# Oracle® Big Data Discovery

Administrator's Guide

Version 1.2.2 • Revision A • October 2016

# Copyright and disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Table of Contents

## Part I: Overview of Big Data Discovery Administration

## Part II: Administering Big Data Discovery

## Part III: Administering Studio

## Part IV: Controlling User Access to Studio

## Part V: Logging for Studio, Dgraph, and Dgraph Gateway

# 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 describes how to administer Oracle Big Data Discovery.

# Audience

This guide is intended for administrators who configure, monitor, and control access to Oracle Big Data Discovery.

# 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.<br>For variables within a code sample, the formatting is ***`Variable`***. |
| `File Path` | This formatting is used for file names and paths. |

## Symbol conventions

The following table describes symbol conventions used in this document.

| Symbol | Description | Example | Meaning |
|---|---|---|---|
| > | The right angle bracket, or greater-than sign, indicates menu item selections in a graphic user interface. | File > New > Project | From the File menu, choose New, then from the New submenu, choose Project. |

### 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*.

# Part I

## Overview of Big Data Discovery Administration

Chapter 1

# Introduction

This section lists administrative tasks and tools that you can use to do these tasks.

*List of administrative tasks*

## List of administrative tasks

This topic lists top-level administrator tasks for Studio, the Dgraph, the Dgraph HDFS Agent, and the Dgraph Gateway.

| Section | Tasks |
|---|---|
| Overview of Big Data Discovery Administration | Learning about available administrative tools and logs used in Big Data Discovery, as well as learning about which files need to be backed up. Also, viewing the diagram of the Big Data Discovery cluster, learning about the cluster behavior, such as routing or requests, handling of data updates, and maintaining high availability. |
| Administering Big Data Discovery | Using the `bdd-admin` script for administering the product — starting, stopping and restarting the components, and checking the status of Big Data Discovery services. Also, performing administrative tasks for the BDD cluster deployment. |
| Administering the Dgraph | • Learning about the Dgraph, its memory consumption, the Dgraph internal cache, and a way to limit the Dgraph memory consumption for expensive queries.<br>• Running the Dgraph administrative operations with the `bdd-admin` script.<br>• Using flags for the Dgraph and for the Dgraph HDFS Agent. |
| Administering Studio | • Configuring framework settings.<br>• Configuring Hadoop settings for file upload.<br>• Managing data sources and viewing summary reports of project usage.<br>• Configuring the locale and email notifications.<br>• Managing projects in the Control Panel. |

| Section | Tasks |
|---|---|
| Controlling User Access to Studio | • Configuring user-related settings in Studio.<br>• Creating and managing users in Studio.<br>• Integrating with an LDAP system to manage users.<br>• Setting up Single Sign-On (SSO). |
| Logging | • Logging options in the `bdd-admin` script.<br>• Studio logs, their format and types, and customization options.<br>• Dgraph Gateway logs, their format, log levels, and customization options.<br>• Dgraph request log and stdout/stderr log. |

Chapter 2

# Cluster Architecture

This section describes the architecture of a Big Data Discovery cluster.

## Cluster components

A Big Data Discovery cluster is a deployment of Big Data Discovery on multiple machines. Such a deployment can be made up of any number of nodes.

### Overview

This topic provides an overview of the components in a Big Data Discovery cluster.

#### What is a BDD cluster?

A BDD cluster is an on-premise deployment of Big Data Discovery, either on commodity hardware or an engineered system, such as Oracle Big Data Appliance (BDA). It can consist of any number of individual nodes, although a production environment requires at least three to ensure enhanced availability of query processing. For example, a production deployment can include six nodes. Each node in the cluster is known as a ***BDD node***.

The cluster performs load-balancing for the Dgraph and routes requests arriving from Studio to it.

#### Nodes

Nodes in the BDD cluster deployment have different roles:

- WebLogic Server nodes host Studio and the Dgraph Gateway, which are Java-based applications. One WebLogic node functions as the Admin Server, which plays an administrative role in the cluster. All other WebLogic nodes are called Managed Servers.
- Dgraph nodes host the Dgraph instances. The Dgraph can be installed on HDFS DataNodes (if the Dgraph databases will be stored in HDFS) or on standalone (non-HDFS) nodes (if the Dgraph databases will be stored in a shared NFS drive). Together, these nodes constitute a Dgraph cluster within the overall

BDD cluster deployment. These nodes communicate with Hadoop and utilize Hadoop ZooKeeper to maintain high availability.

- Data Processing nodes are Hadoop Spark nodes that run data processing jobs for BDD.

For more information on nodes and their roles shown on a diagram, see *Diagram of a Big Data Discovery Cluster on page 15*.

**Note:** These roles are not mutually-exclusive. For example, in demo or learning deployments, you can co-locate Dgraph instances on the same nodes that run WebLogic Server or experiment with other configurations that have nodes serving dual roles. See the *Installation Guide* for information on deployment scenarios and co-location.

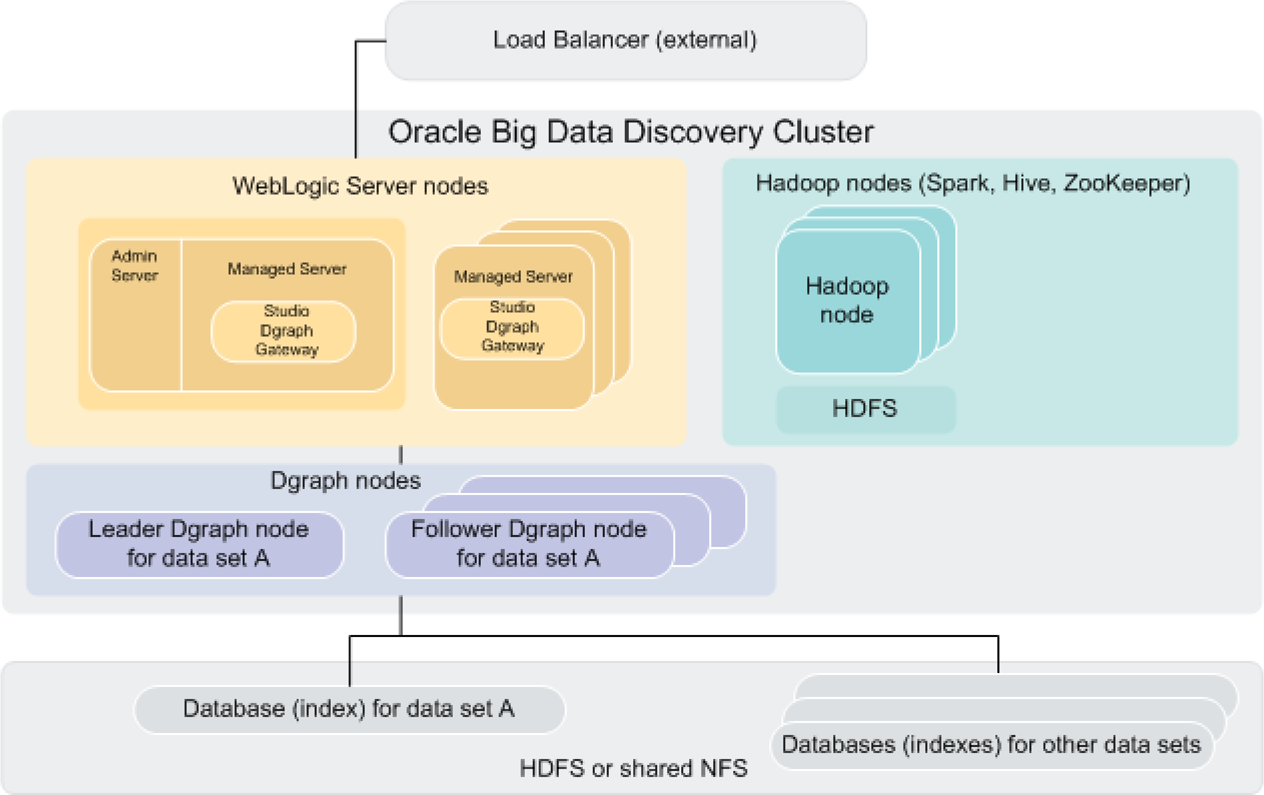
## Types of cluster deployments

BDD supports many different deployment configurations, so you can choose the one that makes the most efficient use of your hardware. The *Installation Guide* describes a few recommended deployment scenarios, including:

- A learning or demo deployment on one or two machines (this deployment is not intended to be turned into a production deployment).
- A production deployment on a set of six machines, with Data Processing, the Dgraph, and WebLogic (including Studio and the Dgraph Gateway) each running on two. The number of nodes in a production deployment can be fewer than six (with some components co-located), or more, depending on your needs.

## Diagram of a Big Data Discovery Cluster

This diagram illustrates a cluster of Big Data Discovery nodes deployed on top of an existing Hadoop cluster.


Load Balancer (external)
Oracle Big Data Discovery Cluster
WebLogic Server nodes
Admin Server
Managed Server
Studio Dgraph Gateway
Managed Server
Studio Dgraph Gateway
Hadoop nodes (Spark, Hive, ZooKeeper)
Hadoop node
HDFS
Dgraph nodes
Leader Dgraph node for data set A
Follower Dgraph node for data set A
Database (index) for data set A
Databases (indexes) for other data sets
HDFS or shared NFS

Note that this is just one supported deployment scenario; many other configurations are possible. For information on staging and learning, demo and production-level deployment topology, see the *Installation Guide*.

The diagram depicts the following BDD cluster components (starting from the top):

- An optional external load balancer serves as the single point of entry to the Big Data Discovery cluster. All browser requests are routed through this load balancer to Studio nodes.

  **Note:** Although it is recommended to use an external load balancer in your deployment, it is optional. For information, see *Load balancing and routing requests on page 17*.

- **WebLogic Server nodes**, which host Studio and the Dgraph Gateway. Note that one node functions as both the Admin Server and a Managed Server.
- **Data Processing nodes**, which run data processing jobs. Data Processing is automatically installed on Hadoop nodes running Spark on YARN, YARN, and HDFS. These nodes represent a subset of the Hadoop cluster BDD is installed on.
- **Dgraph nodes**, which host the Dgraph. These are the main computational modules in BDD, providing search, refinement computation, Guided Navigation, and other features used in Studio.

  The specific nodes the Dgraph is installed on depend on where your Dgraph databases are located. If they're in HDFS, the Dgraph is installed on HDFS DataNodes. If the indexes are on a shared NFS, the Dgraph can be installed on standalone (non-HDFS) nodes.

In the diagram, notice that there is a leader Dgraph node for a data set A, and a set of follower Dgraph nodes for this same data set.

At the same time, the directory holding Dgraph databases may include databases for other data sets. For each of these data sets, at different points in time, a leader Dgraph and follower Dgraph instances may be elected. The other data sets are shown in the diagram, but their leader and follower Dgraph nodes are not shown, for simplicity.

A single Dgraph instance can serve as the leader node for one Dgraph database and a follower for others. Note that there can never be two leader Dgraphs for a single Dgraph database.

ZooKeeper maintains a cluster state for all participating members of the BDD cluster; in particular, it ensures automatic Dgraph leader election for each of the Dgraph databases, in case a leader Dgraph instance fails. Optimally, three Hadoop nodes are required for hosting ZooKeeper instances.

- Additional Hadoop nodes, which are also not shown in the diagram. These run other Hadoop components required by BDD, such as Cloudera Manager/Ambari and ZooKeeper.

## Cluster of Dgraph nodes

A typical BDD cluster deployment includes a set of Dgraph nodes. Together, these nodes form a Dgraph cluster within the BDD cluster.

The ***Dgraph cluster*** handles requests for data sets in Studio. All Studio nodes talk to the same Dgraph cluster. The Dgraph cluster processes all queries to the data stored in a set of Dgraph databases, stored either in HDFS or on a shared NFS.

A Dgraph cluster provides enhanced availability for BDD query processing. If one node in the Dgraph cluster fails, queries are processed by the others. The cluster also increases throughput, as having multiple Dgraph nodes lets you spread the query load across them without having to increase storage requirements.

A BDD cluster can only contain one Dgraph cluster. The Dgraph cluster can have any number of nodes, although a certain number are recommended for a production environment. For more information, see the *Installation Guide*.

## Leader and follower Dgraph nodes

Dgraph nodes can have two roles within the Dgraph cluster: leader and follower.

### Leader Dgraph nodes

A leader Dgraph node receives and processes updates for a specific Dgraph database (i.e., for a specific data set). No other Dgraph node can perform write operations for that database. Note that a given Dgraph can be the leader for multiple databases. Leader Dgraphs are responsible for generating information about the latest versions of their indexes and propagating it to the other Dgraph nodes handling requests to a particular Dgraph database.

A leader is selected for a Dgraph database by Dgraph Gateway the first time a write operation (for example, a transformation from Studio) for that database comes in. Until that point, the database doesn't have a leader. Once a leader has been appointed for a Dgraph database, it remains the leader for as long as it's running.

Leader nodes periodically receive full or incremental database updates, as well as administration or configuration updates. After processing updates, the leaders publish new versions of their data and notify the other Dgraph nodes to start using the updated versions.

### Follower Dgraph nodes

Follower nodes are the Dgraph nodes that aren't the leader for a particular Dgraph database. They have read-only access to that database, meaning they can process queries for it but can't write to it.

Follower nodes process queries against a specific version of each index. When an index is updated, they receive the new version from the index's leader.

# Cluster behavior

There are many possible scenarios of Big Data Discovery deployment clusters. This section describes how the BDD cluster behaves and maintains enhanced availability in various scenarios, such as during node startup, updates to the Dgraph databases, or to individual node failures.

## Load balancing and routing requests

This topic discusses the load balancing and routing requests from Studio nodes to the Dgraph nodes in Oracle Big Data Discovery.

### Load balancing requests

Depending on your deployment strategy, to the external clients, the entry point of contact with the on-premise deployment of the Big Data Discovery cluster could be either any Studio-hosting node in the cluster, or an external load balancer configured in front of Studio instances.

The Big Data Discovery cluster relies on the following two levels of requests load balancing:

1. Load balancing requests across the nodes hosting multiple instances of Studio. This task should be performed by an external load balancer, if you choose to use it in your deployment (an external load balancer is not included in the Big Data Discovery package).

   If you use an external load balancer, it receives all requests and distributes them across all of the nodes in the Big Data Discovery cluster deployment that host the Studio application. Once a request is received from a Studio node, it is routed by BDD to the appropriate Dgraph node.

   If you don't use an external load balancer, external requests can be sent to any Studio node. They are then load-balanced between the nodes hosting the Dgraph.

2. Load balancing requests across the Dgraph nodes. This task is automatically handled by the BDD cluster. The Big Data Discovery software accepts requests from its Studio and Data Processing components on any node hosting the Dgraph and provides their internal load balancing across the other Dgraph-hosting nodes.

### Routing requests

The Big Data Discovery cluster automatically directs requests to the subset of the cluster nodes hosting the Dgraph instances.

Requests are submitted from either Studio or Data Processing to any Dgraph Gateway instance in the cluster, which in turn routes them to an appropriate Dgraph node. For example, an update request (such as a data loading request or a configuration update) is routed to the leader Dgraph for the Dgraph database that needs to be updated. Non-updating requests can be routed to any available Dgraph node. These are load-balanced between the Dgraph nodes using a round-robin algorithm.

The BDD cluster utilizes session affinity for all requests arriving from Studio to the Dgraph, by relying on the session ID in the header of each Studio request. Requests from the same session ID are always routed to the same Dgraph node in the cluster. This improves query processing performance by efficiently utilizing the Dgraph cache, and improves performance of caching for Studio's views.

## How session affinity is used

When a WebLogic Server node hosting Studio and Dgraph Gateway receives a client request, it routes the request to a Dgraph node using session affinity, based on the session ID specified in the header of the request.

When end users issue queries, Studio sets the session ID for the requests in the HTTP headers. Requests with the same session ID are routed to the same Dgraph node. If the BDD software cannot locate the session ID, it relies on a round-robin strategy for deciding which Dgraph node the request should be routed to.

Note that session affinity is enabled by default, via the `endeca-session-id-key` and `endeca-session-id-type` properties in the request headers.

## Startup of Dgraph nodes

Once the Big Data Discovery cluster is started, it activates the Dgraph processes on a subset of the nodes that are hosting the Dgraph instances. This topic discusses the behavior of the Dgraph nodes at startup.

On startup of a Dgraph, the following actions take place:

- A Dgraph starts up without any Dgraph databases mounted.
- If a Dgraph gets a Web service request involving a database that it has not mounted, it tries to mount it. The Dgraph mounts the database as a follower node by default or as a leader node if it has already been appointed leader by the Dgraph Gateway.
- Follower Dgraphs do not alter the Dgraph database in any way; they continue answering queries based on the version of the database to which they have access at startup, even if the leader Dgraph node is in the process of updating, merging, or deleting the database. Follower Dgraph nodes do not receive update requests; they acquire access to the new database once the updates complete.

Note that the Dgraph can start up without ZooKeeper being up. In this case, the Dgraph comes up in a running but not ready-for-requests state, which means that means that the Dgraph will not be able to service any requests involving accessing data. The Dgraph continues to wait and connects with ZooKeeper when ZooKeeper comes up.

## How updates are processed

In a Dgraph cluster, updates to the records or configuration in a specific Dgraph database are routed to that database's leader Dgraph node.

The leader processes the update and commits it to the on-disk Dgraph database for the data set. It then informs the follower nodes that a new version of the Dgraph database is available. The leader Dgraph node and all follower Dgraph nodes can continue to use the previous version of the database to finish query processing that had started against that version.

As each Dgraph node finishes processing queries on the previous version, it releases references to it. Once the follower nodes are notified of the new version, they acquire read-only access to it and start using it.

## Role of ZooKeeper

The ZooKeeper utility provides configuration and state management and distributed coordination services to Dgraph nodes of the Big Data Discovery cluster. It ensures high availability of the query processing by the Dgraph nodes in the cluster.

ZooKeeper is part of the Hadoop package. The Hadoop package is assumed to be installed on all Hadoop nodes in the BDD cluster deployment. Even though ZooKeeper is installed on all Hadoop nodes in the BDD cluster, it may not be running on all of these nodes. To ensure availability of a clustered Dgraph deployment, configure an odd number (at least three) of Hadoop nodes to run ZooKeeper instances. This will prevent ZooKeeper from being a single point of failure.

ZooKeeper has the following characteristics:

- It is a shared information repository that provides a set of distributed coordination services. It ensures synchronization, event notification, and coordination between the nodes. The communication and coordination mechanisms continue to work in the case when connections or Dgraph-hosting nodes fail.
- Zookeeper does not directly ensure automatic Dgraph leader election in the case a leader Dgraph instance fails. It simply informs the Dgraph Gateway of leader Dgraph failure, and the Dgraph Gateway is the component that starts automatic Dgraph leader re-election.

To summarize, in order to run, ZooKeeper requires a majority of its hosting nodes to be active. The optimal number of Hadoop nodes hosting ZooKeeper instances is an odd number that is at least 3.

## How enhanced availability is achieved

This topic discusses how the BDD cluster deployment ensures enhanced availability of query-processing.

**Important:** The BDD cluster deployment provides enhanced availability but does not provide high availability. This topic discusses the cluster behavior that enables enhanced availability and notes instances where system administrators need to take action to restore services.

The following three sections discuss the BDD cluster behavior for providing enhanced availability.

**Note:** This topic discusses BDD deployments with more than one running instance of the Dgraph. Even though you can deploy BDD on a single node, such deployments can only serve development environments, as they do not guarantee the availability of query processing in BDD. Namely, in a BDD deployment where only one node is hosting a single Dgraph instance, a failure of the Dgraph node shuts down the Dgraph process.

## Availability of WebLogic Server nodes hosting Studio

When a WebLogic Server node goes down, Studio also goes down. As long as the BDD cluster utilizes an external load balancer and consists of more than one WebLogic Server node on which Studio is started, this does not disrupt Big Data Discovery operations.

If a WebLogic Studio node hosting Studio fails, the BDD cluster (that uses an external load balancer) stops using it and relies on other Studio nodes, until you restart it.

## Availability of Dgraph nodes

The ZooKeeper ensemble running on a subset of Hadoop (CDH or HDP) nodes ensures the enhanced availability of the Dgraph cluster nodes and services:

- Failure of a leader Dgraph. When the leader Dgraph of a database goes offline, the BDD cluster elects a new leader and starts sending updates to it. During this stage, follower Dgraphs continue maintaining a consistent view of the data and answering queries. You should manually restart this node with the `bdd-admin` script. When the Dgraph that had a leader role is restarted and joins the cluster, it becomes one of the follower Dgraphs. It is also possible that the leader Dgraph is restarted and joins the cluster before the cluster needs to appoint a new leader. In this case, that Dgraph continues to serve as the leader.
- Failure of a follower Dgraph. When a follower Dgraph goes offline, the BDD cluster starts routing requests to other available Dgraphs. You should manually restart this node using the `bdd-admin` script. Once the node is restarted, it rejoins the cluster, and the cluster adjusts its routing information accordingly.

## Availability of ZooKeeper instances

The ZooKeeper instances themselves must be highly available. The following statements describe the requirements in detail:

- Each Hadoop node in the BDD cluster deployment can be optionally configured at deployment time to host a ZooKeeper instance. To ensure availability of ZooKeeper instances, it is recommended to deploy them in a cluster of their own, known as an ensemble. At deployment time, it is recommended that a subset of the Hadoop nodes is configured to host ZooKeeper instances. As long as a majority of the ensemble is running, ZooKeeper services are used by the BDD cluster. Because ZooKeeper requires a majority, the optimal number of Hadoop nodes hosting Zookeeper instances is an odd number that is at least 3.
- A Hadoop node hosting a ZooKeeper instance assumes responsibility for ensuring the ZooKeeper process uptime. It will start ZooKeeper when BDD is deployed and will restart it should it stop running.
- If you do not configure at least three Hadoop nodes to run ZooKeeper, it will be a single point of failure. Should ZooKeeper fail, the data sets served by BDD become entirely unavailable. To recover from this situation, the Hadoop node that was running a failed ZooKeeper must be restarted or replaced (the action required depends on the nature of the failure).

# Part II

## Administering Big Data Discovery

Chapter 3

# The bdd-admin Script

You can use the bdd-admin script to administrate your BDD cluster from the command line. This section describes the script and its commands.

## About the bdd-admin script

The bdd-admin script includes a number of commands that perform different administrative tasks for your cluster, like starting components and updating BDD's configuration. The script is located in the $BDD_HOME/BDD_manager/bin directory.

☆ **Important:** bdd-admin can only be run from the Admin Server by the bdd user. This user requires passwordless sudo enabled on all nodes in the cluster.

bdd-admin has the following syntax:

```
./bdd-admin.sh <command> [options]
```

When you run the script, you must specify a command. This determines the operation it will perform. You can't specify multiple commands at once, and you must wait for a command to complete before running it a second time. Additionally, you can't run the following commands at the same time:

- start
- stop
- restart
- backup
- restore

For example, if you run stop, you can't run start until all components have been stopped.

You can also include any of the specified command's supported options to further control the script's behavior. For example, you can run most commands on all nodes or one or more specific ones. The options each command supports are described later in this chapter.

The commands bdd-admin supports are described below.

## Lifecycle management commands

`bdd-admin` supports the following lifecycle management commands.

| Command | Description |
|---|---|
| `start` | Starts components. |
| `stop` | Stops components. |
| `restart` | Restarts components. |

## System management commands

`bdd-admin` supports the following system management commands.

| Command | Description |
|---|---|
| `autostart` | Enables/disables autostart for components. Components that have autostart enabled will automatically restart after their hosts are rebooted. |
| `backup` | Backs up your cluster's data and metadata to a single tar file. |
| `restore` | Restores your cluster's data and metadata from a backup tar file. |
| `publish-config` | Publishes updated BDD, Hadoop, and Kerberos configuration to all BDD nodes. Can also be used to refresh TLS/SSL certificates on secured Hadoop clusters. |
| `update-model` | Either updates the model files for Data Enrichment modules, or restores them to their original states. |
| `flush` | Flushes component caches. |
| `add-nodes` | Adds new nodes to your BDD cluster. |

## Diagnostics commands

`bdd-admin` supports the following diagnostics commands.

| Command | Description |
|---|---|
| `get-blackbox` | Generates the Dgraph's on-demand tracing blackbox file and returns its name and location. This command is intended for use by Oracle Support only. |
| `status` | Returns either component statuses or the overall health of the cluster. |
| `get-stats` | Returns component statistics. This command is intended for use by Oracle Support only. |

| Command | Description |
|---|---|
| `reset-stats` | Resets component statistics. This command is intended for use by Oracle Support only. |
| `get-log-levels` | Outputs the current levels of component logs. |
| `set-log-levels` | Sets the log levels for components and subsystems. |
| `get-logs` | Generates a zip file of component logs. This command is intended for use by Oracle Support only. |
| `rotate-logs` | Rotates component logs. This command is intended for use by Oracle Support only. |

### Global options

`bdd-admin` supports the following global options. You can include these with any command, or without a command.

| Command | Description |
|---|---|
| `--help` | Prints the usage information for the `bdd-admin` script and its commands. |
| `--version` | Prints version information for your BDD installation. |

For example, to view the usage for the entire `bdd-admin` script, run:

```
./bdd-admin.sh --help
```

To view the usage for a specific command, run the command with the `--help` flag:

```
./bdd-admin.sh <command> --help
```

For the version number of your BDD installation, run:

```
./bdd-admin.sh --version
```

## Lifecycle management commands

You can use the `bdd-admin` script's lifecycle management commands to perform such operations as starting and stopping BDD components.

*start*

*stop*

*restart*

## start

The `start` command starts components.

**Note:** `start` can't be run if `stop`, `restart`, `backup`, or `restore` are currently running.

To start components, run the following from the Admin Server:

```
./bdd-admin.sh start [option <arg>]
```

`start` supports the following options.

| Option | Description |
|---|---|
| `-c, --component` *`<component(s)>`* | A comma-separated list of the components to start:<br>• `agent`: Dgraph HDFS Agent<br>• `dgraph`: Dgraph<br>• `dp`: Data Processing<br>• `bddServer`: Studio and Dgraph Gateway<br>• `transform`: Transform Service<br>Note that if `start` runs on the `bddServer` component (or all components), it will prompt for the WebLogic Server username and password if the `BDD_WLS_USERNAME` and `BDD_WLS_PASSWORD` environment variables aren't set.<br>Additionally, `dp` can't be started if `bddServer` is stopped. |
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script starts all supported components.

### Examples

The following command starts all supported components:

```
./bdd-admin.sh start
```

The following command starts the Dgraph and the HDFS Agent on the `web009.us.example.com` node:

```
./bdd-admin.sh start -c dgraph,agent -n web009.us.example.com
```

## stop

The `stop` command stops components.

**Note:** Never use `SIGKILL`, `kill -9`, or any other OS command to stop BDD components. Always use `bdd-admin` with the `stop` command. If you need to stop a component immediately, run `stop` with `-t 0`.

To stop components, run the following from the Admin Server:

```
./bdd-admin.sh stop [option <arg>]
```

**Note:** `stop` can't be run if `start`, `restart`, `backup`, or `restore` is currently running.

`stop` supports the following options.

| Option | Description |
|---|---|
| `-t, --timeout` *`<minutes>`* | The amount of time to wait (in minutes) before terminating the component(s).<br>If this value is 0, the script forces the component(s) to shut down immediately. If it's greater than 0, the script waits the specified amount of time for the component(s) to shut down gracefully, then terminates them if they don't.<br>If this option isn't specified, the script shuts the component(s) down gracefully, which may take a very long time. If a component is down, a timeout value should be specified or the script will hang. |
| `-c, --component` *`<component(s)>`* | A comma-separated list of the components to stop:<br>• `agent`: Dgraph HDFS Agent<br>• `dgraph`: Dgraph<br>• `dp`: Data Processing<br>• `bddServer`: Studio and Dgraph Gateway<br>• `transform`: Transform Service<br>Note that when `stop` runs on the `bddServer` component (or all components), it will prompt for the WebLogic Server username and password if the `BDD_WLS_USERNAME` and `BDD_WLS_PASSWORD` environment variables aren't set.<br>Additionally, `dp` is automatically shut down when`bddServer` is stopped. |
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script stops all supported components gracefully.

## Note on stopping the dp component

When running on the `dp` component, `stop` performs two actions:

- Stops all active Data Processing jobs.
- Disables the Hive Table Detector cron job (if it's currently enabled).

However, `stop` doesn't actually stop the `dp` component from accepting jobs. For example, if you stop it and then run the `status` command, you'll see that Data Processing is ready to accept jobs:

```
./bdd-admin.sh stop -c dp
[Admin Server] Stopping BDD components...
[b4005.example.com] Stopping active Data Processing jobs.......Success!
[Admin Server] Successfully stopped all components.
...
./bdd-admin.sh status -c dp
[Admin Server] Checking the status of BDD components...
[b4005.example.com] DP is ready to accept jobs. Hive Data Detector is not scheduled to run.
[Admin Server] Successfully checked statuses.
```

The reason for this is that Data Processing isn't a server or service—it's a library that can invoke Spark-on-YARN jobs and is therefore always ready to accept new job requests. It's expected behavior that it can still accept jobs (such as those from manually running the DP CLI) after `stop` has been run. The most important things are that all existing jobs were stopped and that the Hive Table Detector is disabled, so there won't be any automatic job invocation.

### Examples

The following command gracefully shuts down all supported components:

```
./bdd-admin.sh stop
```

The following command waits 10 minutes for the Dgraph HDFS Agent, Dgraph, and Data Processing to shut down gracefully, then terminates any that are still running:

```
./bdd-admin.sh stop -t 10 -c agent,dgraph,dp
```

## restart

The `restart` command restarts components regardless of whether they're currently running or stopped.

**Note:** `restart` can't be run if `start`, `stop`, `backup`, or `restore` is currently running.

To restart components, run the following from the Admin Server:

```
./bdd-admin.sh restart [option <arg>]
```

`restart` supports the following options.

| Option | Description |
|---|---|
| `-t, --timeout` *<minutes>* | The amount of time to wait (in minutes) before terminating the component(s).<br>If this value is 0, the script forces the component(s) to shut down immediately. If it's greater than 0, the script waits the specified amount of time for the component(s) to shut down gracefully, then terminates them if they don't.<br>If this option isn't specified, the script shuts the component(s) down gracefully, which may take a very long time. If a component is down, a timeout value should be specified or the script will hang. |

| Option | Description |
|---|---|
| `-c, --component <component(s)>` | A comma-separated list of the components to restart:<br>• `agent`: Dgraph HDFS Agent<br>• `dgraph`: Dgraph<br>• `dp`: Data Processing<br>• `bddServer`: Studio and Dgraph Gateway<br>• `transform`: Transform Service<br>Note that `restart` runs on the `bddServer` component (or all components), it will prompt for the WebLogic Server username and password if the `BDD_WLS_USERNAME` and `BDD_WLS_PASSWORD` environment variables aren't set.<br>Additionally, `dp` can't be restarted if `bddServer` is stopped. |
| `-n, --node <hostname(s)>` | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script restarts all supported components gracefully.

### Examples

The following command gracefully shuts down and then restarts all supported components:

```
./bdd-admin.sh restart
```

The following command waits 5 minutes for the Dgraph and the HDFS Agent on the `web009.us.example.com` node to shut down gracefully, terminates it if it's still running, then restarts it:

```
./bdd-admin.sh restart -t 5 -c dgraph -n web009.us.example.com
```

## System management commands

You can use the `bdd-admin` script's system management commands to perform such operations as backing up your cluster and updating BDD's configuration.

## autostart

The `autostart` command enables and disables autostart for components. Components that have autostart enabled restart automatically after their hosts are rebooted.

**Note:** `autostart` doesn't restart components that crashed or were stopped by `bdd-admin` before a reboot.

To enable or disable autostart, run the following from the Admin Server:

```
./bdd-admin.sh autostart <operation> [option <arg>]
```

`autostart` requires one of the following operations.

| Operation | Description |
|---|---|
| `on` | Enables autostart for the specified component(s). |
| `off` | Disables autostart for the specified component(s). |
| `status` | Returns the status of autostart for the specified component(s). |

`autostart` also supports the following options.

| Option | Description |
|---|---|
| `-c, --component` *`<component(s)>`* | A comma-separated list of the components to run on:<br>• `agent`: Dgraph HDFS Agent<br>• `dgraph`: Dgraph<br>• `bddServer`: Studio and Dgraph Gateway<br>• `transform`: Transform Service |
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script runs on all supported components.

### Examples

The following command enables autostart for all supported components:

```
./bdd-admin.sh autostart on
```

The following command returns the status of autostart for the HDFS Agent running on the `web009.us.example.com` node:

```
./bdd-admin.sh autostart status -c agent -n web009.us.example.com
```

## backup

The `backup` command creates a backup of the cluster's data and metadata.

It backs up the following data to a single TAR file, which you can later use to restore the cluster:

- Studio database
- Schema and data for Hive tables created in Studio
- Dgraph databases
- Sample files in HDFS
- Configuration files

Partial backups aren't supported. Additionally, the backup doesn't include transient data, like state in Studio. This information will be lost if the cluster is restored.

Before running `backup`, verify the following:

- The `BDD_STUDIO_JDBC_USERNAME` and `BDD_STUDIO_JDBC_PASSWORD` environment variables are set. Otherwise, the script will prompt you for this information at runtime.
- The database client is installed on the Admin Server. For MySQL databases, this should be MySQL client. For Oracle databases, this should be Oracle Database Client, installed with a type of Administrator. The Instant Client isn't supported.
- For Oracle databases, the `ORACLE_HOME` environment variable is set to the directory one level above the `/bin` directory where the `sqlplus` executable is located. For example, if the `sqlplus` executable is located in `/u01/app/oracle/product/11/2/0/dbhome/bin`, `ORACLE_HOME` should be set to `/u01/app/oracle/product/11/2/0/dbhome/bin`.
- The temporary directories used during the backup operation contain enough free space. For more information, see *Space requirements on page 31* below.

**Note:** `backup` can't be run if `start`, `stop`, `restart`, or `restore` is currently running.

To back up the cluster, run the following from the Admin Server:

```
./bdd-admin.sh backup [option <arg>] <file>
```

Where <*file*> is the absolute path to the backup TAR file. This must not exist and its parent directory must be writable.

`backup` supports the following options.

| Option | Description |
|---|---|
| `-o, --offline` | Performs a cold backup. Use this option if your cluster is down. If this option isn't specified, the script performs a hot backup.<br>More information on hot and cold backups is available below. |

| Option | Description |
|---|---|
| `-r, --repeat <num>` | The number of times to repeat the backup process if verification fails. This is only used for hot backups.<br>If this option isn't specified, the script makes one attempt to back up the cluster. If it fails, the script must be rerun.<br>More information on verification is available below. |
| `-l, --local-tmp` | The absolute path to the temporary directory on the Admin Server used during the backup operation. If this option isn't specified, the location defined by `BACKUP_LOCAL_TEMP_FOLDER_PATH` in `bdd.conf` is used. |
| `-d, --hdfs-tmp` | The absolute path to the temporary directory on HDFS used during the backup operation. If this option isn't specified, the location defined by `BACKUP_HDFS_TEMP_FOLDER_PATH` in `bdd.conf` is used. |
| `-v, --verbose` | Enables debugging messages. |

If no options are specified, the script makes one attempt to perform a hot backup and doesn't output debugging messages.

For more information on backing up the cluster, see *Backing up Big Data Discovery on page 52*.

## Space requirements

When the script runs, it verifies that the temporary directories it uses contain enough free space. These requirements only need to be met for the duration of the backup operation.

- The destination of the backup TAR file must contain enough space to store the Dgraph databases, the HDFS sandbox, and the `edpDataDir` (defined in `edp.properties`) at the same time.
- The `local-tmp` directory on the Admin Server also requires enough space to store all three items simultaneously.
- The `hdfs-tmp` directory on HDFS must contain enough free space to accommodate the largest of these items, as it will only store them one at a time.

If these requirements aren't met, the script will fail.

## Hot vs. cold backups

`backup` can perform both hot and cold backups:

- Hot backups are performed while the cluster is running. Specifically, they're performed on the first Managed Server (defined by `MANAGED_SERVERS` in `bdd.conf`), and require that the components on that node are running. This is `backup`'s default behavior.
- Cold backups are performed while the cluster is down. You must include the `-o` option to perform a cold backup.

### Verification

Because hot backups are performed while the cluster is running, it's possible for the data in the backups of the Studio and Dgraph databases and sample files to become inconsistent. For example, something could be added to a Dgraph database after the database was backed up, which would make the data in those locations different.

To prevent this, `backup` verifies that the data in all three backups is consistent. If it isn't, the operation fails.

By default, `backup` only backs up and verifies the data once. However, it can be configured to repeat this process by including the `-r` <*num*> option, where <*num*> is the number of times to repeat the backup and verification steps. This increases the likelihood that the operation will succeed.

**Note:** It's unlikely that verification will fail the first time, so it's not necessary to repeat the process more than once or twice.

### Examples

The following command performs a hot backup with debugging messages:

```
./bdd-admin.sh backup -v /tmp/bdd_backup1.tar
```

The following command performs a cold backup:

```
./bdd-admin.sh backup -o /tmp/bdd_backup2.tar
```

## restore

The `restore` command restores your cluster from an existing backup TAR file.

It completely restores the following from backup:

- Studio database
- Schema and data for Hive tables created in Studio
- Dgraph databases
- Sample files in HDFS

It also restores some of the configuration settings, but not all of them. See below for more information.

**Note:** The script makes a copy of the current Dgraph databases directory in `DGRAPH_INDEX_DIR/.snapshot/old_copy`. This should be deleted if the restored version is kept.

Before running `restore`, verify the following:

- The `BDD_STUDIO_JDBC_USERNAME` and `BDD_STUDIO_JDBC_PASSWORD` environment variables are set. Otherwise, the script will prompt you for this information at runtime.
- The database client is installed on the Admin Server. For MySQL databases, this should be MySQL client. For Oracle databases, this should be Oracle Database Client, installed with a type of Administrator. The Instant Client isn't supported.
- For Oracle databases, the `ORACLE_HOME` environment variable is set to the directory one level above the `/bin` directory where the `sqlplus` executable is located. For example, if the `sqlplus` executable is located in `/u01/app/oracle/product/11/2/0/dbhome/bin`, `ORACLE_HOME` should be set to `/u01/app/oracle/product/11/2/0/dbhome/bin`.

- The temporary directories used during the restore operation contain enough free space. For more information, see *Space requirements on page 33* below.

**Note:** `restore` can't be run if `start`, `stop`, `restart`, or `backup` is currently running.

To restore the cluster, run the following from the Admin Server:

```
./bdd-admin.sh restore [option] <file>
```

Where <*file*> is the absolute path to the backup TAR file to restore from. This must be a TAR file created by the `backup` command. The cluster that was backed up to this file and the current cluster must have the same major version of BDD as well as the same type of database (Oracle or MySQL; `restore` doesn't support Hypersonic databases). They can have different topologies.

`restore` supports the following options.

| Option | Description |
|---|---|
| `-l, --local-tmp` | The absolute path to the temporary directory on the Admin Server used during the restore operation. If this option isn't specified, the location defined by `BACKUP_LOCAL_TEMP_FOLDER_PATH` in `bdd.conf` is used. |
| `-d, --hdfs-tmp` | The absolute path to the temporary directory on HDFS used during the restore operation. If this option isn't specified, the location defined by `BACKUP_HDFS_TEMP_FOLDER_PATH` in `bdd.conf` is used. |
| `-v, --verbose` | Enables debugging messages. |

For more information on restoring your cluster, see *Restoring Big Data Discovery on page 53*.

## Space requirements

When the script runs, it verifies that the temporary directories it uses contain enough free space. These requirements only need to be met for the duration of the restore operation.

- The `local-tmp` directory on the Admin Server must contain enough space to store the Dgraph databases, the HDFS sandbox, and the `edpDataDir` (defined in `edp.properties`) at the same time.
- The `hdfs-tmp` directory on HDFS must contain free space equal to the largest of these items, as it will only store them one at a time.

If these requirements aren't met, the script will fail.

## Configuration restoration

`restore` can't completely restore the configuration files because the current cluster may have a different topology than the backup cluster. Instead, it merges some of them with the ones from the current cluster and leaves others unchanged.

The following table describes the changes the script makes to each configuration file.

| File | Changes |
|---|---|
| `bdd.conf` | The script restores the following properties from backup:<br>• `MAX_RECORDS`<br>• `ENABLE_ENRICHMENTS`<br>• `LANGUAGE`<br>• `SPARK_DRIVER_MEMORY`<br>• `SPARK_DRIVER_CORES`<br>• `SPARK_DYNAMIC_ALLOCATION`<br>• `SPARK_EXECUTOR_MEMORY`<br>• `SPARK_EXECUTOR_CORES`<br>• `SPARK_EXECUTORS`<br>• `YARN_QUEUE`<br>No other properties are modified. |
| `portal-ext.properties` | The script restores the following properties from backup:<br>• `dp.spark.dynamic.allocation`<br>• `sp.spark.driver.cores`<br>• `dp.spark.driver.memory`<br>• `dp.spark.executors`<br>• `dp.spark.executor.cores`<br>• `dp.spark.executor.memory`<br>• `dp.yarn.queue`<br>• `dp.settings.language`<br>No other properties are modified. |
| `esconfig.properties` | The script adds any properties from the backup versions of these files that aren't in the current ones. It doesn't modify any other settings. |
| `edp.properties` | The script restores all settings that don't affect cluster topology. Note that all other Data Processing configuration files will be fully restored. |

## Examples

The following command restores your cluster from the `/tmp/bdd_backup1.tar` file with no debugging messages:

```
./bdd-admin.sh restore /tmp/bdd_backup1.tar
```

## publish-config

The `publish-config` command publishes configuration changes to your BDD cluster.

To update the cluster configuration, run the following from the Admin Server:

```
./bdd-admin.sh publish-config <config type> [option <arg>]
```

**Note:** After `publish-config` runs, the cluster must be restarted for the changes to take effect.

`publish-config` requires one of the following configuration types.

| Configuration type | Description |
|---|---|
| `bdd` *<path>* | Publishes an updated version of `bdd.conf` specified by *<path>* to all BDD nodes. See *bdd on page 35* for more information. |
| `hadoop` *[option <arg>]* | Publishes Hadoop configuration changes to all BDD nodes and performs any other operations defined by the specified options. See *hadoop on page 36* for more information. |
| `kerberos` *<option <arg>>* | Publishes the specified Kerberos principal, `krb5.conf` file, or keytab file to all BDD nodes. See *kerberos on page 37* for more information. |
| `cert` *<path>* | Refreshes the TLS/SSL certificates on clusters secured with TLS/SSL. See *cert on page 38* for more information. |

### bdd

The `bdd` configuration type publishes an updated version of `bdd.conf` to all BDD nodes. This updates the configuration of the entire cluster.

To update the cluster configuration, edit a *copy* of `bdd.conf` on the Admin Server, then run:

```
./bdd-admin.sh publish-config bdd <path>
```

Where *<path>* is the absolute path to the modified copy of `bdd.conf`.

**Note:** It's recommended to edit a *copy* of `bdd.conf` to preserve the original in case the changes need to be reverted.

When the script runs, it makes a backup of the original `bdd.conf` in `$BDD_HOME/BDD_manager/conf` on the Admin Server. The backup is named `bdd.conf.bak`*<num>*, where *<num>* is the number of the backup; for example, `bdd.conf.bak2`. This file can be used to revert the configuration changes, if necessary.

The script then copies the modified version of `bdd.conf` to all BDD nodes in the cluster. When it completes, the cluster must be restarted for the changes to take affect.

**Note:** When `bdd` runs, any component log levels you've set on specific nodes using the `set-log-levels` command will be overwritten by the `DGRAPH_LOG_LEVELS` property in the updated file.

For more information on updating your cluster configuration, see *Updating the cluster configuration on page 50*.

## hadoop

The hadoop configuration type makes changes to BDD's Hadoop configuration.

Depending on the specified options, hadoop can:

- Publish new or updated Hadoop client configuration files to your BDD cluster.
- Reset the HUE_URI property in bdd.conf (HDP clusters only).
- Switch to a different version of your Hadoop distribution without reinstalling BDD.

**Note:** hadoop can be used to switch to a different Hadoop distribution.

To update BDD's Hadoop configuration, run the following from the Admin Server:

```
./bdd-admin.sh publish-config hadoop [option <arg>]
```

hadoop supports the following options.

| Option | Description |
|---|---|
| -u, --hueuri *<host>:<port>* | HDP clusters only. Sets the HUE_URI property in bdd.conf to the specified URI. |
| -l, --clientlibs *<path[,path]>* | Regenerates the Hadoop fat jar from a comma-separated list of client libraries. *<path[,path]>* must be a comma-separated list of the new libraries. This can be used to switch to a different version of your Hadoop distribution.<br>This must be run with --sparkjar. |
| -j, --sparkjar *<file>* | Sets the location of the Spark on YARN jar in all BDD configuration files to the specified path. *<file>* must be the absolute path to the Spark on YARN jar on the Hadoop nodes. This can be used to switch to a different version of your Hadoop distribution.<br>This must be run with --clientlibs. |

If no options are specified, the script publishes the Hadoop client configuration files to all BDD nodes and updates the Hadoop-related properties in all BDD configuration files.

For more information on the actions performed by this configuration type, see:

- *Updating the Hadoop client configuration files on page 55*
- *Setting the Hue URI on page 56*
- *Switching Hadoop versions on page 56*

## kerberos

The `kerberos` configuration type updates to BDD's Kerberos configuration.

Depending on the specified options, `kerberos` can do the following:

- Enable Kerberos
- Update the location of `krb5.conf` in BDD's configuration files
- Update the BDD principal
- Publish a new keytab file to all BDD nodes

To update BDD's Kerberos configuration, run the following from the Admin Server:

```
./bdd-admin.sh publish-config kerberos [operation] <option>
```

`kerberos` requires one of the following operations.

| Operation | Description |
|---|---|
| `on` | Enables Kerberos. The `-k`, `-t`, and `-p` options must also be specified. |
| `config` | Updates BDD's Kerberos configuration. At least one option must be specified.<br>This is the command's default behavior, so this operation is optional. You can only use this if Kerberos is already enabled. |

`kerberos` supports the following options.

| Option | Description |
|---|---|
| `-k, --krb5 <file>` | Updates the location of `krb5.conf` in all BDD configuration files. `<file>` must be the new absolute path to the file.<br>`krb5.conf` must be moved to its new location on all BDD nodes before running this option. |
| `-t, --keytab <file>` | Publishes the specified keytab file to all BDD nodes. `<path>` must be the absolute path to the new keytab file.<br>The script renames this file `bdd.keytab` and copies it to `$BDD_HOME/common/kerberos`. |
| `-p, --principal <principal>` | Publishes the specified principal to all BDD nodes. This option can't be used to change the primary component of the principal. |

For more information on updating your Kerberos configuration, see *Updating BDD's Kerberos configuration on page 58*.

### cert

The `cert` configuration type refreshes the TLS/SSL certificates for the HDFS, YARN, Hive, and KMS services.

Before running this command, you must export the updated certificates from HDFS and copy them to a single location on the Admin Server.

To refresh the certificates, run:

```
./bdd-admin.sh publish-config cert <path>
```

Where *<path>* is the absolute path to the updated certificates.

When the script runs, it imports the certificates to the custom truststore file, then copies the truststore to `$BDD_HOME/common/security/cacerts` on all BDD nodes.

For more information on refreshing your certificates, see *Refreshing TLS/SSL certificates on page 65*.

## update-model

The `update-model` command updates or resets the models used by some of the Data Enrichment modules.

To update or reset the models used by the Data Enrichment modules, run the following command from the Admin Server:

```
./bdd-admin.sh update-model <model_type> [path]
```

`update-model` requires one of the following model types.

| Model type | Description |
|---|---|
| `geonames` | The model for the GeoTagger Data Enrichment modules. |
| `tfidf` | The model for the TF.IDF Data Enrichment module. |
| `sentiment` | The model for the Sentiment Analysis Data Enrichment modules. |

*[path]* is the absolute path to the location of the files to update the model with. This argument is optional. You must move these files to a single directory on the Admin Server before running the script.

If *[path]* is included, the script creates a jar from the files in the specified directory, then replaces the current jar on the YARN worker nodes with the new one. If *[path]* isn't included, the script resets the specified model to its original state.

For details on configuring the input directories and files for the models, see the *Data Processing Guide*.

### Reverting model changes

You can revert the changes made to the models by running the script without the *[path]* argument. For example, the following command resets the `tfidf` model:

```
./bdd-admin.sh update-model tfidf
```

## flush

The `flush` command flushes component caches.

To flush component caches, run the following from the Admin Server:

```
./bdd-admin.sh flush [option <arg>]
```

`flush` supports the following options.

| Option | Description |
|---|---|
| `-c, --component <component(s)>` | A comma-separated list of the component caches to flush:<br>• `dgraph`: Dgraph<br>• `gateway`: Dgraph Gateway<br>When debugging query issues, cold-start or post-update performance can be approximated by cleaning the Dgraph cache before running a request. |
| `-n, --node <hostname(s)>` | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script flushes the caches of all supported components.

### Examples

The following command flushes all Dgraph and Dgraph Gateway caches in the cluster:

```
./bdd-admin.sh flush
```

The following command flushes the Dgraph cache on the `web009.us.example.com` node:

```
./bdd-admin.sh flush -c dgraph -n web009.us.example.com
```

## add-nodes

The `add-nodes` command adds new nodes to your BDD cluster.

When the script runs, it queries your Hadoop manager (Cloudera Manager or Ambari) for newly-added Hadoop nodes, then incorporates them into your BDD cluster. For example, if you add a new YARN NodeManager, the script automatically installs Data Processing on it.

To add nodes to your cluster, run the following from the Admin Server:

```
./bdd-admin.sh add-nodes [option <arg>]
```

`add-nodes` supports the following options.

| Option | Description |
|---|---|
| `-c, --component <component(s)>` | A comma-separated list of the types of nodes to add:<br>• `dp`: Data Processing |

If no options are specified, the script adds nodes for all supported components.

For more information on adding new nodes to your cluster, see *Adding new nodes on page 62*.

### Examples

The following command adds new Data Processing nodes to your cluster:

```
./bdd-admin.sh add-nodes -c dp
```

# Diagnostics commands

You can use the `bdd-admin` script's diagnostics commands to perform such operations as checking the status of your cluster and retrieving component log files.

*get-blackbox*

*status*

*get-stats*

*reset-stats*

*get-log-levels*

*set-log-levels*

*get-logs*

*rotate-logs*

## get-blackbox

The `get-blackbox` command generates the Dgraph's on-demand tracing blackbox file and returns the name and location of the file.

**Note:** This command is intended for use by Oracle Support.

To generate the Dgraph blackbox file, run the following from the Admin Server:

```
./bdd-admin.sh get-blackbox [option <arg>]
```

`get-blackbox` supports the following options.

| Option | Description |
|---|---|
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes the script will run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script generates blackbox files for all Dgraph nodes in the cluster.

### Examples

The following command generates blackbox files for all Dgraph nodes:

```
./bdd-admin.sh get-blackbox
```

The following generates a blackbox file for the Dgraph running on the `web009.us.example.com` node:

```
./bdd-admin.sh get-blackbox -n web009.us.example.com
```

## status

The `status` command checks component statuses and the overall health of the BDD cluster.

`status` can perform two types of checks:

- Ping, which returns the status (up or down) of the specified components. This is the command's default behavior.
- Health check, which returns the overall health of the cluster and the Hive Table Detector.

To check component statuses or cluster health, run the following from the Admin Server:

```
./bdd-admin.sh status [option <arg>]
```

`status` supports the following options.

| Option | Description |
|---|---|
| `-c, --component` *`<component(s)>`* | A comma-separated list of the components to run on:<br>• `agent`: Dgraph HDFS Agent<br>• `dgraph`: Dgraph<br>• `dp`: Data Processing<br>• `gateway`: Dgraph Gateway<br>• `studio`: Studio<br>• `transform`: Transform Service |
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |
| `--health-check` | Returns the health of the cluster and the Hive Table Detector. When specified, the `-c` or `-n` options can't be included.<br>If the healthcheck fails, information on what went wrong can be found in the Studio and Data Processing logs. |

If no options are specified, the script returns the statuses of all supported components.

### Examples

The following command returns the statuses of all supported components:

```
./bdd-admin.sh status
```

The following command returns the health of the cluster and the Hive Table Detector:

```
./bdd-admin.sh status --health-check
```

The output from the above command will be similar to the following:

```
[2015/08/04 11:38:54 -0400] [Admin Server] Checking the health of BDD cluster...
[2015/08/04 11:40:06 -0400] [web009.us.example.com] Check BDD functionality......Pass!
[2015/08
/04 11:40:08 -0400] [web009.us.example.com] Check Hive Data Detector health......Hive Data Detector
has previously run.
[2015/08/04 11:40:10 -0400] [Admin Server] Successfully checked statuses.
```

## get-stats

The `get-stats` command obtains Dgraph statistics.

**Note:** Statistics are intended for use by Oracle Support only.

To obtain the Dgraph statistics, run the following from the Admin Server:

```
./bdd-admin.sh get-stats [option <arg>] <dest>
```

Where <*dest*> is the absolute path to the directory the script will output the requested statistics to. When the script completes, this location will contain a file named <*hostname*>-<*timestamp*>-`dgraph-stats.xml`.

`get-stats` supports the following options.

| Option | Description |
|---|---|
| `-c, --component`<br>*<component(s)>* | A comma-separated list of the components to run on:<br>• `dgraph`: Dgraph |
| `-n, --node`<br>*<hostname(s)>* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script obtains the statistics for all Dgraph instances in the cluster.

For more information on Dgraph statistics, see *About Dgraph statistics on page 79*.

### Examples

The following command outputs the statistics of all Dgraph instances in the cluster to the `/tmp` directory:

```
./bdd-admin.sh get-stats /tmp
```

The following command outputs the statistics of the Dgraph running on the `web009.us.example.com` node to the `/tmp` directory:

```
./bdd-admin.sh get-stats -n web009.us.example.com /tmp
```

## reset-stats

The `reset-stats` command resets the Dgraph statistics.

**Note:** Statistics are intended for use by Oracle Support only.

To reset Dgraph statistics, run the following from the Admin Server:

```
./bdd-admin.sh reset-stats [option <arg>]
```

`reset-stats` supports the following options.

| Option | Description |
|---|---|
| `-c, --component` *`<component(s)>`* | A comma-separated list of the components to run on:<br>• `dgraph`: Dgraph |
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script resets the statistics for all Dgraph instances in the cluster.

For more information on Dgraph statistics, see *About Dgraph statistics on page 79*.

### Examples

The following command resets the statistics for all Dgraph instances in the cluster:

```
./bdd-admin.sh reset-stats
```

The following command resets the statistics for the Dgraph running on the `web009.us.example.com` node:

```
./bdd-admin.sh reset-stats -n web009.us.example.com
```

## get-log-levels

The `get-log-levels` command returns the list of component logs and their current levels.

To obtain component log levels, run the following from the Admin Server:

```
./bdd-admin.sh get-log-levels [option <arg>]
```

`get-log-levels` supports the following options.

| Option | Description |
|---|---|
| `-c, --component <component(s)>` | A comma-separated list of the components to run on:<br>• `dgraph`: Dgraph<br>• `dp`: Data Processing<br>• `gateway`: Dgraph Gateway<br>The `dgraph` option returns the current levels of all Dgraph out log subsystems. For more information, see *Dgraph out log on page 161*. |
| `-n, --node <`*`hostname(s)`*`>` | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script returns the current log levels for all supported components.

If the script completes successfully, its output will be similar to the following:

```
[2015/06/01 22:36:24 -0400] [Admin Server] Retrieving log levels...
[2015/06
/01 22:36:30 -0400] [web009.us.example.com] Retrieving Dgraph Gateway log level.......Success!
   Gateway                     : WARNING
[2015/06/01 22:36:33 -0400] [web009.us.example.com] Retrieving DP log level.......Success!
   DP                          : INCIDENT_ERROR
[2015/06/01 22:36:45 -0400] [web009.us.example.com] Retrieving Dgraph log levels.......Success!
All Dgraph log subsystems:
   background_merging          : ERROR
   bulk_ingest                 : ERROR
   cluster                     : WARNING
   database                    : ERROR
   datalayer                   : ERROR
   dgraph                      : ERROR
   eql                         : ERROR
   eve                         : WARNING
   http                        : ERROR
   lexer                       : ERROR
   splitting                   : ERROR
   ssl                         : ERROR
   task_scheduler              : ERROR
   text_search_rel_rank        : ERROR
   text_search_spelling        : ERROR
   update                      : ERROR
   workload_manager            : ERROR
   ws_request                  : ERROR
   xq_web_service              : ERROR

[2015/06/01 22:36:49 -0400] [Admin Server] Successfully retrieved all log levels.
```

## Examples

The following command prints the current log levels of all supported components:

```
./bdd-admin.sh get-log-levels
```

The following command prints the current log level of the Dgraph Gateway running on the `web009.us.example.com` node:

```
./bdd-admin.sh get-log-levels -c gateway -n web009.us.example.com
```

## set-log-levels

The `set-log-levels` command sets component log levels and updates their configuration files so that the changes persist when the components are restarted.

To set component log levels, run the following from the Admin Server:

```
./bdd-admin.sh set-log-levels [option <arg>]
```

`set-log-levels` supports the following options.

| Option | Description |
|---|---|
| `-c, --component` *`<component(s)>`* | A comma-separated list of the components to run on:<br>• `dgraph`: Dgraph<br>• `dp`: Data Processing<br>• `gateway`: Dgraph Gateway |

| Option | Description |
|---|---|
| `-s, --subsystem` *`<subsystem(s)>`* | A comma-separated list of the Dgraph out log subsystems to run on:<br>• `background_merging`<br>• `bulk_ingest`<br>• `cluster`<br>• `datalayer`<br>• `dgraph` (Note that this is different from the `dgraph` component.)<br>• `eql`<br>• `eve`<br>• `http`<br>• `lexer`<br>• `splitting`<br>• `ssl`<br>• `task_scheduler`<br>• `text_search_rel_rank`<br>• `text_search_spelling`<br>• `update`<br>• `workload_manager`<br>• `ws_request`<br>• `xq_web_service`<br>This option can only be specified when running on the `dgraph` component. If the script runs on the `dgraph` component and this option isn't specified, it runs on all supported subsystems.<br>**Note:** When setting the levels of Dgraph log subsystems, the script also updates the `DGRAPH_LOG_LEVELS` property in `bdd.conf` accordingly. When setting log levels on specific nodes, it only updates `bdd.conf` on those nodes. These settings will be overwritten if the `update-config` command is run.<br>For more information on the Dgraph out log and its subsystems, see *Dgraph out log on page 161*. |

| Option | Description |
|---|---|
| `-l, --level <level>` | The log level to set for the components:<br>• `INCIDENT_ERROR`<br>• `ERROR`<br>• `WARNING`<br>• `NOTIFICATION`<br>• `TRACE`<br>Only one log level can be specified. If this option is omitted, the script sets all specified logs to `NOTIFICATION`. |
| `--non-persistent` | Indicates that the log levels should be reset when the components are restarted. When specified, the script doesn't update the component configuration files.<br>This option is only available for the `dgraph` and `gateway` components. Data Processing log levels are always persistent. |
| `-n, --node <hostname(s)>` | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script sets the log levels of all supported components and Dgraph log subsystems to `NOTIFICATION`. These settings will persist if the components are restarted.

### Examples

The following command sets the log levels of Data Processing and the Dgraph log subsystems `cluster` and `datalayer` to `WARNING`:

```
./bdd-admin.sh set-log-levels -c dgraph,dp -s cluster,datalayer -l WARNING
```

The following command sets the log levels of the Dgraph Gateway and all Dgraph subsystems to `ERROR`, which will not be persistent:

```
./bdd-admin.sh set-log-levels -c dgraph,gateway -l ERROR --non-persistent
```

## get-logs

The `get-logs` command collects requested log files and compresses them to a single zip file.

To obtain components logs, run the following from the Admin Server:

```
./bdd-admin.sh get-logs [option <arg>] <file>
```

Where `<file>` defines the absolute path to the output zip file. This file must not exist and must include the `.zip` file extension.

`get-logs` supports the following options.

| Option | Description |
|---|---|
| `-t, --time <hours>` | When specified, the script returns the logs that were modified within the last *<hours>* hours.<br>If this option is omitted, the script returns the most recently updated log file for each component. |
| `-c, --component <component(s)>` | A comma-separated list of the component logs to collect:<br>• `agent`: Dgraph HDFS Agent logs<br>• `all`: All component logs<br>• `dgraph`: Dgraph logs (includes the FUSE log, if FUSE is enabled)<br>• `dg-on-crash`: Dgraph on-crash tracing logs<br>• `dg-on-demand`: Dgraph on-demand tracing logs<br>• `dp`: Data Processing logs<br>• `gateway`: Dgraph Gateway logs<br>• `spark`: Spark logs<br>• `studio`: Studio logs<br>• `transform`: Transform Service<br>• `weblogic`: WebLogic Server logs<br>• `zk-log`: ZooKeeper logs<br>• `zk-transaction`: ZooKeeper transaction logs<br>Note the following:<br>• The `spark`, `zk-log`, and `zk-transaction` components will prompt for the username and password for Cloudera Manager/Ambari if the `BDD_HADOOP_UI_USERNAME` and `BDD_HADOOP_UI_PASSWORD` environment variables aren't set.<br>• The `dg-on-demand` log is only generated when the `get-blackbox` command is run. This means that if the `-t` option is specified, `get-logs` only returns the `dg-on-demand` log if `get-blackbox` was run during the specified time frame. And if the `-t` option is omitted, `get-logs` won't return the `dg-on-demand` log if `get-blackbox` has never been run. |
| `-n, --node <hostname(s)>` | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script obtains the most recently updated logs for all components except `dg-on-crash`, `dg-on-demand`, and `zk-transaction`.

### Examples

The following command obtains the most recently modified logs for all supported components and outputs them to `/localdisk/logs/all_logs.zip`:

```
./bdd-admin.sh get-logs -c all /localdisk/logs/all_logs.zip
```

The following command obtains all `zk-log` and `zk-transaction` logs modified within the last 24 hours and outputs them to `/localdisk/logs/zk_logs.zip`:

```
./bdd-admin.sh get-logs -t 24 -c zk-log,zk-transaction /localdisk/logs/zk_logs.zip
```

## rotate-logs

The `rotate-logs` command rotates component logs.

**Note:** This command is intended for use by Oracle Support only.

To rotate component logs, run the following from the Admin Server:

```
./bdd-admin.sh rotate-logs [option <arg>]
```

`rotate-logs` supports the following options.

| Option | Description |
|---|---|
| `-c, --component` *`<component(s)>`* | A comma-separated list of the component logs to rotate:<br>• `agent`: Dgraph HDFS Agent logs<br>• `dgraph`: Dgraph logs (includes the FUSE log, if FUSE is enabled)<br>• `gateway`: Dgraph Gateway logs<br>• `studio`: Studio logs<br>• `transform`: Transform Service<br>• `weblogic`: WebLogic Server logs |
| `-n, --node` *`<hostname(s)>`* | A comma-separated list of the nodes to run on. Each must be defined in `bdd.conf`. |

If no options are specified, the script rotates all supported component logs.

### Examples

The following command rotates all supported component logs:

```
./bdd-admin.sh rotate-logs
```

The following command rotates the logs of the Dgraph and Dgraph HDFS Agent running on the `web009.us.example.com` node:

```
./bdd-admin.sh rotate-logs -c dgraph,agent -n web009.us.example.com
```

Chapter 4

# Administering a Big Data Discovery Cluster

This section describes how to perform different administrative tasks for your BDD cluster deployment as a whole, such as backing it up and updating its configuration.

## Updating the cluster configuration

You can update BDD's configuration by editing `bdd.conf` then running the `bdd-admin` script to distribute your changes to the rest of the cluster.

You can edit `bdd.conf` in any text editor. Note that you can't modify all of the properties in the file; for example, you can't change the lists of Dgraph and Managed Server nodes. For the full list of properties you can change, see *Configuration properties that can be modified on page 51*.

**Note:** When you update `bdd.conf`, any component log levels you've set on specific nodes using the `set-log-levels` command will be overwritten by the `DGRAPH_LOG_LEVELS` property in the updated file.

When the script runs, it backs up the original version of `bdd.conf` to `bdd.conf.bak<`*`num`*`>` so you can revert your changes, if necessary. It then copies the updated file to all BDD nodes.

To update your cluster configuration:

1. On the Admin Server, copy `bdd.conf` in `$BDD_HOME/BDD_manager/conf` to a different directory.
2. Open the copy in a text editor and make your desired changes.

   Be sure to save the file before closing.
3. Go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh publish-config bdd <path>
   ```

   Where `<`*`path`*`>` is the absolute path to the modified copy of `bdd.conf`.
4. Restart your cluster so the changes take effect:

```
./bdd-admin.sh restart
```

The above command will shut the cluster down gracefully, which may take a long time. You can optionally specify `-t <minutes>` to force a shutdown sooner.

*Configuration properties that can be modified*

## Configuration properties that can be modified

The table below describes the properties in `bdd.conf` that you can modify. Be sure to read this information carefully before making changes to `bdd.conf`. Don't update any other properties in this file, as this could have negative effects on your cluster.

| Property | Description |
|---|---|
| `DGRAPH_INDEX_DIR` | The path to the Dgraph databases directory. You must prepare the database files in the new location before changing the value of this property. |
| `JAVA_HOME` | The JDK used when starting the BDD components. If you change this value, you must also update the location used by the CLI and Studio. Note that this must be in the same location on all nodes in the cluster. |
| `DGRAPH_THREADS` | The number of threads the Dgraph starts with. Oracle recommends the following:<br>• For machines running only the Dgraph, the number of threads should be equal to the number of CPU cores on the machine.<br>• For machines running the Dgraph and other BDD components, the number of threads should be the number of CPU cores minus 2. For example, a machine with 4 cores should have 2 threads.<br>Be sure that the number you use is in compliance with the licensing agreement. |
| `DGRAPH_CACHE` | The Dgraph cache size, in MB. There is no default value for this property, so you must provide one.<br>For enhanced performance, Oracle recommends allocating at least 50% of the node's available RAM to the Dgraph cache. If you later find that queries are getting cancelled because there is not enough available memory to process them, you should increase this amount. |
| `DGRAPH_OUT_FILE` | The path to the Dgraph's stdout/stderr file. |

| Property | Description |
|---|---|
| DGRAPH_LOG_LEVEL | Optional. Defines the log levels for the Dgraph's out log subsystems. This must be formatted as:<br>`"subsystem1 level1\|subsystem2,subsystem3 level2\|subsystemN levelN"`<br>Be sure to include the quotes. For example:<br>`DGRAPH_LOG_LEVEL = "bulk_ingest WARNING\|cluster ERROR\|dgraph, eql, eve INCIDENT_ERROR"`<br>You can include as many subsystems as you want. Any you don't include will be set to NOTIFICATION. If you enter an unsupported or improperly formatted value, it will default to NOTIFICATION.<br>For more information on the Dgraph's out log subsystems and their supported levels, see *Dgraph out log on page 161*. |
| DGRAPH_ADDITIONAL_ARG | **Note:** This property is only intended for use by Oracle Support.<br>Defines one or more flags to start the Dgraph with. Each flag must be quoted.<br>Note that you cannot include flags that map to properties in bdd.conf. For more information on Dgraph flags, see *Dgraph flags on page 79*. |
| AGENT_OUT_FILE | The path to the HDFS Agent's stdout/stderr file. |

# Backing up Big Data Discovery

Because Big Data Discovery doesn't perform automatic backups, you must back up your system manually. Oracle recommends that, at a minimum, you back up your cluster immediately after deployment.

You back up your cluster by running the bdd-admin script with the backup command. This backs up the following data to a single TAR file, which you can later use to restore your cluster:

- Studio database
- Schema and data for Hive tables created in Studio
- Dgraph databases
- Sample files in HDFS
- Cluster configuration files

**Note:** The script doesn't back up transient data, like state in Studio. This information won't available if you restore your cluster.

Before you back up your cluster, verify that:

- The BDD_STUDIO_JDBC_USERNAME and BDD_STUDIO_JDBC_PASSWORD environment variables are set. Otherwise, the script will prompt you for this information at runtime.
- The database client is installed on the Admin Server. For MySQL databases, this should be MySQL client. For Oracle databases, this should be Oracle Database Client, installed with a type of Administrator. Note that the Instant Client isn't supported.
- If you have an Oracle database, the ORACLE_HOME environment variable is set to the directory one level above the /bin directory that the sqlplus executable is located in. For example, if the sqlplus executable is located in /u01/app/oracle/product/11/2/0/dbhome/bin, ORACLE_HOME should be set to /u01/app/oracle/product/11/2/0/dbhome/bin.
- The temporary directories used during the backup operation contain enough free space. For more information, see *Space requirements on page 31*.

**Note:** Backups aren't supported for Hypersonic databases. You must have an Oracle or MySQL database.

For more information on backup and its supported options, see *backup on page 30*. For instructions on restoring your cluster, see *Restoring Big Data Discovery on page 53*.

To back up BDD:

1. On the Admin Server, go to $BDD_HOME/BDD_manager/bin.
2. Run one of the following commands:
   - If your cluster is running:
     ```
     ./bdd-admin.sh backup -v <file>
     ```
   - If your cluster is down:
     ```
     ./bdd-admin.sh backup -o -v <file>
     ```

   Where <*file*> is the absolute path to the TAR file the script will back up your cluster to. This file must not exist and its parent directory must be writable.

   The -v flag enables debugging messages. This is optional but recommended because the script might take a long time to finish and the output will keep you informed of its current status.
3. If you haven't set the STUDIO_JDBC_USERNAME and STUDIO_JDBC_PASSWORD environment variables, enter the database username and password when prompted.

# Restoring Big Data Discovery

You can restore your cluster from a backup TAR file by running the bdd-admin script with the restore command.

Before restoring your cluster, you should verify that:

- You have access to a backup TAR file created by the backup command.
- Your current cluster and the backup cluster both have the same major version of BDD.
- Both clusters have the same type of database, Oracle or MySQL. Note that restore doesn't support Hypersonic databases.

- The BDD_STUDIO_JDBC_USERNAME and BDD_STUDIO_JDBC_PASSWORD environment variables are set. Otherwise, the script will prompt you for this information at runtime.
- The database client is installed on the Admin Server. For MySQL databases, this should be MySQL client. For Oracle databases, this should be Oracle Database Client, installed with a type of Administrator. Note that the Instant Client isn't supported.
- If you have an Oracle database, the ORACLE_HOME environment variable is set to the directory one level above the /bin directory that the sqlplus executable is located in. For example, if the sqlplus executable is located in /u01/app/oracle/product/11/2/0/dbhome/bin, ORACLE_HOME should be set to /u01/app/oracle/product/11/2/0/dbhome/bin.
- The temporary directories used during the restore operation contain enough free space. For more information, see *Space requirements on page 33*.

Your current cluster can have a different topology than the backup cluster. For example, node IP addresses, the total number of nodes, and the locations of the BDD components can be different between the two.

When the script runs, it restores the Studio database, Hive tables created in Studio, Dgraph databases, and sample files from backup.

Note that the script doesn't completely restore the configuration files from backup—it merges them with the current cluster's configuration files. The restored cluster will contain some of the backup cluster's configuration, but most of it will be from the current cluster.

For more information on the restore command, see *restore on page 32*.

**Important:** The script will overwrite the data on your current cluster with the backed up data and won't roll the restoration back if it fails. Because of this, if your current cluster contains any important data, you should back it up before restoring.

To restore your cluster:

1. On the Admin Server, go to $BDD_HOME/BDD_manager/bin.
2. Stop your cluster if it's running:

   ```
   ./bdd-admin.sh stop
   ```

   The above command will shut the cluster down gracefully, which may take a long time. You can optionally specify -t <*minutes*> to force a shut down sooner.
3. Run the restore command:

   ```
   ./bdd-admin.sh restore <file>
   ```

   Where <*file*> is the absolute path to the backup TAR file you want to restore from.
4. If you haven't set the STUDIO_JDBC_USERNAME and STUDIO_JDBC_PASSWORD environment variables, enter the database username and password when prompted.
5. When the script finishes running, restart your cluster so the changes take effect:

   ```
   ./bdd-admin.sh restart
   ```

   The above command will shut the cluster down gracefully, which may take a long time. You can optionally specify -t <*minutes*> to force a shut down sooner.

When the script runs, it makes a copy of the current Dgraph databases directory in `DGRAPH_INDEX_DIR/.snapshot/old_copy`. You should delete this if you decide to keep the restored version of the Dgraph databases.

# Updating BDD's Hadoop configuration

You can update your BDD cluster's Hadoop configuration with the `bdd-admin` script.

*Updating the Hadoop client configuration files*

*Setting the Hue URI*

*Switching Hadoop versions*

## Updating the Hadoop client configuration files

If you update your Hadoop client configuration files, you can publish your changes to BDD with the `bdd-admin` script. This distributes the Hadoop client configuration files to all BDD nodes and updates the relevant properties in BDD's configuration files.

When the script runs, it obtains the Hadoop client configuration files from Cloudera Manager/Ambari, then updates the following:

- All Hadoop properties in `bdd.conf`
- The following properties in Studio's `portal-ext.properties` file:
  - `dp.settings.hadoop.cluster.host`
  - `dp.settings.hive.metastore.port`
  - `dp.settings.namenode.port`
  - `dp.settings.hive.jdbc.port`
  - `dp.settings.hue.http.port`
- The following properties in Data Processing's `edp.properties`:
  - `hiveServerHost`
  - `hiveServerPort`

When the script finishes running, you must restart your cluster for the changes to take effect.

To update your cluster's Hadoop client configuration files:

1. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh publish-config hadoop
   ```

2. Restart your cluster so the changes take effect:

   ```
   ./bdd-admin.sh restart
   ```

   The above command will shut the cluster down gracefully, which may take a long time. You can optionally specify `-t <`*`minutes`*`>` to force a shutdown sooner.

## Setting the Hue URI

If you have an HDP cluster, you can use the `bdd-admin` script to update the URI of the node running Hue in `bdd.conf`.

When the script runs, it sets the `HUE_URI` property in `bdd.conf` to the hostname and port you specify. It also updates your cluster's Hadoop configuration files and performs the steps described in *Updating the Hadoop client configuration files on page 55*.

After the script finishes, you must restart your cluster for the changes to take effect.

To update the Hue URI:

1. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh publish-config hadoop --hueuri <hostname>:<port>
   ```

   Where *`<hostname>`* and *`<port>`* are the fully qualified domain name and port number of the node running Hue.

2. Restart your cluster so the changes take effect:

   ```
   ./bdd-admin.sh restart
   ```

   The above command will shut the cluster down gracefully, which may take a long time. You can optionally specify `-t` *`<minutes>`* to force a shutdown sooner.

## Switching Hadoop versions

If you want to upgrade to a new version of your Hadoop distribution, you need to update your BDD cluster to integrate with it. You can do this using the `bdd-admin` script.

Before you run the script, you must obtain the new Hadoop client libraries for your distribution and move them to the Admin Server. When the script runs, it uses these libraries to generate a new fat jar, which it then distributes to all BDD nodes.

The script also obtains and distributes the new Hadoop client configuration files as described in *Updating the Hadoop client configuration files on page 55*.

**Note:** You can't use `bdd-admin` to switch to a different Hadoop distribution. For example, you could upgrade from CDH 5.4 to CDH 5.5, but not to HDP 2.3.

To switch to a different Hadoop version:

1. Stop your BDD cluster by running the following from `$BDD_HOME/BDD_manager/bin` on the Admin Server:

   ```
   ./bdd-admin.sh stop [-t <minutes>]
   ```

2. Upgrade your Hadoop cluster according to the instructions in your distribution's documentation.

3. Verify that any configuration changes you made prior to installing BDD (for example, to your YARN settings) weren't reset during the upgrade.

   Additionally, if you have HDP:

   (a) In `mapred-site.xml`, replace all instances of `${hdp.version}` with your HDP version number.

(b) In `hive-site.xml`, remove s from the values of the following properties:

   - `hive.metastore.client.connect.retry.dealay`
   - `hive.metastore.client.cocket.timeout`

4. Obtain the client libraries for the new version of your Hadoop distribution and put them on the Admin Server.

   The location you put them in is arbitrary, as you will provide the `bdd-admin` script with their paths at runtime.

   - If you have a CDH cluster, download the following packages from *http://archive-primary.cloudera.com/cdh5/cdh/5/* and unzip them:
     - `spark-<`*`spark_version`*`>.cdh.<`*`cdh_version`*`>.tar.gz`
     - `hive-<`*`hive_version`*`>.cdh.<`*`cdh_version`*`>.tar.gz`
     - `hadoop-<`*`hadoop_version`*`>.cdh.<`*`cdh_version`*`>.tar.gz`
     - `avro-<`*`avro_version`*`>.cdh.<`*`cdh_version`*`>.tar.gz`
   - If you have an HDP cluster, copy the following directories from your Hadoop nodes to the Admin Server:
     - `/usr/hdp/<`*`version`*`>/pig/lib/h2/`
     - `/usr/hdp/<`*`version`*`>/hive/lib/`
     - `/usr/hdp/<`*`version`*`>/spark/lib/`
     - `/usr/hdp/<`*`version`*`>/spark/external/spark-native-yarn/lib/`
     - `/usr/hdp/<`*`version`*`>/hadoop/`
     - `/usr/hdp/<`*`version`*`>/hadoop/lib/`
     - `/usr/hdp/<`*`version`*`>/hadoop-hdfs/`
     - `/usr/hdp/<`*`version`*`>/hadoop-hdfs/lib/`
     - `/usr/hdp/<`*`version`*`>/hadoop-yarn/`
     - `/usr/hdp/<`*`version`*`>/hadoop-yarn/lib/`
     - `/usr/hdp/<`*`version`*`>/hadoop-mapreduce/`
     - `/usr/hdp/<`*`version`*`>/hadoop-mapreduce/lib/`

5. Start your BDD cluster:

   ```
   ./bdd-admin.sh start
   ```

6. Run the following up update BDD's Hadoop configuration:

   ```
   ./bdd-admin.sh publish-config hadoop -l <path[,path]> -j <file>
   ```

   `<`*`path[,path]`*`>` is a comma-separated list of the absolute paths to each of the client libraries on the Admin Server. For HDP clusters, the libraries *must* be specified in the order they are listed in above.

   `<`*`file`*`>` is the absolute path to the Spark on YARN jar on your Hadoop nodes.

7. Restart your cluster so the changes take effect:

```
./bdd-admin.sh restart
```

The above command will shut the cluster down gracefully, which may take a long time. You can optionally specify `-t` *`<minutes>`* to force a shutdown sooner.

# Updating BDD's Kerberos configuration

You can update your BDD cluster's Kerberos configuration with the `bdd-admin` script.

*Enabling Kerberos*

*Changing the location of the Kerberos krb5.conf file*

*Updating the Kerberos keytab file*

*Updating the Kerberos principal*

## Enabling Kerberos

BDD supports Kerberos 5+ to authenticate its communications with Hadoop. You can enable this for BDD to improve the security of your cluster and data.

Before you can configure Kerberos for BDD, you must install it on your Hadoop cluster. If your Hadoop cluster already uses Kerberos, you must enable it for BDD so it can access the Hive tables it requires.

To enable Kerberos:

1. Install the `kinit` and `kdestroy` utilities on all BDD nodes.
2. Create the following directories in HDFS:
   - `/user/`*`<bdd>`*, where *`<bdd>`* is the name of the `bdd` user.
   - `/user/`*`<HDFS_DP_USER_DIR>`*, where *`<HDFS_DP_USER_DIR>`* is the value of `HDFS_DP_USER_DIR` defined in `bdd.conf`.

   The owner of both directories must be the `bdd` user, and their group must be `supergroup`.
3. Add the `bdd` user to the `hdfs` and `hive` groups on all BDD nodes.
4. If you use HDP, add the group that the `bdd` user belongs to to the `hadoop.proxyuser.hive.groups` property in `core-site.xml`.

   You can do this in Ambari.
5. Create a principal for BDD.

   The primary component must be the name of the `bdd` user and the realm must be your default realm.
6. Generate a keytab file for the BDD principal and move it to the Admin Server.

   The name and location of this file are arbitrary as you will pass this information to the `bdd-admin` script at runtime.
7. Copy your `krb5.conf` file to the same location on all BDD nodes.

   The location is arbitrary, but the default is `/etc`.

8. If your Dgraph databases are stored on HDFS, you must also enable Kerberos for the Dgraph. On the Admin Server, make a copy of bdd.conf and edit the following properties in the copy:

| Property | Description |
|---|---|
| KERBEROS_TICKET_REFRESH_INTERVAL | The interval (in minutes) at which the Dgraph's Kerberos ticket is refreshed. For example, if set to 60, it would be refreshed ever 60 minutes, or every hour. |
| KERBEROS_TICKET_LIFETIME | The amount of time that the Dgraph's Kerberos ticket is valid. This should be given as a number followed by a supported unit of time: s, m, h, or d. For example, 10h (10 hours), or 10m (10 minutes). |

   Then go to $BDD_HOME/BDD_manager/bin and run:

```
./bdd-admin.sh publish-config <path>
```

   Where <*path*> is the absolute path to the modified version copy of bdd.conf.

9. Go to $BDD_HOME/BDD_manager/bin and run:

```
./bdd-admin.sh publish-config kerberos on -k <krb5> -t <keytab> -p <principal>
```

   Where:

   - <*krb5*> is the absolute path to krb5.conf on all BDD nodes
   - <*keytab*> is the absolute path to the BDD keytab file on the Admin Server
   - <*principal*> is the BDD principal

   The script updates BDD's configuration files with the name of the principal and the location of the krb5.conf file. It also renames the keytab file to bdd.keytab and distributes it to $BDD_HOME/common/kerberos on all BDD nodes.

10. If you use HDP, publish the change you made to core-site.xml:

```
./bdd-admin.sh publish-config hadoop
```

11. Restart your cluster for the changes to take effect:

```
./bdd-admin.sh restart [-t <minutes>]
```

12. To enable Kerberos for the Transform Service:

   (a) Copy k5start from $BDD_HOME/dgraph/bin/ on one of your Dgraph nodes to $BDD_HOME/transformservice/ on all of your Transform Service nodes.

   (b) On each Transform Service node, start k5start by running the following command from $BDD_HOME/transformservice/:

```
./k5start -f $KERBEROS_KEYTAB_PATH -K <ticket_refresh>
-l <ticket_lifetime> $KERBEROS_PRINCIPAL -b > <logfile> 2>&1
```

   Where:

   - $KERBEROS_KEYTAB_PATH and $KERBEROS_PRINCIPAL are the values of those properties defined in bdd.conf.

- `<ticket_refresh>` is the rate at which the Transform Service's Kerberos ticket is refreshed, in minutes. For example, a value of 60 would set its ticket to be refreshed every 60 minutes, or every hour. You can optionally use the value for `KERBEROS_TICKET_REFRESH_INTERVAL` in `bdd.conf`.
- `<ticket_lifetime>` is the amount of time the Transform Service's Kerberos ticket is valid for. This should be given as a number followed by a supported unit of time: `s`, `m`, `h`, or `d`. For example, `10h` (10 hours) or `10m` (10 minutes). You can optionally use the value for `KERBEROS_TICKET_LIFETIME` in `bdd.conf`.
- `<logfile>` is the absolute path to the log file you want `k5start` to write to.

(c) Optionally, configure `k5start` to run as a service on all Transform Service nodes.

This will enable it to start automatically after a node reboot. Otherwise, you'll have to rerun the above command each time a Transform Service node is rebooted.

Once Kerberos is enabled, you can use the `bdd-admin` script to update its configuration as needed. For more information, see *kerberos on page 37*.

## Changing the location of the Kerberos krb5.conf file

If you want to change the location of the `krb5.conf` file, you can use the `bdd-admin` script to update BDD's configuration accordingly.

You must provide the script with the absolute path to the `krb5.conf` file on all BDD nodes. When it runs, it updates the location of `krb5.conf` in BDD's configuration files.

For more information on updating your Kerberos configuration with `bdd-admin`, see *kerberos on page 37*.

To change the location of the `krb5.conf` file:

1. On all BDD nodes, move the `krb5.conf` file to the new location.

   The location is arbitrary, but must be the same on all nodes.

2. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh kerberos -k <file>
   ```

   Where `<file>` is the new absolute path to `krb5.conf`.

3. Restart your cluster so the changes take effect:

   ```
   ./bdd-admin.sh restart [-t <minutes>]
   ```

## Updating the Kerberos keytab file

If you update BDD's current keytab file or create a new one, you can use the `bdd-admin` script to publish the new or updated file to the rest of the cluster.

When you run the script, you must provide it with the absolute path to the new or modified file. The script renames the specified file to `bdd.keytab` (if necessary) and copies it to `$BDD_HOME/common/kerberos` on all nodes.

For more information on updating your Kerberos configuration with the `bdd-admin` script, see *kerberos on page 37*.

To update the keytab file:

1. On the Admin Server, edit the current BDD keytab file or create a new one.

   The current file is named `bdd.keytab` and located in `$BDD_HOME/common/kerberos`.

2. Go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh publish-config kerberos -t <file>
   ```

   Where *`<path>`* is the absolute path to the new or modified keytab file.

3. Restart your cluster so the changes take effect:

   ```
   ./bdd-admin.sh restart [-t <minutes>]
   ```

4. On each Transform Service node, restart `k5start` with the new keytab file by running the following command from `$BDD_HOME/transformservice/`:

   ```
   ./k5start -f $KERBEROS_KEYTAB_PATH -K <ticket_refresh>
   -l <ticket_lifetime> $KERBEROS_PRINCIPAL -b > <logfile> 2>&1
   ```

   Where:

   - `$KERBEROS_KEYTAB_PATH` and `$KERBEROS_PRINCIPAL` are the values of those properties defined in `bdd.conf`. Be sure to use the path to the new keytab file.
   - `<ticket_refresh>` is the rate at which the Transform Service's Kerberos ticket is refreshed, in minutes. For example, a value of 60 would set its ticket to be refreshed every 60 minutes, or every hour. You can optionally use the value for `KERBEROS_TICKET_REFRESH_INTERVAL` in `bdd.conf`.
   - `<ticket_lifetime>` is the amount of time the Transform Service's Kerberos ticket is valid for. This should be given as a number followed by a supported unit of time: `s`, `m`, `h`, or `d`. For example, `10h` (10 hours) or `10m` (10 minutes). You can optionally use the value for `KERBEROS_TICKET_LIFETIME` in `bdd.conf`.
   - `<logfile>` is the absolute path to the log file you want `k5start` to write to.

## Updating the Kerberos principal

If you edit the BDD principal or create a new one, you can use the `bdd-admin` script to publish your changes to the rest of the cluster.

When the script runs, it updates the following properties with the new or modified principal:

- `KERBEROS_PRINCIPAL` in `bdd.conf`
- `krb5.principal` in Studio's `portal-ext.properties` file
- `localKerberosPrincipal` and `clusterKererosPrincipal` in the `data_processing_CLI` file

**Note:** You can't change the primary component of the principal.

For more information on updating your Kerberos configuration with the `bdd-admin` script, see *kerberos on page 37*.

To update the Kerberos principal:

1. On the Admin Server, edit the current BDD principal or create a new one.

   Be sure to keep the primary component of the principal the same as the original.

2. Go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh publish-config kerberos -p <principal>
   ```

   Where `<principal>` is the name of the new or modified principal.

3. Restart your cluster so the changes take effect:

   ```
   ./bdd-admin.sh restart [-t <minutes>]
   ```

4. On each Transform Service node, restart `k5start` with the new principal by running the following command from `$BDD_HOME/transformservice/`:

   ```
   ./k5start -f $KERBEROS_KEYTAB_PATH -K <ticket_refresh>
   -l <ticket_lifetime> $KERBEROS_PRINCIPAL -b > <logfile> 2>&1
   ```

   Where:

   - `$KERBEROS_KEYTAB_PATH` and `$KERBEROS_PRINCIPAL` are the values of those properties defined in `bdd.conf`. Be sure to use the name of the new principal.
   - `<ticket_refresh>` is the rate at which the Transform Service's Kerberos ticket is refreshed, in minutes. For example, a value of 60 would set its ticket to be refreshed every 60 minutes, or every hour. You can optionally use the value for `KERBEROS_TICKET_REFRESH_INTERVAL` in `bdd.conf`.
   - `<ticket_lifetime>` is the amount of time the Transform Service's Kerberos ticket is valid for. This should be given as a number followed by a supported unit of time: `s`, `m`, `h`, or `d`. For example, `10h` (10 hours) or `10m` (10 minutes). You can optionally use the value for `KERBEROS_TICKET_LIFETIME` in `bdd.conf`.
   - `<logfile>` is the absolute path to the log file you want `k5start` to write to.

# Adding new nodes

The following sections describe how to add new nodes to your BDD cluster.

*Adding new Dgraph nodes*

*Adding new Data Processing nodes*

## Adding new Dgraph nodes

You can add new Dgraph nodes to BDD to expand your Dgraph cluster.

**Note:** You can also add new Data Processing nodes; for more information, see *Adding new Data Processing nodes on page 65*. You can't add more WebLogic Server nodes without reinstalling.

To add a new Dgraph node:

1. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and stop BDD:

```
./bdd-admin.sh stop [-t <minutes>]
```

2. Select a node in your cluster to move the Dgraph to.

   If your databases are on HDFS, this must be an HDFS DataNode.

3. If BDD is currently installed on the selected node, verify that the following directories are present and copy over any that are missing:

   - `$BDD_HOME/common/edp`
   - `$BDD_HOME/dataprocessing`
   - `$BDD_HOME/dgraph`
   - `$BDD_HOME/logs/edp`

   If BDD isn't installed on the selected node:

   (a) Create a new `$BDD_HOME` directory on the node.

   (b) Set the permissions of `$BDD_HOME` to 755 and the owner to the `bdd` user.

   (c) Copy the following directories from an existing Dgraph node to the new one:

   - `$BDD_HOME/BDD_manager`
   - `$BDD_HOME/common`
   - `$BDD_HOME/dataprocessing`
   - `$BDD_HOME/dgraph`
   - `$BDD_HOME/logs`
   - `$BDD_HOME/uninstall`
   - `$BDD_HOME/version.txt`

   (d) Create a symlink `$ORACLE_HOME/BDD` pointing to `$BDD_HOME`.

   (e) Optionally, remove the `/dgraph` directory from the old Dgraph node, as it's no longer needed.

   Leave the other directories, as they may still be useful.

4. If your databases are on HDFS, install either the HDFS NFS Gateway service or FUSE on the new node.

   The option you should use depends on your Hadoop cluster. You *must* use the NFS Gateway if you have CDH 5.7.1 or HDFS data at rest encryption enabled. In all other cases, you can use either option. More information about each is available in the *Installation Guide*.

   To use the NFS Gateway, install it on the new Dgraph node. For instructions, refer to the documentation for your Hadoop distribution.

   To use FUSE:

   (a) Download FUSE 2.8+ from *https://github.com/libfuse/libfuse/releases*.

   (b) Extract `fuse-<version>.tar.gz`.

   (c) Install FUSE by going to `/fuse-<version>` and running:

```
./configure
make -j8
make install
```

(d) Set the following permissions:

- Add the bdd user to the fuse group.
- Give the bdd user read and execute permissions for fusermount.
- Give the bdd user read and write permissions for /dev/fuse.

5. If you have to host the Dgraph on the same node as Spark (or any other memory-intensive process), set up cgroups so that the Dgraph will have access to the resources it requires.

   For instructions, see *Setting up cgroups on page 76*

6. Clean up the ZooKeeper index.
7. On the Admin Server, copy bdd.conf to a new location. Open the *copy* in a text editor and update the following properties:

| **Property** | **Description** |
|---|---|
| DGRAPH_ SERVERS | The hostnames of all Dgraph servers. Add the new node to this list. Be sure to use its FQDN. |
| DGRAPH_ THREADS | The number of threads the Dgraph starts with. Verify that this setting is still accurate. It should be the number of CPU cores on the Dgraph nodes minus the number required to run HDFS and any other Hadoop services. |
| DGRAPH_CACHE | The size of the Dgraph cache. Verify that this setting is still accurate. It should either be 50% of the machine's RAM or the total amount of free memory, whichever is larger. |
| DGRAPH_ENABLE _ CGROUP | Enables cgroups for the Dgraph. This must be set to TRUE if you created a Dgraph cgroup. You must also set DGRAPH_CGROUP_NAME. |
| DGRAPH_CGROUP _ NAME | The name of the cgroup that controls the Dgraph. This is required if DGRAPH_ENABLE_CGROUP is set to TRUE. |
| NFS_GATEWAY_ SERVERS | The hostnames of all NFS Gateway nodes. If you installed the NFS Gateway service on the new node, add its FQDN to this list. |

8. To populate your configuration changes to the rest of the cluster, go to $BDD_HOME/BDD_manager/bin and run:

```
./bdd-admin.sh publish-config <path>
```

   Where <*path*> is the absolute path to the updated copy of bdd.conf.

9. Start your cluster:

```
./bdd-admin.sh start
```

## Adding new Data Processing nodes

You can add new Data Processing nodes to your BDD cluster to increase your processing power.

**Note:** You can also add more Dgraph nodes; for more information, see *Adding new Dgraph nodes on page 62*. You can't add more WebLogic Server nodes without reinstalling.

To do this, you add one or more qualified YARN NodeManager nodes to your Hadoop cluster, then run the `bdd-admin` script with the `add-nodes` command. The script queries your Hadoop manager (Cloudera Manager or Ambari) for the newly-added nodes and automatically installs Data Processing on them. When the script completes, the new nodes are up and ready to accept new jobs.

**Note:** The `bdd-admin` script requires the username and password for the Hadoop manager to query it. It will prompt you for this information if the `BDD_HADOOP_UI_USERNAME` and `BDD_HADOOP_UI_PASSWORD` environment variables aren't set.

To add a new Data Processing node:

1. Add one or more YARN NodeManager nodes to your Hadoop cluster. To support Data Processing, the following Hadoop components must be installed on each:
   - Spark on YARN
   - YARN
   - HDFS

   For instructions on adding new YARN NodeManager nodes, refer to the documentation for your Hadoop distribution.
2. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and run:
   ```
   ./bdd-admin.sh add-nodes -c dp
   ```
3. Enter the username and password for your Hadoop manager, if prompted.

# Refreshing TLS/SSL certificates

If you have TLS/SSL enabled for BDD, you can use the `bdd-admin` script to refresh your certificates, when needed.

For more information on refreshing your TLS/SSL certificates with `bdd-admin`, see *cert on page 38*.

Before beginning this procedure, verify that the password for `$JAVA_HOME/jre/lib/security/cacerts` is set to `changeit`.

To refresh your TLS/SSL certificates:

1. Export the public key certificates from all Hadoop nodes running TLS/SSL- secured HDFS, YARN, Hive, and/or KMS.

   You can do this with the following command:
   ```
   keytool -exportcert -alias <alias> -keystore <keystore_filename> -file <export_filename>
   ```
   Where:
   - `<alias>` is the certificate's alias.

   - *<keystore_filename>* is the absolute path to your keystore file. You can find this in Cloudera Manager.
   - *<export_filename>* is the name of the file to export the keystore to.

2. Copy all of the exported certificates to a single directory on the Admin Server.
3. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and run:

   ```
   ./bdd-admin.sh publish-config cert <path>
   ```

   Where *<path>* is the absolute path to the location of the updated certificates.

When the script runs, it imports the certificates to the custom truststore file, then copies the truststore to `$BDD_HOME/common/security/cacerts` on all BDD nodes.

Chapter 5

# The Dgraph

This section describes the Dgraph, its administrative operations, and flags. It also describes various Dgraph characteristics and behavior, such as memory consumption, Dgraph cache, and managing the Dgraph core dump files.

## About the Dgraph

The Dgraph is a component of Big Data Discovery that runs search analytical processing of the data sets. It handles query 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. When source data is loaded into Big Data Discovery, the Dgraph creates a separate Dgraph database for each of the data sets. When the Dgraph receives a client request through Studio, the Dgraph queries the appropriate database and returns the results.

An Oracle Big Data Discovery cluster has one or more Dgraph processes that handle end-user query requests accessing the Dgraph databases on shared storage. One of the Dgraphs in a Big Data Discovery cluster is the leader for a particular database and therefore is responsible for handling all write operations (updates, configuration changes) for that database, while the remaining Dgraphs may serve as read-only followers.

### About Dgraph databases

When a data set is created (either from Studio or via the DP CLI), the Dgraph creates a database for it. (A Dgraph database is known also as an ***index***.) The Dgraph database is named:

```
<dataset>_indexes
```

where *dataset* is the name of the data set and "_indexes" is appended to the data set name. For example:

```
edp_cli_edp_256b0c6b-cacf-478c-80bf-b5332f4f37ae_indexes
```

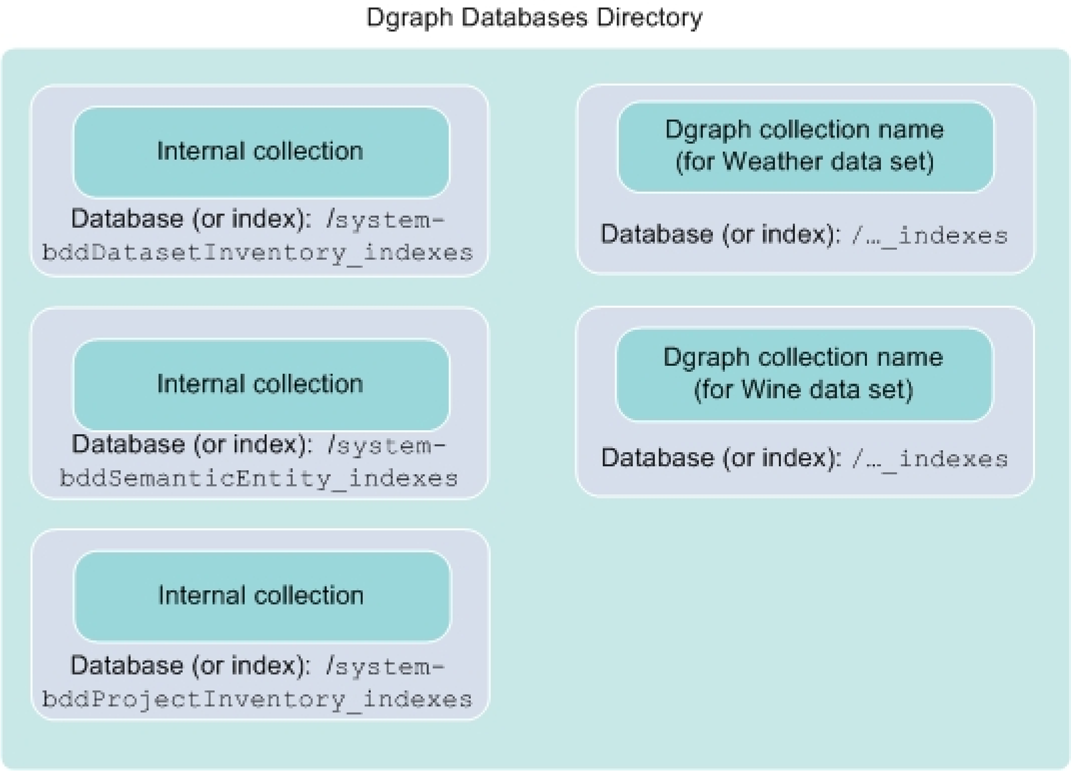
Each data set has its own Dgraph database, and there is only one data set per Dgraph database. The databases are stored in the directory you specify for the `DGRAPH_INDEX_DIR` property in the `bdd.conf` file. This directory is called the ***Dgraph databases directory***.

The Dgraph databases directory also contains three internal, system-created databases that are used by Studio:

- `system-bddProjectInventory_indexes`
- `system-bddDatasetInventory_indexes`
- `system-bddSemanticEntity_indexes`

For example, if you create two data sets, `Wine` and `Weather`, in Studio, the Dgraph databases directory creates five databases (one for each of the two data sets and three internal databases). You may also see other databases in the Dgraph databases directory; they may be created as a result of committing a transformed data set.

This diagram illustrates this example:


Dgraph Databases Directory
Internal collection
Database (or index): /system-bddDatasetInventory_indexes
Dgraph collection name (for Weather data set)
Database (or index): /..._indexes
Internal collection
Database (or index): /system-bddSemanticEntity_indexes
Dgraph collection name (for Wine data set)
Database (or index): /..._indexes
Internal collection
Database (or index): /system-bddProjectInventory_indexes

When a Dgraph database is created, it is automatically mounted by the Dgraph. Unmounted databases are also automatically mounted when the Dgraph receives a query that accesses the database's data. When a database is mounted, a log entry is made in the Dgraph out log, as in this example:

```
DGRAPH    NOTIFICATION      {database}    [0]    Mounting database
edp_cli_edp_256b0c6b-cacf-478c-80bf-b5332f4f37ae
```

Note that the entry is made by the Dgraph `database` log subsystem.

The database name also appears in other BDD component messages. For example, the name of a DP workflow in a YARN log will contain the database name:

```
EDP: ProvisionDataSetFromHiveConfig{hiveDatabaseName=default, hiveTableName=warrantyclaims,
newCollectionId=MdexCollectionIdentifier{databaseName
=edp_cli_edp_256b0c6b-cacf-478c-80bf-b5332f4f37ae,
collectionName=edp_cli_edp_256b0c6b-cacf-478c-80bf-b5332f4f37ae}}
```

You should also see database names in the logs for Studio, Dgraph HDFS Agent, and Transform Service.

### Dgraph support for HDFS Data at Rest Encryption

The HDFS Data at Rest Encryption feature, when enabled, allows data to be stored in encrypted HDFS directories called encryption zones. All files within an encryption zone are transparently encrypted and decrypted on the client side. Decrypted data is therefore never stored in HDFS.

If you have enabled HDFS Data at Rest Encryption, you can store your Dgraph databases in an encryption zone in HDFS. For details on enabling HDFS Data at Rest Encryption, see the *Installation Guide*.

### Dgraph Tracing Utility

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 Tracing Utility stores the Dgraph target trace data it collects in `*.ebb` files, which are useful in analyzing Dgraph crashes. The files are intended for use by Oracle Support. The files are saved in the `$DGRAPH_HOME/bin` directory. You can also manually generate and save the trace data with the `bdd-admin` script's `get-blackbox` command, as described in *get-blackbox on page 40*.

## Memory consumption by the Dgraph

This topic discusses the logic used by the Dgraph to control its memory consumption.

The Dgraph query performance depends on characteristics of your specific deployment: query workload and complexity, the characteristics of the loaded records, and the size of the Dgraph database.

These statements describe how the Dgraph utilizes memory:

- After the installation, when the Dgraph is started it allocates considerable amounts of virtual memory on the system. This is needed for ingesting data and executing queries, including those that are complex. This is an expected behavior and is observable if you use system diagnostic tools.
- If the Dgraph is installed on a machine that is hosting other processes, other memory-intensive processes are present in the operating system and require memory. In this case, the Dgraph releases a significant portion of its physical memory quickly. Without such pressure, that is in cases when the Dgraph is the sole process on the hosting machine, the Dgraph may retain the physical memory indefinitely. This is an expected behavior.

  Because of this, depending on your deployment requirements, such as the size of your deployment, it may be highly desirable to deploy the Dgraph instances on servers dedicated solely to each of the Dgraph processes (this means that these machines are not hosting any other processes, for BDD or other applications).

- If your Dgraph databases are on HDFS, the Dgraph must be deployed on HDFS DataNodes, but this should be the only other process running on those servers. In particular, you shouldn't deploy the Dgraph on servers running Spark, which also requires a lot of memory. If you have to co-locate the Dgraph and Spark, you must use Linux cgroups to ensure the Dgraph has access to the resources it requires; for more information, see *Setting up cgroups on page 76*.
- By default, the memory limit that the Dgraph is allowed to use on the machine is set to 80% of the machine's available RAM. This behavior ensures that the Dgraph does not run out of memory on the machine hosting the Dgraph. In other words, with this limit in place, the Dgraph is protected from running into out-of-memory performance issues.
- In addition to the default memory consumption limit of 80% of RAM, after the installation you can set a custom limit on the amount of memory the Dgraph can consume, using the Dgraph `--memory-limit` flag. If this limit is set, then, upon the Dgraph restart, the amount of memory required by the Dgraph to process all current queries cannot exceed this custom limit.

  **Note:** The Dgraph `--memory-limit` flag is intended for Oracle Support. For information on how to set it, see *Setting Dgraph memory limit on page 71*. Also, a value of `0` for the flag means there is no limit set on the amount of memory the Dgraph can use. In this case, you should be aware that the Dgraph will use all the memory on the machine that it can allocate for its processing without any limit, and will not attempt to cancel any queries that may require the most amount of memory. This, in turn, may lead to out-of-memory page thrashing and require manually restarting the Dgraph.

- Once the Dgraph reaches a memory consumption limit (it could be the default limit of 80% of RAM, or a custom memory limit set with `--memory-limit`), it starts to automatically cancel queries, beginning with the query that is currently consuming the most amount of memory. When the Dgraph cancels a query, it logs the amount of memory the query was using and the time it was cancelled for diagnostic purposes.
- In addition to the memory consumption limit, before you install Big Data Discovery, you can specify the Dgraph cache size, using the `DGRAPH_CACHE` property in the `bdd.conf` file located in your installation directory. The orchestration script uses this value at installation time. You can adjust the size of `DGRAPH_CACHE` later, at any point after the installation. For information, see *Setting the Dgraph cache size on page 71*.
- There is one additional consideration about the Dgraph cache that is useful to keep in mind, before you decide to adjust the cache size:

  While the Dgraph typically operates within the limits of its configured Dgraph cache size, it is possible for the cache to become over-subscribed for short periods of time. During such periods, the Dgraph may use up to 1.5 times more cache than it has configured. It is important to note that the Dgraph does not expect to routinely reach an increase in its configured cache usage. When the cache size reaches the 1.5 times threshold, the Dgraph starts to more aggressively evict entries that consume its cache, so that the cache memory usage can be reduced to its configured limits. This behavior is not configurable by the system administrators.

# Setting Dgraph memory limit

It is possible to specify the custom memory limit the Dgraph is allowed to use for processing. If the memory limit is changed, this overrides the default memory consumption setting in the Dgraph that is set to 80% of the machine's available RAM.

**Note:** It is recommended that Oracle Support change the limit on Dgraph memory consumption.

By default, the memory limit that the Dgraph is allowed to use is 80% of the machine's available RAM. This behavior ensures that the Dgraph never runs out of memory during the course of its query processing or data ingest activity.

You can override the default limit and set a custom limit on the amount of memory the Dgraph can consume in MB, using the `--memory-limit` flag. If this value is set, then the amount of memory required by the Dgraph to process all current queries can't exceed this limit.

Once the Dgraph reaches a memory consumption limit set with this flag, then, similar to how it behaves with the default memory limit of 80%, the Dgraph starts to cancel queries, beginning with the query that is consuming the most amount of memory. When the Dgraph cancels a query, it logs the amount of memory the query was using and the time it was cancelled for diagnostic purposes.

The Dgraph `--memory-limit` can be set after the installation through the `DGRAPH_ADDITIONAL_ARG` parameter in the `bdd.conf` file in the `$BDD_HOME/BDD_manager/conf` directory.

Using the `--memory-limit` flag with a value of `0` means there is no limit set on the amount of memory the Dgraph can use.

For information on all Dgraph flags, see *Dgraph flags on page 79*.

To change the memory limit:

1. Go to `$BDD_HOME/BDD_manager/conf` directory and locate the `bdd.conf` file.
2. In the setting for `DGRAPH_ADDITIONAL_ARG`, specify the `--memory-limit` flag.
3. Save the `bdd.conf` file.
4. Run the `bdd-admin.sh publish-config bdd` command.

   This refreshes the configuration on all the Dgraph hosting machines with the modified settings from the `bdd.conf` file. For information on how to do this, see *Updating the cluster configuration on page 50*.
5. Restart the Dgraph with the `bdd-admin.sh` script.

# Setting the Dgraph cache size

The Dgraph cache size should be configured to be large enough to allow the Dgraph to operate smoothly under normal query load.

For enhanced performance, Oracle recommends allocating at least 50% of the node's available RAM to the Dgraph cache. This is a significant amount of memory that you can adjust if needed. For example, if you later find that queries are getting cancelled because there is not enough available memory to process them, you should decrease this amount.

You configure the Dgraph cache size initially by setting the `DGRAPH_CACHE` value in the `bdd.conf` file in the installation directory. The orchestration script uses this value during the BDD installation process.

After the installation, you can adjust the size of the Dgraph cache by gradually changing the `DGRAPH_CACHE` value in the `bdd.conf` file in the `$BDD_HOME/BDD_manager/conf` directory and use the `bdd-admin publish-config` command to update the configuration for the entire cluster. For more information, see *publish-config on page 35*.

Before you adjust the Dgraph cache, keep the following consideration in mind:

While the Dgraph typically operates within the limits of its configured Dgraph cache size, it is possible for the cache to become over-subscribed for short periods of time. During such periods, the Dgraph may use up to 1.5 times more cache than it has configured. It is important to note that the Dgraph does not expect to routinely reach an increase in its configured cache usage. When the cache size reaches the 1.5 times threshold, the Dgraph starts to more aggressively evict entries that consume its cache, so that the cache memory usage can be reduced to its configured limits.

This means that an occasional spike in Dgraph cache usage should not be the cause of alarm and that you should only consider adjusting the Dgraph cache size after observing Dgraph performance over longer periods of time.

# Moving the Dgraph databases to HDFS

If your Dgraph databases are currently stored on an NFS, you can move them to HDFS.

Because HDFS is a distributed file system, storing your databases there provides increased high availability for the Dgraph. It also increases the amount of data your databases can contain.

When its databases are stored on HDFS, the Dgraph has to run on HDFS DataNodes. If it isn't currently installed on DataNodes, you must move its binaries over when you move its databases.

**Important:** The DataNode service should be the only Hadoop service running on the Dgraph nodes. In particular, you shouldn't co-locate the Dgraph with Spark, as both require a lot of resources. However, if you have to host the Dgraph on nodes running Spark or other Hadoop services, you should use cgroups to ensure it has access to sufficient resources. For more information, see *Setting up cgroups on page 76*.

To move your Dgraph databases to HDFS:

1. On the Admin Server, go to `$BDD_HOME/BDD_manager/bin` and stop BDD:

   ```
   ./bdd-admin.sh stop [-t <minutes>]
   ```

2. Copy your Dgraph databases from their current location to the new one in HDFS.

   The `bdd` user must have read and write access to the new location.

   If you have HDFS data at rest encryption enabled, the new location must be an encryption zone.

3. If the Dgraph isn't currently installed on HDFS DataNodes, select one or more in your Hadoop cluster to move it to.

   If other BDD components are currently installed on the selected nodes, verify that the following directories are present on each, and copy over any that are missing.

   - `$BDD_HOME/common/edp`

- `$BDD_HOME/dataprocessing`
- `$BDD_HOME/dgraph`
- `$BDD_HOME/logs/edp`

If no BDD components are installed on the selected nodes:

(a) Create a new `$BDD_HOME` directory on each node. Its permissions must be 755 an its owner must be the `bdd` user.

(b) Copy the following directories from an existing Dgraph node to the new ones:

- `$BDD_HOME/BDD_manager`
- `$BDD_HOME/common`
- `$BDD_HOME/dataprocessing`
- `$BDD_HOME/dgraph`
- `$BDD_HOME/logs`
- `$BDD_HOME/uninstall`
- `$BDD_HOME/version.txt`

(c) Create a symlink `$ORACLE_HOME/BDD` pointing to `$BDD_HOME`.

(d) Optionally, remove the `/dgraph` directory from the old Dgraph nodes, as it's no longer needed.

Leave any other BDD directories as they may still be useful.

4. To enable the Dgraph to access its databases in HDFS, install either the HDFS NFS Gateway service or FUSE.

The option you should use depends on your Hadoop cluster. You *must* use the NFS Gateway if you have CDH 5.7.1 or HDFS data at rest encryption enabled. In all other cases, you can use either option. More information about each is available in the *Installation Guide*.

To use the NFS Gateway, install it on all Dgraph nodes. For instructions, refer to the documentation for your Hadoop distribution.

To use FUSE:

(a) Download FUSE 2.8+ from *https://github.com/libfuse/libfuse/releases*.

(b) Extract `fuse-<version>.tar.gz`, then copy `/fuse-<version>` to the new Dgraph nodes.

(c) Install FUSE by going to `/fuse-<version>` on each node and running:

```
./configure
make -j8
make install
```

(d) Set the required user permissions on each node:

- Add the `bdd` user to the `fuse` group.
- Give the `bdd` user read and execute permissions for `fusermount`.
- Give the `bdd` user read and write permissions for `/dev/fuse`.

(e) Heavy workloads during parallel ingests can cause socket timeouts on HDFS clients, which can crash the Dgraph. To prevent this, make the following changes to your HDFS configuration:

5. If you're using FUSE, make the following changes to your HDFS configuration to prevent FUSE and the Dgraph from crashing during parallel ingests.

   (a) Open `hdfs-site.xml` in a text editor and add the following lines:

   ```
   <property>
        <name>dfs.client.socket-timeout</name>
        <value>600000</value>
   </property>
   <property>
        <name>dfs.socket.timeout</name>
        <value>600000</value>
   </property>
   <property>
        <name>dfs.datanode.socket.write.timeout</name>
        <value>600000</value>
   </property>
   ```

   (b) If you have CDH, open Cloudera Manager and add the above lines to the following properties:

   - **HDFS Service Advanced Configuration Snippet (Safety Valve) for hdfs-site.xml**
   - **DataNode Advanced Configuration Snippet (Safely Valve) for hdfs-site.xml**
   - **HDFS Client Advanced Configuration Snippet (Safety Valve) for hdfs-site.xml**

   If you Have HDP, open Ambari and set the following properties to 600000:

   - **dfs.client.socket-timeout**
   - **dfs.datanode.socket.write.timeout**
   - **dfs.socket.timeout**

   (c) Restart HDFS to make your changes take effect.

6. If you have to host the Dgraph on the same node as Spark or any other Hadoop processes, set up cgroups to isolate the resources used by Hadoop and the Dgraph.

   For instructions, see *Setting up cgroups on page 76*.

7. For best performance, configure short-circuit reads in HDFS.

   This enables the Dgraph to access local files directly, rather than having to use the HDFS DataNode's network sockets to transfer the data. For instructions, refer to the documentation for your Hadoop distribution.

8. Clean up the ZooKeeper index.

9. On the Admin Server, copy `bdd.conf` to a new location. Open the *copy* in a text editor and update the following properties:

| Property | Description |
|---|---|
| DGRAPH_INDEX_DIR | The absolute path to the new location of the Dgraph databases directory on HDFS.<br>If you have HDFS data at rest encryption enabled, this location must be an encryption zone. |
| DGRAPH_SERVERS | A comma-separated list of the FQDNs of the new Dgraph nodes. These must all be HDFS DataNodes. |

| Property | Description |
|---|---|
| DGRAPH_THREADS | The number of threads the Dgraph starts with. This should be the number of CPU cores on the Dgraph nodes minus the number required to run HDFS and any other Hadoop services running on the new Dgraph nodes. |
| DGRAPH_CACHE | The size of the Dgraph cache. This should be either 50% of the machine's RAM or the total amount of free memory, whichever is larger. |
| DGRAPH_USE_MOUNT_HDFS | Determines whether the Dgraph mounts HDFS when it starts. Set this to TRUE. |
| DGRAPH_HDFS_MOUNT_DIR | The absolute path to the local directory where the Dgraph mounts the HDFS root directory. This location must exist and be empty, and must have read, write, and execute permissions for the bdd user.<br>It's recommended that you use the default location, $BDD_HOME/dgraph/hdfs_root, which was created by the installer and should meet these requirements. |
| KERBEROS_TICKET_REFRESH_INTERVAL | Only required if you have Kerberos enabled. The interval (in minutes) at which the Dgraph's Kerberos ticket is refreshed. For example, if set to 60, the Dgraph's ticket would be refreshed every 60 minutes, or every hour. |
| KERBEROS_TICKET_LIFETIME | Only required if you have Kerberos enabled. The amount of time that the Dgraph's Kerberos ticket is valid. This should be given as a number followed by a supported unit of time: s, m, h, or d. For example, 10h (10 hours), or 10m (10 minutes). |
| DGRAPH_ENABLE_CGROUP | Only required if you set up cgroups for the Dgraph. This must be set to TRUE if you created a Dgraph cgroup. |
| DGRAPH_CGROUP_NAME | Only required if you set up cgroups for the Dgraph. The name of the cgroup that controls the Dgraph. |
| NFS_GATEWAY_SERVERS | Only required if you're using the NFS Gateway. A comma-separated list of the FQDNs of the nodes running the NFS Gateway service. This should include all Dgraph nodes. |
| DGRAPH_USE_NFS_MOUNT | If you're using the NFS Gateway, set this property to TRUE. |

10. To populate your configuration changes to the rest of the cluster, go to $BDD_HOME/BDD_manager/bin and run:

    ```
    ./bdd-admin.sh publish-config <path>
    ```

    Where <*path*> is the absolute path to the updated copy of bdd.conf.

11. Start your cluster:

    ```
    ./bdd-admin.sh start
    ```

# Setting up cgroups

Control groups, or cgroups, is a Linux kernel feature that enables you to allocate resources like CPU time and system memory to specific processes or groups of processes. If you need to host the Dgraph on nodes running Spark, you must use cgroups to ensure sufficient resources are available to it.

**Note:** Because the Dgraph and Spark are both memory-intensive processes, hosting them on the same nodes is not recommended and should only be done if absolutely necessary. Although you can use the `--memory-limit` flag to set Dgraph memory consumption, Spark isn't aware of this and will continue to use as much memory as it needs, regardless of other processes.

To do this, you must enable cgroups in Hadoop and create one for YARN to limit the CPU percentage and amount of memory it can consume. Then, create a separate cgroup for the Dgraph to allocate appropriate amounts of memory and swap space to it.

To set up cgroups:

1. If your system doesn't currently have the `libcgroup` package, install it as root.

   This creates `/etc/cgconfig.conf`, which configures cgroups.

2. Enable the `cgconfig` service to run automatically:

   ```
   chkconfig cgconfig on
   ```

3. Create a cgroup for YARN. This must be done within Hadoop. For instructions, refer to the documentation for your Hadoop distribution.

   The YARN cgroup should limit the amounts of CPU and memory allocated to all YARN containers. The appropriate limits to set depend on your system and the amount of data you will process. At a minimum, you should reserve the following for the Dgraph:

   - 10GB of RAM
   - 2 CPU cores

   The number of CPU cores YARN is allowed to use must be specified as a percentage. For example, on a quad-core machine, YARN should only get two of cores, or 50%. On an eight-core machine, YARN could get up to four of them, or 75%. When setting this amount, remember that allocating more cores to the Dgraph will boost its performance.

4. Create a cgroup for the Dgraph by adding the following to `cgconfig.conf`:

   ```
   # Create a Dgraph cgroup named "dgraph"
   group dgraph {
       # Specify which users can edit this group
       perm {
           admin {
               uid = $BDD_USER;
           }
           # Specify which users can add tasks for this group
           task {
               uid = $BDD_USER;
           }
       }
       # Set the memory and swap limits for this group
       memory {
           # Set memory limit to 10GB
           memory.limit_in_bytes = 10000000000;
   ```

```
        # Set memory + swap limit to 12GB
        memory.memsw.limit_in_bytes = 12000000000;
    }
}
```

Where `$BDD_USER` is the name of the `bdd` user.

**Note:** The values given for `memory.limit_in_bytes` and `memory.memsw.limit_in_bytes` above are the *absolute minimum* requirements. You should use higher values, if possible.

5. Restart `cfconfig` to enable your changes.

# Appointing a new Dgraph leader

You can use the `appointNewDgraphLeader.sh` script to appoint a new Dgraph leader for a database.

The use case for this script is when there is a long-running ingest in progress in the Dgraph HDFS Agent, and the Dgraph goes down for some reason. Instead of waiting until a new write request comes in, the administrator can just run this script to restart the ingest (for the same database) on another machine. (A file is maintained in HDFS that logs the exact progress of the ingest. The newly-appointed Dgraph HDFS Agent leader reads the file and knows at what point to pick up the ingest).

For example, the Dgraph HDFS Agent on Dgraph_A is performing an ingest (on the database named EdpTest) when the Dgraph crashes (which results in the ingest being suspended). When the script is run, the new leader for the EdpTest database can be Dgraph_B, in which case the ingest is picked up at the point when it was stopped (except that Dgraph_B is now performing the ingest instead of Dgraph_A). Because the database is shared among the Dgraphs, the ingest can be resumed by the new leader.

Note that if the script is run but a new leader has been appointed in the interim, then the script basically reappoints the same leader.

The syntax for running the script is:

```
./appointNewDgraphLeader.sh <dg_address> <database_name>
```

where:

- *dg_address* is the FQDN (fully-qualified domain name) and port of the Dgraph Gateway server.
- *database_name* is the name of the database for the ingest.

For example (using the EdpTest database in the example above):

```
./appointNewDgraphLeader.sh web009.us.example.com:7003 EdpTest
```

To appoint a new Dgraph leader for a database:

1. Navigate to the `$DGRAPH_HOME/dgraph-hdfs-agent/bin` directory.
2. Run the `appointNewDgraphLeader.sh` script with the FQDN and port of the Dgraph Gateway and the database name, as in the example above.

If a new Dgraph leader is successfully appointed, the script returns this message:

```
New Dgraph Leader appointed for database <database_name>
```

An unsuccessful operation could return either of these messages:

```
Unable to appoint new Dgraph leader
```

```
Could not reach Dgraph gateway
```

Note that an unsuccessful attempt could be caused by an incorrect address for the Dgraph Gateway.

## Linux ulimit settings for merges

For purposes of generation merging, it is recommended that you set the Linux option `ulimit -v` and `-m` parameters to `unlimited`. You should also set the `-n` parameter to `65536`.

An `unlimited` setting for the `-v` option sets no limit on the maximum amount of virtual memory available to a process and for the `-m` option sets no limit on the maximum resident set size. Setting these options to `unlimited` can help prevent problems when the Dgraph is merging the generation files. Setting the `-n` option to `65536` sets the maximum number of open file descriptors to 64K, which is especially important if the Dgraph and Hadoop are running on the same node.

An example of a merge problem due to insufficient disk space and memory resources is a Dgraph error similar to the following:

```
ERROR 04/03/13 05:24:35.668 UTC (1364966675668) DGRAPH {dgraph} BackgroundMergeTask:
exception thrown: Can't parse generation file, caused by I/O Exception: While mapping file,
caused by mmap failure: Cannot allocate memory
```

In this case, the problem is caused because the Dgraph cannot allocate enough virtual memory for its merging task.

## Managing Dgraph core dump files

In the rare case of a Dgraph crash, the Dgraph writes its core dump files on disk. It is recommended to use the `ulimit -c unlimited` setting for the Dgraph core dump files. Non-limited core files contain all Dgraph data that is resident in memory (RSS of the Dgraph process).

When the Dgraph runs on a very large data set, the size of its database files stored in-memory may exceed the size of the physical RAM. If such a Dgraph fails, it may need to write out potentially very large core dump files on disk. The core files are written to the directory from which the Dgraph was started.

To troubleshoot the Dgraph, it is often useful to preserve the entire set of core files written out as a result of such failures. When there is not enough disk space, only a portion of the files is written to disk until this process stops. Since the most valuable troubleshooting information is contained in the last portion of core files, to make these files meaningful for troubleshooting purposes, it is important to provision enough disk space to capture the files in their entirety.

Two situations are possible, depending on your goal:

- You can afford to provision enough disk space.

  Large applications may take up the entire amount of available RAM. Because of this, the Dgraph core dump files can also grow large and take up the space equal to the size of the physical RAM on disk plus the size of the server data files in memory. To troubleshoot a Dgraph crash, provision enough disk space to capture the entire set of core files. In this case, the files are saved at the expense of potentially filling up the disk.

  **Note:** If you are not setting `ulimit -c unlimited`, you could be seeing the Dgraph crashes that do not write any core files to disk, since on some Linux installations the default for `ulimit -c` is set to 0.

- You would like to limit the amount of disk space allotted for saving core files.

  To prevent filling up the disk, you can limit the size of these files on the operating system level, with the `ulimit -c <size>` command, although this is not recommended. If you set the limit size in this way, the core files cannot be used for debugging, although their presence will confirm that the Dgraph had crashed. In this case, with large Dgraph applications, only a portion of core files is saved on disk. This may limit their usefulness for debugging purposes. To troubleshoot the crash in this case, change this setting to `ulimit -c unlimited`, and reproduce the crash while capturing the entire core file. Similarly, to enable support to troubleshoot the crash, you will need to reproduce the crash while capturing the full core file.

## About Dgraph statistics

The Dgraph statistics page provides information such as startup time, host, port, and process information, data and log paths, and so on. This information is useful to help to tune your Dgraph and useful for Oracle Support.

The statistics page information is valid as long as the Dgraph is running; it is reset upon a Dgraph restart or by resetting the statistics page.

You can view or reset the Dgraph statistics page with these `bdd-admin` script command:

- You can view the Dgraph statistics page with *get-stats on page 42*.
- You can reset the statistics with *reset-stats on page 43*.

## Dgraph flags

Dgraph flags modify the Dgraph's configuration and behavior.

**Important:** Dgraph flags are intended for use by Oracle Support only. They are included in this document for completeness.

You can set Dgraph flags by adding them to the `DGRAPH_ADDITIONAL_ARG` property in `bdd.conf` in `$BDD_HOME/BDD_manager/conf` directory, then using the `bdd-admin publish-config` script to update the cluster configuration. Any flag included in this list will be set each time the Dgraph starts. For more information, see *publish-config on page 35*.

**Note:** Some of the Dgraph flags have the same names as HDFS Agent flags. These must have the same settings as their HDFS Agent counterparts.

| Flag | Description |
|---|---|
| ? | Prints the help message and exits. The help message includes usage information for each Dgraph flag. |
| `-v` | Enables verbose mode. The Dgraph will print information about each request it receives to either its stdout/stderr file (`dgraph.out`) or the file set by the `--out` flag. |

| Flag | Description |
|---|---|
| `--backlog-timeout` | Specifies the maximum number of seconds that a query is allowed to spend waiting in the processing queue before the Dgraph responds with a timeout message.<br>The default is `0` seconds. |
| `--bulk_load_port` | Sets the port on which the Dgraph listens for bulk load ingest requests. This must be the same as the port specified for the HDFS Agent `--bulk_load_port` flag.<br>This flag maps to the `DGRAPH_BULKLOAD_PORT` property in `bdd.conf`. |
| `--cluster_identity` | Specifies the cluster identity of the Dgraph running on this node. The syntax is:<br>`protocol:hostname:dgraph_port:dgraph_bulk_load_port:agent_port`<br>This must be the same as the cluster identity specified for the HDFS Agent `--custer_identity` flag. |
| `--cmem` | Specify the maximum memory usage (in MB) for the Dgraph cache. For more information, see *Setting the Dgraph cache size on page 71*.<br>This flag maps to the `DGRAPH_CACHE` property in `bdd.conf`. |
| `--export_port` | Specifies the port on which the Dgraph listens for requests from the HDFS Agent.<br>This should be the same as the number specified for the HDFS Agent `--export_port` flag. It should be different from the numbers specified for both the `--port` and `--bulk_load_port` flags.<br>This flag maps to the `AGENT_EXPORT_PORT` property in `bdd.conf`. |
| `--help` | Prints the help message and exits. The help message includes usage information for each Dgraph flag. |
| `--host` | Specifies the name of the Dgraph's host server.<br>This flag maps to the `DGRAPH_SERVERS` property in `bdd.conf`. |
| `--log` | Specifies the path to the Dgraph request log file. The default file used is `dgraph.reqlog`. |

| Flag | Description |
|---|---|
| `--log-level` | Specifies the log level for the Dgraph log subsystems. For information on setting this flag, see *Setting the Dgraph log levels on page 164*.<br>This flag maps to the `DGRAPH_LOG_LEVEL` property in `bdd.conf`. |
| `--memory-limit` | Specifies the maximum amount of memory (in MB) the Dgraph is allowed to use for processing.<br>If you do not use this flag, the memory limit is by default set to 80% of the machine's available RAM.<br>If you specify a limit in MB for this flag, this number is used as the memory consumption limit, for the Dgraph, instead of 80% of the machine's available RAM.<br>If you specify `0` for this flag, this overrides the default of 80% and means there is no limit on the amount of memory the Dgraph can use for processing.<br>For a summary of how Dgraph allocates and utilizes memory, see *Memory consumption by the Dgraph on page 69*. |
| `--mount_hdfs` | Specifies that the Dgraph should mount HDFS. The target HDFS is specified by <hdfs config> which is the Hadoop HDFS configuration file (usually named `hdfs-site.xml`) and <core config> which is the Hadoop core configuration file (usually named `core-site.xml`). |
| `--net-timeout` | Specifies the maximum amount of time (in seconds) the Dgraph waits for the client to download data from queries across the network. The default value is `30` seconds. |
| `--out` | Specifies a file to which the Dgraph's stdout/stderr will be remapped. If this flag is omitted, the Dgraph uses its default stdout/stderr file, `dgraph.out`.<br>This file must be different from the one specified by the HDFS Agent's `--out` flag.<br>This flag maps to the `DGRAPH_OUT_FILE` property in `bdd.conf`. |
| `--pidfile` | Specifies the file the Dgraph's process ID (PID) will be written to. The default filename is `dgraph.pid`. |
| `--port` | Specifies the port used by the Dgraph's host server.<br>This flag maps to the `DGRAPH_WS_PORT` property in `bdd.conf`. |
| `--search_char_limit` | Specifies the maximum number of characters that a text search term can contain. The default value is `132`. |

| Flag | Description |
|---|---|
| `--search_max` | Specifies the maximum number of terms that a text search query can contain. The default value is `10`. |
| `--snip_cutoff` | Specifies the maximum number of words in an attribute that the Dgraph will evaluate to identify a snippet. If a match is not found within the specified number of words, the Dgraph won't return a snippet, even if a match occurs later in the attribute value.<br>The default value is `500`. |
| `--snip_disable` | Globally disables snippeting. |
| `--sslcafile` | **Note:** This flag is not used in Oracle Big Data Discovery.<br>Specifies the path to the SSL Certificate Authority file that the Dgraph will use to authenticate SSL communications with other components. |
| `--sslcertfile` | **Note:** This flag is not used in Oracle Big Data Discovery.<br>Specifies the path of the SSL certificate file that the Dgraph will present to clients for SSL communications. |
| `--stat-brel` | **Note:** This flag is deprecated and not used in Oracle Big Data Discovery.<br>Creates dynamic record attributes that indicate the relevance rank assigned to full-text search result records. |
| `--syslog` | Directs all output to syslog. |
| `--threads` | Specifies the number of threads the Dgraph will use to process queries and execute internal maintenance tasks. The value you provide must be a positive integer (2 or greater). The default is 2 threads.<br>The recommended number of threads for machines running only the Dgraph is the number of CPU cores the machine has. For machines co-hosting the Dgraph with other Big Data Discovery components, the recommended number of threads is the number of CPU cores the machine has minus two.<br>This flag maps to the `DGRAPH_THREADS` property in `bdd.conf`. |

| Flag | Description |
|---|---|
| --version | Prints version information and then exits. The version information includes the Oracle Big Data Discovery version number and the internal Dgraph identifier. |
| --wildcard_max | Specifies the maximum number of terms that can match a wildcard term in a wildcard query that contains punctuation, such as ab*c.def*. The default is 100. |
| --zookeeper | Specifies a comma-separated list of ZooKeeper servers. The syntax for each ZooKeeper server is:<br>`<hostname>:<port>`<br>This must be the same as the value specified for the HDFS Agent --zookeeper flag. |
| --zookeeper_auth | Obtains the ZooKeeper authentication password from standard in. Note the following about this flag:<br>• The "ZooKeeper authentication password" corresponds to individual node-level access using ACL described here (Dgraph uses the digest scheme): *https://zookeeper.apache.org/doc/r3.1.2/zookeeperProgrammers.html#sc_ZooKeeperAccessControl*<br>It has nothing to do with Kerberos or the ability of the Dgraph to establish a session with ZooKeeper.<br>• It is imperative that all Dgraphs, Dgraph Gateway, and Dgraph HDFS Agent are using the same "Zookeeper authentication password" because they will not be able to access needed information created by other components if they are using different passwords. If the Dgraph cannot access information in ZooKeeper due to a wrong password, it is a fatal error. |
| --zookeeper_index | Specifies the index of the Dgraph cluster in the ZooKeeper ensemble. ZooKeeper uses this value to identify the Dgraph cluster. This must be the same as the value specified for the HDFS Agent --zookeeper_index flag.<br>This flag maps to the ZOOKEEPER_INDEX property in bdd.conf. |

| Flag | Description |
|---|---|
| `--zookeeper_session_cache` | Specifies the name and (optionally) location of the session cache file used by all Dgraph instances. Each Dgraph uses the file to terminate its last ZooKeeper session if 1) it exits abnormally and 2) if the ZooKeeper was turned off a period longer than session time out and turned back on. The leader uses this file to resume its last session with ZooKeeper if it exits abnormally.<br>This file is created when a session is first established and is deleted when on a normal shutdown of the Dgraphs. On an abnormal shutdown, the file remains on disk so that the Dgraphs can resume the last session.<br>The default file is:<br>`$BDD_HOME/dgraph/zk_session/<managed-server>.session`<br>The file location should always be the same to ensure that all Dgraphs can find it. Additionally, you should not modify the contents of this file. |

## Dgraph HDFS Agent flags

This topic describes the flags used by the Dgraph HDFS Agent.

The Dgraph HDFS Agent requires several flags, which are described in the following table. Note that some flags have the same name as their Dgraph flag counterpart, and (except for `--out`) must have the same settings.

The `startDgraphHDFSAgent.sh` script can use the following flags:

| Dgraph HDFS Agent flag | Description |
|---|---|
| `--agent_port` | Sets the port on which the Dgraph HDFS Agent is listening for HTTP requests. Note that there is no Dgraph version of this flag. |
| `--export_port` | Sets the port on which the Dgraph HDFS Agent is listening for requests from the Dgraph. This port number must be the same as specified for the Dgraph `--export_port` flag. |
| `--port` | Specifies the port on which the Dgraph is listening for HTTP requests. This port number must be the same as specified for the Dgraph `--port` flag. |
| `--bulk_load_port` | Sets the port on which the Dgraph HDFS Agent is listening for bulk load ingest requests. This port number must be the same as specified for the Dgraph `--bulk_load_port` flag. |

| Dgraph HDFS Agent flag | Description |
|---|---|
| `--cluster_identity` | Specifies the cluster identity of the Dgraph running on this node. The syntax is:<br>`protocol:hostname:dgraph_port:dgraph_bulk_load_port:agent_port`<br>This cluster identity must be the same as specified for the Dgraph `--cluster_identity` flag. |
| `--notifications_server_url` | Specifies the URL of the Notification Service. |
| `--out` | Specifies the file name and path of the Dgraph HDFS Agent's stdout/stderr log file. The log name must be different from that specified with the Dgraph `--out` flag. |
| `--principal` | For Kerberos support, specifies the name of the principal. |
| `--keytab` | For Kerberos support, specifies the path to the principal's keytab. |
| `--krb5conf` | For Kerberos support, specifies the path to the `krb5.conf` configuration file. |
| `--hadoop_truststore` | To support TLS-enabled Hadoop services, specifies the location of the Hadoop trust store. |
| `--zookeeper` | Specifies the host and port on which ZooKeeper is running. The syntax is:<br>`host:port`<br>(with a semicolon separating the host name and port). This host:port must be the same as specified for the Dgraph `--zookeeper` flag. |
| `--zookeeper_index` | Specifies the index of the cluster in the ZooKeeper ensemble. This index must be the same as specified for the Dgraph `--zookeeper_index` flag. |
| | |

## Hadoop configuration files

The `core-site.xml` and `hdfs-site.xml` files are used to configure a Hadoop cluster, especially the one machine in the cluster that is designated as the NameNode. The NameNode contains the HDFS file system from which the Dgraph HDFS Agent will read ingest files and write export files.

At start-up, the Dgraph HDFS Agent reads in the `core-site.xml` and `hdfs-site.xml` files so it can determine the location of the NameNode.

## Startup example

The following is an example of using the `startDgraphHDFSAgent.sh` to start the Dgraph HDFS Agent:

```
./startDgraphHDFSAgent.sh --agent_port 7102 --export_port 7101 --port 5555
    --bulk_load_port 5556 --coordinator web04.example.com:2181 --zookeeper_index cluster1
    --cluster_identity http:web04.example.com:5555:5556:7102 --out /tmp/agent.log
```

# Part III

## Administering Studio

Chapter 6

# Managing Data Sources

You can add, configure, and delete database connections and JDBC data sources on the **Control Panel>Big Data Discovery>Data Source Library** page of Studio.

## About database connections and JDBC data sources

Studio users can import data from an external JDBC database and access it from Studio as a data set in the Catalog.

A default installation of Big Data Discovery includes JDBC drivers to support the following relational database management systems:

- Oracle 11g and 12c
- MySQL

To set up this feature, there are both Studio administrator tasks and Studio user tasks.

A Studio administrator goes to the **Data Source Library** page and creates a connection to a database and creates any number of data sources, each with unique log in information, that share that database connection. The administrator configures each new data source with log in information to restrict who is able to create data sets from it. Data sources are not available to Studio users until an administrator sets them up.

Next, a Studio user clicks **Create a data set from a database** to import and filter the JDBC data source. After upload, the data source is available as a data set in the Catalog.

## Creating data connections

To create a data connection, follow the steps below.

To create a data connection:

1. Log in to Studio as an administrator.
2. Click **Configuration Options>Control Panel** and navigate to **Big Data Discovery>Data Source Library**.

3. Click **+ Connection**.
4. On the **New data connection** dialog, provide the name, URL, and authentication information for the data connection.
5. Click **Save**.

## Deleting data connections

If you delete a data connection, the associated data sources also are deleted. Any data sets created from those data sources can no longer be refreshed once the connection has been deleted.

To delete a data connection:

1. Log in to Studio as an administrator.
2. Click **Configuration Options>Control Panel** and navigate to **Big Data Discovery>Data Source Library**.
3. Locate the data source connection and click the delete icon.
4. In the confirmation dialog, click **Delete**.

## Creating a data source

When you create a data source, you specify a SQL query to select the data to include.

To create a data source:

1. Log in to Studio as an administrator.
2. Click **Configuration Options>Control Panel** and navigate to **Big Data Discovery>Data Source Library**.
3. Click **+ data source** for a data connection you created previously.
4. Provide the required authentication information for the data connection, then click **Continue**.
5. Provide a name and description for the data source.
6. In **Maximum number of records**, specify the maximum number of records to include in the data set.

   Studio does not control the order of the records. The SQL statement can indicate the order of records to import using an ORDER BY clause.
7. In the text area, enter the SQL query to retrieve the records for the data source, then click **Next**.

   The next page shows the available columns, with a sample list of records for each.
8. Click **Save**.

Once you have completed this task, the data source displays on the Studio Catalog as a new data set available to users.

# Editing a data source

Once a data source is created, you can change the data or edit it.

## Displaying details for a data source

To display detailed information for a data source, click the data source name. On the details panel:

- The **Data Source Info** tab provides a summary of information about the data source, including tags, the types of attributes, and the current access settings.
- The **Associated Data Sets** tab lists data sets that have been created from the data source.

## Editing a data source

To edit a data source, click the **Edit** link on the data source details panel, or click the name itself.

# Deleting a data source

To delete a data source, follow the steps below.

To delete a data source:

1. Log in to Studio as an administrator.
2. Click **Configuration Options>Control Panel** and navigate to **Big Data Discovery>Data Source Library**.
3. In the Data Connections part of the page, expand the data connection on which your data source is based.
4. Click the information icon for the data source you want to delete.
5. Click the **Delete** link
6. In the confirmation dialog, click **Delete**.

Chapter 7

# Configuring Studio Settings

The **Studio Settings** page on the **Control Panel** configures many general settings for the Studio application.

*Studio settings list*

*Changing the Studio setting values*

## Studio settings list

Studio settings include configuration options for timeouts, default values, and the connection to Oracle MapViewer, for the **Map** and **Thematic Map** components.

**Important:** Except where noted, editing Studio settings is not supported in Big Data Discovery Cloud Service.

The Studio settings are:

| Setting | Description |
|---|---|
| `df.advancedSparkAggregationsEnabled` | Specifies a Boolean value to enable the `set`, `variance`, and `standard dev` operators to the Aggregate transform. These operators are not supported in environments using Spark 1.5. If your environment uses 1.5, set the value to `false`. The operators are supported by Spark 1.6, so if your environment uses 1.6, you can set the value to `true`.<br>The default value is `false`. |
| `df.bddSecurityManager` | The fully-qualified class name to use for the BDD Security Manager. If empty, the Security Manager is disabled. |
| `df.clientLogging` | Sets the logging level for messages logged on the Studio client side. Valid values are ALL, TRACE, DEBUG, INFO, WARN, ERROR, FATAL and OFF. Messages are logged at the set level or above.<br>**Note:** Editing this setting is supported in BDD Cloud Service. |

| Setting | Description |
|---|---|
| `df.countApproxEnabled` | Specifies a Boolean value to indicate that components perform approximate record counts rather than precise record counts. A value of `true` indicates that Studio display approximate record counts using the `COUNT_APPROX` aggregation in an EQL query. A value of `false` indicates precise record counts using the `COUNT` aggregation. Setting this to `true` increases the performance of refinement queries in Studio.<br>The default value is `false`.<br>**Note:** Editing this setting is supported in BDD Cloud Service. |
| `df.dataSourceDirectory` | The directory used to store keystore and certificate files for secured data. |
| `df.defaultAccessForDerivedDataSets` | Controls whether new data sets created by **Export** or **Create new data set** are set to **Private** (restricted to the creator and all Studio Administrators) or made publically available at various access levels. Defaults to **Public (Default Access)**. |
| `df.defaultCurrencyList` | A comma-separated list of currency symbols to add to the ones currently available.<br>**Note:** Editing this setting is supported in BDD Cloud Service. |
| `df.helpLink` | Used to configure the path to the documentation for this release.<br>Used for links to specific information in the documentation. |
| `df.mapLocation` | The URL for the Oracle MapViewer eLocation service.<br>The eLocation service is used for the text location search on the **Map** component, to convert the location name entered by the user to latitude and longitude.<br>By default, this is the URL of the global eLocation service.<br>If you are using your own internal instance, and do not have Internet access, then set this setting to "None", to indicate that the eLocation service is not available. If the setting is "None", Big Data Discovery disables the text location search.<br>If this setting is not "None", and Big Data Discovery is unable to connect to the specified URL, then Big Data Discovery disables the text location search.<br>Big Data Discovery then continues to check the connection each time the page is refreshed. When the service becomes available, Big Data Discovery enables the text location search. |

| Setting | Description |
|---|---|
| `df.mapTileLayer` | The name of the MapViewer Tile Layer.<br>By default, this is the name of the public instance.<br>If you are using your own internal instance, then you must update this setting to use the name you assigned to the Tile Layer. |
| `df.mapViewer` | The URL of the MapViewer instance.<br>By default, this is the URL of the public instance of MapViewer.<br>If you are using your own internal instance of MapViewer, then you must update this setting to connect to your MapViewer instance. |
| `df.mdexCacheManager` | Internal use only.<br>**Note:** Editing this setting is supported in BDD Cloud Service. |
| `df.notificationsMaxDaysToStore` | The maximum number of days to store notifications. This is a setting to prune notifications from displaying in the **Notifications** window. It is a global limit that applies to all Studio users. Notifications that are older than this value are automatically deleted. |
| `df.notificationsMaxToStore` | The maximum number of notifications to store per user. This is a setting to prune notifications from displaying in the **Notifications** window. Notifications that exceed this value are automatically deleted.<br>The default value is 300. |
| `df.stringTruncationLimit` | The maximum number of characters to display for a string value.<br>This value may be overridden when configuring the display of a string value in an individual component.<br>The default value is 10000.<br>**Note:** Editing this setting is supported in BDD Cloud Service. |
| `df.sunburstAnimationEnabled` | Toggles animation and dynamic refinements for the **Chart>Pie / Sunburst** component. |
| `df.performanceLogging` | This property can only be modified from the `portal-ext.properties` file. |

# Changing the Studio setting values

To set the values of Studio settings, you can either use the fields on the **Studio Settings** page, or add the values to `portal-ext.properties`. If you configure a setting in `portal-ext.properties`, then the field on the **Studio Settings** page is locked.

Configuring settings in `portal-ext.properties` makes it easier to migrate settings across different environments. For example, after testing the settings in a development system, you can simply copy the properties file to the production system, instead of having to reset the production settings manually from the **Control Panel**.

To change the Studio setting values:

1. To configure Studio settings:

   (a) From the Control Panel, select **Big Data Discovery>Studio Settings**.

   (b) For each setting you want to update, provide a new value in the setting configuration field.

   **Note:** Take care when modifying these settings, as incorrect values can cause problems with your Studio instance.

   If the setting is configured in `portal-ext.properties`, then you cannot change the setting from this page. You must set it in the file.

   (c) Click **Update Settings**.

   (d) To apply the changes, restart Big Data Discovery.

2. To add a setting to `portal-ext.properties`:

   (a) Stop the server.

   (b) Open a command prompt and change to `$ORACLE_HOME/user_projects/domains/bdd_<version>_domain/config/studio`

   (c) Open `portal-ext.properties` in a text editor and add the setting.

   In the file, the format for adding a setting is:

   ```
   <settingname>=<value>
   ```

   Where:

   - *<settingname>* is the name of the setting from the **Studio Settings** page.
   - *<value>* is the value of the setting.

   For example, to set the maximum number of records to export, the entry would be:

   ```
   df.maxExportRecords=50000
   ```

   (d) Save and close the file.

   (e) Restart Studio.

   On the **Framework Settings** page, the setting is now read only.

Chapter 8

# Configuring Data Processing Settings

In order to upload files and perform other data processing tasks, you must configure the **Data Processing Settings** on Studio's Control Panel.

## List of Data Processing Settings

The settings listed in the table below must be set correctly in order to perform data processing tasks.

Many of the default values for these setting are populated based the values specified in `bdd.conf` during the installation process.

In general, the settings below should match the Data Processing CLI configuration properties which are contained in the script itself. Parameters that must be the same are noted as such in the table below. For information about the Data Processing CLI configuration properties, see the *Data Processing Guide*.

**Important:** Except where noted, editing the Data Processing settings is not supported in Big Data Discovery Cloud Service.

| Hadoop Setting | Description |
|---|---|
| `bdd.enableEnrichments` | Specifies whether to run data enrichments during the sampling phase of data processing. This setting controls the Language Detection, Term Extraction, Geocoding Address, Geocoding IP, and Reverse Geotagger modules. A value of `true` runs all the data enrichment modules and `false` does not run them.<br>You cannot enable an individual enrichment. The default value is `true`.<br>**Note:** Editing this setting is supported in BDD Cloud Service. |

| Hadoop Setting | Description |
|---|---|
| `bdd.maxRecordsToProcess` | Specifies the maximum number of records in the sample size of a data set. This is a global setting controls both the sample size for all files uploaded using Studio, and it also controls the sample size resulting from transform operations such as Join, Aggregate, and FilterRows.<br>For example, you if upload a file that has 5,000,000 rows, you could restrict the total number of sampled records to 1,000,000.<br>The default value is 1,000,000. (This value is approximate. After data processing, the actual sample size may be slightly more or slightly less than this value.)<br>**Note:** Editing this setting is supported in BDD Cloud Service. |
| `bdd.maxSplitSize` | The maximum partition size for Spark jobs measured in MB. This controls the size of the blocks of data handled by Data Processing jobs.<br>Partition size directly affects Data Processing performance — when partitions are smaller, more jobs run in parallel and cluster resources are used more efficiently. This improves both speed and stability.<br>The default is set by the `MAX_INPUT_SPLIT_SIZE` property in the `bdd.conf` file (which is 32, unless changed by the user). The 32MB is amount should be sufficient for most clusters, with a few exceptions:<br>• If your Hadoop cluster has a very large processing capacity and most of your data sets are small (around 1GB), you can decrease this value.<br>• In rare cases, when data enrichments are enabled the enriched data set in a partition can become too large for its YARN container to handle. If this occurs, you can decrease this value to reduce the amount of memory each partition requires.<br>Note that this property overrides the HDFS block size used in Hadoop. |

### Data Processing Topology

In addition to the configurable settings above, you can review the data processing topology by navigating to the **Big Data Discovery>About Big Data Discovery** page and expanding the **Data Processing Topology** drop-down. This exposes the following information:

| Hadoop Setting | Description |
|---|---|
| **Hadoop Admin Console** | The hostname and Admin Console port of the machine that acts as the Master for your Hadoop cluster. |
| **Name Node** | The NameNode internal Web server and port. |
| **Hive metastore Server** | The Hive metastore listener and port. |
| **Hive Server** | The Hive server listener and port. |
| **Hue Server** | The Hue Web interface server and port. |
| **Cluster OLT Home** | The OLT home directory in the BDD cluster. The BDD installer detects this value and populates the setting. |
| **Database Name** | The name of the Hive database that stores the source data for Studio data sets. |
| **EDP Data Directory** | The directory that contains the contents of the `edp_cluster_*.zip` file on each worker node. |
| **Sandbox** | The HDFS directory in which to store the avro files created when users export data from Big Data Discovery. The default value is `/user/bdd`. |

## Changing the data processing settings

You configure the settings on the **Data Processing Settings** page on the **Control Panel**.

To change the Hadoop setting values:

1. Log in to Studio as an administrator.
2. From the **Control Panel**, select **Big Data Discovery>Data Processing Settings**.
3. For each setting, update the value as necessary.
4. Click **Update Settings**.

The changes are applied immediately.

Chapter 9

# Running a Studio Health Check

You check the health and basic functionality of Studio by running a health check URL in a Web browser. This operation is typically only run after major changes to the BDD set up such as upgrading and patching.

You do not need machine access or command line access to run the health check URL. This is especially useful if you do not have machine access and therefore access to a command prompt to run `bdd-admin`.

The health check URL provides a more complete Studio check than running the `bdd-admin status` command. The `bdd-admin` command pings the Studio instance to see whether it is running or not. Whereas, the health check URL does the following:

- Checks that the Studio database is accessible.
- Uploads a file to HDFS.
- Creates a Hive table from that file.
- Ingests a data set from that Hive table.
- Queries the data set to ensure it returns results.

To run a Studio health check:

1. Start a web browser and type the following health check URL:

   `http://<Studio Host Name>:<Studio port>/bdd/health.`

   For example: `http://abcd01.us.oracle.com:7003/bdd/health`.

2. Optionally, check the **Notifications** panel to watch the progress of the check if you are signed into Studio.

The check should return `200 OK` to the browser if the health check succeeds.

Chapter 10

# Viewing Project Usage Summary Reports

Big Data Discovery provides basic reports to allow you to track project usage.

## About the project usage logs

Big Data Discovery stores project creation and usage information in its database.

### When entries are added to the usage logs

Entries are added when users:

- Log in to Big Data Discovery
- Navigate to a project
- Navigate to a different page in a project
- Create a data set from the **Data Source Library**
- Create a project

### When entries are deleted from the usage logs

By default, whenever you start Big Data Discovery, all entries 90 days old or older are deleted from the usage logs.

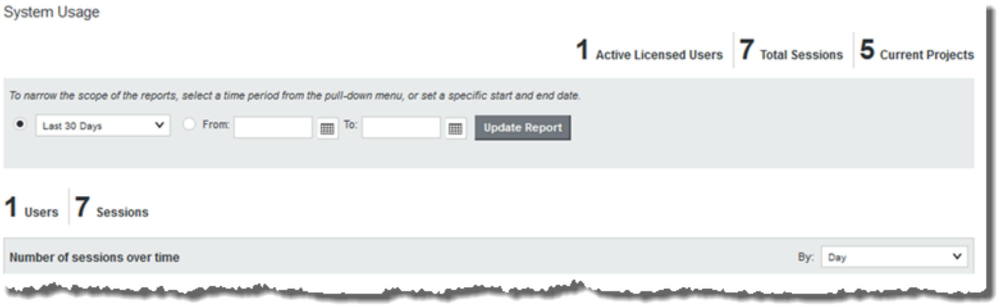
To change the age of the entries to delete, add the following setting to `portal-ext.properties`:

```
studio.startup.log.cleanup.age=entryAgeInDays
```

In addition to the age-based deletions, Big Data Discovery also deletes entries associated with data sets and projects that have been deleted.

# About the System Usage page

The **System Usage** page of the **Control Panel** provides access to summary information on project usage logs.



To narrow the scope of the reports, select a time period from the pull-down menu, or set a specific start and end date.

● Last 30 Days ○ From: To:

Number of sessions over time

By: Day

The page is divided into the following sections:

| Section | Description |
|---|---|
| **Summary totals** | At the top right of the page are the total number of:<br>• Users in the system<br>• Sessions that have occurred<br>• Projects |
| **Date range fields** | Contains fields to set the range of dates for which to display report data. |
| **Current number of users and sessions** | Lists the number of users that were logged in and the number of sessions for the date range that you specify. |
| **Number of sessions over time** | Report showing the number of sessions that have been active for the date range that you specify<br>Includes a list to set the date unit to use for the chart. |
| **User Activity** | Report that initially shows the top 10 number of sessions per user for the selected date range across all projects. You can click on any bars in this chart to drill down into the reporting data.<br>At the top of the report are lists to select:<br>• A specific user, or all users<br>• A specific project, or all projects<br>• Whether to display the top or bottom values (most or least sessions)<br>• The number of values to display |

| Section | Description |
|---|---|
| **Project Usage** | Report that initially shows the top 10 number of sessions per project for the selected date range across all projects. You can click on any bars in this chart to drill down into the reporting data.<br>At the top of the report are lists to select:<br>• A specific project, or all projects<br>• Whether to display the top or bottom values (most or least sessions)<br>• The number of values to display |
| **System** | Contains a pie chart that shows the relative number of sessions by browser type and version for the selected date range. |

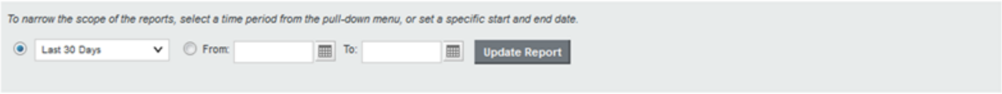
# Using the System Usage page

On the **System Usage** page, you use the fields at the top to set the date range for the report data. You can also change the displayed data on individual reports.

To use the **System Usage** page:

1. To set the date range for the displayed data on all of the reports, you can either set a time frame from the current day, or a specific range of dates.

   By default, the page is set to display data from the last 30 days.

   *To narrow the scope of the reports, select a time period from the pull-down menu, or set a specific start and end date.*

   ● Last 30 Days ○ From: To:

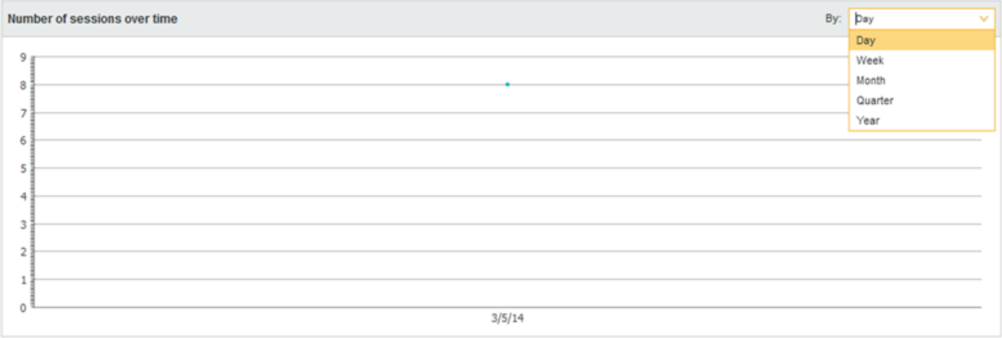
   (a) To select a different time frame, from the list, select the time frame to use.

   (b) To select a specific range of dates, click the other radio button, then in the **From** and **To** date fields, provide the start and end dates.

   (c) After selecting a time frame or range of dates, to update the reports to reflect the new selection, click **Update Report**.

2. For the **Number of sessions over time** report, you can control the date/time unit used to display the results.

   To change the date/time unit, select the new unit from the list.

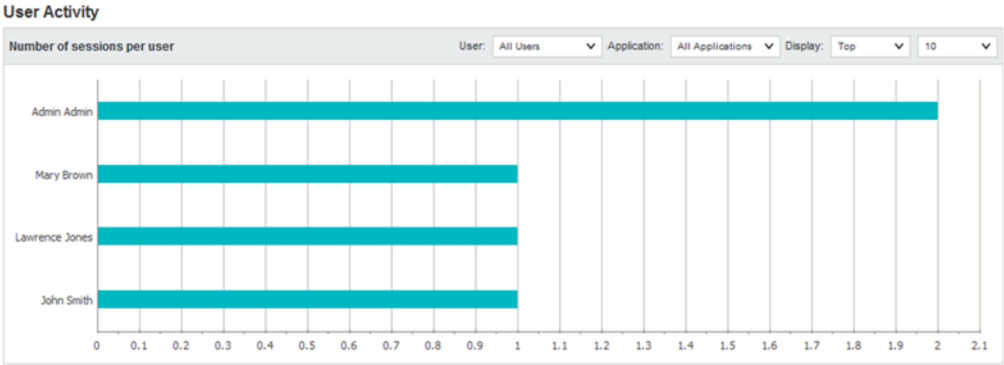


Number of sessions over time

By: Day

   The report is updated automatically to use the new value.

3. By default, the **User Activity** report shows the top 10 number of sessions per user for all projects during the selected time period.



User Activity

Number of sessions per user

User: All Users Application: All Applications Display: Top 10

   You can narrow the report to show values for a specific user or project, and change the number of values displayed.

   (a) To narrow the report to a specific user, from the **User** list, select the user.

   The report is updated to display the top or bottom number of sessions for projects the user has used.

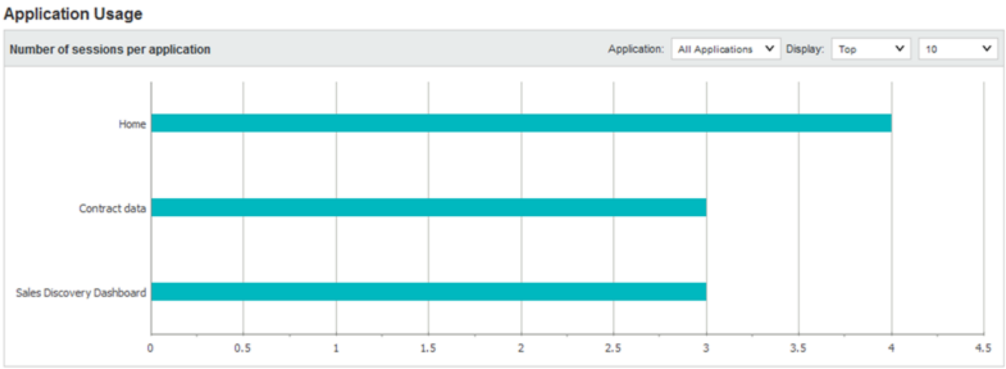
   (b) To narrow the report to a specific project, from the **Project** list, select the project.

   The report is updated to show the users with the top or bottom number of sessions for users.

   If you select both a specific project and a specific user, the report displays a single bar showing the number of sessions for that user and project.

   (c) Use the **Display** settings to control the number of values to display and whether to display the top or bottom values.

4. By default, the **Project Usage** report shows the 10 projects with the most sessions for the selected time range.



Application Usage

Number of sessions per application

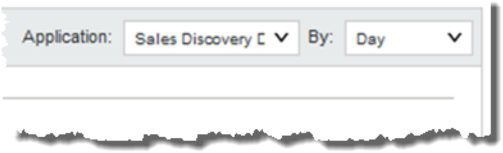
Application: All Applications
Display: Top
10

You can narrow the report to show values for a specific project, and change the number of values displayed.

(a) To narrow the report to a specific project, from the **Project** list, select the project.

The report is changed to a line chart showing the number of sessions per day for the selected project.

A date unit list is added to allow you to select the unit to use.



Application: Sales Discovery D
By: Day

For example, you can display the number of sessions per day, per week, or per month.

(b) If you are displaying the number of sessions for all projects, use the **Display** settings to control the number of values to display and whether to display the top or bottom values.

Chapter 11

# Configuring the Locale and Time Zone

The user interface of Studio and project data can be displayed in different locales and different time zones.

## Locales and their effect on the user interface

The locale determines the language in which to display the user interface. It can also affect the format of displayed data values.

Big Data Discovery is configured with a default locale as well as a list of available locales.

Each user account also is configured with a preferred locale, and the user menu includes an option for users to select the locale to use.

In Big Data Discovery, when a locale is selected:

- User interface labels display using the locale.
- Display names of attributes display in the locale.

  If there is not a version for that locale, then the default locale is used.
- Data values are formatted based on the locale.

### Supported locales

Studio supports the following languages:

- Chinese - Simplified
- English - US
- English - UK
- Japanese
- Korean
- Portuguese - Brazilian
- Spanish

Note that this is a subset of the languages supported by the Dgraph.

# How Studio determines the locale to use

When users log in, Studio determines the locale to use to display the user interface and data.

*Locations where the locale may be set*

*Scenarios for selecting the locale*

## Locations where the locale may be set

The locale is set in different locations.

The locale can come from:

- Cookie
- Browser locale
- Default locale
- User preferred locale, stored as part of the user account
- Locale selected using the **Change locale** option in the user menu, which is also available to users who have not yet logged in.

## Scenarios for selecting the locale

The locale used depends upon the type of user, the Big Data Discovery configuration, and how the user entered Big Data Discovery.

For the scenarios listed below, Big Data Discovery determines the locale as follows:

| Scenario | How the locale is determined |
|---|---|
| A new user is created | The locale for a new user is initially set to **Use Browser Locale**, which indicates to use the current browser locale.<br>This value can be changed to a specific locale.<br>If the user is configured with a specific locale, then that locale is used for the user unless they explicitly select a different locale or enter with a URL that includes a supported locale. |
| A non-logged-in user navigates to Big Data Discovery | For a non-logged-in user, Big Data Discovery first tries to use the locale from the cookie.<br>If there is no cookie, or the cookie is invalid, then Big Data Discovery tries to use the browser locale.<br>If the current browser locale is not one of the supported locales, then the default locale is used. |

| Scenario | How the locale is determined |
|---|---|
| A registered user logs in | When a user logs in, Big Data Discovery first checks the locale configured for their user account.<br>• If the user's locale is set to **Use Browser Locale**, then Big Data Discovery tries to use the locale from the cookie.<br>If there is no cookie, or if the cookie is invalid, then Big Data Discovery tries to use the browser locale.<br>If the current browser locale is not a supported locale, then the default locale is used.<br>• If the user account is configured with a locale value other than **Use Browser Locale**, then Big Data Discovery uses that locale, and also updates the cookie with that locale. |
| A non-logged-in user uses the user menu option to select a different locale | When a non-logged-in user selects a locale, Big Data Discovery updates the cookie with the new locale.<br>Note that this locale change is only applied locally. It is not applied to all non-logged-in users. |
| A logged-in user uses the user menu option to select a different locale | When a logged-in user selects a locale, Big Data Discovery updates both the user's account and the cookie with the selected locale. |

## Setting the available locales

Big Data Discovery is configured with a list of available locales. This list is used to populate the list for configuring the default locale, user default locale, and the available locales displayed for the **Change locale** option.

You can add a the setting to `portal-ext.properties` to constrain the list.

There is an implicit list of the locals that Studio supports:

```
locales=en_US, en_UK, es_ES, ja_JP, ko_KR, pt_BR, zh_CN
```

To constrain this list:

1. Log in to the machine running Studio, locate `portal-ext.properties` and open it in a text editor.
2. Copy the following `locales` parameter to a new line in the file:

   ```
   locales=en_US, en_UK, es_ES, ja_JP, ko_KR, pt_BR, zh_CN
   ```

3. Update the list to remove the locales that you do not want to be available in Studio.

   For example, to only support US English, French, and Japanese, you would update it to:

   ```
   locales=en_US, fr_FR, ja_JP
   ```

4. Save and close the file.
5. Restart Studio for the changes to take effect.

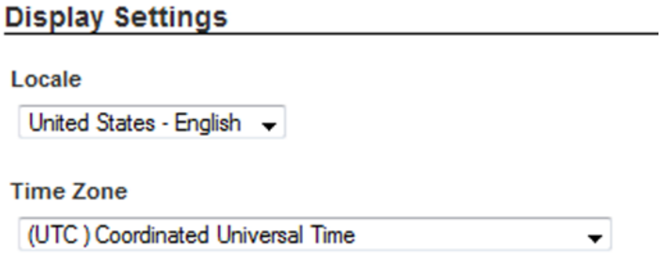
# Selecting the default locale

Studio is configured with a default locale that you can update from the **Control Panel**.

Note that if you have a clustered implementation, make sure to configure the same locale for all of the instances in the cluster.

To select the default locale:

1. From the **Control Panel**, select **Platform Settings>Display Settings**.
2. From the **Locale** list, select a default locale.



Display Settings

Locale: United States - English

Time Zone: (UTC ) Coordinated Universal Time

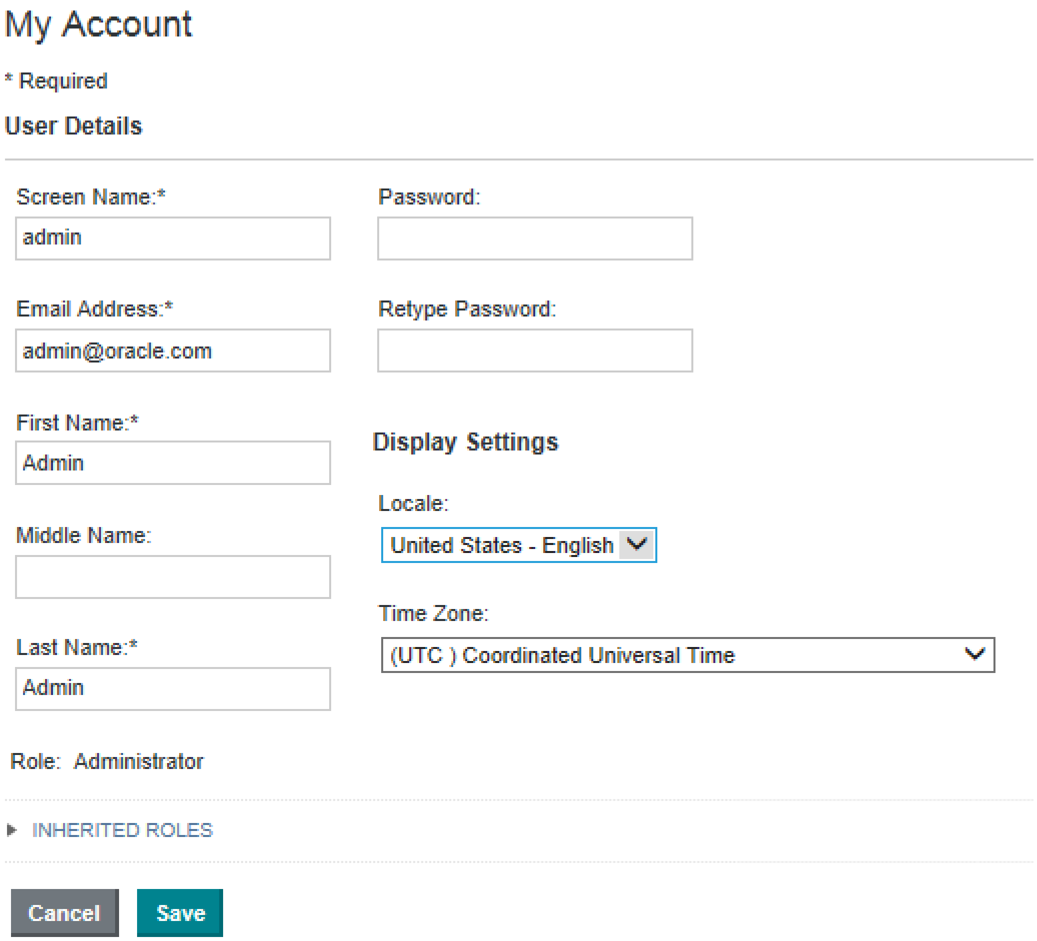
3. Click **Save**.

# Configuring a user's preferred locale

Each user account is configured with a preferred locale. The default value for new users is **Use Browser Locale**, which indicates to use the current browser locale.

To configure the preferred locale for a user:

1. To display the setting for your own account, sign in to Studio, and in the header, select **User Options>My Account**.



My Account

* Required

User Details

Screen Name:* admin

Password:

Email Address:* admin@oracle.com

Retype Password:

First Name:* Admin

Middle Name:

Last Name:* Admin

Display Settings

Locale: United States - English

Time Zone: (UTC ) Coordinated Universal Time

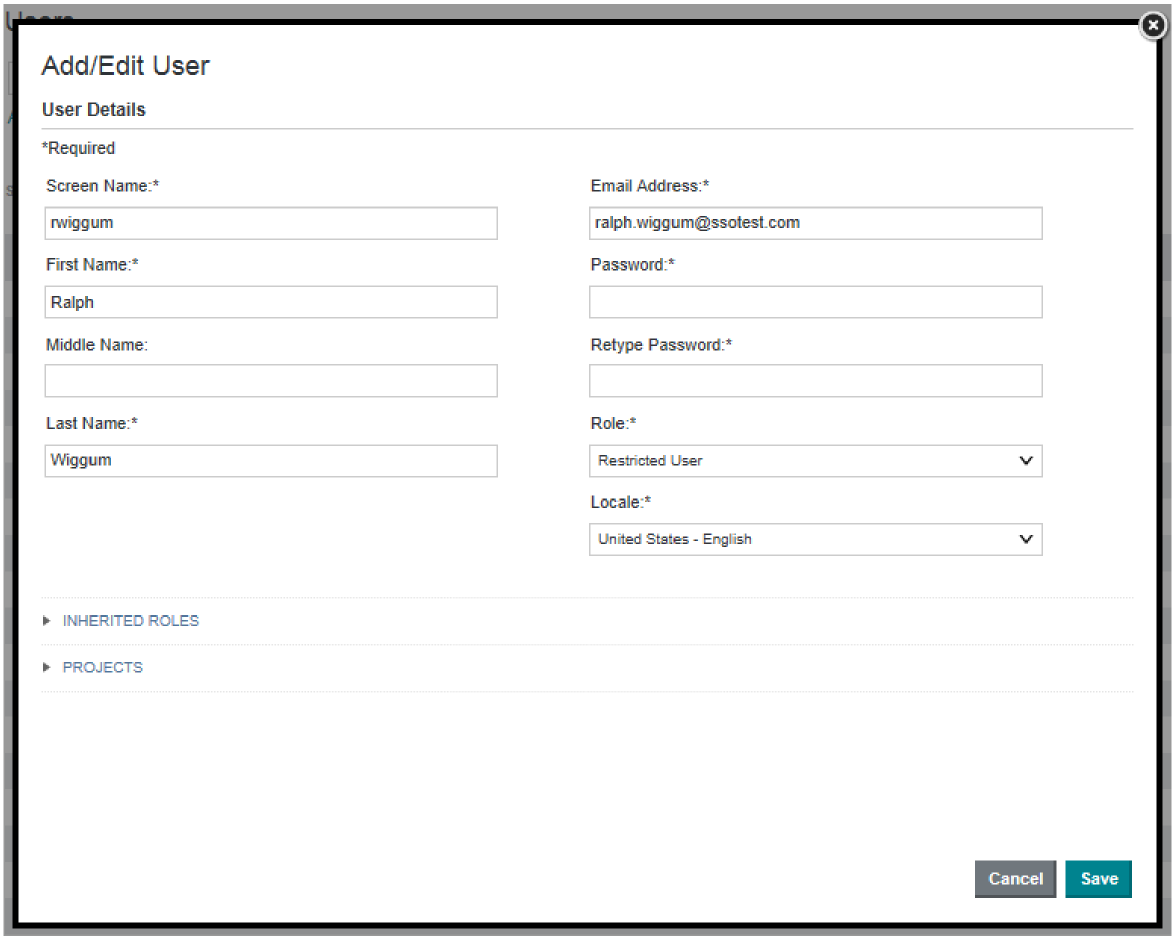
Role: Administrator

2. To display the setting for another user:

   (a) In the Big Data Discovery header, click the **Configuration Settings** icon and select **Control Panel**.

   (b) Select **User Settings>Users**.

(c) Locate the user and click **Actions>Edit**.



Add/Edit User

User Details

*Required

| | |
|---|---|
| Screen Name:* rwiggum | Email Address:* ralph.wiggum@ssotest.com |
| First Name:* Ralph | Password:* |
| Middle Name: | Retype Password:* |
| Last Name:* Wiggum | Role:* Restricted User |
| | Locale:* United States - English |

3. From the **Locale** list, select the preferred locale for the user.

4. Click **Save**.

# Setting the default time zone

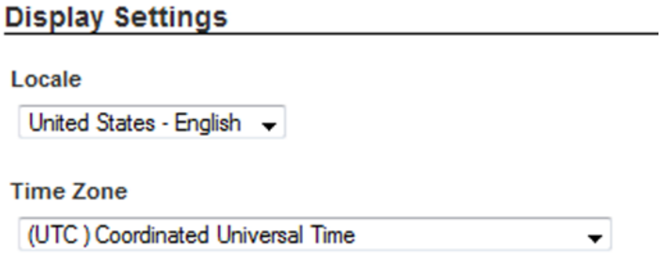
Studio is configured with a default time zone that you can update from the **Control Panel**. By default, the time zone is set to UTC. You might want to set it to your local time zone to reflect accurate time stamps in the **Notifications** panel.

Note that if you have a clustered implementation, make sure to configure the same time zone for all of the instances in the cluster.

To set the default time zone:

1. From the **Control Panel**, select **Platform Settings>Display Settings**.

2. From the **Time Zone** list, select a default time zone.



Display Settings

Locale

United States - English

Time Zone

(UTC ) Coordinated Universal Time

3. Click **Save**.

Chapter 12

# Configuring Settings for Outbound Email Notifications

Big Data Discovery includes settings to enable sending email notifications. Email notifications can include account notices, bookmarks, and snapshots.

## Configuring the email server settings

In order for users to be able to email bookmarks, you must configure the email server settings. The email address associated with the outbound server is used as the From address on the bookmark email message.

To configure the email server settings:

1. In the Big Data Discovery header, click the **Configuration Settings** icon and select **Control Panel**.
2. Select **Platform Settings>Email Settings**.
3. Click the **Sender** tab.
4. Fill out the fields for the incoming mail server:
   (a) In the **Incoming POP Server** field, enter the name of the POP server to use to receive email.

   (b) In the **Incoming Port** field, enter the port number for the POP server.

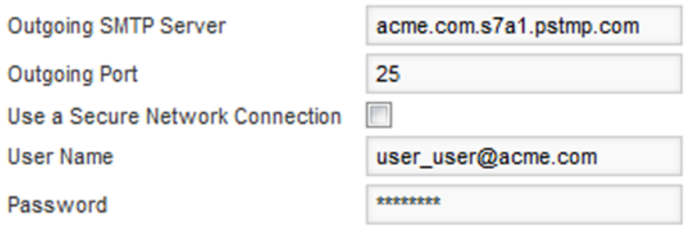
   (c) If you are not using the SMTPS mail protocol to send the email, then you must deselect the **Use a Secure Network Connection**.

   (d) In the **User Name** field, type the email address to associate with the mail server.

   This is the email address used as the **From:** address when end users email bookmarks.

   (e) In the **Password** field, type the email password associated with the email address.

5. Fill out the fields for the outbound mail server:



| | |
|---|---|
| Outgoing SMTP Server | acme.com.s7a1.pstmp.com |
| Outgoing Port | 25 |
| Use a Secure Network Connection | □ |
| User Name | user_user@acme.com |
| Password | ******** |

   (a) In the **Outgoing SMTP Server** field, enter the name of the SMTP server to use to send the email.

   (b) In the **Outgoing Port** field, enter the port number for the SMTP server.

   (c) If you are not using the SMTPS mail protocol to send the email, then the **Use a Secure Network Connection** check box must be deselected.

   (d) In the **User Name** field, type the name to display for the notification sender.

   This is the email address used as the From address when end users email bookmarks.

   (e) In the **Password** field, type the email password associated with the email address.

6. Click **Save**.

# Configuring the sender name and email address for notifications

From the **Email Settings** page of the **Control Panel**, you can configure the sender name an email address to display on outbound notifications.

To configure the sender name and email address:

1. From the **Control Panel**, select **Platform Settings>Email Settings**.
2. On the **Settings** tab, in the **Name** field, type the name to display for the notification sender.
3. In the **Address** field, type the email address to display for the notification sender. The sender address is used as the reply-to address for most notifications. For bookmarks and snapshots, the reply-to address is the email address of the user who creates the request.
4. Click **Save**.

# Setting up the Account Created and Password Changed notifications

From the **Email Settings** page of the **Control Panel**, you can configure the notifications sent when an account is created and when a user's password is changed.

These notifications only apply to users created and managed within Big Data Discovery.

The configuration includes:

- Whether to send the notification

- The subject line of the email message
- The content of the email message

To set up the Account Created and Password Changed notifications:

1. From the **Control Panel**, select **Platform Settings>Email Settings**.
2. To configure the Account Created notification:

   (a) Click the **Account Created Notification** tab.

   (b) By default, the notification is enabled, meaning that when new users are created in Big Data Discovery, they receive the notification. To disable the notification, deselect the **Enabled** check box.

   (c) In the **Subject line** field, type the text of the email subject line.

   The subject line can include any of the dynamic values listed at the bottom of the tab. For example, to include the user's Big Data Discovery screen name in the subject line, include `[$USER_SCREENNAME$]` in the subject line.

   (d) In the **Body** text area, type the text of the email message.

   The message text can include any of the dynamic values listed at the bottom of the tab. For example, to include the user's Big Data Discovery screen name in the message text, include `[$USER_SCREENNAME$]` in the message text.

   (e) To save the message configuration, click **Save**.
3. To configure the Password Changed notification:

   (a) Click the **Password Changed Notification** tab.

   (b) By default, the notification is enabled, meaning that when new users are created in Big Data Discovery, they receive the notification. To disable the notification, deselect the **Enabled** check box.

   (c) In the **Subject line** field, type the text of the email subject line.

   The subject line can include any of the dynamic values listed at the bottom of the tab. For example, to include the user's Big Data Discovery screen name in the subject line, include `[$USER_SCREENNAME$]` in the subject line.

   (d) In the Body text area, type the text of the email message.

   The message text can include any of the dynamic values listed at the bottom of the tab. For example, to include the user's Big Data Discovery screen name in the message text, include `[$USER_SCREENNAME$]` in the message text.

   (e) To save the message configuration, click **Save**.

Chapter 13

# Managing Projects from the Control Panel

The **Control Panel** provides options for Big Data Discovery administrators to configure and remove projects.

## Configuring the project type

The project type determines whether the project is visible to users on the **Catalog**.

The project types are:

| Project Type | Description |
|---|---|
| Private | • The project Creator and Studio Administrators are the only users with access<br>• The **All Big Data Discovery users** group is set to **No Access**<br>Projects are Private by default. Access must be granted by the Creator or by a Studio Administrator. |
| Public | • The **All Big Data Discovery users** group is set to **Project Restricted Users**<br>Public projects grant view access to Studio users. |
| Shared | The project has been modified in any of the following ways:<br>• Users other than the Creator are added to the project<br>• User Groups other than **All Big Data Discovery admins** and **All Big Data Discovery users** are added to the project<br>• The **All Big Data Discovery users** group is set to **Project Authors**<br>Projects are set to Shared to indicate changes from the default Public or Private permissions. |

If you change the project type, then the page visibility type for all of the project pages changes to match the project type.

To change the project type for a project:

1. In the Studio header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **User Settings>Projects**
3. Click the **Actions** link for the project, then select **Edit**
4. From the **Type** drop-down list, select the appropriate project type.

   You cannot explicitly select **Shared** as a project type. Instead, it is assigned if the default permissions have been modified.
5. Click **Save**.

# Assigning users and user groups to projects

You can manage access to projects from the **Project Settings>Sharing** page or from the project details panel in the Catalog. For details, see "Assigning project roles" in the *Studio User's Guide*.

# Certifying a project

Big Data Discovery administrators can certify a project.

Certifying a project can be used to indicate that the project content and functionality has been reviewed and the project is approved for use by all users who have access to it.

Note that only Big Data Discovery administrators can certify a project. Project Authors cannot change the certification status.

To certify a project:

1. From the **Control Panel**, select **User Settings>Projects**.
2. Click the **Actions** link for the project, then click **Edit**.
3. On the project configuration page, to certify the project, select the **Certified** check box.
4. Click **Save**.

# Making a project active or inactive

By default, a new project is marked as active. From the **Control Panel**, Big Data Discovery administrators can control whether a project is active or inactive. Inactive projects are not displayed on the **Catalog**.

Note that this option only available to Big Data Discovery administrators.

To make a project active or inactive:

1. In the Studio header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **User Settings>Projects**
3. Click the **Actions** link for the project, then click **Edit**.

4. To make the project inactive, deselect the **Active** check box. If the project is inactive, then to make the project active, check the **Active** check box.
5. Click **Save**.

# Deleting projects

From the **Control Panel**, Big Data Discovery administrators can delete projects.

To delete a project:

1. From the **Control Panel**, select **User Settings>Projects**.
2. Click the **Actions** link for the project you want to remove.
3. Click **Delete**.

# Part IV

## Controlling User Access to Studio

Chapter 14

# Configuring User-Related Settings

You configure settings for passwords and user authentication in the Studio **Control Panel**.

*Configuring authentication settings for users*

*Configuring the password policy*

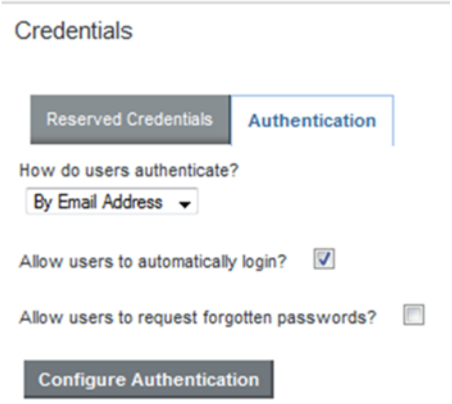
*Restricting the use of specific screen names and email addresses*

## Configuring authentication settings for users

Each user has both an email address and a screen name. By default, users log in to Studio using their email addresses.

To configure the authentication settings for users:

1. In the Studio header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **Platform Settings>Credentials** .
3. On the **Credentials** page, click the **Authentication** tab.



Credentials

How do users authenticate? By Email Address

Allow users to automatically login? ☑

Allow users to request forgotten passwords? □

4. From the **How do users authenticate?** list, select the name used to log in.

   To enables users log in using their email address, select **By Email Address**. This is the default.

   To enable users log in using their screen name, select **By Screen Name**.
5. To enable the **Remember me** option on the login page, so that login information is saved when users log in, select the **Allow users to automatically login?** check box.
6. To enable the **Forgot Your Password?** link on the login page, so that users can request a new password if they forget it, select the **Allow users to request forgotten passwords?** check box.
7. Click **Save**.

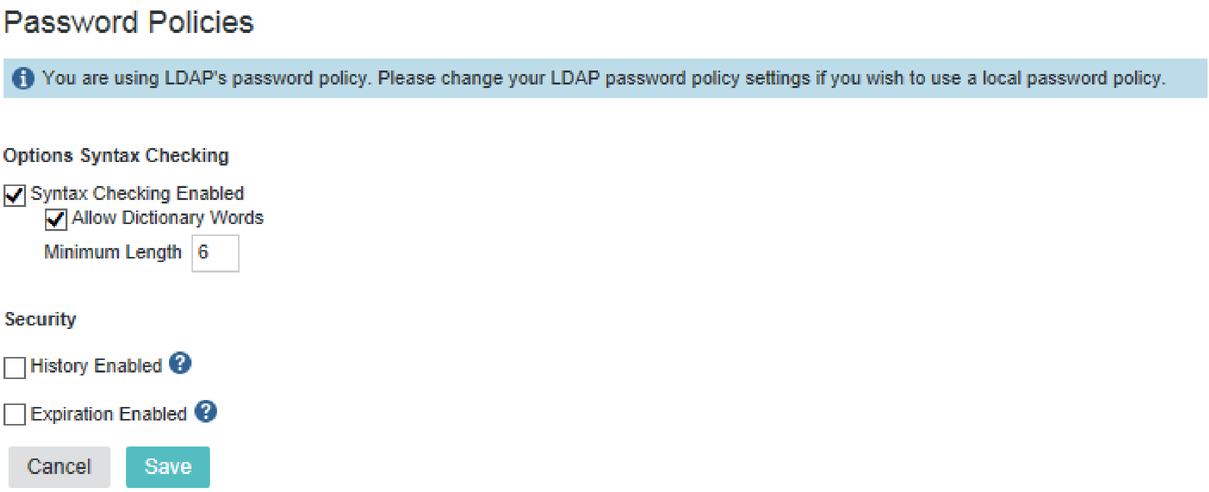
# Configuring the password policy

The password policy sets the requirements for creating and setting Studio passwords. These options do not apply to Studio passwords managed by an LDAP system.

To configure the password policy:

1. Select **Configuration Options>User Settings>Password Policies**.

   The **Password Policies** page displays.

   Password Policies

   You are using LDAP's password policy. Please change your LDAP password policy settings if you wish to use a local password policy.

   Options Syntax Checking

   ☑ Syntax Checking Enabled

   ☑ Allow Dictionary Words

   Minimum Length 6

   Security

   □ History Enabled

   □ Expiration Enabled

2. Under **Options Syntax Checking** to enable syntax checking (enforcing password requirements), select **Syntax Checking Enabled**.

   If the box is not selected, then there are no restrictions on the password format.

3. If syntax checking is enabled, then:

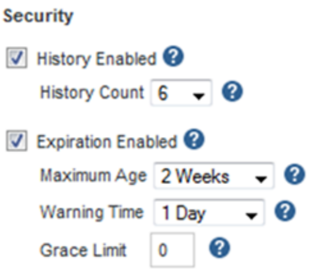
   (a) To allow passwords to include words from the dictionary, select the **Allow Dictionary Words** check box.

   If the box is not selected, then passwords cannot include words.

   (b) In the **Minimum Length** field, type the minimum length of a password.

4. To prevent users from using a recent previous password:

   (a) Under **Security**, select the **History Enabled** check box.

   Security

   ☑ History Enabled

   History Count 6

   ☑ Expiration Enabled

   Maximum Age 2 Weeks

   Warning Time 1 Day

   Grace Limit 0

   (b) From the **History Count** list, select the number of previous passwords to save and prevent the user from using.

   For example, if you select 6, then users cannot use their last 6 passwords.

5. To enable password expiration:

   (a) Select the **Expiration Enabled** check box.

   You should not enable expiration if users cannot change their passwords in Big Data Discovery.

   (b) From the **Maximum Age** list, select the amount of time before a password expires.

   (c) From the **Warning Time** list, select the amount of time before the expiration to begin displaying warnings to the user.

   (d) In the **Grace Limit** field, type the number of times a user can log in using an expired password.

6. Click **Save**.

# Restricting the use of specific screen names and email addresses

If needed, you can configure lists of screen names and email addresses that should not be used for Studio users.

To restrict the user of specific screen names and email addresses:

1. In the Studio header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **Platform Settings>Credentials** .
3. On the **Reserved Credentials** tab, in the **Screen Names** text area, type the list of screen names that cannot be used.

   Put each screen name on a separate line.

4. In the **Email Addresses** text area, type the list of email addresses that cannot be used.

   Put each email address on a separate line.

Chapter 15

# Creating and Editing Studio Users

In Studio, roles are used to control access to general features as well as to access specific projects and data. The **Users** page on the **Control Panel** provides options for creating and editing Studio users.

## About user roles and access privileges

Each Studio user is assigned a user role. The user role determines a user's access to features within Studio.

### User roles and project roles

Studio roles are divided into Studio-wide user roles and project-specific roles. The user roles are Administrator, Power User, Restricted User, and User. These roles control access to Studio features in data sets, projects, and Studio administrative configuration. The project-specific roles are Project Author and Project Restricted User. These roles control access to project-specific configuration and project data. All Studio users have a user role, and they may also have project-specific roles that have been assigned to them individually or to any of their user groups.

Administrators can assign user roles. They also have Project Author access to all projects, which allows them to assign project roles as well.

### Inherited roles

A Studio user might have a number of assigned roles. In addition to a user role, they may have a project-specific role and belong to a user group that grants additional roles. In these cases, the highest privileges apply to each area of Studio, regardless of if these privileges have been assigned directly or inherited from a user group.

## User Roles

The user roles are as follows:

| Role | Description |
|---|---|
| **Administrator** | Administrators have full access to all features in Studio.<br>Administrators can:<br>• Access the **Control Panel**<br>• Create, reload, and delete data sets and edit public metadata<br>• Modify access to all data sets<br>• Create and delete and projects<br>• Transform data within a project<br>• View, configure, and manage all projects |
| **Power User** | Power users can:<br>• Create, reload, and delete data sets and edit public metadata<br>• Modify access to any data sets they can view<br>• Create and delete and projects<br>• Transform data within a project<br>• Export data to HDFS and create new data sets<br>• View, configure, and manage projects for which they have a project role<br>• Edit their account information<br>Power users cannot:<br>• Access the **Control Panel** |
| **User** | Users can:<br>• Create and delete data sets and projects<br>• Transform data within a project<br>• View, configure, and manage projects for which they have a project role<br>• Edit their account information<br>Users cannot:<br>• Access the **Control Panel**<br>• Export data to HDFS |

| Role | Description |
|---|---|
| **Restricted User** | This is the default user role for new users. It has the most restricted privileges and is essentially a read-only role. This is the default user role for new users.<br>Restricted users can:<br>• Create new projects<br>• View data sets in the Catalog<br>• View, configure, and manage projects for which they have a project role<br>Restricted users cannot:<br>• Edit their account information<br>• Access the **Control Panel**<br>• Create new data sets<br>• Transform data within a project<br>• Export data to HDFS |

**Note:** Power Users, Users, and Restricted Users have no project roles by default, but they can access any projects that grant roles to the **All Big Data Discovery users** group. They can also access projects for which they have a project role, outlined below.

## Project Roles

Project roles grant access privileges to project content and configuration. You can assign project roles to individual users or to user groups, and they define access to a given project regardless of a user's user role in Big Data Discovery Studio. The roles are:

| Role | Description |
|---|---|
| **Project Author** | Project authors can:<br>• Configure and manage a project<br>• Add or remove users and user groups<br>• Assign user and user group roles<br>• Transform project data<br>• Export project data<br>Project authors cannot:<br>• Create new data sets<br>• Access the Big Data Discovery Control Panel |

| Role | Description |
|---|---|
| **Project Restricted User** | Project Restricted Users can:<br>• View a project and navigate through the configured pages<br>• Add and configure project pages and components<br>Project restricted users cannot:<br>• Access **Project Settings**<br>• Create new data sets<br>• Transform data<br>• Export project data |

## Data set access levels

In addition to the global feature access and project level access controlled by user roles and project roles, some deployments may require access controls at the data set level. Since data sets are a fundamental component of Big Data Discovery, this requires granting or denying access to data sets on a case-by-case basis.

**Note:** You cannot set permissions to "Default Access" or "No Access" for individual users, only for user groups.

| Access Level | Description |
|---|---|
| **No Access** (User Groups only) | The user group cannot access the data set. The data set does not show up for this user or group in the Catalog. |
| **Default Access** (User Groups only) | The user group has default access to the data set. The "default" access level is set via the `df.defaultAccessForDerivedDataSets` setting on the Studio Settings page in the Control Panel. See *Studio settings list on page 91* for more information. |
| **Read-only** | Users with Read access to a data set can<br>• See the data set in search results or by browsing the Catalog<br>• Explore the data set<br>• Add the data set to a project and modify it within the project<br>**Note:** Power Users with either Read-only or Read/Write access to a data set can also modify the data set permissions. |
| **Read/Write** | In addition to Read permissions, users with Write access to a data set can modify a data set's public metadata such as description, searchable tags, and global attribute metadata. |

Users have No Access to any data set uploaded from a file by another user; only the file uploader and Studio Administrators have access, and both have Read/Write permissionsas well as Reload/Delete/Modify permissions.

As an example of using these access levels, you may wish to restrict default data set access "Read-only" and assign the "Default Access" level to all non-Administrative user groups. This gives all users the ability to add data sets to a project and modify them there. You can then create a "Data Curators" group that has Read/Write access to data sets in order to configure attribute metadata and data set details globally to make it easier for your users to navigate the Catalog. The group effectively becomes an additional level of permissions on top of whatever other access its users have.

**Important:** A user without any access to a data set can still explore the data they are a Project Restricted User or Project Author on a project that uses the data set. Project Authors can use the Transform operations to create a duplicate data set and gain access to the new data set. Similarly, a user with Read-only access to a data set can create a project using that data set and then execute transformations against the data if the default data set permissions include Write access. If you are working with sensitive information, consider this when assigning project roles and data set permissions.

# Creating a new Studio user

If you are not using LDAP, you may want to create Studio users manually.

For example, for a small development instance, you may just need a few users to develop and test projects. Or if your LDAP users for a production site are all end users, you may need a separate user account for administering the site.

To create a new Studio user:

1. In the Studio header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **User Settings>Users** .
3. Click **Add**.

   The **Details** page for the new user displays.
4. In the **Screen Name** field, type the screen name for the user.

   The screen name must be unique, and cannot match the screen name of any current active or inactive user.
5. In the **Email Address** field, type the user's email address.
6. For the user's name, enter values for at least the **First Name** and **Last Name** fields.

   The **Middle Name** field is optional.
7. To create the initial password for the user:

   (a) In the **Password** field, enter the password to assign to the new user.

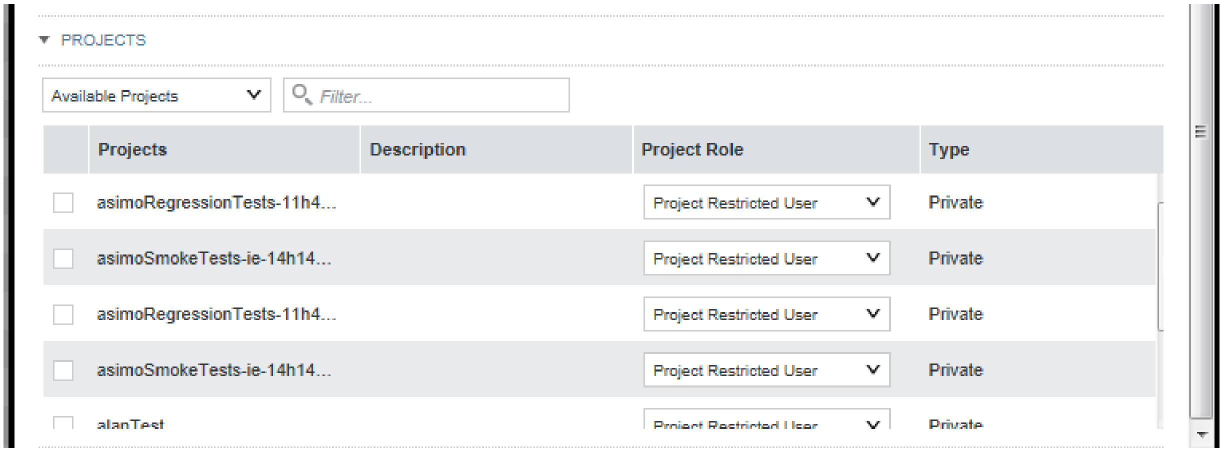
   (b) In the **Retype Password** field, type the password again.

   By default, the Studio password policy requires users to change their password the first time they log in.
8. From the **Locale** list, select the preferred locale for the user.

9. From the **Role** list, select the user role to assign to the user.

   For details, see *About user roles and access privileges on page 121*.

10. From the **Projects** section at the bottom of the dialog, to assign the user to projects:



|  | Projects | Description | Project Role | Type |
|---|---|---|---|---|
| □ | asimoRegressionTests-11h4... |  | Project Restricted User | Private |
| □ | asimoSmokeTests-ie-14h14... |  | Project Restricted User | Private |
| □ | asimoRegressionTests-11h4... |  | Project Restricted User | Private |
| □ | asimoSmokeTests-ie-14h14... |  | Project Restricted User | Private |
| □ | alanTest |  | Project Restricted User | Private |

   (a) Select the check box next to each project you want the new user to be a member of.

   (b) For each project, from the **Role** list, select the project role to assign to the user.

11. Click **Save**.

    The user is added to the list of users.

# Editing a Studio user

The **Users** page also allows you to edit a user's account.

From the **Users** page, to edit a user:

1. In the Studio header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **User Settings>Users**
3. Click the **Actions** button next to the user.
4. Click **Edit**.
5. To change the user's password:

   (a) In the **Password** field, type the new password.

   (b) In the **Retype Password** field, re-type the new password.
6. To change the user role, from the **Role** list, select the new role.
7. Under **Projects**, to add a user as an project member:

   (a) Make sure the list is set to **Available Projects**. These are projects the user is not yet a member of.

   (b) Select the check box next to each project you want to add the user to.

   (c) For each project, from the **Role** list, select the project role to assign to the user.

8. Under **Projects**, to change the project role for or remove the user from a project:

   (a) From the list, select **Assigned Projects**.

   The list shows the projects the user is currently a member of.

   (b) To change the user's project role, from the **Role** drop-down list, select the new project role.

   (c) To remove the user from a project, deselect the check box.

9. Click **Save**.

# Deactivating, reactivating, and deleting Studio users

From the **Users** page of the **Control Panel**, you can make an active user inactive. You can also reactivate or delete inactive users.

Note that you cannot make your own user account inactive, and you cannot delete an active user.

From the **Users** page, to change the status of a user account:

1. To make an existing user inactive:

   (a) In the users list, select the check box for the user you want to deactivate.

   (b) Click **Deactivate**.

   Big Data Discovery prompts you to confirm that you want to deactivate the user.

   The user is then removed from the list of active users.

   Note that inactive users are not removed from Big Data Discovery.

2. To reactivate or delete an inactive user:

   (a) Click the **Advanced** link below the user search field.

   Big Data Discovery displays additional user search fields.

   (b) From the **Active** list, select **No**.

   Note that if you change the **Match type** to **Any**, you must also provide search criteria in at least one of the other fields.

   (c) Click **Search**.

   The users list displays only the inactive users.

   (d) Select the check box for the user you want to reactivate or delete.

   (e) To reactivate the user, click **Restore**.

   (f) To delete the user, click **Delete**.

Chapter 16

# Integrating with an LDAP System to Manage Users

If you have an LDAP system, you can allow users to use those credentials to log in to Big Data Discovery.

## About using LDAP

Integrating Studio with Lightweight Directory Access Protocol (LDAP) allows users to sign in to Studio using their existing LDAP user accounts, rather than creating separate user accounts from within Studio. LDAP is also used when integrating with a single sign-on (SSO) system.

You can integrate Studio with one LDAP directory but not multiple LDAP directories.

Users in LDAP must be contained in LDAP groups for Studio to properly map roles and permissions.

You can set up mixed authentication systems with both LDAP and manually created Studio users. In such a scenario, Studio pulls users and groups from an LDAP directory, and you can supplement those LDAP users with additional Studio users that you create.

If Studio uses LDAP for user management, you are notified in a blue banner across the **Password Policies** page. In this scenario, Studio relies entirely on the LDAP system for user names, passwords, syntax checking, minimum length settings, and so on. The settings on the **Password Policies** page do not apply to your LDAP users. However, if you create users directly in Studio, you can modify some basic settings about the password configuration on the **Password Policies** page.

## Configuring the LDAP settings and server

The LDAP settings on the **Control Panel>Credentials** page include whether LDAP is enabled and required for authentication, the connection to the LDAP server, and whether to support batch import or export to or from the LDAP directory. The method for processing batch imports is set in `portal-ext.properties`.

In `portal-ext.properties`, the setting `ldap.import.method` determines how to perform batch imports from LDAP. This setting is only applied if batch import is enabled. The available values for `ldap.import.method` are:

| Value | Description |
|---|---|
| `user` | Specifies a user-based import. This is the default value.<br>User-based batch import uses the import search filter configured in the **User Mapping** section of the **LDAP** tab.<br>For user-first import, Big Data Discovery:<br>1. Uses the user import search filter to run an LDAP search query.<br>2. Imports the resulting list of users, including all of the LDAP groups the user belongs to.<br>The group import search filter is ignored. |
| `group` | Specifies a group-based import.<br>Group-based import uses the import search filter configured in the **Group Mapping** section of the **LDAP** tab.<br>For group-based import, Big Data Discovery:<br>1. Uses the group import search filter to run an LDAP search query.<br>2. Imports the resulting list of groups, including all of the users in those groups.<br>The user import search filter is ignored. |

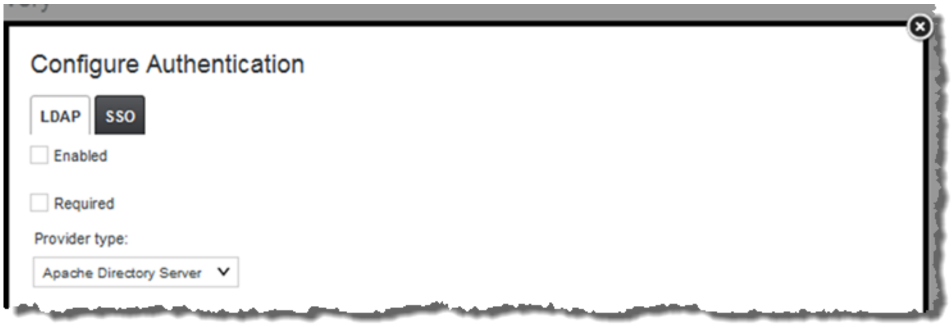
The value you should use depends partly on how your LDAP system works. If your LDAP directory only provides user information, without any groups, then you have to use user-based import. If your LDAP directory only provides group information, then you have to use group-based import.

To configure the LDAP settings:

1. In the Big Data Discovery header, click the **Configuration Options** icon and select **Control Panel**.
2. Click **Credentials>Authentication**

3. Click the **Configure Authentication** button.

   The **Configure Authentication** dialog displays, with the **LDAP** tab selected.



Configure Authentication

□ Enabled

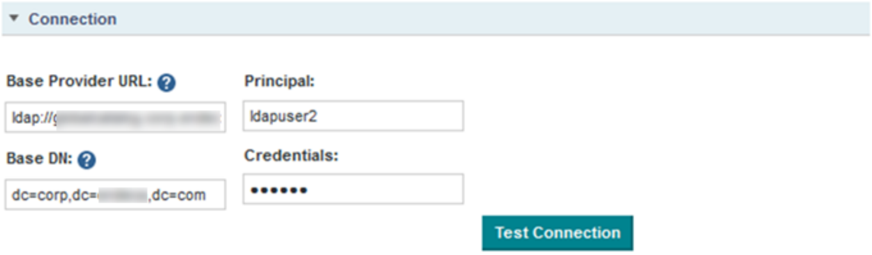
□ Required

Provider type: Apache Directory Server

4. To enable LDAP authentication, select the **Enabled** check box.
5. To only allow users to log in using an LDAP account, select the **Required** check box.

   If this box is selected, then any users that you create manually in Big Data Discovery cannot log in. To make sure that users you create manually can log in, make sure that this box is deselected.
6. To populate the LDAP server configuration fields with default values based on a specific type of provider, select the type of server you are using from the **Provider type** list.

   If you select the **Custom** option, then the fields are cleared.
7. The **Connection** settings cover the basic connection to LDAP:



Base Provider URL: ldap://[redacted]

Principal: ldapuser2

Base DN: dc=corp,dc=[redacted],dc=com

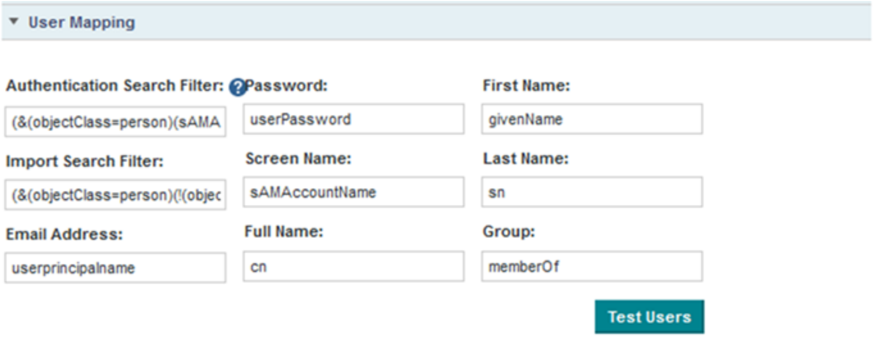
Credentials: ••••••

| Field | Description |
|---|---|
| **Base Provider URL** | The location of your LDAP server.<br>Make sure that the machine on which Big Data Discovery is installed can communicate with the LDAP server.<br>If there is a firewall between the two systems, make sure that the appropriate ports are opened. |
| **Base DN** | The Base Distinguished Name for your LDAP directory.<br>For a commercial organization, it may look something like:<br>`dc=companynamehere,dc=com` |

| Field | Description |
|---|---|
| **Principal** | The user name of the administrator account for your LDAP system. The principal must be a user distinguished name (DN), for example:<br>`CN=bddldap,OU=Service Accounts,DC=company,DC=com`<br>This ID is used to synchronize user accounts to and from LDAP. |
| **Credentials** | The password for the administrative user. |

After providing the connection information, to test the connection to the LDAP server, click **Test Connection**.

8. Under **User Mapping**:



Authentication Search Filter: (&(objectClass=person)(sAMA
Password: userPassword
First Name: givenName
Import Search Filter: (&(objectClass=person)(!(objec
Screen Name: sAMAccountName
Last Name: sn
Email Address: userprincipalname
Full Name: cn
Group: memberOf

(a) Use the search filter fields to configure the filters for finding and identifying users in your LDAP directory.

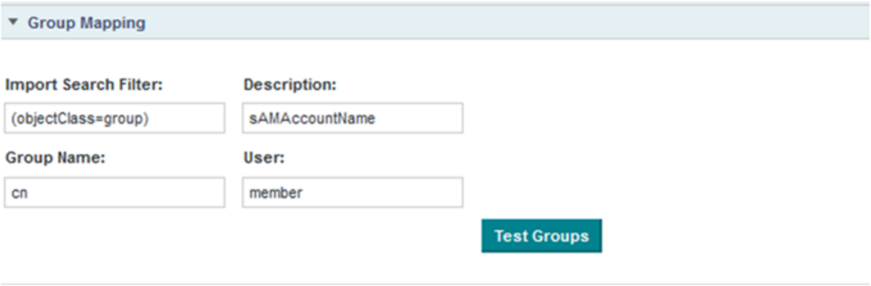
| Field | Description |
|---|---|
| **Authentication Search Filter** | The search criteria for user logins.<br>If you do not enable batch import of LDAP users, then the first time a user tries to log in, Big Data Discovery uses this authentication search filter to search for the user in the LDAP directory.<br>By default, users log in using their email address. If you have changed this setting, you must modify the search filter here.<br>For example, if you changed the authentication method to use the screen name, you would modify the search filter so that it can match the entered login name:<br>`(cn=@screen_name@)` |

| Field | Description |
|---|---|
| **Import Search Filter** | The search filter to use for batch import of users.<br>This filter is used if:<br>• You enable batch import of LDAP users<br>• In `portal-ext.properties`, `ldap.import.method` is set to `user`<br>Depending on the LDAP server, there are different ways to identify the user.<br>The default setting (`objectClass=inetOrgPerson`) usually is fine, but to search for only a subset of users or for users that have different object classes, you can change this. |

(b) Use the remaining fields to map your LDAP attributes to the Big Data Discovery user fields.

(c) After setting up the attribute mappings, to test the mappings, click **Test Users**.

9. Under **Group Mapping**, map your LDAP groups.



Import Search Filter: (objectClass=group)

Description: sAMAccountName

Group Name: cn

User: member

(a) In the **Import Search Filter** field, type the filter for finding LDAP groups.

This filter is used if:

- You enable batch import of LDAP users
- In `portal-ext.properties`, `ldap.import.method` is set to `group`

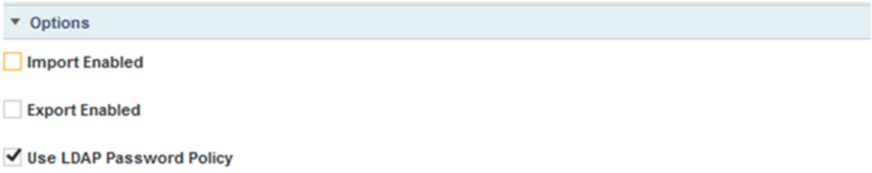
(b) Map the following group fields:

- Group Name
- Description
- User

(c) To test the group mappings, click **Test Groups**.

The system displays a list of the groups returned by your search filter.

10. The **Options** section is used to configure importing and exporting of LDAP user data and to select the password policy:



▾ Options

□ Import Enabled

□ Export Enabled

☑ Use LDAP Password Policy

   (a) If you selected the **Import Enabled** check box, then batch import of LDAP users is enabled.

   If you did not select this box, then Big Data Discovery synchronizes each user as they log in. It is recommended that you leave this box deselected.

   If you do enable batch import, then the import process is based on the value of `ldap.import.method`.

   Note also that when using batch import, you cannot filter both the imported users and imported groups at the same time. For user-based batch import mode, you cannot filter the LDAP groups to import. For group-based batch import mode, you cannot filter the LDAP users to import.

   (b) If the **Export Enabled** check box is selected, then any changes to the user in Big Data Discovery are exported to the LDAP system.

   It is recommended that you leave this box deselected.

   (c) To use the password policy from your LDAP system, instead of the Big Data Discovery password policy, select the **Use LDAP Password Policy** check box.

# Preventing encrypted LDAP passwords from being stored in BDD

By default, when you use LDAP for user authentication, each time a user logs in, Big Data Discovery stores a securely encrypted version of their LDAP password. For subsequent logins, Big Data Discovery can then authenticate the user even when it cannot connect to the LDAP system. For even stricter security, you can configure Big Data Discovery to prevent the passwords from being stored.

To prevent Big Data Discovery from storing the encrypted LDAP passwords:

1. Stop Big Data Discovery.
2. Add the following settings to `portal-ext.properties`:

```
ldap.password.cache.hashed=false
ldap.auth.required=true
auth.pipeline.enable.liferay.check=false
```

3. Restart Big Data Discovery.

Big Data Discovery no longer stores the encrypted LDAP passwords for authenticated users. If the LDAP system is unavailable, Big Data Discovery cannot authenticate previously authenticated users.

# Assigning roles based on LDAP user groups

For LDAP integration, it is recommended that you assign roles based on your LDAP groups.

To ensure that users have the correct roles as soon as they log in, you create groups in Big Data Discovery that have the same name as your LDAP groups, but in lowercase, and assign the correct roles to each group.

To create a user group, and assign roles to that group:

1. In the Big Data Discovery header, click the **Configuration Options** icon and select **Control Panel**.
2. Select **User Settings>User Groups**.
3. On the **User Groups** page, to add a new group, click **Add**.

   The **Add Group** dialog displays.
4. In the **Name** field, type the name of the group.

   Make sure the name is the lowercase version of the name of a group from your LDAP system. For example, if the LDAP group is called `SystemUsers`, then the user group name would be `systemusers`.
5. In the **Description** field, type a description of the group.
6. To assign roles to the group, from the **Role** list, select the user role to assign to the group.

   The selected roles are assigned to all of the users in the group. For details on the available user roles, see *About user roles and access privileges on page 121*.
7. Click **Save**.

   The group is added to the **User Groups** list.

Chapter 17

# Setting Up Single Sign-On (SSO)

You can provide user access by integrating with an SSO system.

## About using single sign-on

Integrating with single sign-on (SSO) allows Studio users to be logged in to Big Data Discovery automatically once they are logged in to your SSO system.

Note that once Big Data Discovery is integrated with SSO, you cannot create and edit users from within Big Data Discovery. All users get access to Big Data Discovery using their SSO credentials. This means that you can no longer use the default administrative user provided with Big Data Discovery. You will need to make sure that there is at least one SSO user with an Administrator user role for Big Data discovery.

The officially supported method for integrating with SSO is to use Oracle Access Manager, with an Oracle HTTP Server in front of the Big Data Discovery application server. While you may be able to use another SSO tool that supports passing the user name in an HTTP header, you would have to use the documentation and support materials for that tool in order to set up the integration.

The information in this guide focuses on the details and configuration that are specific to the Big Data Discovery integration. For general information on installing Oracle Access Manager and Oracle HTTP Server, see the associated documentation for those products.

## Overview of the process for configuring SSO with Oracle Access Manager

Here is an overview of the steps for using Oracle Access Manager to implement SSO in Big Data Discovery.

1. Install Oracle Access Manager 11g, if you haven't already. See the Oracle Access Manager documentation for details.
2. Install Oracle HTTP Server (OHS) 11g. See the Oracle HTTP Server documentation for details.
3. Install OHS Webgate 11g. See the Webgate documentation for details.

4. Create an instance of OHS and confirm that it is up and running. See the OHS documentation for details.
5. Configure the reverse proxy module for the Big Data Discovery application server in Oracle HTTP Server. See *Configuring the reverse proxy module in OHS on page 136*.
6. Install the Webgate module into the Oracle HTTP Server. See *Registering the Webgate with the Oracle Access Manager server on page 137*.
7. In Big Data Discovery, configure the LDAP connection for your SSO implementation. See *Configuring the LDAP connection for SSO on page 139*.
8. In Big Data Discovery, configure the Oracle Access Manager SSO settings. See *Configuring the Oracle Access Manager SSO settings on page 140*.
9. Configure Big Data Discovery's web server settings to use the OHS server. See *Completing and testing the SSO integration on page 141*.
10. Disable direct access to the Big Data Discovery application server, to ensure that all traffic to Big Data Discovery is routed through OHS.

# Configuring the reverse proxy module in OHS

For WebLogic Server, you need to update the file `mod_wl_ohs.conf` to add the logout configuration for SSO.

Here is an example of the file with the `/bdd/oam_logout_success` section added:

```
LoadModule weblogic_module   "${ORACLE_HOME}/ohs/modules/mod_wl_ohs.so"
<IfModule weblogic_module>
      WebLogicHost hostName
      WebLogicPort portNumber
</IfModule>

<Location /bdd/oam_logout_success>
      PathTrim /bdd/oam_logout_success
      PathPrepend /bdd/c/portal
      DefaultFileName logout
      SetHandler weblogic-handler
</Location>

<Location />
      SetHandler weblogic-handler
</Location>
```

The `/bdd/oam_logout_success Location` configuration is special for Big Data Discovery. It redirects the default Webgate Logout Callback URL (`/bdd/oam_logout_success`) to an application tier logout within Big Data Discovery. With this configuration, when users sign out of SSO from another application, it is reflected in Big Data Discovery.

# Registering the Webgate with the Oracle Access Manager server

After you have installed the OHS Webgate, you use the remote registration (RREG) tool to register the OHS Webgate with the OAM server.

To complete the registration:

1. Obtain the RREG tarball (`rreg.tar.gz`) from the Oracle Access Manager server.
2. Extract the file to the OHS server.
3. Modify the script `oamreg.sh`.

   Correct the `OAM_REG_HOME` and `JAVA_HOME` environment variables.

   `OAM_REG_HOME` should point to the extracted `rreg` directory created in the previous step.

   You may not need to change `JAVA_HOME` if it's already set in your environment.
4. In the `input` directory, create an input file for the RREG tool. The file can include the list of resources secured by this Webgate.

   You can omit this list if the application domain already exists.

   Here is an example of an input file where the resources have not been set up for the application domain and host in Oracle Access Manager:

   ```
   <?xml version="1.0" encoding="UTF-8"?>

   <OAM11GRegRequest>

   <serverAddress>http://oamserver.us.mycompany.com:7001</serverAddress>
   <hostIdentifier>myserver-1234</hostIdentifier>
   <agentName>myserver-1234-webgate</agentName>
   <applicationDomain>Big Data Discovery</applicationDomain>
   <protectedResourcesList>
     <resource>/bdd</resource>
     <resource>/bdd/.../*</resource>
   </protectedResourcesList>
   <publicResourcesList>
     <resource>/public/index.html</resource>
   </publicResourcesList>
   <excludedResourcesList>
     <resource>/excluded/index.html</resource>
   </excludedResourcesList>

   </OAM11GRegRequest>
   ```

   In this example, the resources have already been set up in Oracle Access Manager:

   ```
   <?xml version="1.0" encoding="UTF-8"?>

   <OAM11GRegRequest>

   <serverAddress>http://oamserver.us.mycompany.com:7001</serverAddress>
   <hostIdentifier>myserver-1234</hostIdentifier>
   <agentName>myserver-1234-webgate</agentName>
   <applicationDomain>Big Data Discovery</applicationDomain>

   </OAM11GRegRequest>
   ```

In the input file, the parameter values are:

| Parameter Name | Description |
|---|---|
| `serverAddress` | The full address (`http://`*`host`*`:`*`port`*) of the Oracle Access Manager administrative server.<br>The port is usually 7001. |
| `hostIdentifier` | The host identifier string for your host.<br>If you already created a host identifier in the Oracle Access Manager console, use its name here. |
| `agentName` | A unique name for the new Webgate agent.<br>Make sure it doesn't conflict with any existing agents in the application domain. |
| `applicationDomain` | A new or existing application domain to add this agent into.<br>Each application domain may have multiple agents.<br>An application domain associates multiple agents with the same authentication and authorization policies. |

5. Run the tool:

   ```
   ./bin/oamreg.sh inband input/inputFileName
   ```

   For example:

   ```
   ./bin/oamreg.sh inband input/my-webgate-input.xml
   ```

   When the process is complete, you'll see the following message:

   ```
   Inband registration process completed successfully! Output artifacts are created in the
   output folder.
   ```

6. Copy the generated output files from the `output` directory to the OHS instance `config` directory (under `webgate/config/`).
7. Restart the OHS instance.
8. Test your application URL via OHS.

   It should forward you to the SSO login form.

   Check the OAM console to confirm that the Webgate is installed and has the correct settings.

## Testing the OHS URL

Before continuing to the Big Data Discovery configuration, you need to test that the OHS URL redirects correctly to Big Data Discovery.

To test the OHS URL, use it to browse to Big Data Discovery.

You should be prompted to authenticate using your SSO credentials.

Because you have not yet configured the Oracle Access Manager SSO integration in Big Data Discovery, after you complete the authentication, the Big Data Discovery login page displays.

Log in to Big Data Discovery using an administrator account.

# Configuring Big Data Discovery to integrate with SSO via Oracle Access Manager

In Big Data Discovery, you configure the LDAP connection and Oracle Access Manager connection settings.

*Configuring the LDAP connection for SSO*

*Configuring the Oracle Access Manager SSO settings*

## Configuring the LDAP connection for SSO

The SSO implementation uses LDAP to retrieve and maintain the user information. For the Oracle Access Manager SSO, you configure Big Data Discovery to use Oracle Internet Directory for LDAP.

In Big Data Discovery, to configure the LDAP connection for SSO:

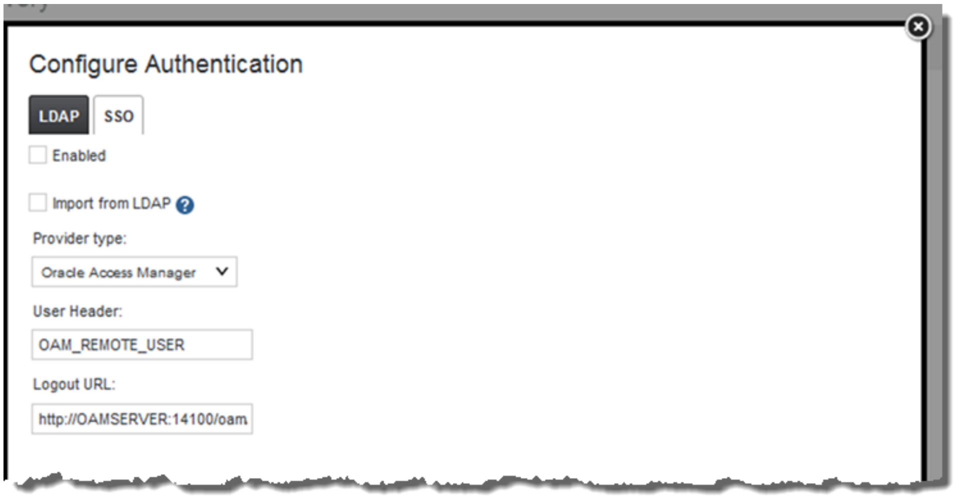
1. From the **Control Panel**, select **Platform Settings>Credentials**.
2. On the **Credentials** page, click **Authentication**.
3. On the **Authentication** tab, click the **Configure Authentication** button.

   The **Configure Authentication** dialog is displayed, with the **LDAP** tab selected.
4. On the **LDAP** tab, check the **Enabled** check box. Do not check the **Required** check box.
5. From the **Provider type** drop-down list, select **Oracle Internet Directory**.
6. Configure the LDAP connection, users, and groups as described in *Configuring the LDAP settings and server on page 129*.
7. Configure the user roles for your user groups as described in *Assigning roles based on LDAP user groups on page 134*.
8. To save the LDAP connection information, click **Save**.

## Configuring the Oracle Access Manager SSO settings

After you configure the LDAP connection for your SSO integration, you configure the Oracle Access Manager SSO settings.

The settings are on the **SSO** tab on the **Configure Authentication** dialog.



□ Enabled

□ Import from LDAP

Provider type: Oracle Access Manager

User Header: OAM_REMOTE_USER

Logout URL: http://OAMSERVER:14100/oam

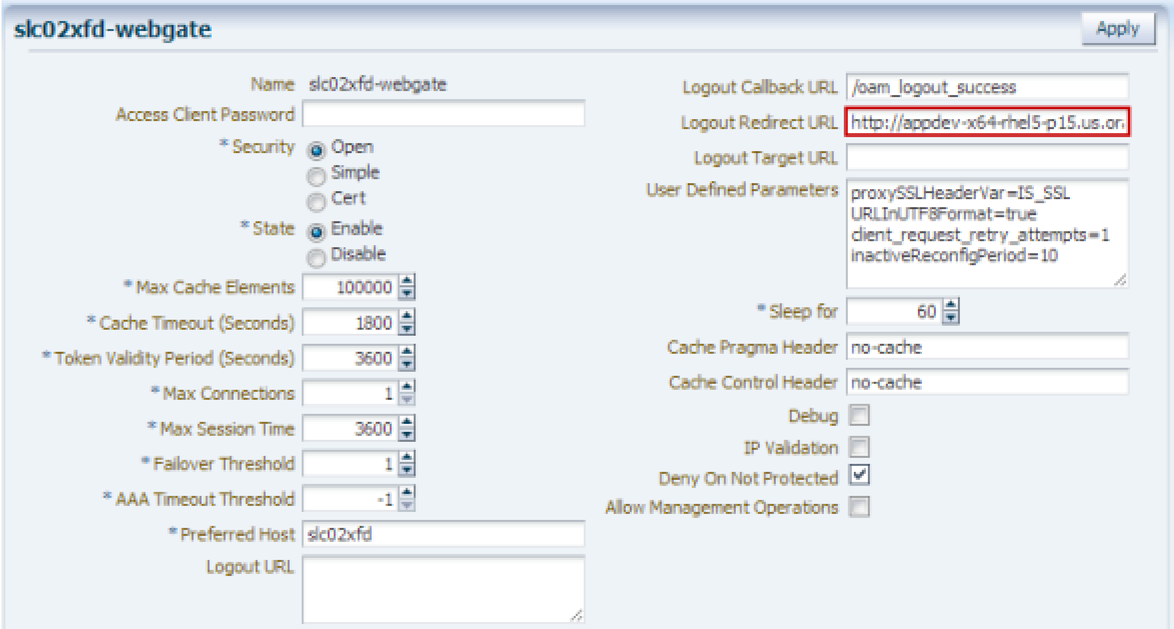
To configure the SSO settings:

1. From the **Control Panel**, select **Platform Settings>Credentials**.
2. In the **Credentials** page, click **Authentication**.
3. On the **Authentication** tab, click **Configure Authentication**.
4. On the **Configure Authentication** dialog, click **SSO**.
5. Select the **Enabled** check box.
6. Select the **Import from LDAP** check box.
7. From the **Provider Type** list, select **Oracle Access Manager**.

   Note that the only other option is **Custom**, which clears the fields. You would use the **Custom** option if you are using some other tool that passes the user name in an HTTP header. For information on setting up an SSO tool other than Oracle Access Manager, see the documentation and support materials for that tool.
8. Leave the default user header `OAM_REMOTE_USER`.

9. In the **Logout URL** field, provide the URL to navigate to when users log out.

   Make sure it is the same logout redirect URL you have configured for the Webgate:



slc02xfd-webgate

Name: slc02xfd-webgate
Access Client Password:
* Security: ● Open ○ Simple ○ Cert
* State: ● Enable ○ Disable
* Max Cache Elements: 100000
* Cache Timeout (Seconds): 1800
* Token Validity Period (Seconds): 3600
* Max Connections: 1
* Max Session Time: 3600
* Failover Threshold: 1
* AAA Timeout Threshold: -1
* Preferred Host: slc02xfd
Logout URL:

Logout Callback URL: /oam_logout_success
Logout Redirect URL: http://appdev-x64-rhel5-p15.us.or
Logout Target URL:
User Defined Parameters:

```
proxySSLHeaderVar=IS_SSL
URLInUTF8Format=true
client_request_retry_attempts=1
inactiveReconfigPeriod=10
```

* Sleep for: 60
Cache Pragma Header: no-cache
Cache Control Header: no-cache
Debug: □
IP Validation: □
Deny On Not Protected: ☑
Allow Management Operations: □

   For the logout URL, you can add an optional `end_url` parameter to redirect the browser to a final location after users sign out. To redirect back to Big Data Discovery, configure `end_url` to point to the OHS host and port.

   For example:

```
http://oamserver.us.mycompany.com:14100/oam/server/logout?end_url=http:/
/bddhost.us.company.com:7777/
```

10. To save the configuration, click **Save**.

# Completing and testing the SSO integration

The final step in setting up the SSO integration is to add the OHS server host name and port to `portal-ext.properties`.

To complete and test the SSO configuration:

1. In `portal-ext.properties`:

   If OHS is not using SSL, then add the following lines:

```
web.server.host=ohsHostName
web.server.http.port=ohsPortNumber
```

   If OHS is using SSL, then add the following lines:

```
web.server.protocol=https
web.server.host=ohsHostName
web.server.https.port=ohsPortNumber
```

Where:

- `ohsHostName` is the fully qualified domain name (FQDN) of the server where OHS is installed. The name must be resolvable by Big Data Discovery users.

  For example, you would use `webserver01.company.com`, and not `webserver01`.

  You need to specify this even if OHS is on the same server as Big Data Discovery.
- `ohsPortNumber` is the port number used by OHS.

2. Restart Big Data Discovery.

   Make sure to completely restart the browser to remove any cookies or sessions associated with the Big Data Discovery user login you used earlier.
3. Navigate to the Big Data Discovery URL. The Oracle Access Manager SSO form displays.
4. Enter your SSO authentication credentials.

   You are logged in to Big Data Discovery.

   As you navigate around Big Data Discovery, make sure that the browser URL continues to point to the OHS server and port.

# Part V

## Logging for Studio, Dgraph, and Dgraph Gateway

Chapter 18

# Overview of BDD Logging

This topic provides a logging overview of the BDD components.

*List of Big Data Discovery logs*

*Gathering information for diagnosing problems*

*Retrieving logs*

*Rotating logs*

## List of Big Data Discovery logs

This topic provides a list of all the logs generated by a BDD deployment.

The list also includes a summary of where to find logs for each BDD component and tells you how to access logs.

### List of BDD logs

| Log | Purpose | Default Location |
|---|---|---|
| WebLogic Admin Server domain log | Provides a status of the WebLogic domain for the Big Data Discovery deployment. See *Dgraph Gateway logs on page 167*. | `$BDD_DOMAIN/servers/AdminServer/logs/bdd_domain.log` |
| WebLogic Admin Server server log | Contains messages from the WebLogic Admin Server subsystems. For both server logs, see *Dgraph Gateway logs on page 167*. | `$BDD_DOMAIN/servers/AdminServer/logs/AdminServer.log` |
| WebLogic Managed Server server log | Contains messages from the WebLogic Managed Server subsystems and applications. | `$BDD_DOMAIN/servers/<serverName>/logs/<serverName>.log` |
| Dgraph Gateway application log | WebLogic log for the Dgraph Gateway application. See *Dgraph Gateway log entry format on page 169* | `$BDD_DOMAIN/servers/<serverName>/logs/<serverNamem>-diagnostic.log` |
| Dgraph stdout/stderr log | Contains Dgraph operational messages, including startup messages. See *Dgraph out log on page 161*. | `$BDD_HOME/logs/dgraph.out` |

| Log | Purpose | Default Location |
|---|---|---|
| Dgraph request log | Contains entries for Dgraph requests. See *Dgraph request log on page 159*. | `$BDD_HOME/dgraph/bin/dgraph.reqlog` |
| Dgraph tracing ebb logs | Dgraph Tracing Utility files, which are especially useful for Dgraph crashes. See *get-blackbox on page 40*. | `$BDD_HOME/dgraph/bin/dgraph-<serverName>-*.ebb` |
| Dgraph HDFS Agent stdout/stderr log | Contains startup messages, as well as messages from operations performed by the Dgraph HDFS Agent (such as ingest operations). See the *Data Processing Guide*. | `$BDD_HOME/logs/dgraphHDFSAgent.out` |
| FUSE stdout/stderr log | Contains FUSE operational messages. See *FUSE out log on page 165*. | `$BDD_HOME/logs/hdfs_fuse_client.out` |
| Studio application log in Log4j format | Studio application log (in Log4j format). For both Studio application logs, see *About the main Studio log file on page 153*. | `$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio.log` |
| Studio application log in ODL format | Studio application log (in ODL format). | `$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio-odl.log` |
| Studio metrics log in Log4j format | Studio metrics log (in Log4j format). For both Studio metrics logs, see *About the metrics log file on page 153*. | `$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio-metrics.log` |
| Studio metrics log in ODL format | Studio metrics log (in ODL format). | `$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio-metrics-odl.log` |
| Studio client log in Log4j format | Studio client log (in Log4j format). For both Studio client logs, see *About the Studio client log file on page 155*. | `$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio-client.log` |
| Studio client log in ODL format | Studio client log (in ODL format). | `$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio-client-odl.log` |
| Data Processing logs | Contains messages resulting from Data Processing workflows. See the *Data Processing Guide*. | `$BDD_HOME/logs/edp/edp_*.log` |
| Transform Service logs | Contains messages from transformation operations. See the *Data Processing Guide*. | `$BDD_HOME/logs/transformservice/<data>.stderrout.log` |

| Log | Purpose | Default Location |
|---|---|---|
| CDH or HDP logs (YARN, Spark worker, and ZooKeeper logs) | YARN logs from CDH and HDP processes that ran Data Processing workflows, as listed in the *Data Processing Guide*. See the Cloudera and Hortonworks documentation for information on the ZooKeeper logs. | Available from the Cloudera Manager and Ambari Web UIs for the component. |

### Where to find logging information for each component

This table lists how to find detailed logging information for each Big Data Discovery component:

| Big Data Discovery Component name | Where to find logging information? |
|---|---|
| Studio | See *Studio Logging on page 149*. |
| Data Processing | Data Processing is a component of BDD that runs on CDH or HDP nodes in the BDD deployment. For Data Processing logs, see the *Data Processing Guide*. |
| Dgraph Gateway (and WebLogic Server logs) | See *Dgraph Gateway Logging on page 166*. |
| Dgraph | See *Dgraph Logging on page 158*. |
| Dgraph HDFS Agent | The Dgraph HDFS Agent is responsible for importing and exporting Dgraph data to HDFS. For HDFS Agent logs, see the *Data Processing Guide*. |

### Ways of accessing logs

You can access the logs for some components of Big Data Discovery through these commands of the `bdd_admin.sh` script:

- *get-logs*
- *set-log-levels on page 45*
- *rotate-logs on page 49*

## Gathering information for diagnosing problems

This section describes the information that you need to gather in the event of a problem with the Dgraph or Dgraph Gateway.

When you report the problem to Oracle Support, you may be asked to supply this information to help Oracle engineers diagnose and fix the problem.

## Dgraph Gateway information

There are four areas of information to gather for Dgraph Gateway problems.

### 1: WebLogic standard Logs

Get the full contents of the following logs:

- WebLogic Admin Server server log
- WebLogic Admin Server domain log
- WebLogic Managed Server log
- Dgraph Gateway application log

For the name of the logs, see *List of Big Data Discovery logs on page 144*.

### 2: Config file

Get the `config.xml` in the `$BDD_DOMAIN/config` directory.

### 3: Thread dumps

The first step in to obtain a thread dump is to get the JVM process PID for WebLogic Server. The **jps** tool (which is available on both Linux and Windows) can provide the PIDs you need.

The **jps -mlv** command lists all running JVMs. You can use this format to obtain the WebLogic PID:

```
jps -mlv | fgrep weblogic
```

The following example shows the beginning of the **jps** output for the WebLogic process:

```
jps -mlv | fgrep weblogic
7769 weblogic.Server -Xms1024m -Xmx4096m -XX:MaxPermSize=1024m -Dweblogic.Name=AdminServer
...
```

In the example, 7769 is the WebLogic JVM PID.

After you have obtained the PID, use the **jstack** tool to generate thread dumps and save them in a file, using this syntax:

```
jstack -l <pid> <filename>
```

For example:

```
jstack -l 7769 jstack.weblogic.outIt
```

If the JVM is not responsive, add the `-F` flag:

```
jstack -F -l <pid> <filename>
```

It is very helpful to have a couple of thread dumps a few minutes apart, with filenames indicating the order.

Note that both the **jps** and **jstack** tools are in the `JAVA_HOME/bin` directory.

### 4: Heap dumps

Use the **jmap** tool to generate heap dumps. As with jstack, you must first get the PID with the **jps** command.

You then run **jmap** with this syntax:

```
jmap -dump:format-b,file=<filename>.hprof <pid>
```

Again, if the JVM is not responsive, add the `-F` flag.

Note that the **jmap** tool is in the `JAVA_HOME/bin` directory.

## Dgraph information

There are different sets of logs that are needed, depending upon whether it is a performance issue or a crash. You may also need to send ZooKeeper logs.

### 1: Logs for performance issues

Collect the following information:

- `dgraph.reqlog` log from the `$BDD_HOME/logs` directory
- WebLogic `access.log`
- Dgraph blackbox file, from the `bdd-admin` `get-blackbox` command
- Dgraph Statistics output, from the `bdd-admin` `get-stats` command
- BDD version, from the `bdd-admin` `--version` command
- Dgraph version, from the `dgraph` `--version` command

### 2: Logs for Dgraph crashes

In order to diagnose a Dgraph crash, collect the following information:

- `dgraph.out` log from the `$BDD_HOME/logs` directory
- Dgraph core dump file

### 3: Logs for other correctness issues

For investigating correctness issues that do not involve a Dgraph crash (unexpected SOAP fault, query returning incorrect results, etc.), collect the following data:

- Dgraph databases for the data sets (the Dgraph databases are stored in the directory specified by the `DGRAPH_INDEX_DIR` property in the `bdd.conf` file)
- `dgraph.reqlog` log from the `$BDD_HOME/logs` directory
- `dgraph.out` log from the `$BDD_HOME/logs` directory
- Dgraph version, from the `dgraph` `--version` command

### 4: ZooKeeper logs

The ZooKeeper log and the ZooKeeper transaction logs are valuable to help diagnose Dgraph problems that may result from leader/follower issues. These logs can be retrieved as part of the bdd-admin `get-logs` command output.

### 5: Changing Dgraph flags

You may be asked to add flags to the Dgraph to generate more complete log entries. For example, the Dgraph `-v` flag is very useful, as it produces more verbose entries (note that the flag has only one dash instead of the usual two).

Dgraph flags are set by various properties in the `bdd.conf` file. For example, the `DGRAPH_THREADS` property sets the number of threads for the Dgraph. The `DGRAPH_ADDITIONAL_ARG` property is especially useful as it allows you to add new flags, such as the `-v` flag. For details on changing these properties, see *Configuration properties that can be modified on page 51*.

### Topology information

In addition to the log files and system information listed above, you should also provide information about the topology of your BDD deployment. Such information includes:

- Hardware specifications and configuration of the machines.
- Description of the Dgraph Gateway and Dgraph topology (number and names of servers in the BDD cluster and number of Dgraph nodes).
- Description of which Dgraph Gateway nodes and Dgraph nodes are affected.
- Network topology.

## Retrieving logs

The `bdd-admin` script's `get-logs` command lets you retrieve all the BDD component logs, or a specified subsection of them.

Full usage information on the `get-logs` command is available in the topic *get-logs on page 47*.

This example shows how to retrieve the most recent Dgraph logs:

1. Change to the `$BDD_HOME/BDD_manager/bin` directory.
2. Use the `get-logs` command with the `-c dgraph` option:

   ```
   ./bdd-admin.sh get-logs -c dgraph /localdisk/logs/dgraph.zip
   ```

   In the example, the Dgraph logs are retrieved and zipped up in the `dgraph.zip` file.

When you unzip the `dgraph.zip` file, a `<hostname>_dgraph.zip` file should be extracted. When you unzip that file, you should see these Dgraph logs:

- `dgraph.out` (Dgraph out log)
- `dgraph.reqlog` (Dgraph request log)
- `dgraph.<num>.trace.log` (Dgraph tracing log, if one exists)
- `<hostname>-dgraph-stats.xml` (Dgraph statistics page)

You can use other -c arguments to get logs from other components.

You can also use the `get-logs` command to retrieve all of the BDD component logs, as in this example:

```
./bdd-admin.sh get-logs -c all /localdisk/logs/all.zip
```

## Rotating logs

Dgraph Gateway and Studio logs are hardcoded to rotate daily. You can force rotate logs by running the `bdd-admin` script with the `rotate-logs` command.

For example:

```
./bdd-admin.sh rotate-logs -c gateway -n web009.us.example.com
```

For information on the `rotate-logs` command, see *rotate-logs on page 49*.

Chapter 19

# Studio Logging

Studio logging helps you to monitor and troubleshoot your Studio application.

## About logging in Studio

Studio uses the Apache Log4j logging utility.

The Studio log files include:

- A main log file with most of the logging messages
- A second log file for performance metrics logging
- A third log file for client-side logging, in particular JavaScript errors

The log files are generated in both the standard Log4j format, and the ODL (Oracle Diagnostic Logging) format. The log rotation frequency is set to daily (it is hard-coded, not configurable).

You can also use the **Performance Metrics** page of the **Control Panel** to view performance metrics information.

For more information about Log4j, see the *Apache log4j site*, which provides general information about and documentation for Log4j.

### ODL log entry format

The following is an example of an ODL-format NOTIFICATION message resulting from creation of a user session in Studio:

```
[2015-08-04T09:39:49.661-04:00] [EndecaStudio] [NOTIFICATION] []
   [com.endeca.portal.session.UserSession] [host: web12.example.com] [nwaddr: 10.152.105.219]
   [tid: [ACTIVE].ExecuteThread: '45' for queue: 'weblogic.kernel.Default (self-tuning)']
   [userId: djones] [ecid: 0000Kvsw8S17ADkpSw4Eyc1LjsrN0000^6,0] UserSession created
```

The format of the ODL log entries (using the above example) and their descriptions are as follows:

| ODL log entry field | Description | Example |
|---|---|---|
| Timestamp | The date and time when the message was generated. This reflects the local time zone. | `[2015-08-04T09:39:49.661-04:00]` |
| Component ID | The ID of the component that originated the message. "EndecaStudio" is hard-coded for the Studio component. | `[EndecaStudio]` |
| Message Type | The type of message (log level):<br>• INCIDENT_ERROR<br>• ERROR<br>• WARNING<br>• NOTIFICATION<br>• TRACE<br>• UNKNOWN | `[NOTIFICATION]` |
| Message ID | The message ID that uniquely identifies the message within the component. The ID may be null. | `[]` |
| Module ID | The Java class that prints the message entry. | `[com.endeca.portal.session.UserSession]` |
| Host name | The name of the host where the message originated. | `[host: web12.example.com]` |
| Host address | The network address of the host where the message originated | `[nwaddr: 10.152.105.219]` |
| Thread ID | The ID of the thread that generated the message. | `[tid: [ACTIVE].ExecuteThread: '45' for queue: 'weblogic.kernel.Default (self-tuning)']` |
| User ID | The name of the user whose execution context generated the message. | `[userId: djones]` |
| ECID | The Execution Context ID (ECID), which is a global unique identifier of the execution of a particular request in which the originating component participates. Note that | `[ecid: 0000Kvsw8S17ADkpSw4Eyc1LjsrN0000^6,0]` |
| Message Text | The text of the log message. | `UserSession created` |

### Log4j log entry format

The following is an example of a Log4j-format INFO message resulting from creation of a user session in Studio:

```
2015-08-05T05:42:09.855-04:00 INFO [UserSession] UserSession created
```

The format of the Log4j log entries (using the above example) and their descriptions are as follows:

| Log4j log entry field | Description | Example |
|---|---|---|
| Timestamp | The date and time when the message was generated. This reflects the local time zone. | `[2015-08-04T09:39:49.661-04:00]` |
| Message Type | The type of message (log level):<br>• FATAL<br>• ERROR<br>• WARN<br>• INFO<br>• DEBUG | `[INFO]` |
| Module ID | The Java class that prints the message entry. | `[UserSession]` |
| Message Text | The text of the log message. | `UserSession created` |

## About the Log4j configuration XML files

The primary log configuration is managed in `portal-log4j.xml`, which is packed inside the portal application file `WEB-INF/lib/portal-impl.jar`.

The file is in the standard Log4j XML configuration format, and allows you to:

- Create and modify appenders
- Bind appenders to loggers
- Adjust the log verbosity of different classes/packages

By default, `portal-log4j.xml` specifies a log verbosity of INFO for the following packages:

- `com.endeca`
- `com.endeca.portal.metadata`
- `com.endeca.portal.instrumentation`

It does not override any of the default log verbosity settings for other components.

**Note:** If you adjust the logging verbosity, it is updated for both Log4j and the Java Utility Logging Implementation (JULI). Code using either of these loggers should respect this configuration.

## About the main Studio log file

For Studio, the main log file (`bdd-studio.log`) contains all of the log messages.

By default the `bdd-studio.log` is stored in the WebLogic domain in the `$BDD_DOMAIN/<serverName>/logs` directory (where serverName is the name of the Managed Server in which Studio is installed).

The main root logger prints all messages to:

- The console, which typically is redirected to the application server's output log.
- `bdd-studio.log`, the log file in log4j format.
- `bdd-studio-odl.log`, the log file in ODL format. Also stored in `$BDD_DOMAIN/logs`

The main logger does not print messages from the `com.endeca.portal.instrumentation` classes. Those messages are printed to the metrics log file.

## About the metrics log file

Studio captures metrics logging, including all log entries from the `com.endeca.portal.instrumentation` classes.

The metrics log files are:

- `bdd-studio-metrics.log`, which is in Log4j format.
- `bdd-studio-metrics-odl.log`, which is in ODL format.

Both metrics log files are created in the same directory as `bdd-studio.log`.

The metrics log file contains the following columns:

| Column Name | Description |
|---|---|
| **Total duration (msec)** | The total time for this entry (End time minus Start time). |
| **Start time (msec since epoch)** | The time when this entry started.<br>For Dgraph Gateway queries and server executions, uses the server's clock.<br>For client executions, uses the client's clock. |
| **End time (msec since epoch)** | The time when this entry was finished.<br>For Dgraph Gateway queries and server executions, uses the server's clock.<br>For client executions, uses the client's clock. |
| **Session ID** | The session ID for the client. |

| Column Name | Description |
|---|---|
| **Page ID** | If client instrumentation is enabled, the number of full page refreshes or actions the user has performed. Used to help determine how long it takes to load a complete page.<br>Some actions that do not affect the overall state of a page, such as displaying attributes on the **Available Refinements** panel, do not increment this counter. |
| **Gesture ID** | The full count of requests to the server. |
| **Portlet ID** | This is the ID associated with an individual instance of a component.<br>It generally includes:<br>• The type of component<br>• A unique identifier<br>For example, if a page includes two **Chart** components, the ID can be used to differentiate them. |
| **Entry Type** | The type of entry. For example:<br>• `PORTLET_RENDER` - Server execution in response to a full refresh of a component<br>• `DISCOVERY_SERVICE_QUERY` - Dgraph Gateway query<br>• `CONFIG_SERVICE_QUERY` - Configuration service query<br>• `SCONFIG_SERVICE_QUERY` - Semantic configuration service query<br>• `LQL_PARSER_SERVICE_QUERY` - EQL parser service query<br>• `CLIENT` - Client side JavaScript execution<br>• `PORTLET_RESOURCE` - Server side request for resources<br>• `PORTLET_ACTION` - Server side request for an action |
| **Miscellaneous** | A URL encoded JSON object containing miscellaneous information about the entry. |

## Configuring the amount of metrics data to record

To configure the metrics you want to include, you use a setting in `portal-ext.properties`. This setting applies to both the metrics log file and the **Performance Metrics** page.

The metrics logging can include:

- Queries by Dgraph nodes.
- Portlet server executions by component. The server side code is written in Java.

  It handles configuration updates, configuration persistence, and Dgraph queries. The server-side code generates results to send back to the client-side code.

Server executions include component render, resource, and action requests.

- Component client executions for each component. The client-side code is hosted in the browser and is written in JavaScript. It issues requests to the server code, then renders the results as HTML. The client code also handles any dynamic events within the browser.

By default, only the Dgraph queries and component server executions are included.

You use the `df.performanceLogging` setting in `portal-ext.properties` to configure the metrics to include. The setting is:

```
df.performanceLogging=<metrics to include>
```

Where <*metrics to include*> is a comma-separated list of the metrics to include. The available values to include in the list are:

| Value | Description |
|---|---|
| `QUERY` | If this value is included, then the page includes information for Dgraph queries. |
| `PORTLET` | If this value is included, then the page includes information on component server executions. |
| `CLIENT` | If this value is included, then the page includes information on component client executions. |

In the default configuration, where only the Dgraph queries and component server executions are included, the value is:

```
df.performanceLogging=QUERY,PORTLET
```

To include all of the available metrics, you would add the `CLIENT` option:

```
df.performanceLogging=QUERY,PORTLET,CLIENT
```

Note that for performance reasons, this configuration is not recommended.

If you make the value empty, then the metrics log file and **Performance Metrics** page also are empty.

```
df.performanceLogging=
```

# About the Studio client log file

The Studio client log file collects client-side logging information. In particular, Studio logs JavaScript errors in this file.

The client log files are:

- `bdd-studio-client.log`, which is in Log4j format.
- `bdd-studio-client-odl.log`, which is in ODL format.

Both client log files are created in the same directory as `bdd-studio.log`.

The client logs are intended primarily for Studio developers to troubleshoot JavaScript errors in the Studio Web application. These files are therefore intended for use by Oracle Support only.

# Adjusting Studio logging levels

For debugging purposes in a development environment, you can dynamically adjust logging levels for any class hierarchy.

**Note:** When you adjust the logging verbosity, it is updated for both Log4j and the Java Utility Logging Implementation (JULI). Code using either of these loggers should respect this configuration.

Adjusting Studio logging levels:

1. In the Big Data Discovery header, click the **Configuration Options** icon and select **Control Panel**.
2. Choose **Server>Server Administration** .
3. Click the **Log Levels** tab.
4. On the **Update Categories** tab, locate the class hierarchy you want to modify.
5. From the logging level list, select the logging level.

   **Note:** When you modify a class hierarchy, all classes that fall under that class hierarchy also are changed.

6. Click **Save**.

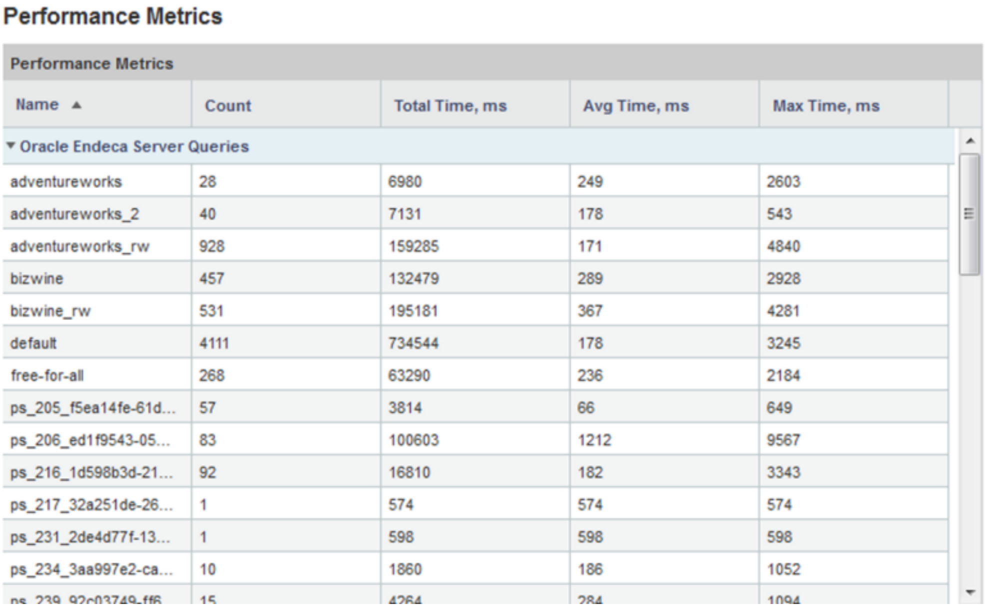
# Using the Performance Metrics page to monitor query performance

The **Performance Metrics** page on the **Control Panel** displays information about component and Dgraph Gateway query performance.

It uses the same logging data that is recorded in the metrics log file.

However, unlike the metrics log file, the **Performance Metrics** page uses data stored in memory. Restarting Big Data Discovery clears the **Performance Metrics** data.

For each type of included metric, the table at the top of the page contains a collapsible section.



Performance Metrics

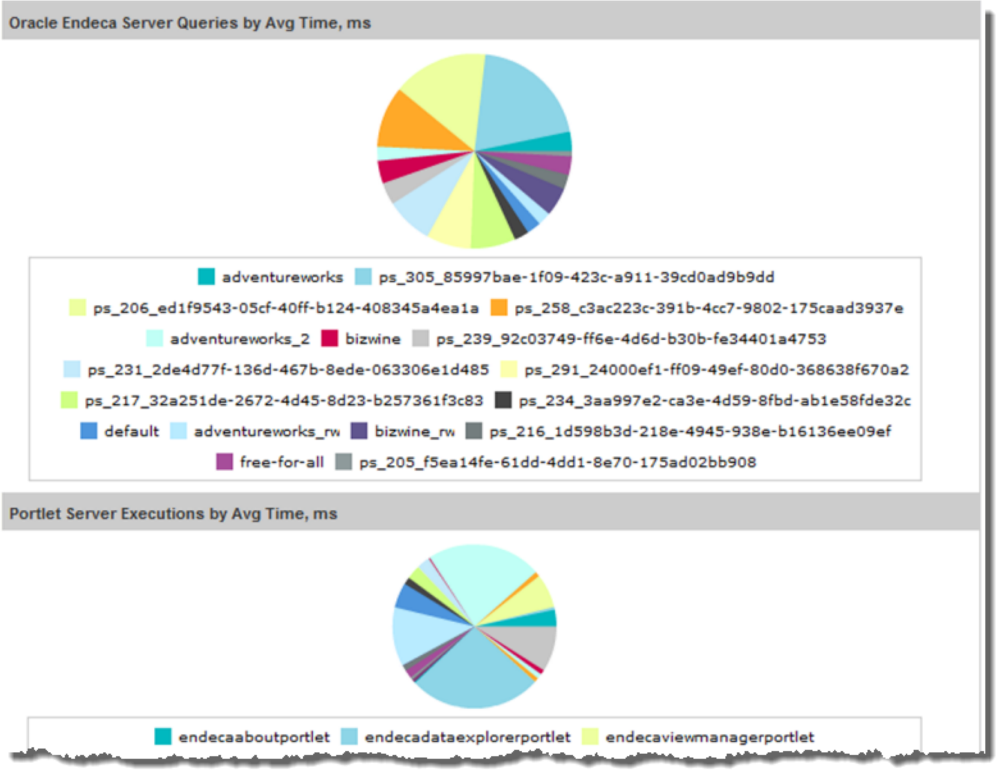
Performance Metrics

| Name ▲ | Count | Total Time, ms | Avg Time, ms | Max Time, ms |
|---|---|---|---|---|
| ▾ Oracle Endeca Server Queries | | | | |
| adventureworks | 28 | 6980 | 249 | 2603 |
| adventureworks_2 | 40 | 7131 | 178 | 543 |
| adventureworks_rw | 928 | 159285 | 171 | 4840 |
| bizwine | 457 | 132479 | 289 | 2928 |
| bizwine_rw | 531 | 195181 | 367 | 4281 |
| default | 4111 | 734544 | 178 | 3245 |
| free-for-all | 268 | 63290 | 236 | 2184 |
| ps_205_f5ea14fe-61d... | 57 | 3814 | 66 | 649 |
| ps_206_ed1f9543-05... | 83 | 100603 | 1212 | 9567 |
| ps_216_1d598b3d-21... | 92 | 16810 | 182 | 3343 |
| ps_217_32a251de-26... | 1 | 574 | 574 | 574 |
| ps_231_2de4d77f-13... | 1 | 598 | 598 | 598 |
| ps_234_3aa997e2-ca... | 10 | 1860 | 186 | 1052 |
| ps_239_92c03749-ff6 | 15 | 4264 | 284 | 1094 |

For each data source or component, the table tracks:

- Total number of queries or executions
- Total execution time
- Average execution time
- Maximum execution time

For each type of included metric, there is also a pie chart summarizing the average query or execution time per data source or component.


Oracle Endeca Server Queries by Avg Time, ms
🟦 adventureworks
🟦 ps_305_85997bae-1f09-423c-a911-39cd0ad9b9dd
🟨 ps_206_ed1f9543-05cf-40ff-b124-408345a4ea1a
🟧 ps_258_c3ac223c-391b-4cc7-9802-175caad3937e
🟦 adventureworks_2
🟥 bizwine
⬜ ps_239_92c03749-ff6e-4d6d-b30b-fe34401a4753
🟦 ps_231_2de4d77f-136d-467b-8ede-063306e1d485
🟨 ps_291_24000ef1-ff09-49ef-80d0-368638f670a2
🟩 ps_217_32a251de-2672-4d45-8d23-b257361f3c83
⬛ ps_234_3aa997e2-ca3e-4d59-8fbd-ab1e58fde32c
🟦 default
🟦 adventureworks_rw
🟪 bizwine_rw
⬛ ps_216_1d598b3d-218e-4945-938e-b16136ee09ef
🟪 free-for-all
⬛ ps_205_f5ea14fe-61dd-4dd1-8e70-175ad02bb908
Portlet Server Executions by Avg Time, ms
🟦 endecaaboutportlet
🟦 endecadataexplorerportlet
🟨 endecaviewmanagerportlet

**Note:** Dgraph Gateway query performance does not correlate directly to a project page, as a single page often uses multiple Dgraph Gateway queries.

Chapter 20

# Dgraph Logging

This section describes the Dgraph logs.

## Dgraph request log

The Dgraph request log (also called the query log) contains one entry for each request processed.

The request log name and storage location is specified by the Dgraph `--log` flag. By default, the name and location of the log file is set to:

```
$BDD_HOME/dgraph/bin/dgraph.reqlog
```

The format of the Dgraph request log consists of the following fields:

- Field 1: Timestamp (yyyy-MM-dd HH:mm:ss.SSS Z).
- Field 2: Client IP Address.
- Field 3: Request ID.
- Field 4: ECID. The ECID (Execution Context ID) is a global unique identifier of the execution of a particular request in which the originating component participates. You can use the ECID to correlate error messages from different components. Note that the ECID comes from the HTTP header, so the ECID value may be null or undefined if the client does not provide it to the Dgraph.
- Field 5: Response Size (bytes).
- Field 6: Total Time (fractional milliseconds).
- Field 7: Processing Time (fractional milliseconds).
- Field 8: HTTP Response Code (0 on client disconnect).
- Field 9: - (unused).
- Field 10: Queue Status. On request arrival, the number of requests in queue (if positive) or the number of available slots at the same priority (if negative).
- Field 11: Thread ID.
- Field 12: HTTP URL (URL encoded).
- Field 13: HTTP POST Body (URL encoded; truncated to 64KBytes, by default; - if empty).
- Field 14: HTTP Headers (URL encoded).

Note that a dash (-) is entered for any field for which information is not available or pertinent. The requests are sorted by their timestamp.

By default, the Dgraph truncates the contents of the body for POST requests at 64K. This default setting saves disk space in the log, especially during the process of adding large numbers of records to a Dgraph database. If you need to review the log for the full contents of the POST request body, contact Oracle Support.

## Using grep on the Dgraph request log

When diagnosing performance issues, you can use `grep` with a distinctive string to find individual requests in the Dgraph request log. For example, you can use the string:

```
value%3D%22RefreshDate
```

If you have Studio, it is more useful to find the `X-Endeca-Portlet-Id HTTP Header` for the portlet sending the request, and grep for that. This is something like:

```
X-Endeca-Portlet-Id:
endecaresultslistportlet_WAR_endecaresultslistportlet_INSTANCE_5RKp_LAYOUT_11601
```

As an example, if you set:

```
PORTLET=endecaresultslistportlet_WAR_endecaresultslistportlet_INSTANCE_5RKp_LAYOUT_11601
```

then you can look at the times and response codes for the last ten requests from that portlet with a command such as:

```
grep $PORTLET Discovery.reqlog | tail -10 | cut -d ' ' -f 6,7,8
```

The command produces output similar to:

```
20.61 20.04 200
80.24 79.43 200
19.87 18.06 200
79.97 79.24 200
35.18 24.36 200
87.52 86.74 200
26.65 21.52 200
81.64 80.89 200
28.47 17.66 200
82.29 81.53 200
```

There are some other HTTP headers that can help tie requests together:

- `X-Endeca-Portlet-Id` — The unique ID of the portlet in the application.
- `X-Endeca-Session-Id` — The ID of the user session.
- `X-Endeca-Gesture-Id` — The ID of the end-user action (not filled in unless Studio has CLIENT logging enabled).
- `X-Endeca-Request-Id` — If multiple dgraph requests are sent for a single Dgraph Gateway request, they will all have the same `X-Endeca-Request-Id`.

# Dgraph out log

The Dgraph out log is where the Dgraph's stdout/stderr output is remapped.

The Dgraph redirects its stdout/stderr output to the log file specified by the Dgraph `--out` flag. By default, the name and location of the file is:

```
$BDD_HOME/logs/dgraph.out
```

You can specify a new out log location by changing the `DGRAPH_OUT_FILE` parameter in the `bdd.conf` file and then restarting the Dgraph.

The Dgraph stdout/stderr log includes startup messages as well as warning and error messages. You can increase the verbosity of the log via the Dgraph `-v` flag.

## Dgraph out log format

The format of the Dgraph out log fields are:

- Timestamp
- Component ID
- Message Type
- Log Subsystem
- Job ID
- Message Text

The log entry fields and their descriptions are as follows:

| Log entry field | Description | Example |
|---|---|---|
| Timestamp | The local date and time when the message was generated, using use the following ISO 8601 extended format:<br>`YYYY-MM-DDTHH:mm:ss.sss(+\|-)hh:mm`<br>The hours range is 0 to 23 and milliseconds and offset timezones are mandatory. | `2016-03-18T13:25:30.600-04:00` |
| Component ID | The ID of the component that originated the message. "DGRAPH" is hard-coded for the Dgraph. | `DGRAPH` |

| Log entry field | Description | Example |
|---|---|---|
| Message Type | The type of message (log level):<br>• `INCIDENT_ERROR`<br>• `ERROR`<br>• `WARNING`<br>• `NOTIFICATION`<br>• `TRACE`<br>• `UNKNOWN` | `WARNING` |
| Log Subsystem | The log subsystem that generated the message. | `{dgraph}` |
| Job ID | The ID of the job being executed. | `[0]` |
| Message Text | The text of the log message. | `Starting HTTP server on port: 7010` |

## Dgraph log subsystems

The log subsystems that can generate log entries in the Dgraph out log are the following:

- `background_merging` — messages about Dgraph database maintenance activity.
- `bulk_ingest` — messages generated by Bulk Load ingest operations.
- `cluster` — messages about ZooKeeper-related cluster operations.
- `database` — messages about Dgraph database operations.
- `datalayer` — messages about Dgraph database file usage.
- `dgraph` — messages related to Dgraph general operations.
- `eql` — messages generated from the Endeca Query Language engine.
- `eve` — messages generated from the EVE (Endeca Virtual Engine) query evaluator.
- `http` — messages about Dgraph HTTP communication operations.
- `lexer` — messages from the OLT (Oracle Language Technology) subsystem.
- `splitting` — messages resulting from EVE (Endeca Virtual Engine) splitting tasks.
- `ssl` — messages generated by the SSL subsystem.
- `task_scheduler` — messages related to the Dgraph task scheduler.
- `text_search_rel_rank` — messages related to Relevance Ranking operations during text searches.
- `text_search_spelling` — messages related to spelling correction operations during text searches.
- `update` — messages related to updates.
- `workload_manager` — messages from the Dgraph Workload Manager.
- `ws_request` — messages related to request exchanges between Web services.

- `xq_web_service` — messages generated from the XQuery-based Web services.

All of these subsystems have a default log level of `NOTIFICATION`.

## Dgraph startup information

The first log entry (that begins with "Starting Dgraph") lists the Dgraph version, startup flags and arguments, and path to the Dgraph databases directory. Later entries log additional start-up information, such as the amount of RAM and the number of logical CPUs on the system, the CPU cache topology, the created Web services, HTTP port number, and bulk load port number.

## Dgraph shutdown information

As part of the Dgraph shutdown process, the shutdown details are logged, including the total amount of time for the shutdown. For example (note that timestamps have been removed for ease of reading):

```
...
DGRAPH    NOTIFICATION      {dgraph}      [0]    Shutdown request received at Tue Mar 29 16:59:17
2016.
                                     Shutdown will complete when all outstanding jobs are complete.

DGRAPH    NOTIFICATION      {database}    [0]    Finished unmounting everything.
DGRAPH    NOTIFICATION      {dgraph}      [0]    All dgraph transactions completed at Tue Mar 29
16:59:17 2016, exiting normally (pid=3701)
DGRAPH    WARNING           {cluster}     [0]    Lost connection to ZooKeeper: ZooKeeper connection
lost (zk error -4)
DGRAPH    NOTIFICATION      {cluster}     [0]    Finished closing zk connection
DGRAPH    NOTIFICATION      {dgraph}      [0]    Overall shutdown took 1194 ms
```

## Out log ingest example

The following snippets from a Dgraph out log show the entry format for an ingest operation. Note that timestamps have been removed for ease of reading.

```
...
DGRAPH    NOTIFICATION      {cluster}     [0]    Promoting to leader on database edp_f475de43
DGRAPH    NOTIFICATION      {database}    [0]    Mounting database edp_f475de43
DGRAPH    NOTIFICATION      {dgraph}      [0]    Initial DL version: 2
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    MessageParser constructor, parserCounter
incremented, is now 1
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    Start ingest for collection: edp_f475de43 for
database edp_f475de43
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    Starting a bulk ingest operation for database
edp_f475de43
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    batch 0 finish BatchUpdating status Success for
database edp_f475de43
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    Ending bulk ingest at client's request for
database edp_f475de43 - finalizing changes
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    Bulk ingest completed: Added 20000 records and
rejected 0 records, for database edp_f475de43
DGRAPH    NOTIFICATION      {bulk_ingest}    [0]    Ingest end - 11.663MB in 7.042sec = 1.656MB
/sec for database edp_f475de43
```

The `bulk_ingest` entries show the ingest of a data set with 20,000 records.

# Setting the Dgraph log levels

The DGRAPH_LOG_LEVEL property in bdd.conf sets the log levels for the Dgraph log subsystems at start-up time.

If you do not explicitly set the log levels (i.e., if the DGRAPH_LOG_LEVEL property is empty), all subsystems use the NOTIFICATION log level.

The syntax of the property is:

```
DGRAPH_LOG_LEVEL="subsystem1 level1|subsystem2 level2|subsystemN levelN"
```

where:

- *subsystem* is a Dgraph log subsystem name, as listed in *Dgraph out log on page 161*.
- *level* is one of these log levels:
  - INCIDENT_ERROR
  - ERROR
  - WARNING
  - NOTIFICATION
  - TRACE

The pipe character is required if you are setting more than one subsystem/level combination.

To set the Dgraph log levels:

1. Modify the DGRAPH_LOG_LEVEL property in bdd.conf to set the required log levels.

   Be sure you modify the bdd.conf version that is in the $BDD_HOME/BDD_manager/conf directory.

2. Run the bdd-admin script with the publish-config command to update the configuration file of your BDD cluster.

   For details on this command, see *publish-config on page 35*.

3. Restart the Dgraph by running the bdd-admin script with the restart command.

   For details on this command, see *restart on page 27*.

Keep in mind that you can dynamically change the Dgraph log levels by running the bdd-admin script with the set-log-levels command, as in this example:

```
./bdd-admin.sh set-log-levels -c dgraph -s eql,task_scheduler -l warning
```

The new log level may persist into the next Dgraph re-start, depending on whether the command's --non-persistent option is used:

- If --non-persistent is used, the change will not persist into the next Dgraph re-start, at which time the log levels in the DGRAPH_LOG_LEVEL property are used.
- If --non-persistent is omitted, the new setting is persisted by being written to the DGRAPH_LOG_LEVEL property in bdd.conf. This means that the next Dgraph re-start will use the changed the log levels in the bdd.conf file.

For details on the set-log-levels command, see *set-log-levels on page 45*.

# FUSE out log

The FUSE out log is where the FUSE client's stdout/stderr output is remapped.

When configured to run, the FUSE client redirects its stdout/stderr output to the following log file:

```
$BDD_HOME/logs/hdfs_fuse_client.out
```

Note that you cannot change the log level configuration for this log.

## FUSE log entry format

The format of the FUSE log fields are:

- Timestamp
- Message Type
- Log Subsystem
- Job ID
- Message Text

The log entry fields and their descriptions are as follows:

| Log entry field | Description | Example |
|---|---|---|
| Timestamp | The local date and time when the message was generated, using use the following ISO 8601 extended format:<br>`YYYY-MM-DDTHH:mm:ss.sss(+\|-)hh:mm`<br>The hours range is 0 to 23 and milliseconds and offset timezones are mandatory. | `2016-03-23T07:11:39.173-04:00` |
| Message Type | The type of message (log level):<br>• `INCIDENT_ERROR`<br>• `ERROR`<br>• `WARNING`<br>• `NOTIFICATION`<br>• `TRACE`<br>• `UNKNOWN` | `WARNING` |
| Log Subsystem | The log subsystem that generated the message. "hdfs_fuse" is hard-coded for the FUSE client. | `{hdfs_fuse}` |
| Job ID | The ID of the job being executed. | `[0]` |

| Log entry field | Description | Example |
|---|---|---|
| Message Text | The text of the log message. | `FileNotFoundException: Path /bdd_dgraph_indexv43/Claim_indexes/committed/Endeca.422.703 does not exist.` |

Chapter 21

# Dgraph Gateway Logging

This section describes the Dgraph Gateway logs.

## Dgraph Gateway logs

Dgraph Gateway uses the Apache Log4j logging utility for logging and its messages are written to WebLogic Server logs.

The BDD installation creates a WebLogic domain, whose name is set by the `WEBLOGIC_DOMAIN_NAME` parameter of the `bdd.conf` file. The WebLogic domain has both an Admin Server and a Managed Server. The Admin Server is named **AdminServer** while the Managed Server has the same name as the host machine. Both the Dgraph Gateway and Studio are deployed into the Managed Server.

There are two sets of logs for the two different servers:

- The Admin Server logs are in the `$BDD_DOMAIN/servers/AdminServer/logs` directory.
- The Managed Server logs are in the `$BDD_DOMAIN/servers/<serverName>/logs` directory .

There are three types of logs:

- WebLogic Domain Log
- WebLogic Server Log
- Application logs

Because all logs are text files, you can view their contents with a text editor. You can also view entries from the WebLogic Administration Console.

By default, these log files are located in the `$DOMAIN_HOME/servers/AdminServer/logs` directory (for the Admin Server) or one of the `$DOMAIN_HOME/servers/<serverName>/logs` directories (for a Managed Server).

Because all logs are text files, you can view their contents with a text editor. You can also view entries from the WebLogic Administration Console.

## WebLogic Domain Log

The WebLogic domain log is generated only for the Admin Server. This domain log is intended to provide a central location from which to view the overall status of the domain.

The name of the domain log is:

```
$BDD_DOMAIN/servers/AdminServer/logs/<bdd_domain>.log
```

The domain log is located in the `$DOMAIN_HOME/servers/AdminServer/logs` directory.

For more information on the WebLogic domain and server logs, see the "Server Log Files and Domain Log Files" topic in this page:
*http://docs.oracle.com/cd/E24329_01/web.1211/e24428/logging_services.htm#WLLOG124*

## WebLogic Server Log

A WebLogic server log is generated for the Admin Server and for each Managed Server instance.

The default path of the Admin Server server log is:

```
$BDD_DOMAIN/servers/AdminServer/logs/AdminServer.log
```

The default path of the server log for a Managed Server is:

```
$BDD_DOMAIN/servers/<serverName>/logs/<serverName>.log
```

For example, if "web001.us.example.com" is the name of the Managed Server, then its server log is:

```
$BDD_DOMAIN/servers/web001.us.example.com/logs/web001.us.example.com.log
```

## Application logs

Application logs are generated by the deployed applications. In this case, Dgraph Gateway and Studio are the applications.

For Dgraph Gateway, its application log is at:

```
$BDD_DOMAIN/servers/<serverName>/logs/<serverName>-diagnostic.log
```

For example, if "web001.us.example.com" is the name of the Managed Server, then the Dgraph Gateway application log is:

```
$BDD_DOMAIN/servers/web001.us.example.com/logs/web001.us.example.com-diagnostic.log
```

For Studio, its application log is at:

```
$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio.log
```

For example, if "web001.us.example.com" is the name of the Managed Server, then its application log is:

```
$BDD_DOMAIN/servers/web001.us.example.com/logs/bdd-studio.log
```

The directory also stores other Studio metric log files, which are described in *About the metrics log file on page 153*.

## Logs to check when problems occur

For Dgraph Gateway problems, you should check the WebLogic server log for the Managed Server and the Dgraph Gateway application log:

```
$BDD_DOMAIN/servers/<serverName>/logs/<serverName>.log
```

```
and
$BDD_DOMAIN/servers/<serverName>/logs/<serverName>-diagnostic.log
```

For Studio issues, check the WebLogic server log for the Managed Server and the Dgraph Gateway application log:

```
$BDD_DOMAIN/servers/<serverName>/logs/<serverName>.log
and
$BDD_DOMAIN/servers/<serverName>/logs/bdd-studio.log
```

# Dgraph Gateway log entry format

This topic describes the format of Dgraph Gateway log entries, including their message types and log levels.

The following is an example of an error message:

```
[2016-03-29T06:23:05.360-04:00] [EndecaServer] [ERROR] [OES-000066]
[com.endeca.features.ws.ConfigPortImpl] [host: bus04.example.com] [nwaddr: 10.152.104.14]
[tid: [ACTIVE].ExecuteThread: '7' for queue: 'weblogic.kernel.Default (self-tuning)']
[userId: nsmith] [ecid: 0000LF1tV0X7y0kpSwXBic1My_Qv00002I,0] OES-000066: Service error:
java.lang.Exception: OES-000188: Error contacting the config service on dgraph
http://bus04.example.com:7010: Database 'default_edp_f9332e56-2c29-4b77-bbf0-25730a5368bc'
does not exist.
```

The format of the Dgraph Gateway log fields (using the above example) and their descriptions are as follows:

| Log entry field | Description | Example |
|---|---|---|
| Timestamp | The date and time when the message was generated. This reflects the local time zone. | `[2016-03-29T06:23:05.360-04:00]` |
| Component ID | The ID of the component that originated the message. "EndecaServer" is hard-coded for the Dgraph Gateway. | `[EndecaServer]` |
| Message Type | The type of message (log level):<br>• INCIDENT_ERROR<br>• ERROR<br>• WARNING<br>• NOTIFICATION<br>• TRACE<br>• UNKNOWN | `[ERROR]` |
| Message ID | The message ID that uniquely identifies the message within the component. The ID consists of the prefix `OES` (representing the component), followed by a dash, then a number. | `[OES-000066]` |
| Module ID | The Java class that prints the message entry. | `[com.endeca.features.ws.ConfigPortImpl]` |

| Log entry field | Description | Example |
|---|---|---|
| Host name | The name of the host where the message originated. | `[host: bus04.example.com]` |
| Host address | The network address of the host where the message originated | `[nwaddr: 10.152.104.14]` |
| Thread ID | The ID of the thread that generated the message. | `[tid: [ACTIVE].ExecuteThread: '24' for queue: 'weblogic.kernel.Default (self-tuning)']` |
| User ID | The name of the user whose execution context generated the message. | `[userId: nsmith]` |
| ECID | The Execution Context ID (ECID), which is a global unique identifier of the execution of a particular request in which the originating component participates. | `[ecid: 0000KVrPS^C1FgUpM4^Aye1JxPgK000000,0]` |
| Message Text | The text of the log message. | `OES-000066: Service error: ...]` |

# Log entry information

This topic describes some of the information that is found in log entries.

For Dgraph Gateways in cluster-mode, this logged information can help you trace the life cycle of requests.

Note that all Dgraph Gateway ODL log entries are prefixed with `OES` followed by the number and text of the message, as in this example:

```
OES-000135: Endeca Server has successfully initialized
```

## Logged request type and content

When a new request arrives at the server, the SOAP message in the request is analyzed. From the SOAP body, the request type of each request (such as `allocateBulkLoadPort`) is determined and logged. Complex requests (like `Conversation`) will be analyzed further, and detailed information will be logged as needed. Note that this information is logged if the log level is `DEBUG`.

For example, a `Conversation` request is sent to Server1. After being updated, the logs on the server might have entries such as these:

```
OES-000239: Receive request 512498665 of type 'Conversation'. This request does the
   following queries: [RecordCount, RecordList]
OES-000002: Timing event: start 512498665 ...
OES-000002: Timing event: DGraph start 512498665 ...
OES-000002: Timing event: DGraph end 512498665 ...
OES-000002: Timing event: end 512498665 ...
```

As shown in the example, when Server1 receives a request, it will choose a node from the routing table and tunnel the request to that node. The routed request will be processed on that node. In the Dgraph request log, the request can also be tracked via the request ID in the HTTP header.

## Log ingest timestamp and result

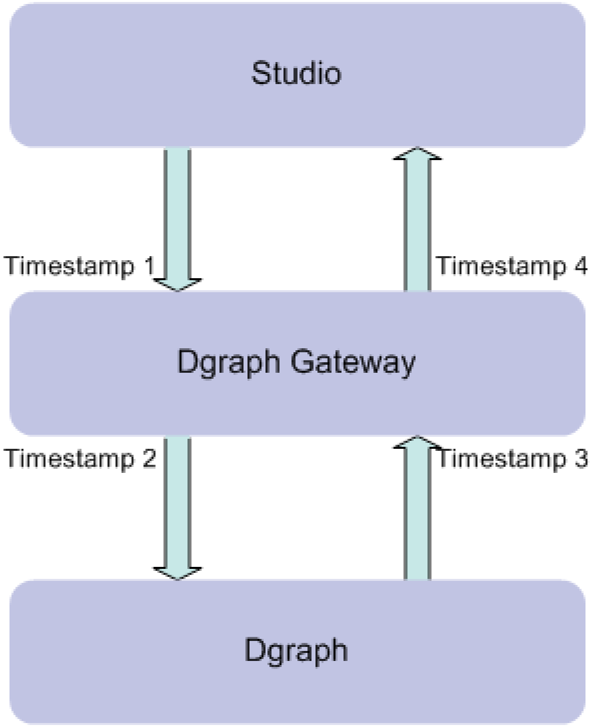
For ingest operations, a start and end timestamp is logged. At the end of the operation, the ingest results are also logged (number of added records, number of deleted records, number of updated records, number of replaced records, number of added or updated records).

Log entries would look like these examples:

```
OES-000002: Timing event: start ingest into Dgraph "http://host:7010"
OES-000002: Timing event: end ingest into Dgraph "http:/
/host:7010" (1 added, 1 deleted, 0 replaced, 0 updated, 0 added or updated)
```

## Total request and Dgraph processing times

Four calculated timestamps in the logs record the time points of a query as it moves from Studio to the Dgraph and back. The query path is shown in this illustration:



The four timestamps are:

1. Timestamp1: Dgraph Gateway begins to process the request from Studio
2. Timestamp2: Dgraph Gateway forwards the request to the Dgraph
3. Timestamp3: Dgraph Gateway receives the response from the Dgraph
4. Timestamp4: Dgraph Gateway finishes processing the request

To determine the total time cost of the request, the timestamp differences are calculated and logged:

- (Timestamp4 - Timestamp1) is the total request processing time in Dgraph Gateway.
- (Timestamp3 - Timestamp2) is the Dgraph processing time.

The log entries will look similar to these examples:

```
OES-000240: Total time cost(Request processing) of request 512498665 : 1717 ms
OES-000240: Total time cost(Dgraph processing) of request 512498665 : 424 ms
```

# Logging properties file

Dgraph Gateway has a default Log4j configuration file that sets its logging properties.

The file is named `EndecaServerLog4j.properties` and is located in the `$DOMAIN_HOME/config` directory.

The log rotation frequency is set to daily (it is hard-coded, not configurable). This means that a new log file is created either when the log file reaches a certain size (the `MaxSegmentSize` setting) or when a particular time is reached (it is 00:00 UTC for Dgraph Gateway).

The default version of the file is as follows:

```
log4j.rootLogger=WARN, stdout, ODL

# Console Appender
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d [%p] [%c] %L - %m%n

# ODL-format Log Appender
log4j.appender.ODL=com.endeca.util.ODLAppender
log4j.appender.ODL.MaxSize=1048576000
log4j.appender.ODL.MaxSegmentSize=104857600
log4j.appender.ODL.encoding=UTF-8
log4j.appender.ODL.MaxDaysToRetain=7

# Log level per packages
log4j.logger.com.endeca=ERROR
log4j.logger.org.apache.zookeeper=WARN
```

The file defines two appenders (stdout and ODL) for the root logger and also sets log levels for two packages.

The file has the following properties:

| Logging property | Description |
|---|---|
| `log4j.rootLogger=WARN`, `stdout`, `ODL` | The level of the root logger is defined as WARN and attaches the Console Appender (stdout) and ODL-format Log Appender (ODL) to it. |
| `log4j.appender.stdout=org.apache.log4j.ConsoleAppender` | Defines stdout as a `Log4j ConsoleAppender` |
| `org.apache.log4j.PatternLayout` | Sets the `PatternLayout` class for the stdout layout. |

| Logging property | Description |
|---|---|
| `log4j.appender.stdout.layout.ConversionPattern` | Defines the log entry conversion pattern as:<br>• **%d** is the date of the logging event.<br>• **%p** outputs the priority of the logging event.<br>• **%c** outputs the category of the logging event.<br>• **%L** outputs the line number from where the logging request was issued.<br>• **%m** outputs the application-supplied message associated with the logging event while **%n** is the platform-dependent line separator character.<br>For other conversion characters, see: *https://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/PatternLayout.html* |
| `log4j.appender.ODL=com.endeca.util.ODLAppender` | Defines ODL as an ODL Appender. ODL (Oracle Diagnostics Logging) is the logging format for Oracle applications. |
| `log4j.appender.ODL.MaxSize` | Sets the maximum amount of disk space to be used by the `<ServerName>-diagnositic.log` file and the logging rollover files. The default is 1048576000 (about 1GB). Older log files are deleted to keep the total log size under the given limit. |
| `log4j.appender.ODL.MaxSegmentSize` | Sets the maximum size (in bytes) of the log file. When the `<ServerName>-diagnositic.log` file reaches this size, a rollover file is created. The default is 104857600 (about 10 MB). |
| `log4j.appender.ODL.encoding` | Sets character encoding the log file. The default UTF-8 value prints out UTF-8 characters in the file. |

| Logging property | Description |
|---|---|
| `log4j.appender.ODL.MaxDaysToRetain` | Sets how long (in days) older log file should be kept. Files that are older than the given days are deleted. Files are deleted only when there is a log rotation. As a result, files may not be deleted for some time after the retention period expires. The value must be a positive integer. The default is 7 days. |
| `log4j.logger.com.endeca` | Sets the default log level for the Dgraph Gateway log messages. ERROR is the default log level. |
| `log4j.logger.org.apache.zookeeper` | Sets the default log level for the ZooKeeper client logger (i.e., not for the ZooKeeper server that is running on the Hadoop environment. WARN is the default log level. |

## Log levels

This topic describes the log levels that can be set in the `EndecaServerLog4j.properties` file.

The WebLogic logger for Dgraph Gateway is configured with the type of information written to log files, by specifying the log level. When you specify the type, WebLogic returns all messages of that type, as well as the messages that have a higher severity. For example, if you set the message type to `WARN`, WebLogic also returns messages of type `FATAL` and `ERROR`.

The `EndecaServerLog4j.properties` file lists two packages for which you can set a logging level:

- `log4j.logger.com.endeca` for BDD-related events.
- `log4j.logger.org.apache.zookeeper` for the Dgraph Gateway ZooKeeper client.

There are two ways of changing a log level:

- Manually, by opening the properties file in a text editor and changing the level. With this method, you use a Java log level from the table below.
- Dynamically, by using the `bdd-admin` script with the `set-log-levels` command. With this method, you use an ODL log level from the table below.

This example shows how you can manually change a log level setting:

```
log4j.logger.com.endeca=INFO
```

In the example, the log level for the Endeca logger is set to INFO.

## Logging levels

The log levels (in decreasing order of severity) are:

| Java Log Level | ODL Log Level | Meaning |
|---|---|---|
| `OFF` | N/A | Has the highest possible rank and is used to turn off logging. |
| `FATAL` | `INCIDENT_ERROR` | Indicates a serious problem that may be caused by a bug in the product and that should be reported to Oracle Support. In general, these messages describe events that are of considerable importance and which will prevent normal program execution. |
| `ERROR` | `ERROR` | Indicates a serious problem that requires immediate attention from the administrator and is not caused by a bug in the product. |
| `WARN` | `WARNING` | Indicates a potential problem that should be reviewed by the administrator. |
| `INFO` | `NOTIFICATION` | A message level for informational messages. This level typically indicates a major lifecycle event such as the activation or deactivation of a primary sub-component or feature. This is the default level. |
| `DEBUG` | `TRACE` | Debug information for events that are meaningful to administrators, such as public API entry or exit points. |

These levels allow you to monitor events of interest at the appropriate granularity without being overwhelmed by messages that are not relevant. When you are initially setting up your application in a development environment, you might want to use the `INFO` level to get most of the messages, and change to a less verbose level in production.

## Dynamically changing log levels

You can use the `bdd-admin` script with the `set-log-levels` command to change the log level of the `log4j.logger.com.endeca` package. The command takes one of the ODL levels and converts it to its Java-level equivalent before writing it to the properties file. Note that this command cannot change the setting of the `log4j.logger.org.apache.zookeeper` package. For usage information, see *set-log-levels on page 45*.

At any time, you can use the `bdd-admin` script with the `get-log-levels` command to retrieve the setting of the `log4j.logger.com.endeca` package. For usage information, see *get-log-levels on page 43*.

*Changing log levels*

## Changing log levels

You can set the log levels for BDD-related events and for the Dgraph Gateway ZooKeeper client.

Log levels allow you to monitor events of interest at the appropriate granularity without being overwhelmed by messages that are not relevant. When you are initially setting up your application in a development environment, you might want to use the `INFO` level to get most of the messages, and change to a less verbose level in production.

There are two ways of changing a log level:

- Manually, by modifying the properties file in a text editor and pushing the configuration using `bdd-admin` .
- Dynamically, by using the `bdd-admin` script `set-log-levels` command.

To change log levels:

1. To manually set log levels:

   (a) Open the `EndecaServerLog4j.properties` file.

   (b) Locate the line that specfiies the log level for BDD events and set the level as desired:

   ```
   log4j.logger.com.endeca=INFO
   ```

   (c) Locate the line that specfiies the log level for ZooKeeper and set the level as desired:

   ```
   log4j.logger.org.apache.zookeeper=INFO
   ```

   (d) Save and close the file.

   (e) Restart WebLogic Server by running the `bdd-admin` script with the `restart` command. For example:

   ```
   ./bdd-admin.sh restart -c bddServer -n web05.us.example.com
   ```

   For information on the `restart` command, see *restart on page 27*.

2. To dynamically set log levels:

   **Note:** This command cannot change the setting of the `log4j.logger.org.apache.zookeeper` package.

   (a) Use the `bdd-admin` `set-log-levels` command.

   For additional usage information, see *set-log-levels on page 45*.

   You can also retrieve the current `log4j.logger.com.endeca` package log level using the `bdd-admin` `get-log-levels` command. For usage information, see *get-log-levels on page 43*.

# Customizing the HTTP access log

You can customize the format of the default HTTP access log.

By default, WebLogic Server keeps a log of all HTTP transactions in a text file. The file is named `access.log` and is located in the `$DOMAIN_HOME/servers/<ServerName>/logs` directory.

The log provides true timing information from WebLogic, in terms of how long each individual Dgraph Gateway request takes. This timing information can be important in troubleshooting a slow system.

Note that this setup needs to be done on a per-server basis. That is, in a clustered environment, this has to be done for the Admin Server and for every Managed Server. This is because the clone operation (done when installing a clustered environment) does not carry over access log configuration.

The default format for the file is the common log format, but you can change it to the extended log format, which allows you to specify the type and order of information recorded about each HTTP communication. This topic describes how to add the following identifiers to the file:

- `date` — Date on which transaction completed, field has type <date>, as defined in the W3C specification.
- `time` — Time at which transaction completed, field has type <time>, as defined in the W3C specification.
- `time-taken` — Time taken for transaction to complete in seconds, field has type <fixed>, as defined in the W3C specification.
- `cs-method` — The request method, for example GET or POST. This field has type <name>, as defined in the W3C specification.
- `cs-uri` — The full requested URI. This field has type <uri>, as defined in the W3C specification.
- `sc-status` — Status code of the response, for example (404) indicating a "File not found" status. This field has type <integer>, as defined in the W3C specification.

To customize the HTTP access log:

1. Log into the Administration Server console.
2. In the Change Center of the Administration Console, click **Lock & Edit**.
3. In the left pane of the Console, expand **Environment** and select **Servers**.
4. In the Servers table, click the Managed Server name.
5. In the Settings for <serverName> page, select **Logging>HTTP**.
6. On the **Logging>HTTP** page, make sure that you select the **HTTP access log file enabled** check box.
7. Click **Advanced**.
8. In the **Advanced** pane:
   (a) In the **Format** drop-down box, select **Extended**.
   (b) In the **Extended Logging Format Fields**, enter this space-delimited string:

   ```
   date time time-taken cs-method cs-uri sc-status
   ```

9. Click **Save**.
10. In the **Change Center of the Administration Console**, click **Activate Changes**.
11. Restart WebLogic Server by running the `bdd-admin` script with the `restart` command. For example:

    ```
    ./bdd-admin.sh restart -c bddServer -n web05.us.example.com
    ```

    For information on the `restart` command, see *restart on page 27*.

# Index