Oracle® Big Data Discovery

Getting Started Guide

Version 1.4.0 • October 2016

ORACLE

Copyright and disclaimer

Table of Contents

Preface

Oracle Big Data Discovery is a set of end-to-end visual analytic capabilities that leverage the power of Apache Spark to turn raw data into business insight in minutes, without the need to learn specialist big data tools or rely only on highly skilled resources. The visual user interface empowers business analysts to find, explore, transform, blend and analyze big data, and then easily share results.

About this guide

This guide introduces Oracle Big Data Discovery, and orients you how to use it for your needs, from loading data to exploring, transforming, and updating it.

Along the way, the guide introduces key terms, components and user interfaces in Big Data Discovery. This guide is complementary to the Getting Started video series.

Audience

This guide is for business analysts, data scientists, and data engineers who work with big data.

Conventions

The following conventions are used in this document.

Typographic conventions

The following table describes the typographic conventions used in this document.

Typeface	Meaning
User Interface Elements	This formatting is used for graphical user interface elements such as pages, dialog boxes, buttons, and fields.
`Code Sample`	This formatting is used for sample code segments within a paragraph.
Variable	This formatting is used for variable values. For variables within a code sample, the formatting is *`Variable`*.
`File Path`	This formatting is used for file names and paths.

Path variable conventions

This table describes the path variable conventions used in this document.

Path variable	Meaning
`$ORACLE_HOME`	Indicates the absolute path to your Oracle Middleware home directory, where BDD and WebLogic Server are installed.
`$BDD_HOME`	Indicates the absolute path to your Oracle Big Data Discovery home directory, `$ORACLE_HOME/BDD-<version>`.
`$DOMAIN_HOME`	Indicates the absolute path to your WebLogic domain home directory. For example, if your domain is named `bdd-<version>_domain`, then `$DOMAIN_HOME` is `$ORACLE_HOME/user_projects/domains/bdd-<version>_domain`.
`$DGRAPH_HOME`	Indicates the absolute path to your Dgraph home directory, `$BDD_HOME/dgraph`.

Contacting Oracle Customer Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. This includes important information regarding Oracle software, implementation questions, product and solution help, as well as overall news and updates from Oracle.

You can contact Oracle Customer Support through Oracle's Support portal, My Oracle Support at *https://support.oracle.com*.

Chapter 1

Welcome to Big Data Discovery

This section introduces Big Data Discovery (BDD), and includes overviews of its components.

Ways of using Big Data Discovery

In your organization, use BDD as the center of your data lab, as a unified environment for navigating and exploring all of your data sources in Hadoop, and to create projects and BDD applications.

Use BDD in the data lab

The data lab is a complete set of analytics solutions. It lets analysts collaborate and have access to many data sets. It serves as a sandbox for data experiments and supports many data projects. It is often integrated with the production environment, so that it can integrate feedback, reiterate and produce new solutions that can be taken into production.

When used in the data lab, Big Data Discovery lets you:

- Define projects in BDD and share them with others in your research data lab.
- Use BDD to develop ideas, build prototypes and models, and invent ways of deriving value from data. For example, you can try out new approaches, quickly discard the ones that are not working, and move on to try new ways of working with your data.
- Shape data in your projects in multiple ways, from making single-row changes, such as trimming, editing, splitting, or null-filling, to advanced data shaping techniques, such as aggregations, joins, and custom transformations.
- Let others in your organization consume models created in the data lab, and publish insights to decision-making groups inside your organization.

Navigate and explore data sets in Hadoop

When used as the navigator on top of your data sources in Hadoop, BDD visually represents all data available to you in your environment. You see all the data in Studio's Catalog and can find interesting data sets quickly. You can filter, edit metadata on data sets, and create new data sets for others in your team.

Create BDD applications

BDD lets you create ***BDD applications*** for well-established and known problems, to suit your business needs. For example, you can:

- Create BDD projects from scratch, improve them, and share them with wider groups of users in the organization.
- Serve as the author of all discovery solution elements: data sets, information models, discovery applications, and transformation scripts.
- Run discovery repeatedly and transparently to others in your group, and on different large-scale data sets arriving periodically from various sources.

Value of using Big Data Discovery

In BDD, a wider number of people can work with big data, compared with traditional analytics tools. You spend less time on data loading and updates, and can focus on actual data analysis of big data.

In traditional analytics projects you must first predict possible business questions, model the data to match, locate and get the data sources, and manipulate feeds to fit into the model. You also build pipelines for extracting, transforming and loading data (ETL). Only after these tasks you can engage with the data. As a result, only a minimal effort is actually focused on data analysis.

The complexity of big data compounds the up-front costs of data loading. Big data increases the amount and kinds of data. You need more time to understand and manipulate new source data, especially unstructured sources, before it is ready for analysis.

Big data also changes fast. You must update your existing analytics projects with newer data on a regular basis. Such data loading and update tasks need qualified engineers with expert skills. As a result, you spend more time and money on up-front data loading and data updates.

Big Data Discovery addresses these problems:

- More users can engage with the data and contribute to big data analytics projects.
- Data preparation becomes a small part of your effort. You don't have to prepare data up-front, and can focus your time and effort on analyzing data, and improving your business.
- You can update BDD projects with new data, refresh already loaded data, or add newer data to existing projects.
- You can work with data as the data scientist, and rely on a range of ways for data shaping and visual analysis in BDD.

Addressing your goals and needs

Big Data Discovery makes your job faster and easier than traditional analytics tools. This topic discusses your goals and needs in data analysis and shows how Big Data Discovery can address them.

As a data scientist or analyst, you:

- **Solve complex questions using a set of disjointed tools**. You start with many dynamic imprecise questions. You often have unpredictable needs for data, visualization, and discovery capabilities. To address them, you rely on open source and custom discovery tools. You use tools in combination with other tools, and often need to reopen the same tools several times.

  In Big Data Discovery, this fragmented workflow between tools is replaced with a single workflow that is natively part of your ecosystem.

- **Need to collaborate**. Together with your team, you work with big data from many external and internal sources. Other team members may consume the results of your findings. You also improve and publish insights and prototypes of new offerings.

  In Big Data Discovery, you can create personal projects, or create and share a project with your team.

- **Want to make sense of data**. You often need to create and use insights. You do this by collecting, cleaning and analyzing your data.

  Big Data Discovery lets you make sense of data. You use it to collect, join, shape, and analyze your data.

- **Generate ideas and insights**. You want to create insights that lead to changes in business, or you want to enhance existing products, services, and operations. You also need to define and create prototypes (or models) for new data-driven products and services.

  In Big Data Discovery, you arrive at insights by using many data visualization techniques. They include charts, statistical plots, maps, pivot tables, summarization bars, tag clouds, timelines, and others. You can save and share results of your discovery using snapshots and bookmarks.

- **Validate, trace back, tweak, and share your hypotheses**. You often need to provide new perspectives to address old problems. The path to the solution is usually exploratory and involves:

  - Hypotheses revision. You must explore several hypotheses in parallel. They are often based on many data sets and interim data products.
  - Hypotheses validation. You need to frame and test hypotheses. This requires validation and learning of experimental methods and results done by others.
  - Data recollection and transparency. You would like all stages of the analytical effort to be transparent so that your team can repeat them. You want to be able to recreate analytical workflows created before. You also would like to share your work. This requires linear history of all steps and activities.

  Big Data Discovery helps you by saving your BDD projects, data sets, and transformation scripts. This lets you improve your projects, share them, and apply saved transformation scripts to other data sets.

Results of working with Big Data Discovery

In BDD you can create an insight, an exploratory project, or a tool for long-term analysis of your data.

For example, you can create:

- Another data set that illustrates the business hypothesis, or answers a specific business question.

- A product, such as a set of transformations useful for similar data sets.
- One or more ad-hoc exploratory projects.
- An insight, such as a conclusion that you illustrate with several saved visualizations and share with your team.
- A long-term BDD application for further analysis of future data sets.
- An infrastructure (a set of tools that includes Big Data Discovery as a component).

As an outcome, you:

- reach insights for action
- gain understanding for building more sophisticated mathematical models
- create data products, such as new features and services for your organization.

About Studio

Studio is a component in Big Data Discovery. Studio provides a business-user friendly user interface for a range of operations on data.

It combines search, guided navigation, and many data visualizations in a single environment.

Some aspects of what you see in Studio always appear. For example, Studio always includes search, **Explore**, **Transform**, and **Discover** areas. You can add other parts of your interface as needed. These include many types of data visualization components. For example, you can add charts, maps, pivot tables, summarization bars, timelines, and other components. You can also create custom visualization components.

Studio provides tools for loading, exploration, updating, and transforming data sets. It lets you create projects with one or more data sets that are linked. You can load more data into existing projects. This increases the corpus of analyzed data from a sample to a fully loaded set. You can also update data sets. You can transform data with simple and more complex transforms, and write transformation scripts.

Studio launches data set workflows that apply transformations to the data, and also lets you preview them before saving. Studio relies on a sub-component, called the Transform Service, for these actions.

The ***Transform Service*** is a sub-component in BDD serving Studio. It processes transformations for Studio, and lets you preview the effects of transformations before they are applied. It runs in a Jetty container, and is often co-located with Studio on the same node, during installation.

Studio's administrators can control data set access and access to projects. They can set up user roles, and configure other Studio settings.

For more information on Studio, see *Taking a Tour of Studio on page 12*.

About Data Processing

The ***Data Processing*** component of BDD runs a set of processes and jobs. This set is called ***data processing workflows***. Many of these processes run natively in Hadoop.

The Data Processing component controls several workflows in BDD. For example, you can create workflows for data loading, data updates, and others. The Data Processing component discovers new Hive tables and loads data into BDD. It runs data refresh operations and incremental updates. It also keeps BDD data sets in synch with Hive tables that BDD creates.

For example, during a data loading workflow, the Data Processing component performs these tasks:

- Discovery of data in Hive tables
- Creation of data sets in BDD
- Running a select set of enrichments on discovered data sets
- Profiling of the data sets
- Indexing of the data sets, by streaming data to the Dgraph.

The Workflow Manager Service

The ***Workflow Manager*** is a service in BDD that acts as an intermediary between Spark and the BDD clients: Studio and Data Processing CLI. The service receives data set workflow requests from Studio or Data Processing CLI, and delegates the sequence of Spark jobs needed for each workflow, such as sampling, discovery or transformations, to run in YARN. The Spark jobs run asynchronously of each other and the service notifies Studio of the job status. The Workflow Manager also delegates jobs to other components in BDD, such as the Dgraph and the Dgraph HDFS Agent. Here are two examples of these interactions:

- Studio submits its workflow request to the Workflow Manager Service. The service submits sampling or transformation Spark jobs to YARN. Depending on the workflow, it also indicates to the Dgraph HDFS Agent and the Dgraph to start data loading and indexing. Finally, the Workflow Manager notifies Studio when the workflow finishes.
- The Data Processing CLI submits its workflow request to the Workflow Manager Service. This could be a request to load data from Hive, to refresh data or to run incremental data updates (add more data to existing data). The Workflow Manager delegates a sequence of Spark jobs to YARN, and also notifies Studio of the status when the workflow finishes.

Data Processing CLI

The ***DP CLI*** is a shell Linux utility that submits data processing workflows to the Workflow Manager, which starts them in YARN. You can run DP CLI manually or from a `cron` job. The DP CLI submits the workflow requests and uses the Workflow Manager for the delegation of workflows to YARN. The workflows can run on an individual Hive table, all tables within a Hive database, or all tables within Hive. This depends on DP CLI settings, such as a blacklist and a whitelist.

Here are some of the workflow requests you can submit with DP CLI:

- Load data from Hive tables after installing Big Data Discovery (BDD). When you first install BDD, your existing Hive tables are not processed. You must use the DP CLI to launch a data processing operation on your tables.
- Run data updates. They include:
  - An operation to refresh data. It reloads an existing data set in a Studio project, replacing the contents of the data set with the latest data from Hive in its entirety.
  - An incremental update. It adds newer data to existing data sets in a Studio's project.
- Launch the BDD ***Hive Table Detector***, which is another sub-component related to data processing in BDD. The Hive Table Detector (HDT) detects if a new table is added to Hive. It then checks the whitelist and blacklist. If the table passes them, it creates a data set in BDD. It also deletes any BDD data set that does not have a corresponding source Hive table. This keeps BDD data sets in synch with data sets in

Hive. For detailed information on how data sets are managed in BDD, see *Data set lifecycle in Studio on page 25*.

For information on Data Processing workflows, the Workflow Manager, the DP CLI, and the Hive Table Detector utility, see the *Data Processing Guide*.

About the Dgraph

The Dgraph uses data structures and algorithms to issue real-time responses to queries.

The Dgraph stores the ingested data in the Dgraph databases it creates for the indexed data sets. The Dgraph receives client requests from Studio, queries its databases, and returns the results. The Dgraph is stateless. This design requires that a complete query is sent to it for each request. The stateless design facilitates the addition of Dgraph instances for load balancing and redundancy. Any instance of a Dgraph can reply to queries independently of other instances.

A Big Data Discovery cluster can include two or more Dgraph instances. They handle end-user query requests and store Dgraph databases on HDFS/MapR-FS, or a shared NFS. For each data set, the leader Dgraph (once it is automatically appointed in BDD) handles all write operations, such as updates and configuration changes, while other instances serve as read-only followers for that data set.

The Dgraph Gateway performs the routing of requests to the Dgraph nodes. It handles query caching, business logic, and cluster services for the Dgraph nodes.

The Dgraph Tracing Utility is a Dgraph diagnostic program used by Oracle Support. It stores the Dgraph trace data, which are useful in troubleshooting the Dgraph. It starts when the Dgraph starts and keeps track of all Dgraph operations. It stops when the Dgraph shuts down. You can save and download trace data to share it with Oracle Support.

The Dgraph HDFS Agent loads data into the Dgraph. It reads Avro files from the data processing workflows and formats them for data loading. It then sends them to the Dgraph.

To learn about the Dgraph and its logs, see the *Administrator's Guide*. To learn about the Dgraph HDFS Agent, its logs and behavior, see the *Data Processing Guide*.

Chapter 2

Taking a Tour of Studio

This section walks you through Studio. It describes how you move through areas within Studio when working with big data.

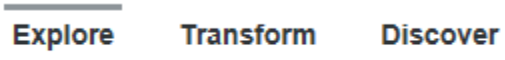
Workflow in Studio

Studio is the visual face of Big Data Discovery. It enables exploratory data analysis and is structured around **Explore**, **Transform** and **Discover** areas.

- After you log in, you see the Catalog. It lets you browse and discover the contents of the data lake. (A ***data lake*** is a storage repository that holds raw data in its native format until it is needed.)

  In Catalog you can find projects you created before, or projects shared by other users. You can also browse data sets discovered in a Hive data base, or data sets that other users uploaded from a file or imported from an external JDBC data source:

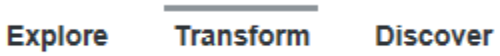
- Next, if there are no **Discover** pages created yet by other users, the **Explore** page for the first data set opens. If **Discover** pages already have been created by other users, the first **Discover** page opens.
- If you move to **Explore**, you can pick data sets that are of interest to you for further exploration, and use the results of data profiling and enrichments to grasp basic data characteristics. For example, you can look for outliers. Outliers are values that are distant from other values for the attribute. You can also explore scatter plots to see correlations between values for two attributes, or link data sets.



- If some pages by other users have already been created and you have access to them, then instead of **Explore**, you move to **Discover** for the project that is shared with you.
- If you started with **Explore**, then to move to **Transform**, or **Discover**, you must first add the data set to a project. Alternatively, you can find and select an existing project:

  Note that once your data set is in a project, you can continue using **Explore**.
- You can then move to **Transform**, where you transform the data and remove inconsistencies. For example, you can change data types, or create custom transformation scripts.

- In **Discover** you arrive at insights about the data and save your projects for sharing with others.

Finding data sets

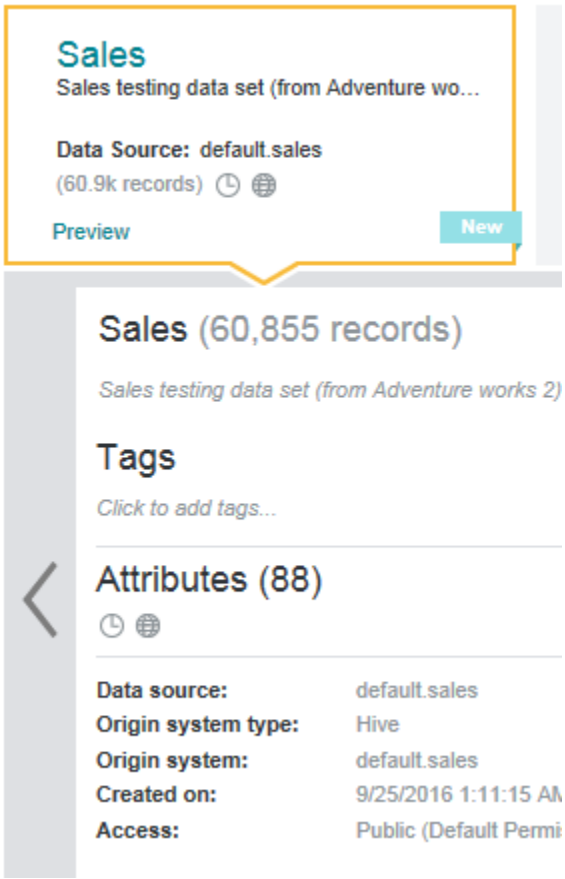
The number of data sets in any data repository can be huge. In Studio's Catalog, you can use search and Guided Navigation to find data sets and existing projects relevant to your analytical purpose.

After you log in, you see the **Catalog**:

Here, you can:

- Find data sets
- Find existing projects
- Add a data set to BDD

You can quickly look at a data set by clicking the tile to expand the Data Set Details panel. This displays the data set's enriched metadata to provide a summary to help you decide if you'd like to explore this data set further:



Through the Catalog, you can:

- Navigate data sets using a familiar keyword search and Guided Navigation experience similar to that found on major commerce sites.
- Browse data sets based on name, metadata, attribute names, topics, dates, locations, or key terms.
- Find related data sets that are potentially relevant to your interests based on overlapping data or frequent use by other users.
- Filter displayed data sets based on data set metadata (such as the creation date, tags, notes, number of records, or contents of the data) and also by attribute metadata (such as semantic types).

Adding tags and notes to data sets

When you expand a data set to view the data set details, you can optionally add tags or notes that describe the data. All users can then filter data sets based on those tags, or run keyword searches that include text from the data set notes.

To add notes, click the description field below the data set name.

To add searchable tags, click the text field below **Tags**. To help you choose consistent tag names, autocomplete results may display as you type.

Profiling and Enriching Data

This topic summarizes how Big Data Discovery helps to profile and enrich new data.

You can add new data as files in HDFS using data integration technologies. You can also use Studio to upload files, such as Excel and CSV, or pull data from a database using your credentials, and import it into a personal sandbox in HDFS. In either case, when it loads data, Big Data Discovery performs these activities for you:

- Profiles the data, inferring data types, classifying content, and understanding value distributions.
- Lists most interesting data sets first and indicates what they contain.
- Decorates the data with metadata, by adding profile information.
- Enriches the data, by extracting terms, locations, sentiment, and topics, and storing them as new data in HDFS.
- Takes a random sample of the data of the specified size. (You can increase the sample size, or load full data.)
- Indexes the data and prepares it for fast search and analysis in BDD.

As an outcome, the data you need resides in HDFS, is indexed by the Dgraph and enriched by BDD, and is ready for your inspection and analysis.

Using Explore

Explore provides you with an out-of-the-box guided analytics experience. It configures the canvas with the right visualizations to explain the data based on your goals. This allows you to get a summary of the data and explore data quality.

Explore provides an attribute-focused visual summary of the data, summarizing value distributions, data quality gaps, and relationships.

Explore presents visualizations that give you the most insight into the data set, selecting the types most suitable to each attribute's data type and value distribution.

Visualizations are automatically composed, to save you time and effort at this early stage in the process. When you have a better understanding of the data set, you can compose your own visualizations on data that you have identified as worthy of further analysis.

You can think of the **Explore** area as a guided tour of new data sets, freeing you from the need to manually query the data or configure your own visualizations. It helps you immediately extract meaning from the results to better understand what's inside the data set.

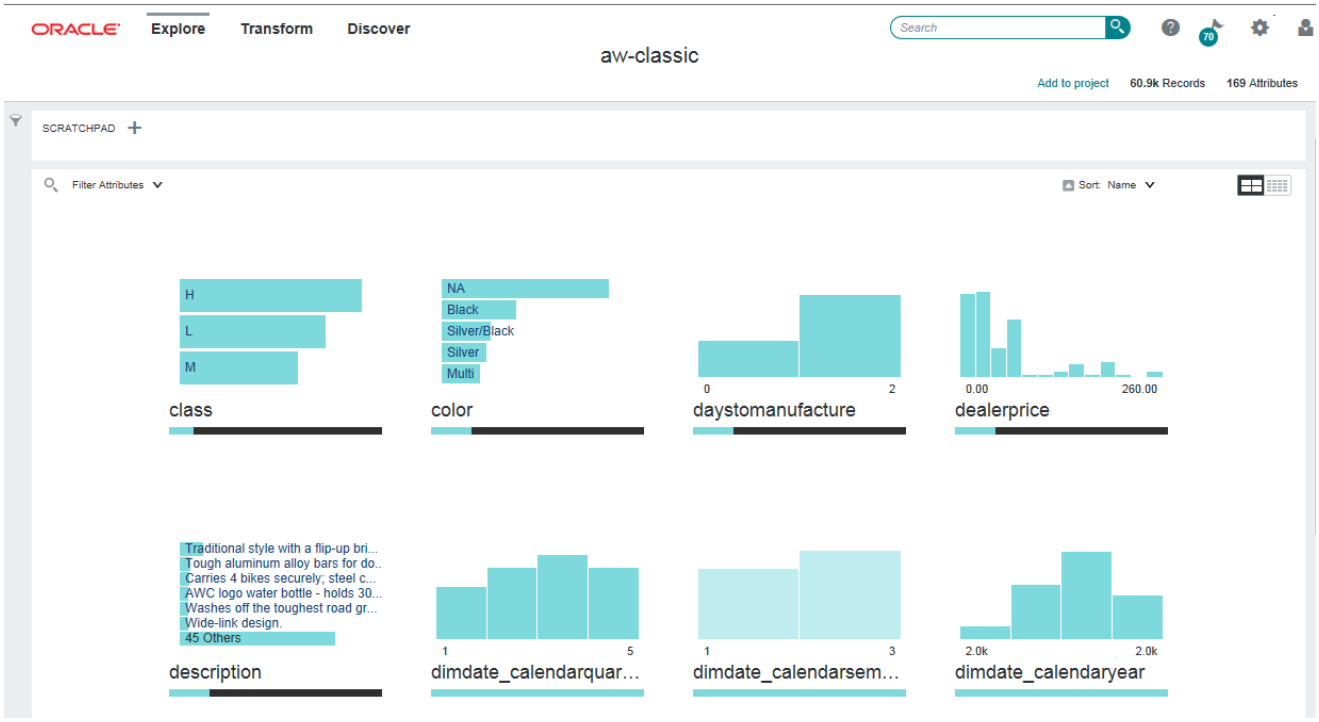
Here are some of the questions that Big Data Discovery provides answers to, when you use **Explore**:

- What fields are in my data? Do I understand what they are? **Explore** shows these fields as a series of visualizations.
- How are values distributed? **Explore** uses descriptive statistics (mean, median, mode, quintiles) and visuals (histograms, box plots).
- How dirty/messy is the data? Are there missing values or outliers? **Explore** uses the bar underneath each attribute, showing you at a glance whether any inconsistencies exist.
- What relationships exist between fields? **Explore** uses numeric correlations and visuals.

In addition, **Explore** lets you add your own notes and tags to data sets. You can later sort or search on those tags, or review your notes for a quick summary without having to walk through the data a second time.

As an outcome, you understand the data sets you've been exploring.

Here are some examples of **Explore** visualizations:



This image shows **Explore** for a data set with 60.9K records, with attributes sorted by name.

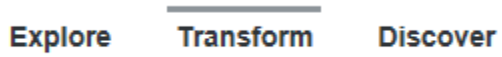
Note: Even if you could explore a large data set at full scale, you always want to start by exploring a representative sample, and later confirm your hypotheses or expand your analysis at full data scale.

Using Transform

Through exploration, data quality issues often come to the surface. You must triage them and, in many cases, correct, clean, or change data in some way. The interactive **Transform** area in Studio helps you turn your data into data that is fit for advanced analysis.

Transform helps you isolate data quality problems and lets you quickly apply a number of data transformations to clean or restructure the data.

You can find **Transform** in Studio here:

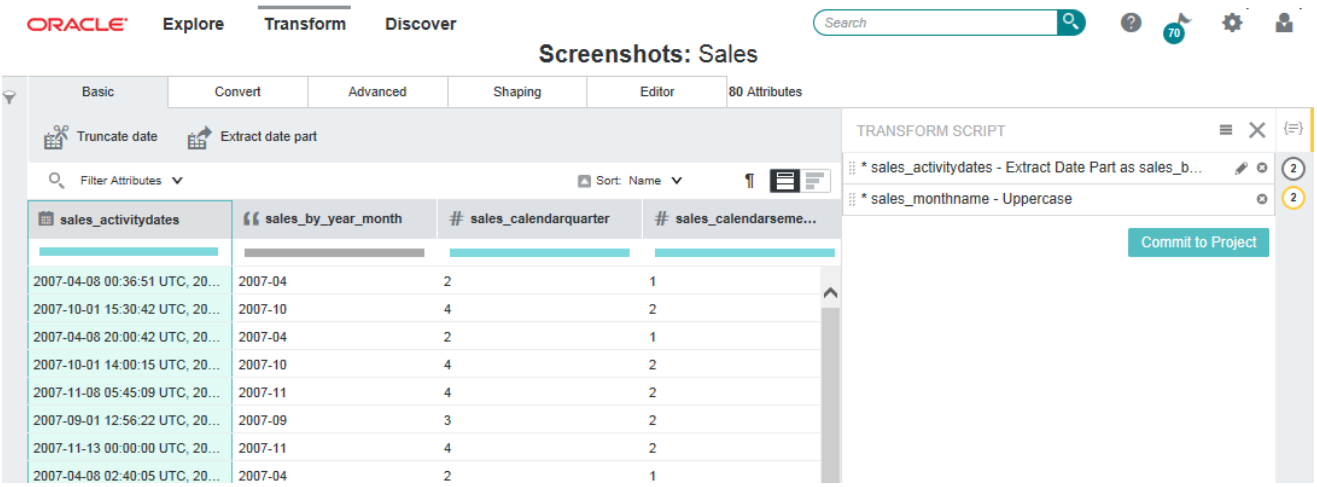


Transform presents a spreadsheet-style view of data, organized either by attribute's value list view, or row view. Here are examples of how you can transform data:

- You can focus on just a few basic transformations needed to support your visualization or analysis. You do this by running common data transformations, such as filling missing values, converting data types, collapsing values, reshaping, pivoting, grouping, merging, and extracting.
- You can join one or more data sets together within a project and also aggregate attribute values to create new derived attributes that sum, average, find minimum/maximum values, and run other operations.
- You can use live **Preview** to see the set of transformations in action before applying them to the entire data set in Studio.

  Because data transformation and data exploration are two integral aspects of any discovery project, you can seamlessly transition between **Transform** and **Explore** within a unified interface, which promotes a smooth workflow.
- You can also share your transformation scripts with other BDD users in your team.

Here is an example of what **Transform** looks like in a project:



In this image, you can see the **Transform** area of Studio. Notice the drop-down attribute menu in the second column. It includes options to hide an attribute, mark it as favorite, edit it, and sort it. To the right, you can also see the **Transform Script** panel, with three transforms created by the project's user.

Also notice the transform menu just under the header. The Transform menu is made up of the tabs: **Basic**, **Convert**, **Advanced**, **Shaping**, and **Editor**.

The transforms that display under each tab vary depending on the data type of the attribute you select. For example:

- If you select a date time attribute, the **Basic** tab displays the **Truncate date** and **Extract date part** transforms. These are both transforms that are contextually appropriate for date time attributes.
- If you select a numeric attribute, the **Basic** tab displays the **Absolute value** transform. The **Absolute value** transform is contextually appropriate for numeric attributes.

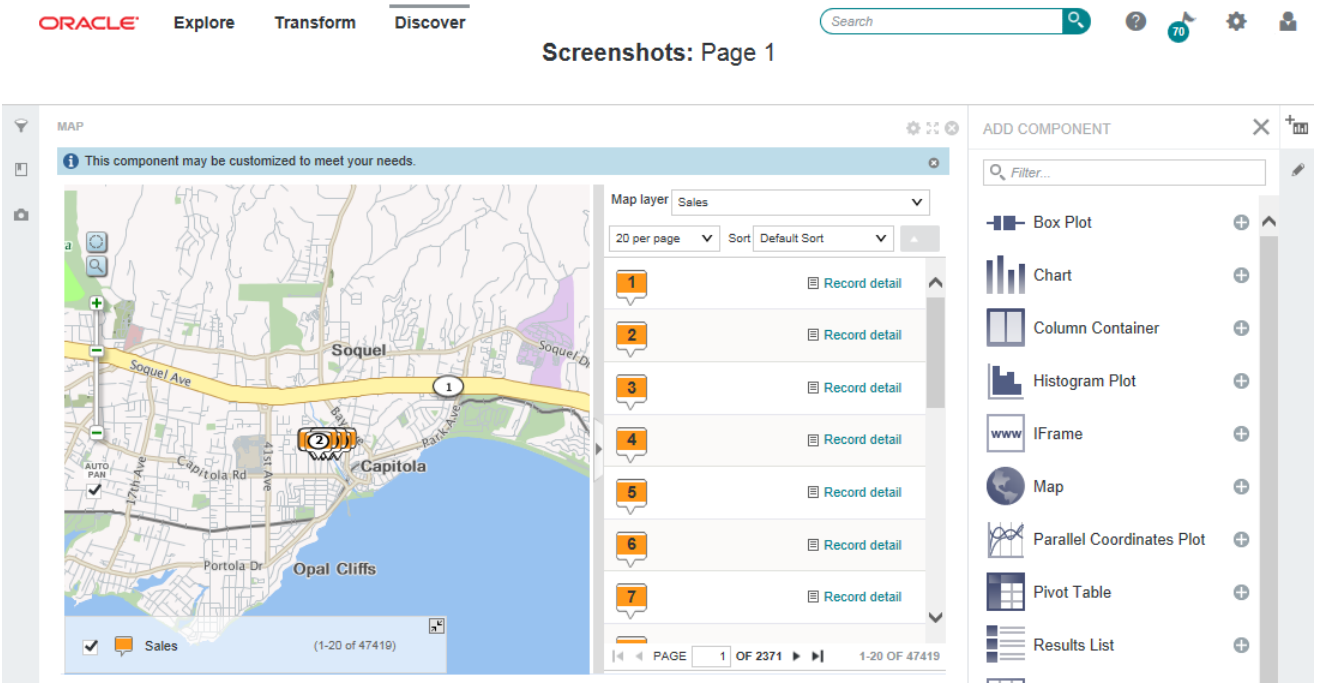
After you run the transform script by clicking **Commit to Project**, the data is transformed and ready for further analysis.

Using Discover

Using **Discover** in Studio, you analyze the data and discover insights hidden in data. **Discover** lets you identify and pick meaningful visualizations and capture the results of the data exploration phase.

You can find the **Discover** area of Studio here:

Here is an example of how **Discover** might look for your project:



In this diagram, you can see the **Discover** area, with the Map visualization added. To the right, notice some of the other visualization components you can add to your project.

Some of the examples of the insights you can create with **Discover** are:

- Use search and Guided Navigation to identify subsets of data for deeper analysis.
- Use interactive visualizations to let the data speak for itself, in raw or aggregated forms, in text or in numbers. Statistical techniques (regression, correlation) and machine learning tools (classification, clustering) are used in Big Data Discovery to drive these visualizations.
- Join (or combine) multiple data sets for new insights.
- Create projects that serve as models for different outcomes. These models automatically understand the weight of different factors in contributing to given outcomes without requiring experts to hand-code rules or heuristics.
- Add new data to existing projects or run your projects on existing data that has been altered to examine new scenarios. (This is known as "what if" analysis.)
- Export your data sets from Big Data Discovery back into Hadoop, so that you can run complex models built in other tools (such as R, or Python) on your custom-created data sub-sets, such as on different customer segments.

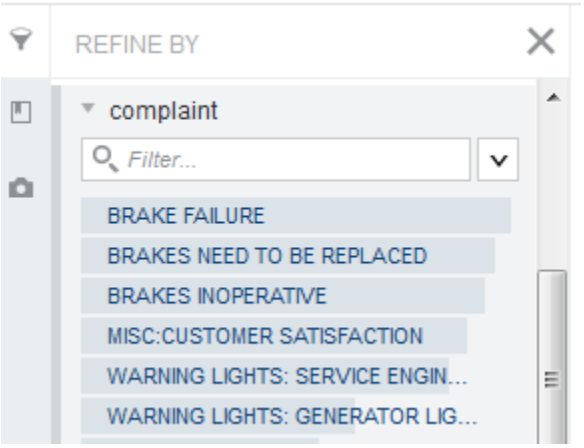
Using filtering, Guided Navigation, and Search

The following examples explain how search filtering and Guided Navigation work in Big Data Discovery.

The Catalog, **Explore**, **Transform**, and **Discover** areas in Studio make use of powerful search and data-driven Guided Navigation capabilities of Big Data Discovery.

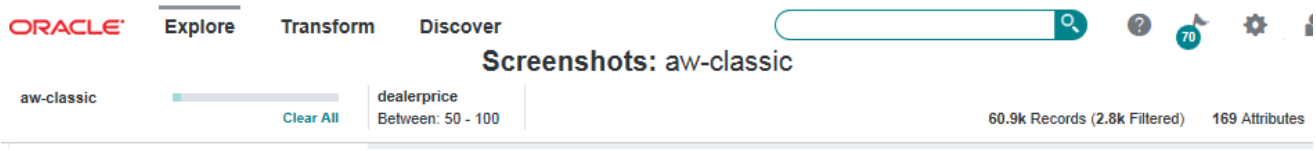
In the Catalog, refinements are always available on the left sidebar.

For data sets in projects, you can expand or collapse the sidebar using the funnel icon. If you expand it, it looks like this:



Using refinements, you can filter on attributes or on available metadata such as creation date. In the Catalog, this also includes any tags that users add to projects or data sets.

Selected refinements represent applied filters. They show up as breadcrumbs at the top of your screen. Here is an example of a selected refinement from the attribute `dealerprice` where the values are between 50 - 100. Notice that applying the refinement reduces the number of matching records to only 2.8k of the 60.9k total records in the data set.



Attribute filters apply across all parts of BDD. If you highlight or exclude attributes, you project reflects these selections when you move from the Catalog to **Explore** to **Transform** and back. For example, if you select all numeric attributes in a data set in **Explore**, and then switch to **Transform**, **Transform** is filtered by all numeric attributes.

You can also search data sets at any stage in your workflow using the search box:

In addition to searching across project and data set content, keyword search results from the Catalog also include any notes that users add to describe projects or data sets.

Like refinement filters, keyword search filters apply across both **Explore** and **Transform** if you switch between pages.

Chapter 3

Data Sets in Big Data Discovery

This section describes what is a data set in BDD, discusses sampled and full data sets, and access to data sets. It also discusses a data set's lifecycle in Studio, and BDD projects and applications.

About data sets

Data sets in BDD are called ***BDD data sets***. This helps you to distinguish them from ***source data sets*** in Hive tables.

A BDD data set is the central concept in the product architecture. Data sets in BDD originate from these sources:

- Many data sets are loaded as a result of the data processing workflow for loading data. It runs when you launch DP CLI, after installing Big Data Discovery. This process adds data sets to Studio's Catalog.
- Other data sets appear in Studio because you load them from a personal file or a JDBC data source.
- Also, you can create a new BDD data set by transforming an existing data set, or by exporting a data set to HDFS as a new Hive table. Such data sets are called ***derived data sets***.

When the Data Processing component runs data loading, or when you add a data set by uploading a file or data from a JDBC source, it appears in Studio's Catalog. You can use **Preview** to see the details for all data sets in Catalog.

You can also use Studio's **Data Set Manager** to see information about all data sets in a project. For each data set in a project, you can see its record count, when it was added, whether it is private to you, and other details.

Data sets in Catalog versus data sets in projects

It is important to distinguish between data sets in the Catalog, and data sets you add to a project.

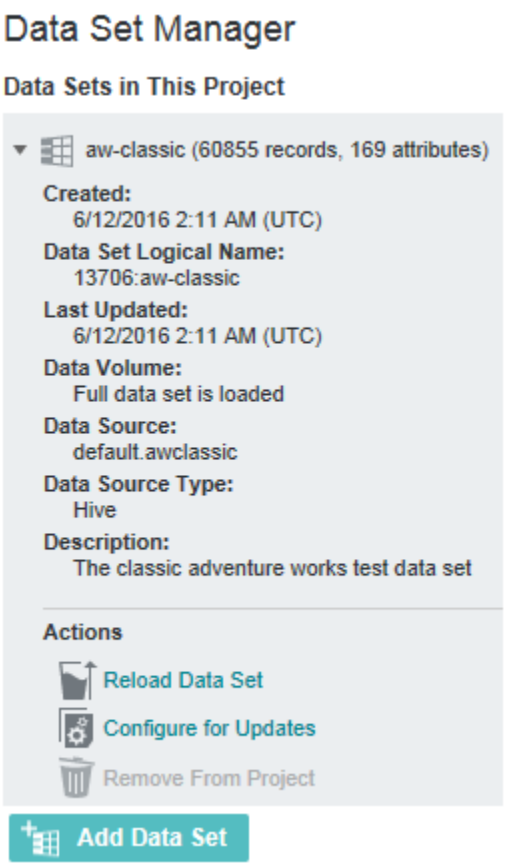
Here is why:

- The Catalog only contains data sets that are discovered in a data lake, or that you add to BDD from a personal file or import from a JDBC source. A data set in a project represents your own modified version of the data set. Once you move a data set to a project, it is similar to creating a personal branch version of the data set.
- You can edit data set metadata and some attribute metadata from the Catalog or the **Explore** view, but data set altering operations, such as transformations, enrichments, and other attribute metadata changes, require editing the data set in a project.
- You can perform some updates of data sets from the Catalog, such as reloading a newer version of a data set. In order to run scheduled updates using the DP CLI, a data set must be part of a project. This is beneficial if you want to run updates periodically, and automate them, by adding them to your scripts that use DP CLI.

Data Set Manager

In Studio, the **Data Set Manager** includes information about all project data sets. You access Data Set Manager once you are in a project, from **Project Settings**.

Here is an example of data set's details in the Data Set Manager:



You can see that the project includes one data set. You can also see the number of records and attributes in it.

The ***Data Source Type*** shows the type of the source file, such as Excel, or CSV, or it shows "Hive", if the data originated from a Hive table. In this example, the Data Source Type is Excel. This means the data set originated from the spreadsheet that was loaded in Studio and then was reloaded. You can tell this by observing the dates for creation and update.

The **Data Source** field shows the name of the source file. In this example, it is `WarrantyClaims.xls`.

Notice the ***Data Set Logical Name***. This is the name that each data set has, whether it exists in Catalog or belongs to a project. When a data set is in Catalog, it has one data set logical name. When you move it into a project, the data set's logical name changes. To run scripted updates with DP CLI, you need to note the correct data set logical name, so that you know which data set you are going to update. For information on scripted updates, see the *Data Processing Guide*.

Sampled and full data sets

Data sets in BDD can be sampled, or they can represent a full data set.

Sampled data sets

A *sample data set* in BDD represents a random sample of a source data set in Hive. If the data set originates in Hive, you use the Data Processing CLI to load it. The DP CLI uses the default sample size of 1 million records. You can specify a different sample size at data loading time.

- If you specify a sample size that is less than the size of the source data set in Hive, a sample data set is loaded into BDD.
- If you specify a sample size that is greater than or equal to the size of the source data set, a full data set is loaded.

Full data sets

A *full data set* in BDD represents a data set that contains all records, if you compare it to the source it was loaded from. For example, if a data set originates in Hive, and the sample size in DP CLI is greater than the record count in the source Hive table, this data set is loaded in full.

For a summary of how to get from a sample to a full data set, see *Data loading and sample size on page 31*

For more information on sampling and data set loading during data processing, see the *Data Processing Guide*.

For information on adding and managing data sets in Studio, including loading a full data set, see the *Studio User's Guide*.

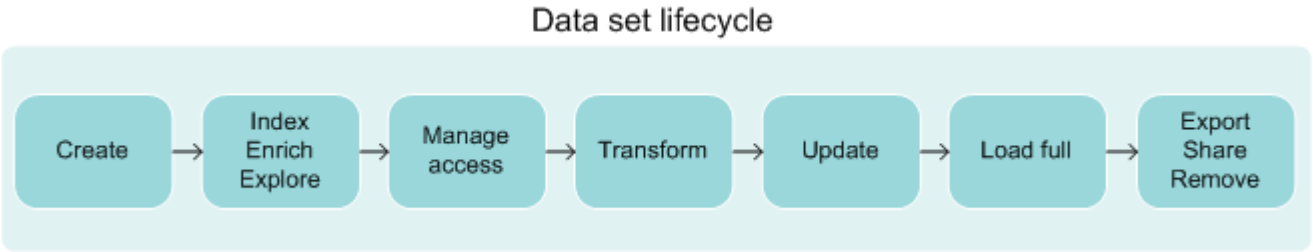
Data set lifecycle in Studio

As a data set flows through Big Data Discovery, it is useful to know what happens to it along the way.

Before we describe the data set lifecycle, here's how BDD interacts with source data sets it finds in Hive:

- BDD does not update or delete source Hive tables. When BDD runs, it only creates new Hive tables to represent BDD data sets. This way, your source Hive tables remain intact if you want to work with them outside of Big Data Discovery.
- Most of the actions in the BDD data set lifecycle take place because you select them. You control which actions you want to run. Indexing in BDD is a step that runs automatically.

This diagram shows stages in the data set's lifecycle as it flows through BDD:



In this diagram, the data set goes through these stages:

1. **Create** the data set. You create a data set in BDD in one of two ways:
   - Uploading source data using Studio. You can upload source data in delimited files and upload from a JDBC data source. When you upload source data, BDD creates a corresponding Hive source table based on the source data file.
   - Running the Data Processing CLI to discover Hive tables and create data sets in Studio based on source Hive tables. Each source Hive table has a corresponding data set in Studio.

   The Catalog of data sets in Studio displays two types of data sets. Some data sets originate from personally-loaded files or a JDBC source. Other data sets are loaded by Data Processing from source Hive tables.
2. Optionally, you can choose to **enrich** the data set. The data enrichment step in a data processing workflow samples the data set and runs the data enrichment modules against it. For example, it can run the following enrichment modules: Language Detection, Term Frequency/Inverse Document Frequency (TF/IDF), Geocoding Address, Geocoding IP, and Reverse Geotagger. The results of the data enrichment process are stored in the data set in Studio and not in the Hive tables.

   Note: The Data Processing (DP) component of BDD optionally performs this step as part of creating the data set.
3. **Create an index** of the data set. The Dgraph process creates binary files, called the Dgraph database, that represent the data set (and other configuration). The Dgraph accesses its database for each data set, to respond to Studio queries. You can now explore the data set.
4. **Manage** access to a data set. If you uploaded a data set, you have private access to it. You can change it to give access to other users. Data sets that originated from Hive are public. Studio's administrators can change these settings.
5. **Transform** the data set. To do this, you use various transformation options in **Transform**. In addition, you can create a new data set (this creates a new Hive table), and commit a transform script to modify an existing data set.

   If you commit a transform script's changes, Studio writes the changes to the Dgraph, and stores the changes in the Dgraph's database for the data set. Studio does not create a new Hive table for the data set. You are modifying the data set in the Dgraph, but not the source Hive table itself.
6. **Update** the data set. To update the data set, you have several options. For example, if you loaded the data set from a personal data file or imported from a JDBC source, you can reload a newer version of this data set in Catalog. If the data set was loaded from Hive, you can use DP CLI to refresh data in the data set.

   Also, you can **Load Full Data Set**. This option is useful for data sets that represent a sample. If a data set is in a project, you can also configure the data set for incremental updates with DP CLI.
7. **Export** the data set. When your data set is in a project, you can export it. For example, you can export a data set to HDFS, after you have applied transformations to it. This way, you can continue working with this data set using other tools. Also, you can export a data set and create a new data set in Catalog. Even though on this diagram **Export** is shown as the last step, you can export the data set at any stage in its lifecycle, after you add the data set to a project.
8. **Share** the data set. At any stage in the data set's lifecycle, you can share the data set with others.

9. **Remove** the data set. When you delete a data set from Studio, the data set is removed from the Catalog and is no longer accessible in Studio. However, deleting the data set does not remove the corresponding source Hive table that BDD created, when it loaded this data set.

   It's important to note that BDD does not update or delete original source Hive tables. BDD only creates new Hive tables to represent BDD data sets. You may need to ask your Hive data base administrator to remove old tables as necessary to keep the Hive data base clean. If the Hive data base administrator deletes a Hive table from the data base, the Hive Table Detector detects that the table was deleted and removes the corresponding data set from the Catalog in Studio. The Hive Table Detector is a utility in the Data Processing component of BDD.

Projects and applications in Big Data Discovery

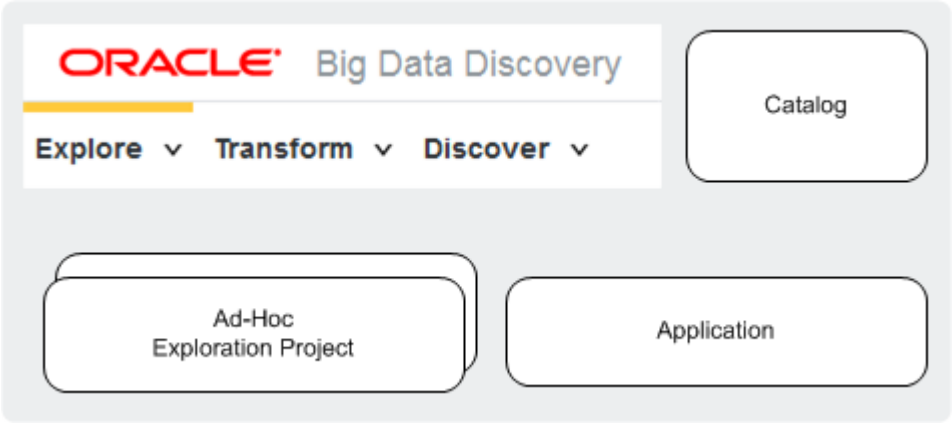
When you work with data sets in BDD, you add them in projects in Studio. Some BDD projects can become BDD applications.

BDD projects let you perform ad-hoc exploration and discovery using standard and advanced analytic techniques.

BDD applications expose an interactive analytic dashboard to a broad set of users.

While each BDD application is also a BDD project in Studio, it is referred to as a BDD application because it has a number of characteristics that make it special. In other words, you always start with projects in Studio; you can turn some projects into BDD applications.

The following diagram illustrates that BDD can contain one or more projects and an application:



In the diagram, the Catalog is also shown. It contains data sets that you select to add to your projects. Some of the projects you can later turn into BDD applications.

BDD projects

BDD projects are created by each user and serve as personal sandboxes. Each BDD deployment supports many BDD projects at once. This lets everyone in the BDD analyst community explore their own data, try different sample data sets, and identify interesting data sets for future in-depth analysis.

BDD projects often, but not always, run on sample data and allow you to load newer versions of sample data into them. Each BDD deployment can support dozens of ad-hoc, exploratory BDD projects. You can turn the most interesting or popular BDD projects into BDD applications.

Here are characteristics of a BDD project. It:

- is often a short-lived project, used to try out an idea
- is owned by a single user who can maintain, edit, and delete the project once no longer needed
- typically, but not always, runs on a sample of data
- answers a simple analytics question, or helps with initial exploration of the data set in Catalog
- lets you identify and select data set candidates for in-depth analysis.

As opposed to BDD projects that any user in BDD can create, BDD administrators own and certify BDD analytic applications, which they can share with their teams.

BDD applications

A ***BDD application*** includes a list of data sets that have been tweaked, linked and transformed, based on the goals of business analysis. Data analysts with power user permissions create and configure it. They can share it with other users in the BDD business analyst community who can use it for analysis.

Such an application answers a set of predefined questions and lets a group of users analyze the findings in it. A BDD application is often built on at least one full data set, with other data sets linking to it.

It is configured for periodic scripted data updates. You can either load newer versions of data onto it, or keep the existing data while periodically adding new data.

Each BDD deployment can support several BDD applications. The number of applications a BDD deployment can support depends on the capacity and sizing of your BDD deployment.

You can think of a BDD application as a "productized" or "certified" project that gained greater significance and is now maintained by power users (or by your IT organization) for use by a group of analysts within your team.

Here are the characteristics of a BDD application. It:

- is a longer-term project as opposed to other BDD projects
- often includes joins from multiple data sets
- often contains full data from at least one of the comprising data sets
- includes a predefined set of visualizations aimed at answering a set of specific questions
- allows you to run periodic data updates implemented with DP CLI
- has user access for a BDD application that is split between owners, curators and the team. Owners create and configure it, curators maintain it; the team can view and use visualizations. The team members may have different access to data sets in the application.
- has results and findings that are intended for sharing and viewing by a team (as opposed to exploratory BDD projects used by individual users who create them).

Data set access, access to projects, and user roles

Studio Administrators and project authors manage access to data sets and projects in Studio.

User roles

In Studio, two sets of user roles exist: a set of global permissions, and project-specific permission sets. For information on Studio's project access, data set access and user roles, see the *Administrator's Guide*.

Data set access

If you have Read access to a data set, you can see it in search results, or by browsing the Catalog. You can also explore it or add it to a project and then modify the project-specific version of the data set.

Write access allows you to set data set metadata or change data set permissions.

For information on changing access to a data set, see the *Studio User's Guide*.

For information on changing the default Studio settings for data set access, see the *Administrator's Guide*.

Project access

Projects are private by default. Only the project author and the Studio Administrators group can access them. If you are an Administrator or the project author, you can modify access to add user groups or individual users to your project. For information on changing access to a project, see the *Studio User's Guide*.

Chapter 4

Data Loading and Updates

This section discusses options for initial data loading and data updates. It illustrates how you can load files in Studio, or using Data Processing CLI.

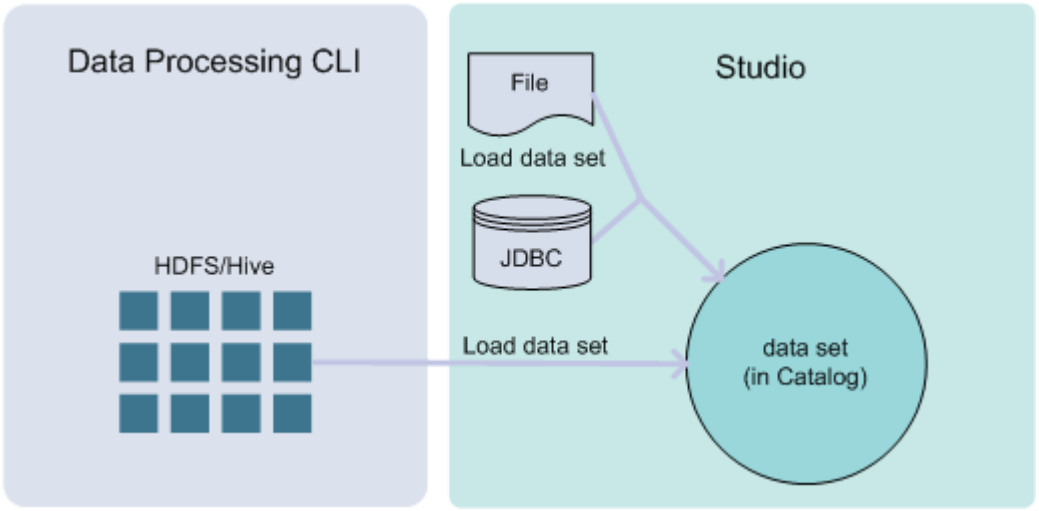
Data loading options

BDD offers several options for data loading. You can load data by running the data loading workflow with DP CLI. Also, in Studio, you can upload a personal file or import data from a JDBC source.

For initial loading of data into BDD, three methods exist:

1. **Load of Hive tables**. When you run DP CLI, it runs the data processing workflow and loads data from Hive tables into BDD.
2. **Personal file upload**. In Studio you can upload a data set from a personal file.
3. **Import from a JDBC source**. In Studio you can import a data set from a JDBC source.

This diagram illustrates data loading options:



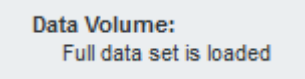
Also, you can load a sample or full data. Or, you can also start with a sample and then load full data. See *Data loading and sample size on page 31*.

Data loading and sample size

You can load either a sample or a full data set. If you load a sample, you can go to a full data set later. This topic summarizes how to get from a sample to a full data set.

These options are high-level summaries only. For detailed steps, see the referenced documentation for each option.

- **Controlling sample size when loading from Data Processing CLI**. DP CLI has a parameter for data size sample. The default sample size is 1 million records. When you use DP CLI for data loading, you can customize this parameter:
  - If it is less than the record count of source records in Hive, a full data set is loaded. In this case you already loaded a full data set. This is indicated in the Data Set Manager in Studio, in the **Data Volume** field:

  - If it is greater than the number of records in Hive, a sampled data set is loaded, based on the sample size you specify. In this case, you can use DP CLI with `--Incremental update` flag, or you can use **Load Full Data Set** in Studio, to load the entire source data set from Hive. You will then have a full data set in BDD.

  For detailed information on specifying the sample size with DP CLI, see the *Data Processing Guide*.
- **Controlling data set size when loading from a file or a JDBC source**.

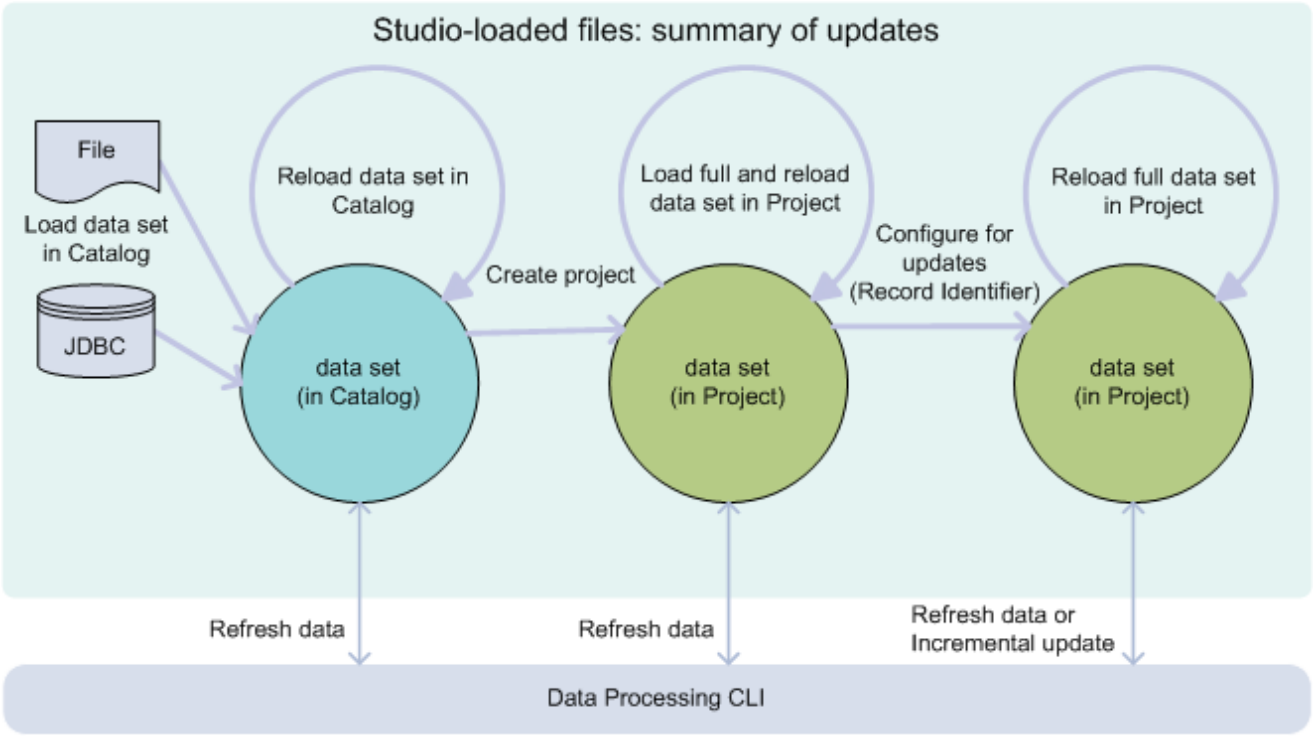
  If you load a data set from a personal file or import it from a JDBC source, then all data is loaded. However, it may still be a sample if you compare it to the source data you may also have elsewhere on your system.

  If you later want to add full data from the source, you can locate the Hive data set that BDD created when you loaded a file. Next, use the `drop` command to place that data set in Hue, and replace it with a production Hive table. You can then run **Load Full Data Set** on this table in Studio. This will load a full data set.

  This process is known as creating a BDD application. For detailed steps on this procedure, see the topic on creating a BDD application in the *Studio User's Guide*.

Studio-loaded files: data update diagram

The diagram in this topic shows data sets loaded in Studio by uploading a personal file or importing data from a JDBC source. It illustrates how you can reload this data set in Studio. Also, you can update the data set with DP CLI, and increase its size from sample to full.



In this diagram, the following actions take place, from left to right:

- You load data with the Studio's **Add data set** action. This lets you load a personal file, or data from a JDBC data source. Once loaded, the data sets appear in Catalog. BDD creates a Hive table for each data set. This Hive table is associated with the data set in Catalog.
- If you later have a newer version of these files, you can reload them with **Reload data set** action, found in data set's details, in Catalog.
- Next, if you'd like to transform, or to use **Discover** area to create visualizations, you should create a project. This is shown in the diagram in the middle circle. Notice that in this circle, the data set is now in your project and is no longer in Catalog. This data set can still be shown in Catalog, for others to use, if they have permissions to see it. However, now you have your own "version" of this data set, inside your project.
- Next, you can choose to load full data into this data set with **Load full data**, found in Data Set Manager. This replaces the data set sample with a fully loaded data set within your project. Or, if the data set was already fully loaded, you can reload the full data set in your project, by using this option again. This is shown as a circular arrow above the middle circle.
- Alternatively, you can use the options from Data Processing CLI to run scripted updates. There are two types of scripted updates: `Refresh Data` and `Incremental update`.

- To run `Refresh Data` with DP CLI, identify the data set's logical name in the properties for this data set in Studio. You must use that data set's logical name as a parameter for the data refresh command in DP CLI.
- To run `Incremental update` with DP CLI, you must specify a primary key (also known as record identifier in Studio). To add a primary key in Studio, go to **Control Panel**, then **Data Set Manager**, then **Configure for Updates**. You also need the data set's logical name for the incremental update.
- Be careful to use the correct data set logical name. Notice that a data set in Catalog differs from the data set in a project.
- Notice that you can run **Reload data set** in Catalog only for data sets in Catalog. If you want to update a data set that is already added to a project, you need to use one of the scripted updates from DP CLI.

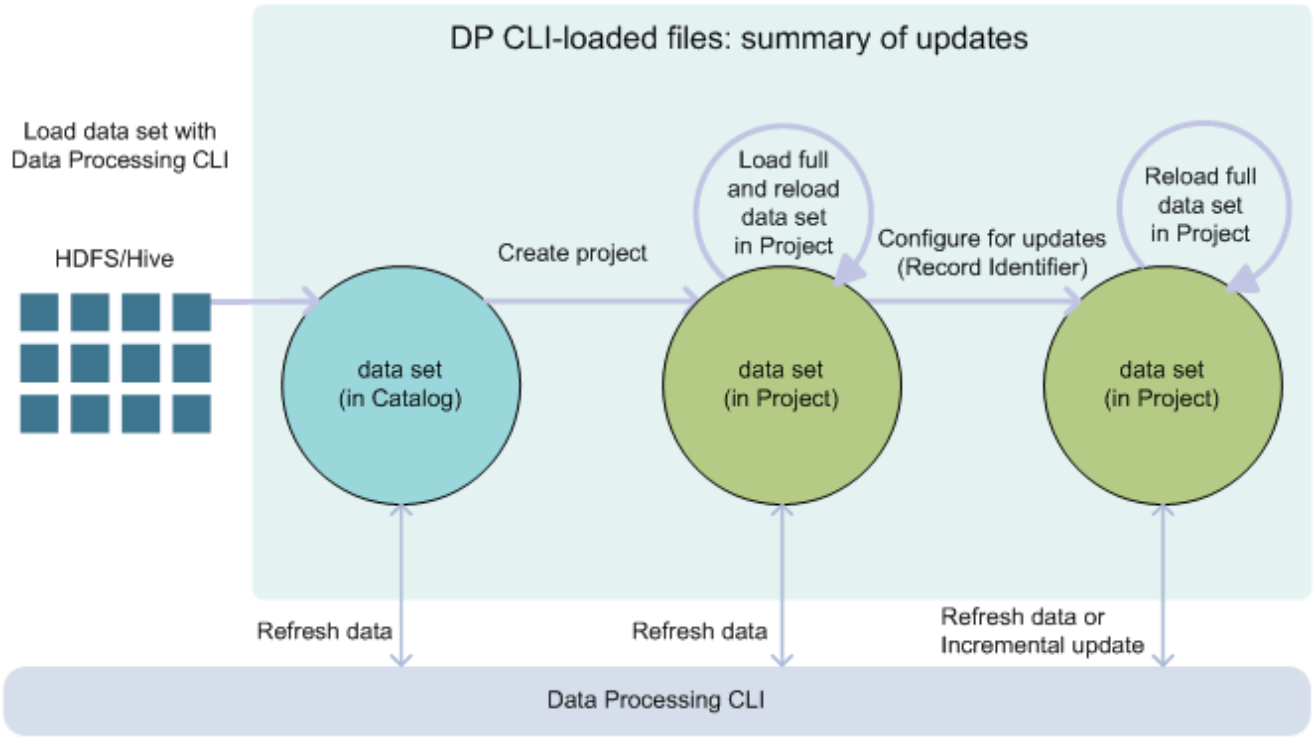
  If you reload a data set in Catalog that has a private copy in a project, Studio alerts you to this when you open the project, and gives you a chance to accept these updates. If any transformations were made to the data set in the project, you can apply them to the data set in Catalog.

Note the following about this diagram:

- You can `Refresh Data` with DP CLI for data sets in Catalog and in projects. Typically, you use DP CLI `Refresh Data` for data sets in a project.
- You can run an `Incremental Update` with DP CLI after you specify a record identifier. For this, you must move the data set into a project in Studio.
- Once you load full data, this does not change the data set that appears in Catalog. Moving a data set to a project is similar to creating your personal version of the data set. Next, you can load full data and write scripted updates to this data set using DP CLI commands `Refresh data` and `Incremental update`. You can run these updates periodically, as `cron` jobs. The updates will run on your personal version of this data set in this project. This way, your version of this data set is independent of the data set's version that appears in Catalog.

DP CLI-loaded files: data update diagram

The diagram in this topic shows data sets loaded by Data Processing component of BDD, from Hive. The diagram illustrates how you can update this data set using DP CLI, and increase its size from sample to full.



In this diagram, from left to right, the following actions take place:

- You load a data set from Hive using the Data Processing workflow (DP CLI). The data set appears in Catalog in Studio.
- You can now create a BDD project from this data set. Notice that a data set in your project does not eliminate this data set in Catalog. That is, other users may still see this data set in Catalog, if they have permissions. However, you have now taken this data set with you into your project. You can think of it as a different version of the data set, or your project's private version of the data set.
- In a project, you can load full data into it with the **Load full data set** action in Studio.
- Using Data Processing CLI, you can also run scripted updates to this data set. Two types of scripted updates exist: `Refresh Data` and `Incremental Update`.

  To run scripted updates, you need a data set logical name, found in the data set's properties in Studio. It is important to provide the correct data set logical name to the DP CLI. If the data set is in Catalog, it is not the same data set that you have in your project. Note the correct data set logical name.

Note the following about this diagram:

- You cannot run an `Incremental Update` with DP CLI for a data set in Catalog; you can only run it when you add the data set to a project.
- You can run an `Incremental Update` with DP CLI after you specify a record identifier. For this, you must move the data set into a project in Studio.

- Because this data set arrived from Hive, you cannot update it within Studio. Instead, you can use DP CLI for data set updates.
- Once you load full data, this does not change the data set that appears in Catalog. Moving a data set to a project and loading full data is similar to creating a personal version of the data set. Next, you can write scripted updates to this data set, using DP CLI commands `Refresh data` and `Incremental update`. You can run these updates periodically, as `cron` jobs. The updates will run on your personal version of this data set in this project. This way, your version of this data set is independent of the data set's version that appears in Catalog.

With this workflow, you create a project of your own, based on this data set, where you can run scripted updates with DP CLI. This approach works well for BDD projects that you want to keep around and populate with newer data.

This way, you can continue using the configuration and visualizations you built in Studio before, and analyze newer data as it arrives.

Data update options

Here is a summary of how you can update data loaded into BDD, and when each type of update is useful to use.

Options for data updates

To update already loaded data, you have these options:

- Reload data set in Studio
- Refresh data with DP CLI
- Run an incremental update with DP CLI
- When to use each type of update

Updates that you run with DP CLI are also called ***scripted updates***.

Reload data set in Studio

The **Reload data set** option in Studio is useful when you want to reload a newer version of the data than you loaded before. It applies to personally uploaded files and to data imported from a JDBC source. Note that this option works only for data sets in Studio's Catalog.

For a diagram of updating data sets that were loaded in Studio, see *Studio-loaded files: data update diagram on page 32* in this guide.

For detailed procedures for loading and reloading data in Studio, see the sections in the *Studio User's Guide*.

Refresh data with DP CLI

The `Refresh data` operation from DP CLI reloads an existing data set in a Studio project, replacing the contents of the data set with the latest data from Hive in its entirety. If the schema in the source Hive table changes, so does the newly-referenced data set. In this type of update, old data is removed and is replaced with new data. New attributes may be added, or attributes may be deleted. Also, the data type for an attribute may change.

For a diagram of updating data sets that were loaded with DP CLI, see *DP CLI-loaded files: data update diagram on page 34* in this guide.

For detailed information on how to run scripted updates with DP CLI, see the *Data Processing Guide*.

Run an incremental update with DP CLI

The `Incremental update` operation from DP CLI lets you add newer data to an existing BDD application, without removing already loaded data. In this type of update, the records' schema cannot change. An incremental update is most useful when you keep already loaded data, but would like to continue adding new data. For example, you can add more recent twitter feeds to the ones that are already loaded.

For a diagram of updating data sets that were loaded with DP CLI, see *DP CLI-loaded files: data update diagram on page 34* in this guide.

For detailed information on how to run scripted updates with DP CLI, see the *Data Processing Guide*.

When to use each type of update

This table summarizes when it is useful to use each type of update.

Type of data update	Useful when...
Reload data set in Catalog (in Studio)	This update is useful when you want to replace the loaded file with an updated version. Similarly, if data in a JDBC source was updated, you can reload it this way.
Scripted updates with DP CLI (`Refresh data` and `Incremental update`)	You can run scripted updates on files that originated from a Studio's upload, and on files that BDD discovers in Hive, when you run its data processing workflow for loading data using DP CLI. You can run either type of the scripted updates periodically, by writing update scripts and `cron` jobs on the Hadoop machines that utilize options for these updates from Data Processing CLI. Depending on characteristics of your source data, you may need to periodically run both types of scripted updates, or only one of them. For example, you may want to create a `cron` job that runs an incremental update nightly. This adds data from that day to the existing data set in a project in Studio. In addition to a periodic incremental update, you can run a `Refresh data` update weekly, to replace the data in the project wholesale with the new data that was collected in Hive during the week. A `Refresh data` update is also useful to run weekly because it lets you handle deletes from the source data set.

Chapter 5

What's New and Changed in this Release

This section describes the changes made for this release of BDD, including new, deprecated, and unsupported features.

New and updated features

The following features have been added, improved, or updated for the Oracle Big Data Discovery 1.4.x release.

Transform-related features

This release includes these transform-related features and improvements:

- You can use a new **Get Sentiment** transformation in Studio to extract positive or negative sentiment from attributes in your data, or label some attributes as having a positive or negative sentiment.

Improvements in data scale and management

This release includes these improvements to data and scale management:

- Continued data ingest parallelization further improves performance for data loading.
- The Data Processing component of BDD uses Parquet format for its sample files, instead of the Avro format used in the previous releases. During an upgrade to this release, Avro files are converted to Parquet. Parquet is an efficient storage format natively supported in Hive that separates storing data from metadata and improves query performance for columnar-based data in analytics applications.

BDD as a complete toolkit in your data lab

This release includes these changes:

- As the BDD administrator, you can better manage job status for data processing jobs in BDD. You can use new command line flags to get the job status, list currently running jobs, and cancel jobs by job ID.
- The stability and robustness of the indexing process in BDD has been improved: the Dgraph process validates all of its provided command line options and ports, reports inconsistencies, uses defaults in cases when invalid values are specified, and exits with errors in cases of port conflicts.
- The Workflow Manager Service is added in this release. Studio and the DP CLI use the Workflow Manager Service to run several data processing workflows in BDD. You can control the Workflow Manager Service settings in the **edp.properties** file in `$BDD_HOME/workflowmanager/dp/config`. This file is also new in this release. The introduction of the Workflow Manager Service separates the front-

end Studio processing from data processing, indexing, and other tasks within BDD, further improving maintenance of the entire pipeline. For information on the Workflow Manager Service and its configuration, see the *Data Processing Guide*.

Data set settings controlled by edp-cli.properties or in Studio

In this release, for the most part, the data set's workflow configuration is controlled by the Workflow Manager Service and its configuration file, **edp.properties**. However, there are a few exceptions, where you control the data set's workflow characteristics in other locations. This topic summarizes these exceptions.

In this release, three locations exist that set a data set's configuration:

1. The **edp.properties** file in the Workflow Manager Service directory, `$BDD_HOME/workflowmanager/dp/config`. This file is newly added in this release, along with the Workflow Manager Service.
2. The **edp-cli.properties** file in the CLI directory, `$BDD_HOME/dataprocessing/edp_cli/config`.

   Note: In this release, this file is renamed. In the previous release, its name was **edp.properties**.

3. The **Data Processing Settings** page in the **Control Panel** in Studio.

This table indicates exceptions and lists those data set's workflow settings that you change in **edp-cli.properties**, or in Studio. (All *other* data set workflows and their settings are set in the **edp.properties** file in the Workflow Manager directory.)

Data set workflow setting	If loaded or reloaded in Studio	If created or refreshed from DP CLI
Sample size	Change `bdd.sampleSize` in **Data Processing Settings** in the **Control Panel** on each Studio node.	Change `maxRecordsForNewDataSet` in `edp-cli.properties`, or override it with the `--maxRecords` flag on the command line.
Language	Change the data set's language when loading a new file in Studio.	Change the `defaultLanguage` property in `edp-cli.properties`.
Whether to run enrichments	Change `bdd.enableEnrichments` in **Data Processing Settings** page in the **Control Panel** on each Studio node.	Change `runEnrichment` in `edp-cli.properties`, or override it with the `--runEnrichment` flag on the command line.
Access type	The data set is always loaded as private and you cannot change this setting. However, you can include the data set into a project and share the project with others in your group.	Change `datasetAccessType` in `edp-cli.properties`.

For complete information on using these configuration files, see the *Data Processing Guide*.

For information on changing Data Processing settings in Studio, see the *Administrator's Guide*.

Chapter 6

Where You Go from Here

Quick reference to areas of interest

Quick reference to areas of interest

Use this table to locate information on a specific subject of BDD.

BDD feature	Where to find information on it?
List of addressed and known issues and most recent patches	Release notes for Big Data Discovery, available at the Oracle download site.
Installing BDD	*Installation Guide*
Upgrading BDD from a previous version	*Upgrade Guide*
Loading data into BDD with DP CLI	*Data Processing Guide*
Loading data in Studio from a personal file or a JDBC source	*Studio User's Guide*
Updating data sets with DP CLI: `Refresh data` and `Incremental update`	*Data Processing Guide*
Updating data sets in Studio	*Studio User's Guide*
Transforming data	*Studio User's Guide* and *Transform API Reference*
Data processing information in BDD, including default attribute types at loading time, sample size, DP CLI options, Dgraph HDFS Agent, and logs from the Data Processing component.	*Data Processing Guide*
Writing your own Custom Visualization Component	*Extensions Guide*
Administering BDD, including using `bdd-admin` script for starting, stopping and other lifecycle operations	*Administrator's Guide*
Understanding security features in BDD	*Security Guide*

Appendix A
Glossary

attribute

An ***attribute*** consists of a name and values on a record.

Just like columns describe rows in a table, attributes describe records in Big Data Discovery. Each set of attributes is specific to a particular data set of records. For example, a data set consisting of a store's products might contain attributes like "item name", "size", "color", and "SKU" that records can have values for. If you think of a table representation of records, a record is a row and an attribute name is a column header. Attribute values are values in each column.

An attribute's configuration in the schema controls three characteristics of each attribute: required (or not), unique (or not), and having a single or multiple assignments. In other words, an attribute's configuration in the schema determines whether an attribute is:

- Required. For a required attribute, each record must have at least one value assigned to it.
- Unique. For unique attributes, two records can never have the same assigned value.
- Single-value or multi-value (also known as single-assign and multi-assign). Indicates whether a record may have at most one value, or it can have more than one value assignments for the same attribute. Single-value attributes can at most have one assigned value on a record. For example, each item can have only one SKU. Multi-value attributes allow for multiple assigned values on a single record. For example, a Color attribute may allow multiple values for a specific record.

These characteristics of attributes, along with the attribute type, originate in the schema maintained in the Dgraph. In addition, Studio has additional attribute characteristics, such as refinement mode, or a metric flag. Studio also lets you localize attribute descriptions and display names.

Most attributes that appear in Big Data Discovery appear in the underlying source data. Also, in Big Data Discovery you can create new attributes, change, or delete attributes within a project. These changes are not persisted to the source data in Hive. Some attributes are the result of enrichments that Big Data Discovery runs on the data it discovers.

See also data set, Dgraph database, schema, record, type (for attributes), and value.

base view

The base view for a data set represents the fundamental attributes in a project data set. Base views represent the data "as is". You can create custom views, which are useful for data aggregation, computation, and visualization.

Custom views include only data from specific selected attributes (or columns) in the underlying data. They provide different ways of looking at data. Each custom view has a definition expressed as an EQL statement. Custom views do not eliminate the base view, which is always present in the system. You can create multiple custom views in your project in parallel to each other.

Linked views are created automatically when you join data sets. They are broadened views of data. Linked views widen a base view by joining the original data set with another data set.

BDD Application

A ***BDD Application*** is a type of BDD project that has special characteristics. An application often contains one or more data sets, where at least one of them may be loaded in full. You can transform and update data in a BDD application. Data updates can be done periodically. You maintain a BDD application for long-lasting data analysis and reporting on data that is kept up-to-date.

As opposed to ad-hoc exploratory BDD projects which any user in BDD can create, BDD administrators own and certify BDD analytic applications, which they can share with their teams.

See also project.

Big Data Discovery Cluster

A ***Big Data Discovery Cluster*** is a deployment of Big Data Discovery components on any number of nodes.

Nodes in the deployment have different roles:

- **DP nodes** represent machines in a pre-existing Hadoop cluster on which components of Big Data Discovery (those that require YARN) are deployed.
- **Studio nodes** are machines on which Java-based components of Big Data Discovery, such as Studio, run.
- **Dgraph nodes** represent machines in the Big Data Discovery cluster which the Dgraph instance is running on. They themselves form a Dgraph cluster within the Big Data Discovery cluster deployment.

You have multiple options for deploying Big Data Discovery in order to use hardware efficiently. For example, you can co-locate various parts of the Big Data Discovery on the same nodes. For information on BDD cluster deployment options, see the *Installation and Deployment Guide*.

Catalog

A Catalog is an area of the Studio application that lists:

- Data sets available to you
- Projects that you have access to

The Catalog contains options to create a new data set, explore a data set, or navigate to an existing project.

When the Data Processing component of Big Data Discovery runs, the available data sets are discovered by Big Data Discovery in the Hive database, profiled, and then presented as a list in the Catalog.

You can then use the Catalog to identify data sets of interest to you, by navigating and filtering data sets and projects based on data set metadata and various characteristics of projects. You can also display additional details about each data set or project, for further exploration.

The first time you log in to Big Data Discovery, the Catalog may display discovered data sets only, but not projects. After you or members of your group create and share projects, the Catalog displays them when you log in, in addition to the available data sets.

See also data set and project.

Custom Visualization Component

A ***Custom Visualization Component*** is an extension to Studio that lets you create customized visualizations in cases where the default components in Studio do not meet your specific data visualization needs.

custom views

Custom views are useful for data aggregation, computation, and visualization. Compared with base views that contain fundamental data for records, custom views include only data from specific selected attributes (or columns) in the underlying data. This way, custom views provide different ways of looking at data. Each custom view has a definition expressed as an EQL statement.

Custom views do not eliminate the base view, which is always present in the system. You can create multiple custom views in your project in parallel to each other.

See also base view and linked view.

data loading

Data loading is a process for loading data sets into BDD. Data loading can take place within Studio, or with Data Processing CLI.

In Studio, you can load data by uploading a personal file or data from a JDBC source. You can also add a new data set as the last step in transforming an existing data set.

Using DP CLI, you can load data by manually running a data loading workflow, or by adding it to a script that runs on your source data in Hive, and uses whitelists and blacklists, as well as other DP CLI parameters, to discover and load source data into BDD.

Often, you load a sample of data into BDD. You can use an option in DP CLI to change the sample size. Also, in Studio, you can load a full data set into your project created with sampled data. For information on loading full data, see the *Data Exploration and Analysis Guide*.

See also sampling, and data updates.

Data Processing (component of BDD)

Data Processing is a component in Big Data Discovery that runs various data processing workflows. Data processing includes several sub-components, such as the Transform Service used by Studio (to submit transform workflows and to let you preview transform results before applying them), and the Workflow Manager Service. The Workflow Manager serves as an intermediary between Spark jobs running in Yarn and the clients of BDD (Studio and Data Processing CLI), which submit their workflow requests to the Workflow Manager Service.

Here are some tasks involved in the data loading workflow:

- Discovery of data in Hive table
- Creation of a data set in BDD
- Running a select set of enrichments on discovered data sets
- Profiling of the data sets
- Indexing (by running the Dgraph process that creates the Dgraph database)

To submit this and other data processing workflows to the Workflow Manager when Big Data Discovery starts, you use the Data Processing Command Line Interface (DP CLI). For information, see the *Data Processing Guide*.

See also Dgraph database, enrichments, sampling, and profiling.

data set

In the context of Big Data Discovery, a ***data set*** is a logical unit of data, which corresponds to source data such as a delimited file, an Excel file, a JDBC data source, or a Hive table.

The data set becomes available in Studio as an entry in the Catalog. A data set may include enriched data and transformations applied to it from **Transform**. Each data set has a corresponding set of files in the Dgraph database.

A data set in Big Data Discovery can be created in different ways:

- When Big Data Discovery starts and you run its data processing workflow for loading data
- When you load personal data files (delimited or Excel files) using Studio
- When you load data from a JDBC data source using Studio
- When you use Transform features to create a new data set after running a transformation script
- When you export a data set from BDD into Hive, and BDD discovers it and adds it to Catalog.

See also sampling, attribute, database, schema, record, type (for attributes), and value.

data set import (personal data upload)

Data set import (or ***personal data upload***) is the process of manually creating a data set in Studio by uploading data from an Excel or delimited (CSV) file.

data updates

A ***data update*** represents a change to the data set loaded into BDD. Several types of updates are supported.

In Studio's Catalog, you can run **Reload data set** for a data set that you loaded from a personal file, or from a JDBC source. This is an update to a personally loaded file or to a sample from JDBC.

Using DP CLI, you can run two types of updates: `Refresh data` and `Incremental update`. These updates are also called scripted updates, because you can use them in your scripts, and run them periodically on data sets in a project in Studio.

The `Refresh data` operation from DP CLI reloads an existing data set in a Studio project, replacing the contents of the data set with the latest data from Hive in its entirety. In this type of update, old data is removed and is replaced with new data. New attributes may be added, or attributes may be deleted. Also, data type for an attribute may change.

The `Incremental update` operation from DP CLI lets you add newer data to an existing BDD application, without removing already loaded data. In this type of update, the records schema cannot change. An incremental update is most useful when you keep already loaded data, but would like to continue adding new data. For example, you can add more recent twitter feeds to the ones that are already loaded.

Dgraph

The ***Dgraph*** is a component of Big Data Discovery that runs search analytical processing of the data sets. It handles requests users make to data sets. The Dgraph uses data structures and algorithms to provide real-time responses to client requests for analytic processing and data summarization.

The Dgraph stores the database created after source data is loaded into Big Data Discovery. After the database is stored, the Dgraph receives client requests through Studio, queries its database, and returns the results.

The Dgraph is designed to be stateless. This design requires that a complete query is sent to it for each request. The stateless design facilitates the addition of Dgraph processes (during the installation) for load balancing and redundancy — any replica of a Dgraph can reply to queries independently of other replicas. The Dgraph Gateway routes requests to the Dgraph instances. The Dgraph HDFS Agent loads data into the Dgraph. Finally, the Dgraph Tracing Utility is a diagnostic service for the Dgraph.

Se also Dgraph database and the Dgraph Gateway.

Dgraph database

The ***Dgraph database*** represents the contents of a data set that can be queried by Dgraph, in Big Data Discovery. Each data set has its own Dgraph database. The Dgraph database is what empowers analytical processing. It exists both in persistent files in memory and on disk. The database refers to the entire set of files of the data set and to the logical structures into which the information they contain is organized internally. Logical structures describe both the contents and structure of the data set (schema).

The Dgraph database stores data in way that allows the query engine (the Dgraph) to effectively run interactive query workloads, and is designed to allow efficient processing of queries and updates. (The Dgraph database is sometimes referred to as an index).

When you explore data records and their attributes, Big Data Discovery uses the schema and its databases to allow you to filter records, identify their provenance (profiling), and explore the data using available refinements.

See also attribute, data set, schema, record, refinement, type (for attributes), and value.

Dgraph Gateway

The ***Dgraph Gateway*** is a sub-component in BDD. It is a Java-based interface in Big Data Discovery for the Dgraph, providing:

- Routing of requests to the Dgraph instances
- Caching
- Handling of cluster services for the Dgraph instances, using the ZooKeeper package from Hadoop

In BDD, the Dgraph Gateway and Studio are two Java-based applications co-hosted on the same WebLogic Server.

Discover

Discover is one of the three main modes, or major areas of Studio, along with **Explore** and **Transform**. As a user, you work within one of these three modes at a time.

Discover provides an intuitive visual discovery environment where you can compose and share discovery dashboards using a wide array of interactive data visualization components. It lets you link disparate data sources to find new insights, and publish them across the enterprise using snapshots.

Discover is where you create persistent visualizations of your data and share them with other users of your project.

See also **Explore** and **Transform**.

enrichments

Enrichments are modules in Big Data Discovery that extract semantic information from raw data to enable exploration and analysis. Enrichments are derived from a data set's additional information such as terms, locations, the language used, sentiment, and key phrases. As a result of enrichments, additional derived attributes (columns) are added to the data set, such as geographic data, or a suggestion of the detected language.

For example, BDD includes enrichments for finding administrative boundaries, such as states or counties, from Geocodes and IP addresses. It also has text enrichments that use advanced statistical methods to pull out entities, places, key phrases, sentiment, and other items from long text fields.

Some enrichments let you add additional derived meaning to your data sets. For example, you can derive positive or negative sentiment from the records in your data set. Other enrichments let you address invalid or inconsistent values.

Some enrichments run automatically during the data processing workflow for loading data. This workflow discovers data in Hive tables, and performs data set sampling and initial data profiling. If profiling determines that an attribute is useful for a given enrichment, the enrichment is applied as part of the data loading workflow.

The data sets with applied enrichments appear in Catalog. This provides initial insight into each discovered data set, and lets you decide whether the data set is a useful candidate for further exploration and analysis.

In addition to enrichments that may be applied as part of data loading by Data Processing, you can apply enrichments to a project data set from the **Transformation Editor** in **Transform**. From **Transform**, you can configure parameters for each type of enrichment. In this case, an enrichment is simply another type of available transformation.

See also transformations.

Explore

Explore is an area in Studio where you analyze the attributes and their values for a single data set. You can access **Explore** from the Catalog, or from within a project. You can use **Explore** to analyze attributes and their value distributions for a single data set at a time.

The attributes in **Explore** are initially sorted by name. You can filter displayed attributes, and change the sort order.

For each attribute, **Explore** provides a set of visualizations that are most suitable for that attribute's data type and value distribution. These visualizations let you engage with data to find interesting patterns and triage messy data.

Exploring a data set does not make any changes to that data set, but allows you to assemble a visualization using one or more data set attributes, and save it to a project page.

See also **Discover** and **Transform**.

exporting to HDFS/Hive

Exporting to HDFS/Hive is the process of exporting the results of your analysis from Big Data Discovery into HDFS/Hive.

From the perspective of Big Data Discovery, you are exporting the files from Big Data Discovery into HDFS/Hive. From the perspective of HDFS, you are importing the results of your work from Big Data

Discovery into HDFS. In Big Data Discovery, the ***Dgraph HDFS Agent*** is responsible for exporting to HDFS and importing from it.

The exporting to HDFS process is not to be confused with data set import, also known as personal data upload, where you add a data set to BDD, by uploading a file in Studio (in which case BDD adds a data set to Hive).

linked view

Linked views are created automatically when you join data sets. They are broadened views of data. Linked views widen a base view by joining the original data set with another data set.

See also base view and custom views.

metadata

Each data set includes various types of ***metadata*** — higher-level information about the data set attributes and values.

Basic metadata is derived from the characteristics of data sets as they are registered in Hive during data processing. This is called ***data profiling***. Big Data Discovery performs initial data profiling and adds metadata, such as Geocode values, derived from running various data enrichments.

As you explore and analyze the data in Big Data Discovery, additional metadata is added, such as:

- Which projects use this data set
- Whether the source data has been updated

Some metadata, such as attribute type, or multi-value and single-value for attributes, can be changed in **Transform**. Other metadata uses the values assigned during data processing.

In addition, various types of attribute metadata are available to you in Studio. They include:

- Attribute display names and descriptions
- Formatting preferences for an attribute
- Available and default aggregation functions for an attribute

Oracle Big Data Discovery

Oracle Big Data Discovery is a set of end-to-end visual analytic capabilities that leverage the power of Hadoop to transform raw data into business insight in minutes, without the need to learn complex products or rely only on highly skilled resources.

It lets you find, explore, and analyze data, as well as discover insights, decide, and act.

The Big Data Discovery software package consists of these main components:

- **Studio**, which is the front-end Web application for the product, with a set of unified interfaces for various stages of your data exploration:

  You use the Catalog to find data sets and **Explore** to explore them.

  You can then add data sets to projects, where you can analyze them, or use **Transform** to apply changes to them.

You can also export data to Hive, for further analysis by other tools such as Oracle R. Both **Explore** and **Transform** are part of the area in the user interface known as **project**. Note that you can explore data sets that are part of a project, as well as source data sets that are not included into any project but appear in **Explore**.

- The **Dgraph** is the query engine of Big Data Discovery.
- **Data Processing**, which runs various data processing workflows for BDD. For example, for data loading workflow, the Workflow Manager Service (that is part of data processing) receives a workflow request from Studio or Data Processing CLI and submits Spark jobs for discovery, sampling, profiling, and enrichments on source data found in Hive.

profiling

Profiling is a step in the data loading workflow run by the Data Processing component.

It discovers the characteristics of source data, such as a Hive table or a CSV file, and the attributes it contains, and creates metadata, such as attribute names, attribute data types, attribute's cardinality (a number of distinct values a record has from the attribute), and time when a data set was created and updated. For example, a specific data set can be recognized as a collection of structured data, social data, or geographic data.

Using **Explore**, you can look deeper into the distribution of attribute values or types.

Using **Transform**, you can adjust or change some of these metadata. For example, you can replace null attribute values with actual values, or fix other inconsistencies, such as change an attribute that profiling judged to be a String value into a number.

project

A BDD ***project*** is a container for data sets and user-customized pages in Studio. When you work with data sets in BDD, you put them in projects in Studio. In a project, you can create pages with visualizations, such as charts and tables.

As a user in Studio, you can create your own project. It serves as your individual sandbox for exploring your own data. In a project you can try adding different sample data sets, and identify interesting data sets for future in-depth analysis.

BDD projects often, but not always, run on sample data and allow you to load newer versions of sample data into them. Each BDD deployment can support dozens of ad-hoc, exploratory BDD projects for all Studio users. You can turn the most interesting or popular BDD projects into BDD applications.

From within a project, you can:

- Try out an idea on a sample of data
- Explore a data set and answer a simple analytics question
- Transform a data set
- Link data sets
- Create custom views of data set data
- Save and share it with others

See also BDD application.

record

A ***record*** is a collection of assignments, known as values, on attributes. Records belong to data sets.

For example, for a data set containing products sold in a store, a record can have an item named "T-shirt", be of size "S", have the color "red", and have an SKU "1234". These are ***values*** on attributes.

If you think of a table representation of records, a record is a row and an attribute name is a column header, where attribute values are values in each column.

record identifier (Studio)

A ***record identifier*** in Studio is one or more attributes from the data set that uniquely identify records in this data set.

To run an incremental update against a project data set, you must provide a ***record identifier*** for a data set so that the data processing workflow can determine the incremental changes to update, and you must load the full data set into the project. It is best to choose a record identifier with the highest percentage of key uniqueness (100% is the best).

refinement state

A ***refinement state*** is a set of filter specifications (attribute value selections, range selections, searches) to narrow a data set to a subset of the records.

sample

A ***sample*** is an indexed representative subset of a data set that you interact with in Studio. As part of its data loading workflow, Data Processing draws a simple random sample from the underlying Hive table, then creates a database for the Dgraph to allow search, interactive analysis, and exploration of data of unbounded size.

The default size of the sample is one million records. You can change the sample size.

sampling

Sampling is a step in the data loading workflow that Data Processing runs. Working with data at very large scales causes latency and reduces the interactivity of data analysis. To avoid these issues in Big Data Discovery, you can work with a sampled subset of the records from large tables discovered in HDFS. Using sample data as a proxy for the full tables, you can analyze the data as if using the full set.

During its data loading workflow, Data Processing takes a random sample of the data. The default sample size is one million records. You can adjust the sample size. If a source Hive table contains fewer records than the currently specified sample size, then all records are loaded. This is referred to as "data set is fully loaded". Even if you load a sample of records, you can later load a full data set in BDD, using an option in Studio's Data Set Manager.

schema

Schema defines the attributes in the data set, including characteristics of each attribute.

See also attribute, data set, Dgraph database, record, type (for attributes), and value.

scratchpad

The ***scratchpad*** is a part of **Explore** that lets you quickly compose visualizations using multiple attributes. When you add an attribute to the scratchpad, either by clicking on a tile, or by using typeahead within the scratchpad itself, the scratchpad renders a data visualization based on the attributes in the scratchpad. This lets you concentrate on the data, rather than on configuring this visualization yourself.

In addition to rendering a visualization, the scratchpad provides several alternative visualizations for the attributes, allowing you to quickly switch to an alternative view, without having to change any configuration. From within a project, you can save a scratchpad visualization to a page in **Discover** where you can apply a more fine-grained configuration.

semantic type

A ***semantic type*** is a setting in Studio that provides additional information about an attribute. It is a logical addition to an attribute that refines how an attribute is used in Studio. You add a semantic type to an attribute and then you can search and navigate based on the semantic type. A semantic type does not change an attribute's data type.

A semantic type can indicate whether an attribute represents an entity (places, people, organizations), personal information (SSN, phone numbers, emails, etc.), units of measure (currency, temperature, etc.), date times (year, month, day, etc.), and digital info (OS versions, IP addresses, etc.) For example, you could add a semantic type of Currency to an attribute named Price and then search and refine the data set by the keyword or value of Currency.

For details about creating semantic types, see the *Studio User's Guide*.

source data

Source data can be a CSV file, an Excel file, a JDBC data source, or a Hive table. All source data is visible in Hadoop, is stored in HDFS, and is registered as a Hive table.

Any source Hive table can be discovered by the data loading workflow that Data Processing (DP) component runs. As part of data loading, DP takes a random sample of a specific size and creates a data set in Dgraph, visible in the Catalog, for possible exploration and selection.

Once a sampled source data appears in the Catalog, it becomes a Big Data Discovery data set, and already represents a sample of the source Hive table.

Here is how BDD interacts with source data sets it finds in Hive:

- BDD does not update or delete source Hive tables. When BDD runs, it only creates new Hive tables to represent BDD data sets. This way, your source Hive tables remain intact if you want to work with them outside of Big Data Discovery.
- Most of the actions in the BDD data set lifecycle take place because you select them. You control which actions you want to run. Indexing in BDD is a step that runs automatically.

Studio

Studio is a component of Big Data Discovery. Studio provides a business-user friendly user interface for a range of operations on data.

Some aspects of what you see in Studio always appear. For example, Studio always includes search, **Explore**, **Transform**, and **Discover** areas. Other parts of your interface you can add as needed. These

include many types of data visualization components. For example, you can add charts, maps, pivot tables, summarization bars, timelines, and other components. You can also create custom visualization components.

Studio provides tools for loading, exploration, updating, and transformation of data sets. It lets you create projects with one or more data sets, and link data sets. You can load more data into existing projects. This increases the corpus of analyzed data from a sample to a fully loaded set. You can also update data sets. You can transform data with simple transforms, and write transformation scripts.

Studio's administrators can control data set access and access to projects. They can set up user roles, and configure other Studio settings.

Projects and settings are stored in a relational database.

token (in Studio)

A ***token*** is a placeholder (or variable) that Studio uses in the EQL query that powers a custom visualization. It lets you write an abstract EQL query once and provide a way for other users of your Studio project to swap in different values on demand, in place of a token.

Tokens can represent various aspects of an EQL query, such as attributes, views, sorts, or data. For example, using a view token in an EQL query allows project's users to employ the same query multiple times, to visualize different views. In the EQL query syntax in Studio's custom visualizations editor, tokens are strings enclosed by percentage signs (%).

After writing an EQL query, you can request Studio to detect tokens in the EQL script you wrote. You can then designate which of the tokens are intended to represent attributes, views or sorts. Until you designate what each token is for, tokens are unassigned. All tokens except data must be assigned to a query role before the visualization is complete.

See also Custom Visualization Component.

Transform

Transform is one of the three main areas of Studio, along with **Explore** and **Discover**. **Transform** is where you make changes to your project data set. It allows you to edit the data set values and schema, either to clean up the data, or to add additional values.

Transform unlocks data cleansing, manipulation and enrichment activities that are commonly restricted to rigid ETL processes. **Transform** lets you easily construct a transformation script through the use of quick, user-guided transformations as well as a robust, Groovy-based list of custom transform functions.

In **Transform**, you interactively specify a list of default enrichments and transformations, or you can write your own custom transformations. You can preview the results of applying the transformations, then add them to a transformation script that you can edit, run against the project data set, and save.

See also Explore and Discover.

Transformation Editor

The ***Transformation Editor*** is a part of **Transform** in Studio where you transform your data and often create derived attributes. Along with Groovy support, the **Transformation Editor** gives you access to a list of easy-to-use default transformations (based on Groovy) that let you speed up the process of data conversion, manipulation and enrichment.

Transform Service

The ***Transform Service*** processes transformations for Studio, and lets you preview the effects of transformations before they are applied. It runs in a Jetty container, and is often co-located with Studio on the same node, during installation.

Transformation script

A ***Transformation script*** is a sequential set of transformations organized into a script that you run against a project data set. When you run the transformation script against a project data set, no new entry is created in the Catalog, but the current project does reflect the effects of each transformation step in the script.

After you run a transformation script against project data set, you can also create a new version of the project data set, and publish it to the Catalog. This creates a new full data set in Hadoop, thus unlocking the transformed data for exploration in Big Data Discovery as well as in other applications and tools within Hadoop.

If a transformation script is useful to other Studio users, you can share the script by publishing it, so that it is available to load and run in other projects.

transformations

Transformations (also called transforms) are individual changes to a project data set. For example, you can apply any of the following transformations:

- Change data types
- Change capitalization of values
- Remove records
- Split columns into new ones (by creating new attributes)
- Group or bin values
- Extract information from values

Use transformations to shape and clean your data, such as to overwrite an existing attribute, modify attributes, or create new attributes. Most transformations are available directly as unique editors in **Transform**. Some transformations are enrichments.

You can you use the Groovy scripting language and a list of custom, predefined Groovy-based transform functions available in Big Data Discovery, to create a custom transformation.

See also enrichment, Transform, and Transformation Editor.

type (for an attribute)

An attribute's ***type*** determines the possible values that can be assigned to an attribute. Examples of attribute types include Boolean, Integer, String, Date, Double, and Geocode.

String attributes have additional characteristics related to text search.

See also attribute, data set, Dgraph database, schema, record, and value.

value (for an attribute)

An attribute's ***value*** is an assignment to an attribute for a specific record.

For example, for a data set containing products sold in a store, a record may have:

- For an attribute with the name "Item Name", an assignment to the attribute's value "t-shirt"
- For an attribute named "Color", an assignment to the attribute's value "red"
- For an attribute named "SKU", an assignment to the attribute's value "1234"

See also attribute, data set, Dgraph database, schema, record, and type (for attributes).

Workflow Manager Service

The ***Workflow Manager*** is a service in BDD that acts as an intermediary between Spark and the BDD clients: Studio and Data Processing CLI. The service receives data set workflow requests from Studio or Data Processing CLI, and delegates the sequence of Spark jobs needed for each workflow, such as sampling, discovery or transformations, to run in YARN. The Spark jobs run asynchronously of each other and the service notifies Studio of the job status. The Workflow Manager also delegates jobs to other components in BDD, such as the Dgraph and the Dgraph HDFS Agent.

It is installed in its own Jetty container and is often co-located on one of the Studio nodes.

Index