with Anzo subcommands such as find, get, query, call, and analyze.

For example, the following get subcommand returns data set details in XML format:

```
anzo get -o xml http://csi.com/FileBasedLinkedDataSet/059060234accd1d2d44b6bbb4207ee54
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:ld="http://cambridgesemantics.com/ontologies/2009/05/LinkedData#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://csi.com/DataLocation/059060234accd1d2d44b6bbb4207ee54">
<fileConnection xmlns="http://cambridgesemantics.com/ontologies/DataSources#"
    rdf:resource="http://cambridgesemantics.com/File_Connection/local"/>
<filePath xmlns="http://cambridgesemantics.com/ontologies/DataSources#">
    /nfs/data/store/LoadMovies_223d3/</filePath>
<isPrimary xmlns="http://cambridgesemantics.com/ontologies/DataSources#"
    rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">true</isPrimary>
<rdf:type
rdf:resource="http://cambridgesemantics.com/ontologies/DataSources#DataLocation"/>
<rdf:type
rdf:resource="http://cambridgesemantics.com/ontologies/DataSources#PathConnection"/>
</rdf:Description>
```

Developer Guide

The Developer Guide provides information about using the Anzo Java software development kit (SDK) to develop custom extensions for Anzo.

The Anzo system, including the SDK, is built using the Open Service Gateway Initiative (OSGi) as a packaging mechanism. OSGi is a Java packaging and runtime environment that enables Anzo to load and unload extensions easily. Certain components, such as Anzo Semantic Services, are packaged into an OSGi bundle and then loaded into the server. For an introductory description of OSGi, see What is OSGi? on the OSGi Alliance website. Note that a deep understanding of OSGi is not necessary for creating Anzo extensions with the Anzo SDK.

The topics in this section list the SDK requirements and provide instructions for deploying, testing, and using the Anzo SDK.

Deploying the Anzo Java SDK

This topic provides instructions for setting up an Anzo development environment using the Anzo software development kit (SDK) and Eclipse integrated development environment (IDE). The sample instructions below deploy the Anzo SDK in a Windows environment with Eclipse IDE for Java Developers Version 4.12.0. Anzo SDK and Eclipse can also be deployed on Linux and Mac operating systems.

Requirements

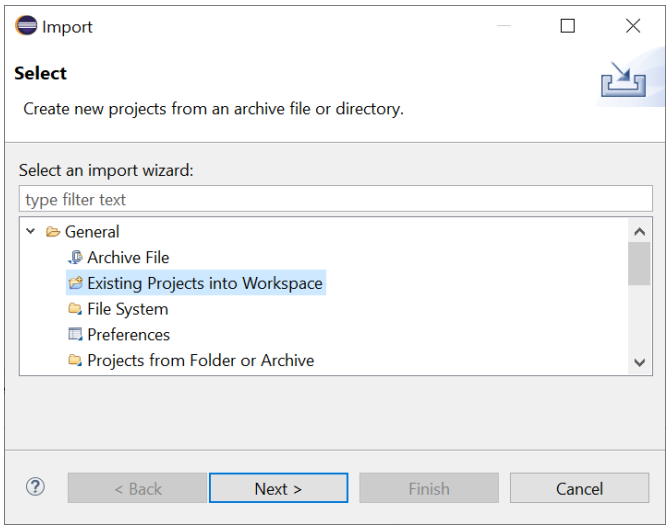
Make sure that the Anzo development server meets the requirements in Anzo Requirements. In addition, install the following programs for working with the Anzo Java SDK:

- Eclipse for Java Developers Version 4.7.3+: Install the **Eclipse IDE for Java Developers** or **Eclipse IDE for Enterprise Java Developers**.
- Java Runtime Environment Version 8: Eclipse and the Anzo SDK require JDK version 8. Cambridge Semantics tests with jdk1.8.0_181.

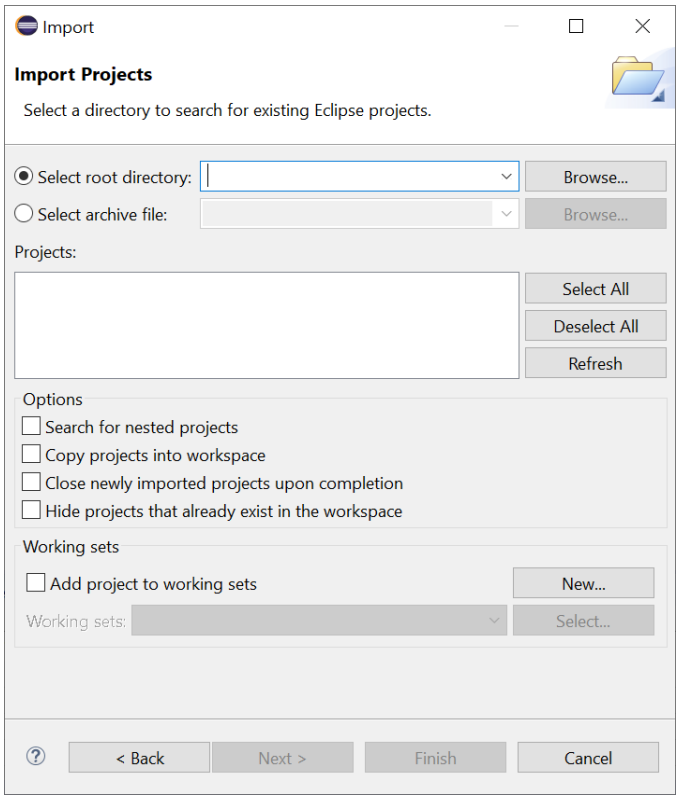
Deploying the Anzo SDK with Eclipse

Follow the instructions below to import the Anzo Java SDK to Eclipse and configure and test the environment.

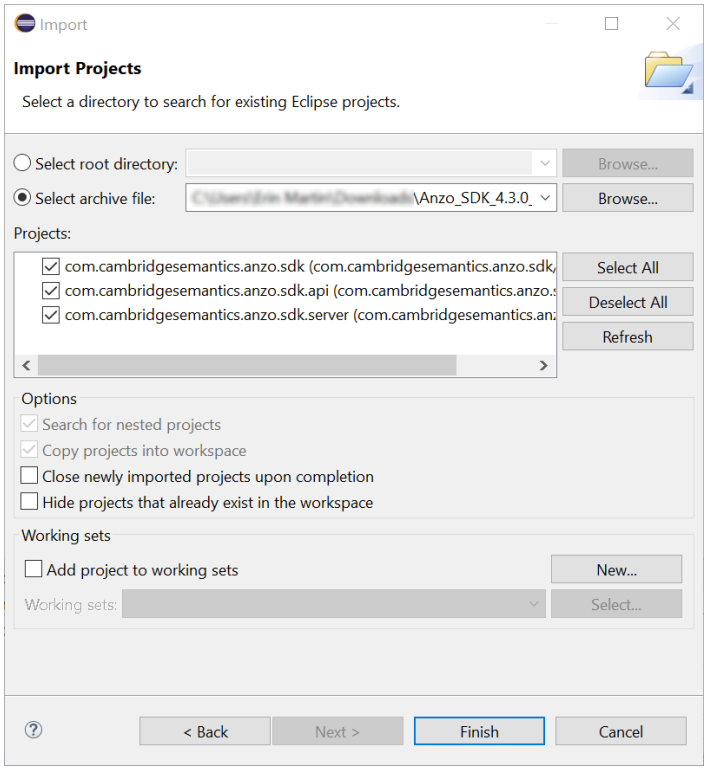
1. Download the Anzo SDK .zip file to the host server. Do not unpack the file.
2. In Eclipse, click the **File** menu and select **Import**. Eclipse opens the Import dialog box. For example:



3. In the Import dialog box, expand the **General** folder and select **Existing Projects into Workspace** and click **Next**. Eclipse opens the Import Projects dialog box. For example:



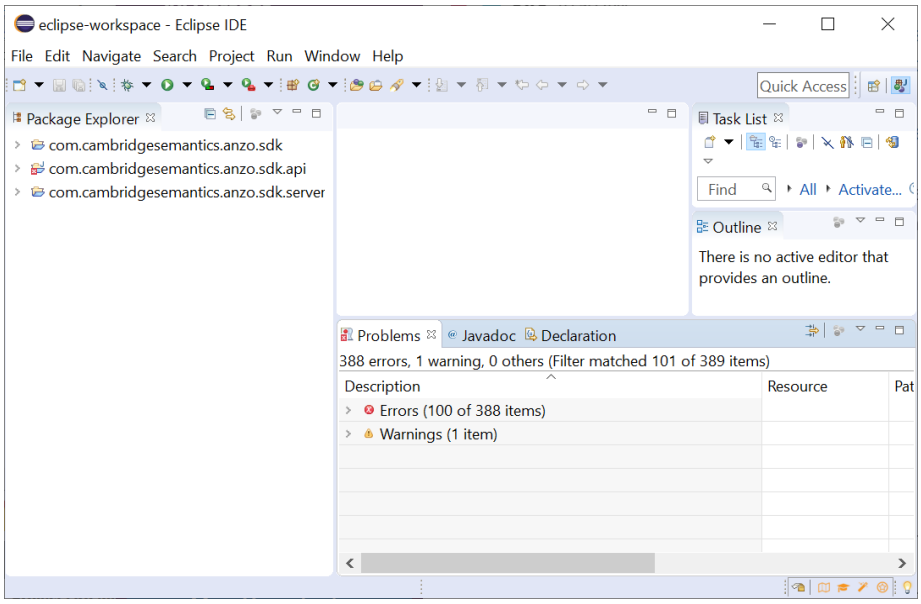
4. Select the **Select archive file** radio button and then browse to and select the Anzo SDK .zip file. Eclipse loads the .zip file and lists the contents in the Projects field. For example:



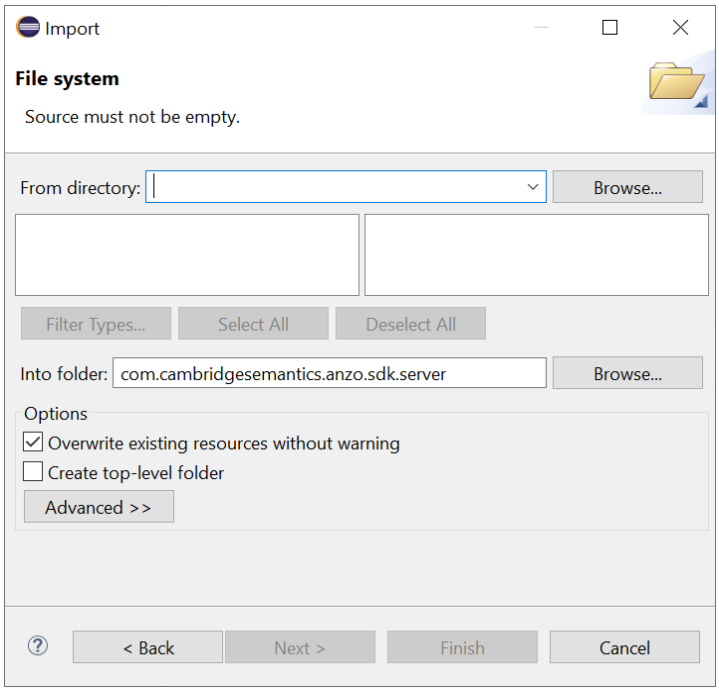
The Anzo SDK contains three projects:

- **com.cambridgesemantics.anzo.sdk**: This core project is required for creating solutions. It contains the Anzo libraries that provide the Anzo APIs and extension points as well as the libraries that enable Anzo to run in the development environment.
- **com.cambridgesemantics.anzo.sdk.server** This core project is required for creating solutions. It contains configuration files for running Anzo as well as a launcher for starting the Anzo server.
- **com.cambridgesemantics.anzo.sdk.api**: This is an example project that contains sample Java programs that illustrate several aspects of the Anzo client APIs. Each program is a simple example that demonstrates how to communicate with the Anzo server to read, write, and query data. See the comments in each example for an explanation of what each one demonstrates.

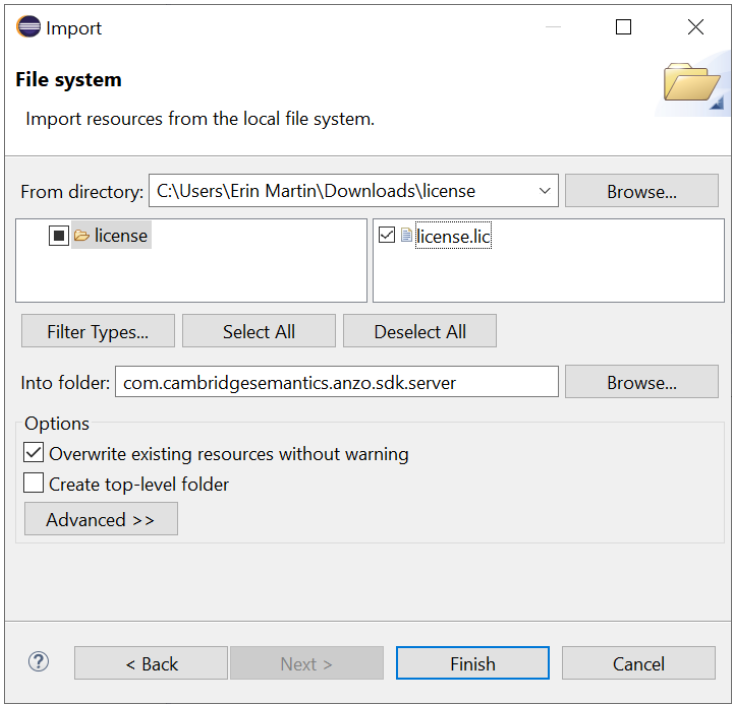
5. Click **Finish** to import the Anzo SDK .jar files. The process may take a few minutes. When the import is complete, Eclipse opens the workspace. At this point in the process, expect to see several errors in the workspace. For example:



6. Import your Anzo license:
   a. Make sure that you have a copy of the Anzo license on the server. If necessary, you can view and download a copy from the Cambridge Semantics Support Center.
   b. Rename the license file so its file extension is .lic. For example, **license.lic**.
   c. In the Eclipse Package Explorer, right-click **com.cambridgesemantics.anzo.sdk.server** and select **Import**.
   d. In the Import dialog box, expand the **General** folder and select **File System**. Then click **Next**. Eclipse opens the File System Import dialog box. For example:

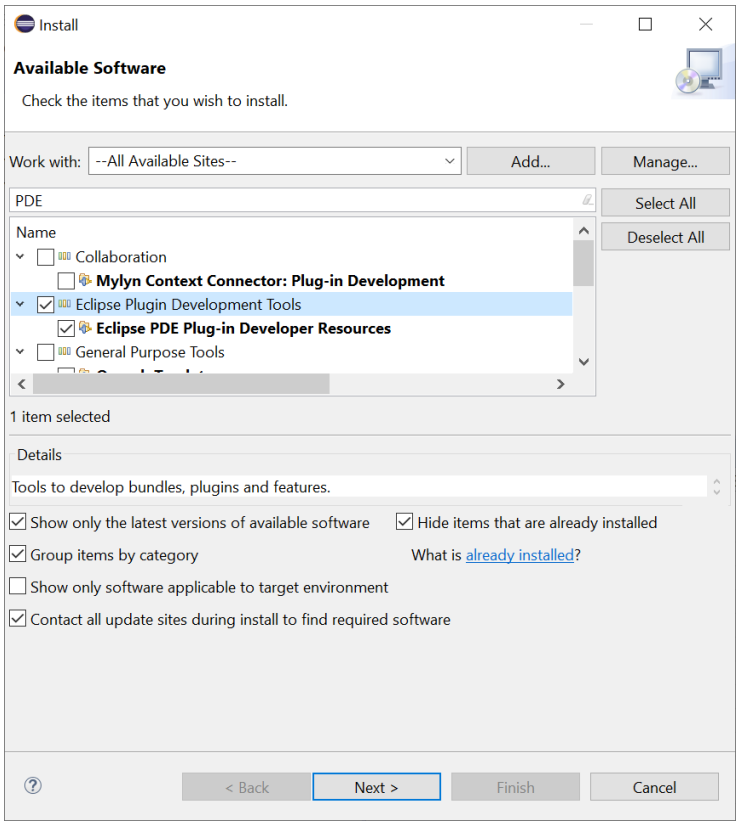


e. Click the **Browse** button next to the From directory field and select the directory that contains the license file. Eclipse displays the directory and its contents.



f. Select the license file in the right pane, and then click **Finish**.

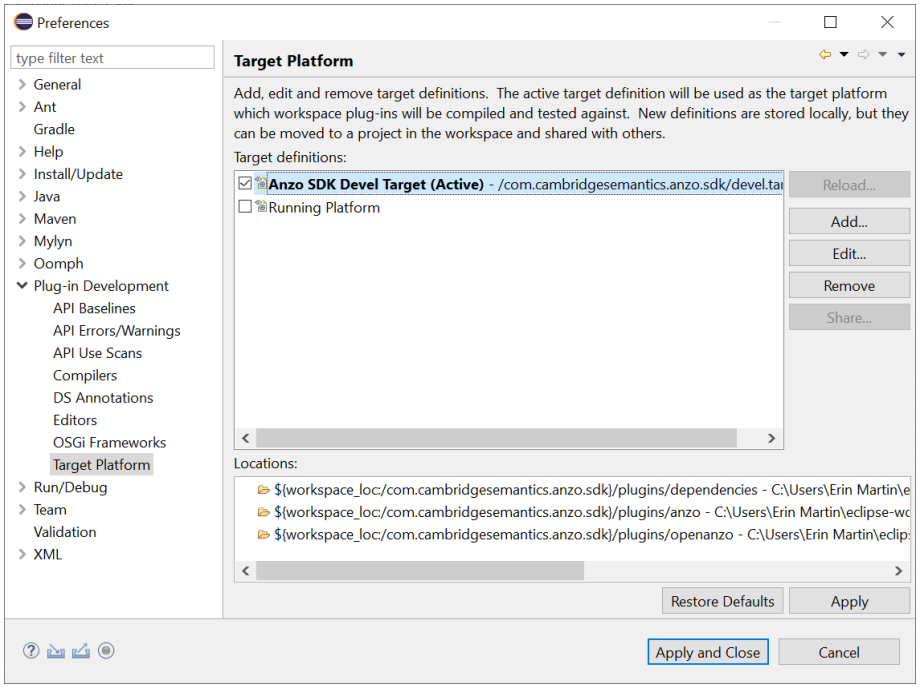
7. Install the Eclipse Plugin Development Tools:
    a. Click the **Help** menu and select **Install New Software**. Eclipse opens the Install dialog box.
    b. In the Install dialog box, click the **Work with** drop-down list and select **All Available Sites**. In the search field below the Work with field, type "PDE" and wait for Eclipse to find the plugin tools. Select the check-box next to **Eclipse Plugin Development Tools**, including **Eclipse PDE Plug-in Developer Resources**. For example:



    c. Click **Next** and accept the license agreement, then click **Finish**. Eclipse installs the software and then prompts you to restart the application.

8. After restarting Eclipse, load the Anzo SDK Target Platform:
    a. Click the **Window** menu and select **Preferences**.
    b. In the Preferences dialog box, expand **Plug-in Development** and select **Target Platform**.

c. In the Target Platform definitions, select the **Anzo SDK Devel Target** checkbox. For example:

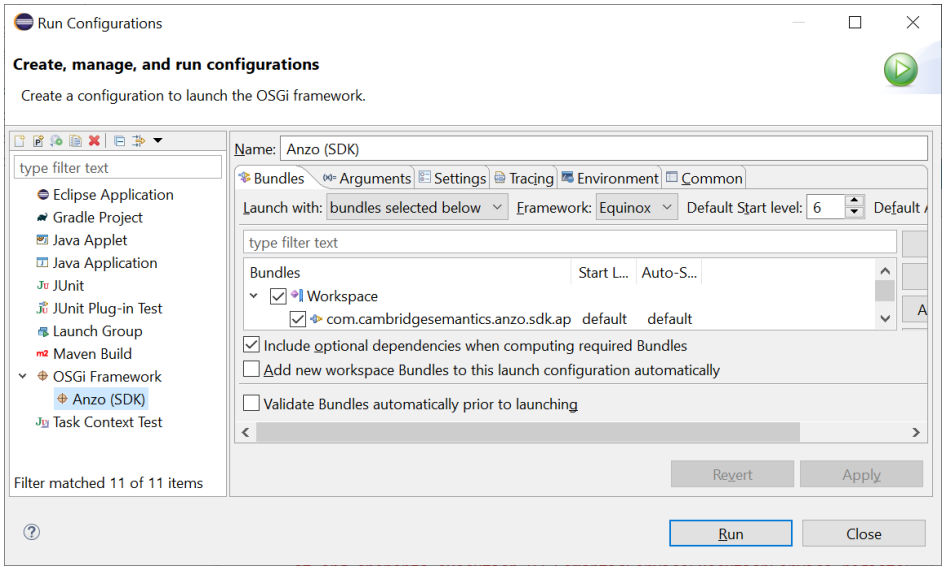


d. Click **Apply and Close**. Eclipse loads the Anzo SDK Target Platform.

9. Test the environment:

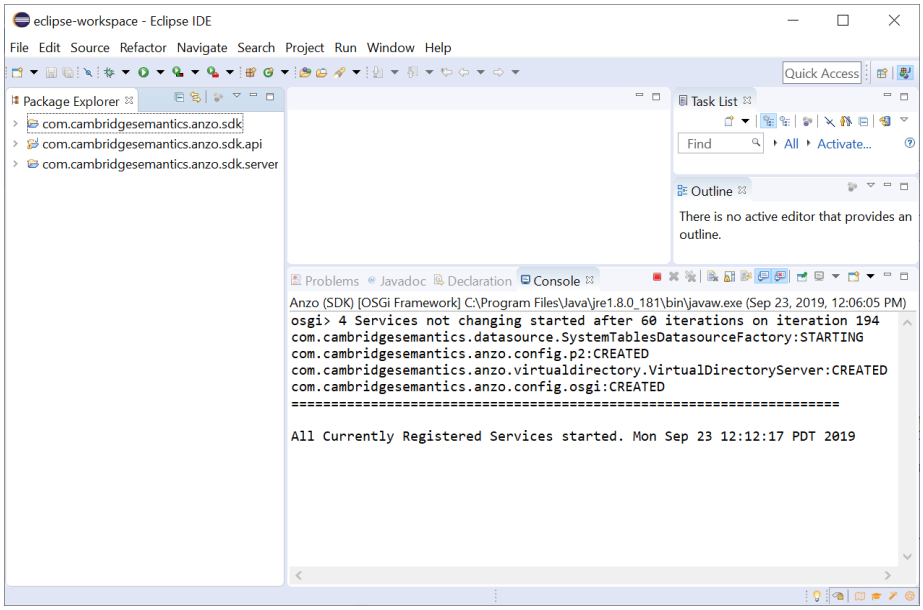
a. In the Eclipse workspace, click the **Run** menu and select **Run Configurations**. Eclipse opens the Run Configurations dialog box.

b. On the left side of the dialog box, expand the **OSGi Framework** folder and select **Anzo (SDK)**. For example:



c. Click **Run** to run the Anzo SDK target platform. A Console tab opens in Eclipse and shows the status messages. When Anzo starts, the console displays the message "All Currently Registered Services

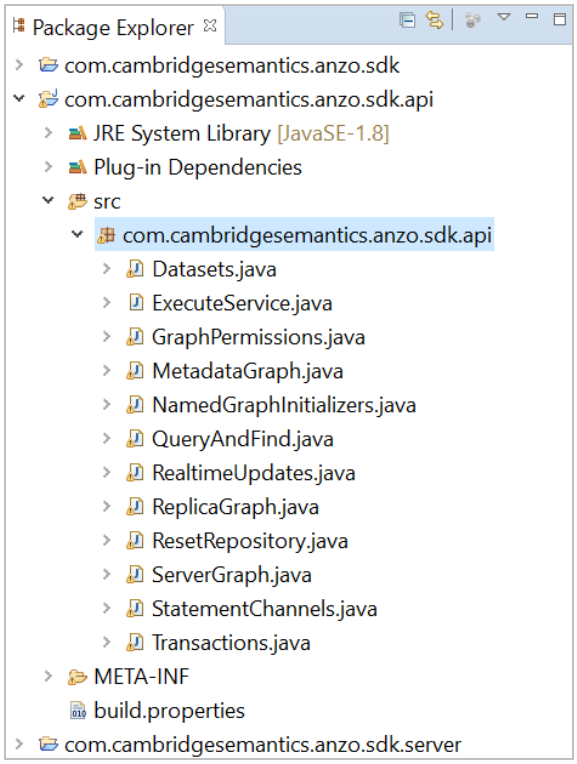
started." For example:



If Anzo fails to start, one of the common reasons for the failure is that one or more of the Anzo ports are in use by other software. See Firewall Requirements for information about the ports that Anzo uses.

Note If you deployed the Anzo SDK on Windows, Eclipse displays Spark-related error messages such as "`java.io.FileNotFoundException: Source '...\com.cambridgesemantics.anzo.sdk.server\spark' does not exist.`" The errors occur because Spark is not supported on Windows operating systems. You cannot run ETL jobs locally, but the errors do not affect the ability to develop Anzo extensions.

To explore the sample Java programs that are included in the Anzo SDK, expand the **com.cambridgesemantics.anzo.sdk.api** package in the Package Explorer. In the package, expand the **src** directory and then the **com.cambridgesemantics.anzo.sdk.api** directory to see the list of sample programs. For example:



To run a program, right-click the .java file and select **Run As** > **Java Application**. For more information about using the Anzo SDK, see the **Anzo Java SDK Guide.pdf** that is distributed in the SDK .zip file.

Troubleshooting

The topics in this section provide troubleshooting information for Anzo components.

Getting Information from Anzo Log Files

You can review the Anzo log files to get more detailed information about errors or to obtain more granular information about server operations. The server writes logs to the `Anzo/Server/logs` directory and adds timestamps to all logged statements. Major issues are logged in files with the suffix "error," and other server information is logged in files with the suffix "info." For information about viewing and managing Anzo logs, see Managing Anzo Server Logging.

Error Message Reference

This topic provides information about Anzo and AnzoGraph and error messages.

Anzo Error Messages

This section includes the possible causes and solutions for Anzo error messages. Click a message in the list below to view details about that error:

Application Service Failure

This message indicates that the Anzo server cannot bind to the Application Port defined on the Server Settings page in the Anzo console. The problem has two likely causes:

- Another program is bound to the defined Anzo Server Application Port.
- You are not running as the root user and lack the required permission.

To resolve this issue, make sure that no other application is running on the defined Application port and log in as the root user if Anzo is installed on a UNIX operating system.

Elasticsearch exception [type=circuit_breaking_exception, reason=[parent] Data too large, data for [<http_request>]...

This message indicates that the Elasticsearch heap size is not large enough to process the request. By default, Elasticsearch is configured to use a maximum heap size of 1 GB. Cambridge Semantics recommends that you increase the amount to 50% of the memory that is available on the server. To change the configuration, open the *elasticsearch_install_dir*`/config/jvm.options` file in an editor. At the top of the file, modify the **Xms** and **Xmx** values to replace the **1** with the new value. For example:

```
# Xms represents the initial size of total heap space
# Xmx represents the maximum size of total heap space

-Xms15g
-Xmx15g
```

Sparkler Exception: java.io.IOException: Unable to connect to provided ports 10000~10010

This message indicates that the Sparkler Livy RSC client ran out of the ports that it uses internally for running jobs. Increase the range of ports by adjusting the **livy.rsc.launcher.port.range** value in the **livy-client.conf** file. If you use the embedded Anzo Sparkler compiler, the file is in the `install_path/Server/spark/csi-livy-spark/conf` directory.

Cambridge Semantics recommends that you set **livy.rsc.launcher.port.range = 10000~10110**. Restart the Livy server after changing the configuration file.

AnzoGraph Error Messages

This section includes the possible causes and solutions for AnzoGraph error messages. Click a message in the list below to view details about that error:

- Exiting: Error - Cannot execute as user 'root'. To override this security protection, set 'enable_root_user=true': Invalid user id
- Invalid Certificate

Exiting: Error - Cannot execute as user 'root'. To override this security protection, set 'enable_root_user=true': Invalid user id

This message indicates that you tried to start AnzoGraph as the root user and root access is disabled. You can try the command again as a non-root user, or you can enable root access. For instructions, see Enabling Root User Access.

Invalid Certificate

This message indicates that you replaced the default AnzoGraph certificates with your own trusted certificates and the certificates are invalid. Certificates can be invalid because they expired or they were generated or signed incorrectly.

FAQ

This topic provides answers to frequently asked questions and includes references to more detailed information.

What is an Anzo data store?

An Anzo data store, also known as a graph data source, is a designated directory on the file store where Anzo can save the AnzoGraph load files that are generated during the ETL process. All installations require at least one graph data source. You can create one graph data source and configure all pipelines to write to that graph source (each ETL run automatically creates a new sub-directory under the graph data source) or you can create multiple graph data sources to use for different data sets. For information about creating graph sources, see Creating an Anzo Data Store.

What is the difference between a Dataset Pipeline and an ETL Pipeline?

Dataset pipelines are used to ingest data into Anzo. They produce new data sets in the Catalog and generate RDF files for loading data to AnzoGraph. All auto-ingested projects are dataset pipelines. For more information, see Creating a New Dataset Pipeline to Ingest Data into Anzo.

ETL pipelines do not ingest data into Anzo; they are used to ingest data into a file or another database. ETL pipelines do not generate a new data set entry in the Catalog or produce RDF load files for AnzoGraph. When you create mappings for ETL pipelines, you define a file schema or database as the target. For more information, see Creating a New ETL Pipeline to Move Data to a File or Database.

How do I update the data in Anzo if a file in my CSV data source changes?

If the data in a CSV file changes, the way that you update the data set in Anzo depends on the type of changes that were made to the file and the file system where the file is hosted. The table below provides guidance on the steps to take to update Anzo based on the type of content updates and the file location.

Note The instructions below assume that the updated file has the same name and location as the file that was originally uploaded.

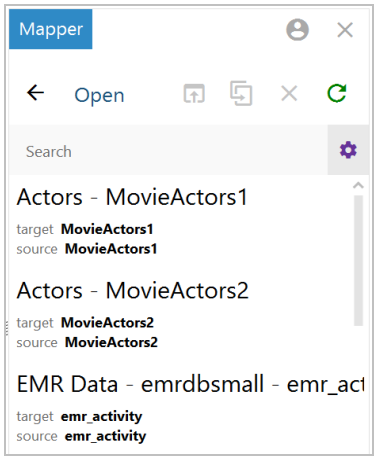
Update Type	File Location	Update Process
Added, deleted, or changed **rows** – Columns did not change	Uploaded from your Computer	1. In the Anzo application, view the CSV data source that contains the file to update. 2. On the Tables tab, click **Add New File** and select the file. The file is uploaded to Anzo and appears twice (or more if you have previously updated the file and re-uploaded it). This is normal; Anzo does not overwrite the original file. 3. Click **Import Pending Files** to import the updated file. 4. Re-publish the job for this file or the entire pipeline to update the existing data set. 5. Reload any graphmarts that include the updated data set and then refresh the affected Hi-Res Analytics dashboards to view the updated data.
	Selected from the File Store	1. Replace the file on the file store with the updated version of the file. 2. Re-publish the job for this file or the entire pipeline to update the existing data set. 3. Reload any graphmarts that include the updated data set and then refresh the affected Hi-Res Analytics dashboards to view the updated data.

Added, deleted, or changed **columns** and rows	Uploaded from your Computer	1. In the Anzo application, view the CSV data source that contains the file to update. 2. On the Tables tab, click **Add New File** and select the file. The file is uploaded to Anzo and appears twice (or more if you have previously updated the file and re-uploaded it). This is normal; Anzo does not overwrite the original file. 3. Click **Import Pending Files** to import the updated file. 4. Click **Ingest** and re-ingest the data source. 5. Re-publish the job for this file or the entire pipeline to update the existing data set. 6. Reload any graphmarts that include the updated data set and then refresh the affected Hi-Res Analytics dashboards to view the updated data.
	Selected from the File Store	1. Replace the file on the file store with the updated version of the file. 2. In the Anzo application, view the CSV data source that contains the file to update. On the Tables tab, select the checkbox next to the file to re-import. Then click the **Import Selected** button to import the updated file. 3. Click **Ingest** and re-ingest the data source. 4. Re-publish the job for this file or the entire pipeline to update the existing data set. 5. Reload any graphmarts that include the updated data set and then refresh the affected Hi-Res Analytics dashboards to view the updated data.

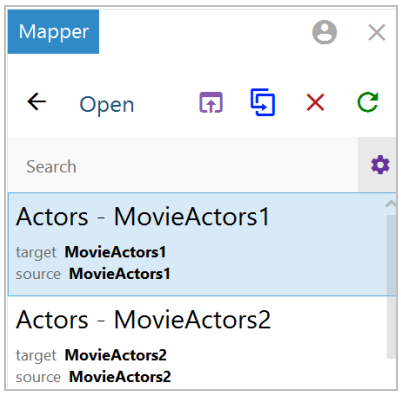
How do I duplicate a mapping?

Using the Anzo for Office Excel plugin, users can duplicate mappings to use as a template for a new mapping. Follow the instructions below to duplicate a mapping.

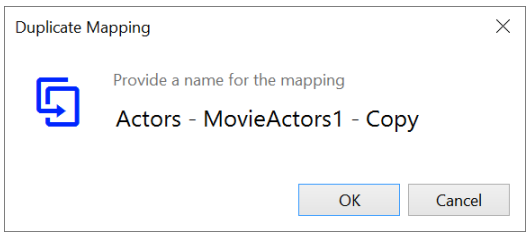
1. In Excel, open the Anzo Mapper tool and connect to the Anzo server.
2. In the Mapper menu, click the folder icon ( ) to list the mappings that are available to open. By default, the mapping tool lists only the mappings that you created. To display additional mappings, such as auto-generated files, type a term in the **Search** field, and then click the cog icon ( ) to display the files. For example:



3. Click the mapping that you want to duplicate. Selecting a mapping activates the buttons at the top of the screen. For example:



4. Click the **Duplicate** icon ( ) to copy the selected mapping. Anzo displays the Duplicate Mapping dialog box. For example:

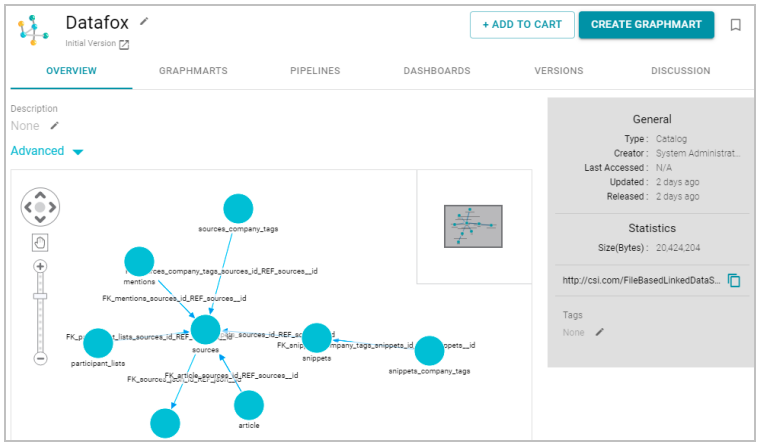


5. Edit the mapping name and then click **OK** to create the duplicate. The new mapping is added to the list of mappings that are available to open.

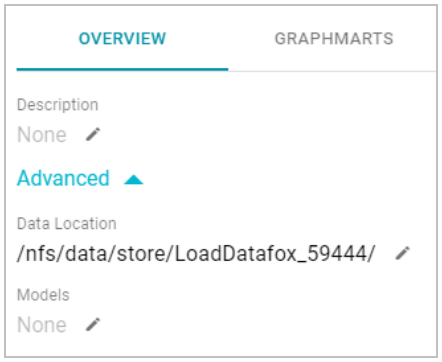
How do I associate a model with an existing data set?

Follow the instructions below to associate a model that is in Anzo with an onboarded data set.

1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Dataset catalog, which lists the existing data sets.
2. Click the data set that you want to add a model to. Anzo displays the Overview screen for the data set. For example:



3. On the Overview screen under the Description field, click **Advanced** to display the advanced options. For example:

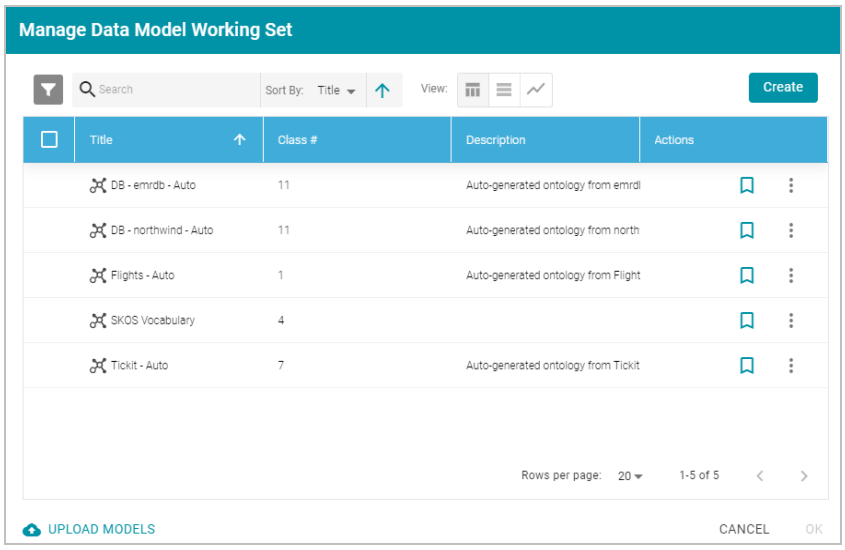


4. Click the Edit icon (✎) next to **Models** to open the Models drop-down list, and then select the model that you want to use for this data set. To include a system model, select the **Include System Data** checkbox. To select multiple models, click the drop-down list again and select another model.
5. When you have finished selecting models, click the checkmark icon (✓) to save the change and associate the model or models with the data set.

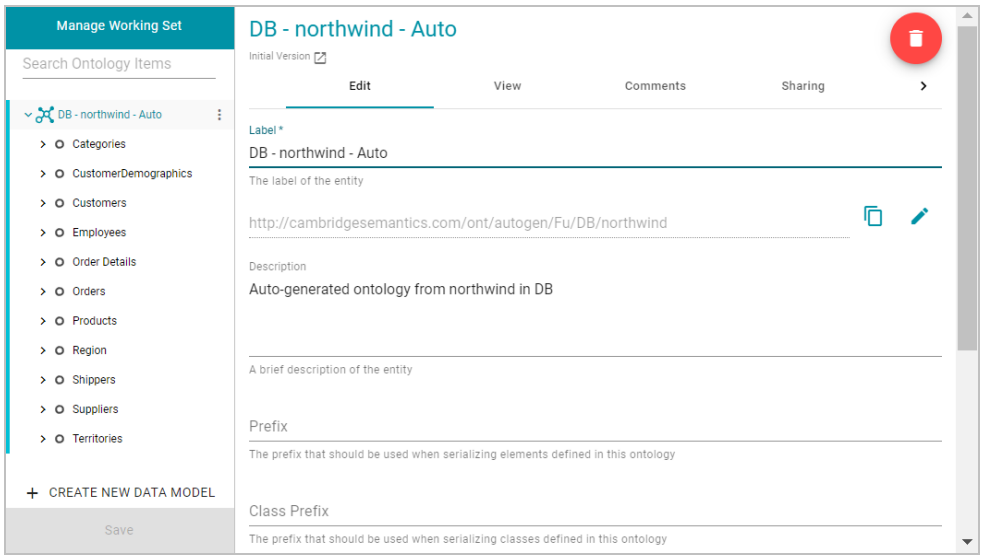
How do I download a model?

Follow the instructions below to download a data model to your computer.

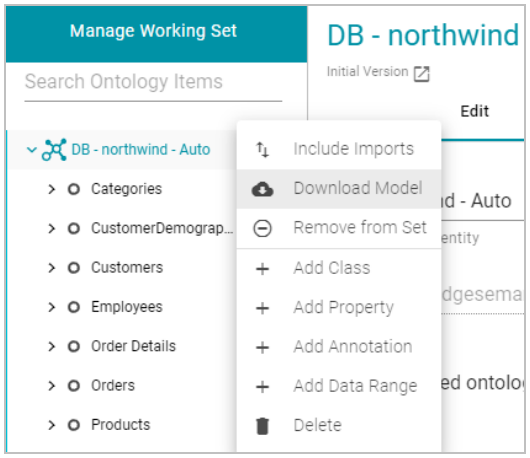
1. In the Anzo console, click **Model**. Anzo displays the Manage Data Model Working Set screen. For example:



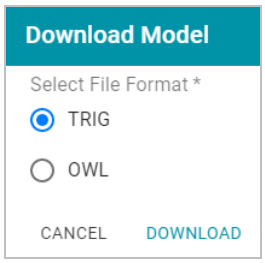
2. On the Manage Working Set screen, select the checkbox next to the model that you want to export, and then click **OK**. Anzo opens the selected model in the editor. For example:



3. Open the model menu by clicking the menu icon (⋮) to the right of the model name. Then select **Download Model**.



Anzo displays the Download Model dialog box:



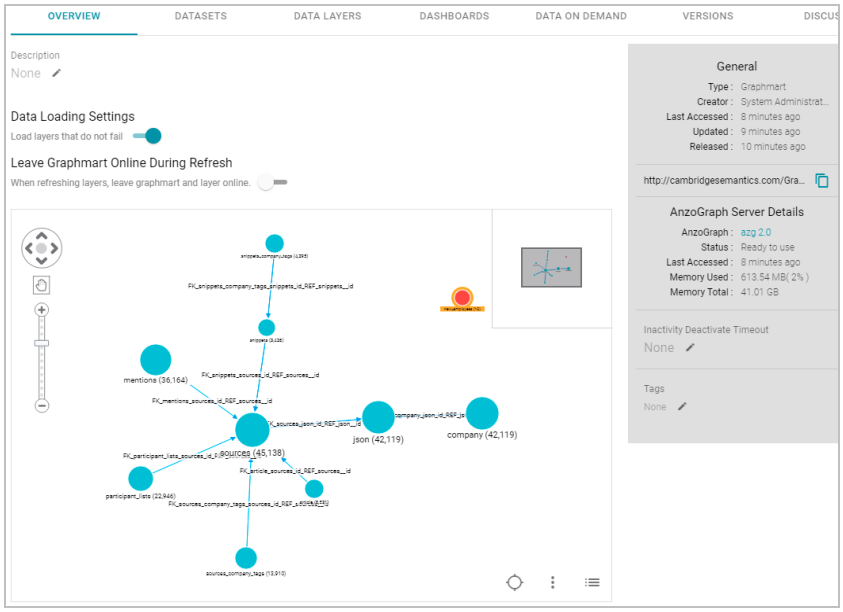
4. In the Download Model dialog box, select the format to save the model in. By default Anzo saves models in **TRIG** format. If you want to save the file in OWL format, select the **OWL** radio button. Then click **Download**.

Anzo downloads the model to your computer in the selected format.

How do I see which models are included in a graphmart?

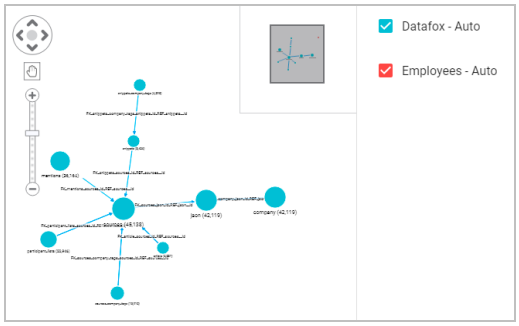
Anzo displays graphmart details, such as a list of the models in the graphmart, on the Overview screen for the graphmart. Follow the steps below to view the list of models in a graphmart.

1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**. Anzo displays the Graphmarts screen, which lists the existing graphmarts.
2. In the list of graphmarts, click the name of the graphmart for which you want to view the included models. Anzo displays the graphmart overview. For example:



In the bottom right corner of the graph view in the center of the screen, there are three icons:

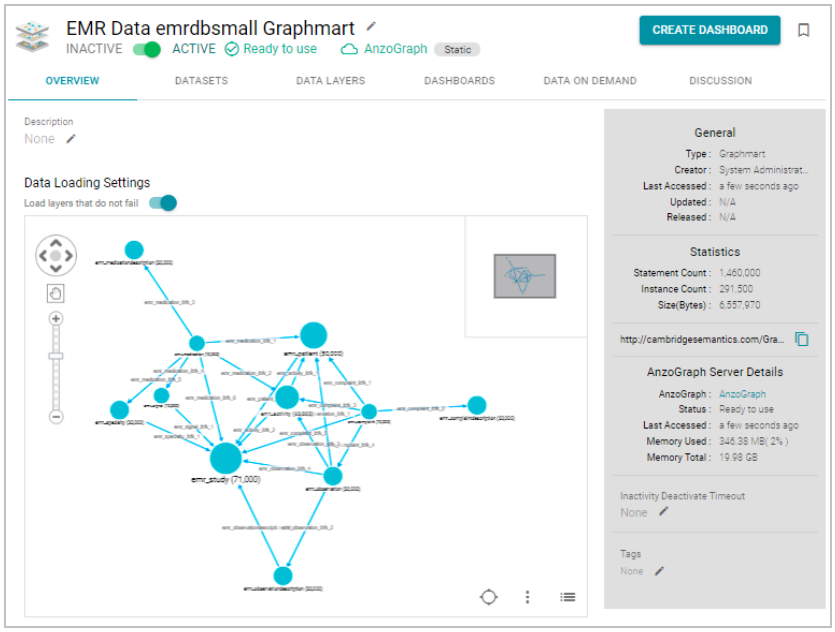
3. To view the associated models, click the contents icon ( ) on the right. Anzo displays the list of models. For example, the image below shows a graphmart with two models:



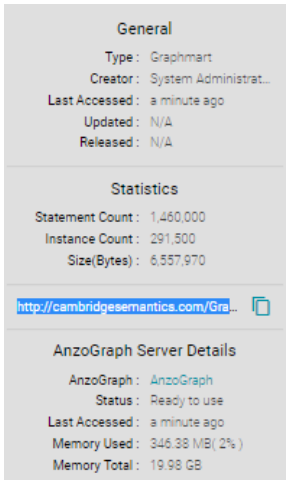
How do I find the URI for a graphmart?

Anzo displays graphmart details, such as a graphmart's URI, on the Overview screen for the graphmart. Follow the steps below to view and copy a graphmart URI.

1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**. Anzo displays the Graphmarts screen, which lists the existing graphmarts.
2. In the list of graphmarts, click the name of the graphmart for which you want to view or copy the URI. Anzo displays the graphmart overview. For example:

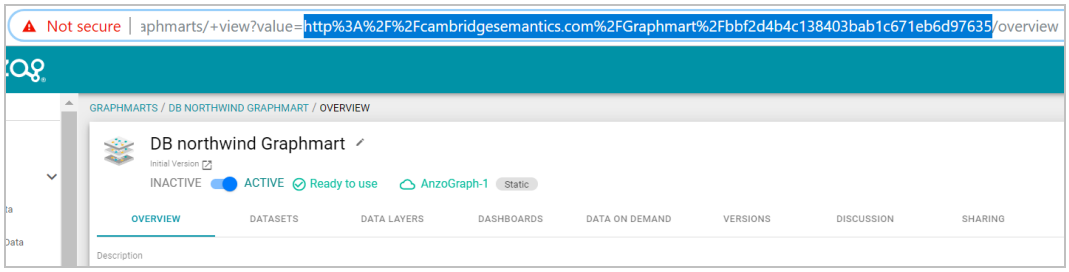


3. View the graphmart URI in the statistics section on the right side of the screen. For example:



   You can click the clipboard icon ( ) to copy the graphmart URI to your clipboard.

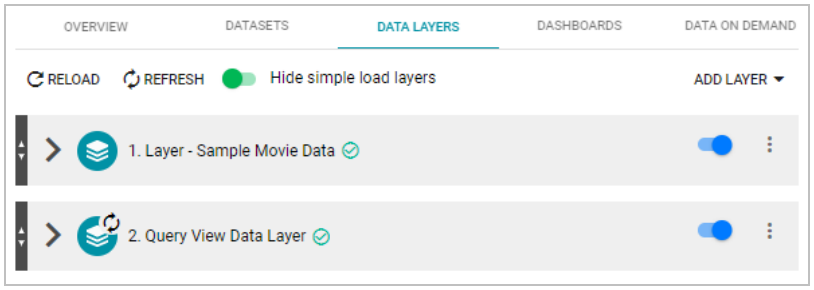
You can also copy a URL-encoded version of the graphmart URI from the address bar in the browser when viewing the graphmart overview screen. For example:



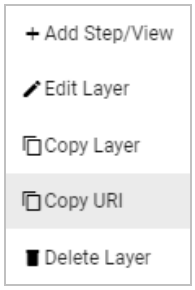
How do I find the graph URI for a layer in a graphmart?

You can retrieve a graph or data layer URI on the Data Layers screen for a graphmart. Follow the steps below to copy a graph URI.

1. In the Anzo console, expand the **Blend** menu and click **Graphmarts**. Anzo displays the Graphmarts screen, which lists the existing graphmarts.
2. In the list of graphmarts, click the name of the graphmart that contains the graph or data layer for which you want to copy the URI. Anzo displays the graphmart overview.
3. Click the **Data Layers** tab. Anzo displays the data layers in the graphmart. Each data layer is a graph. For example:



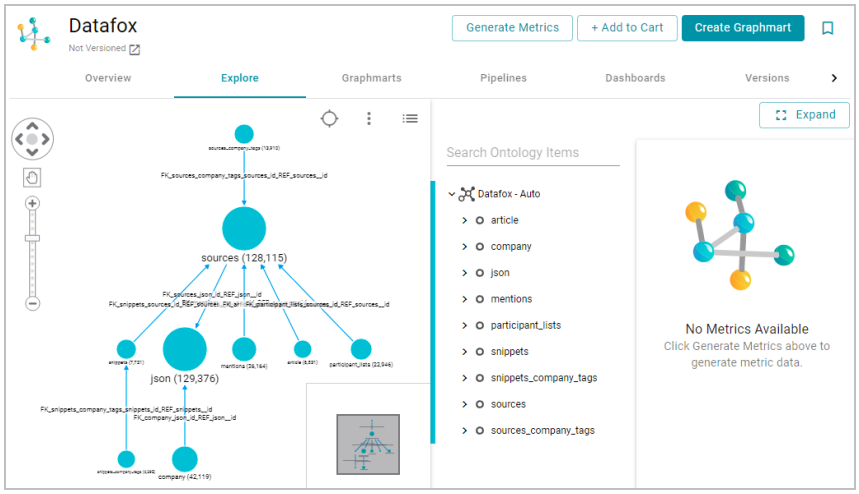
4. To copy the URI for a layer to your clipboard, click the menu icon (⋮) on the right side of the layer and click **Copy URI**.



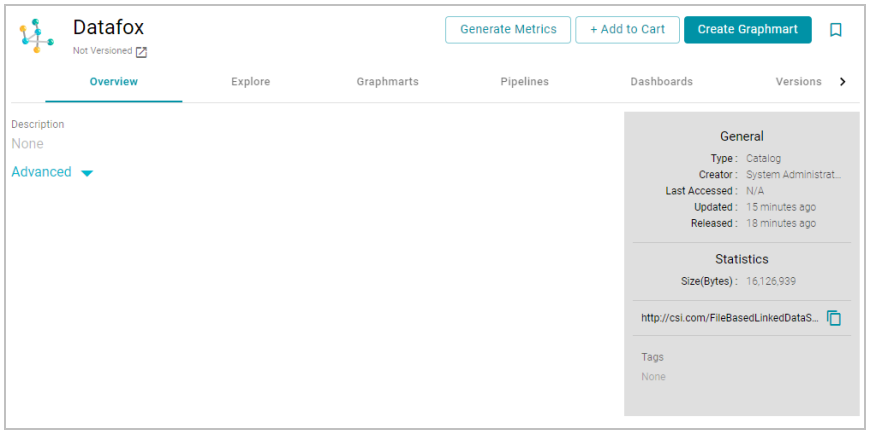
How do I find the URI for an LDS in the Dataset Catalog?

Follow the steps below to view and copy the URI for a linked data set (LDS) in the catalog.

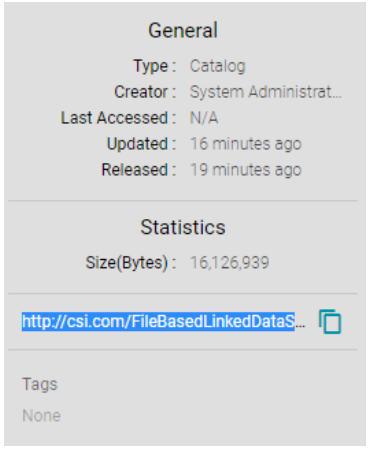
1. In the Anzo console, expand the **Blend** menu and click **Datasets**. Anzo displays the Dataset catalog, which lists the data sets that have been onboarded to Anzo.
2. In the list of data sets, click the name of the data set for which you want to copy the URI. Anzo displays the Explore tab. For example:



3. Click the **Overview** tab to view the general information for the data set. For example:



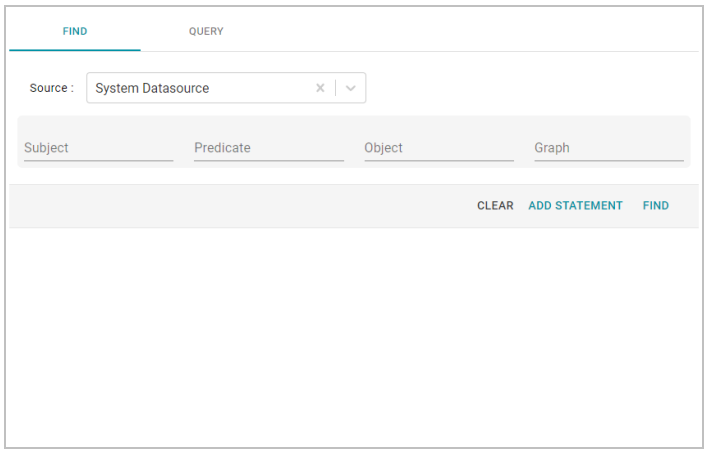
4. In the statistics section on the right side of the screen, click the clipboard icon ( ) to copy the LDS URI to your clipboard. For example, the image below shows the URI highlighted:



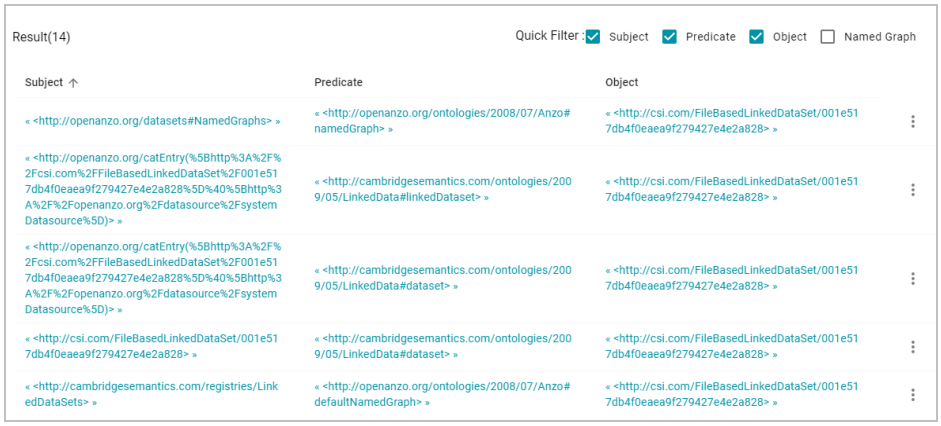
How do I find the catalog entry URI for an LDS?

To query from a remote client (such as over the SPARQL endpoint) a linked data set (LDS) that is stored in a local volume, you need to specify the catalog entry URI for that LDS as the target data set. The catalog entry URI uniquely identifies an LDS because it encodes both the LDS and its data source (local volume) in the URI. Follow the steps below to find the catalog entry for an LDS.

1. First, retrieve the URI for the LDS whose catalog entry URI you want to find. For instructions, see How do I find the URI for an LDS in the Dataset Catalog? above.
2. Next, open the Find tab in the Query Builder. In the Anzo application, expand the **Access** menu and click **Query Builder**. Then click the **Find** tab. The Find screen opens and the **System Datasource** is selected as the target data source.



3. If the LDS is in a different volume, click the **Source** drop-down list and select the appropriate volume. Typically, linked data sets are stored in the system volume.
4. Paste in the **Object** field the LDS URI that you copied in the first step. Then click **Find**. Anzo returns the set of quads for which the LDS URI is the object. For example:



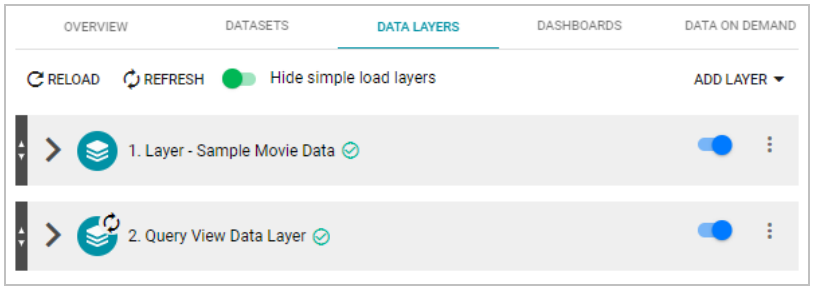
5. In the **Subject** field in the results, look for a URI that begins with **http://openanzo.org/catEntry**. The entire value is the catalog entry URI for the LDS. For example:

```
« <http://openanzo.org/catEntry(%5Bhttp%3A%2F%2Fcsi.com%2FFile
BasedLinkedDataSet%2F001e517db4f0eaea9f279427e4e2a828%5D%
40%5Bhttp%3A%2F%2Fopenanzo.org%2Fdatasource%2FsystemDatas
ource%5D)> »
```

6. Copy the entire URI. This is the URI to use as the target data source for SPARQL endpoint queries against the LDS. For more information about the SPARQL endpoint, see Accessing Data via the SPARQL Endpoint.

What is the difference between the graphmart Reload and Refresh options?

When you make modifications to data layers in a graphmart, Anzo displays **Reload** and **Refresh** buttons on the top of the Data Layers screen. For example:



The Refresh option becomes available when changes have been made to one or more data layers. Clicking **Refresh** resets (deletes from AnzoGraph) and reloads only the data layers that have changed. Clicking **Reload** resets and reloads the entire graphmart to AnzoGraph, including the data layers that have not changed.

What happens to the existing data in an FLDS when I run an incremental ETL job?

When you publish a pipeline that includes a job that onboards data incrementally, Anzo preserves the previously onboarded data by moving the existing RDF files to a hidden directory within the FLDS. The new RDF files for the current pipeline are then written to the FLDS. Since hidden directories are not loaded to AnzoGraph, only the most current data is loaded into memory. For information about onboarding data incrementally, see Creating an Incremental Schema.