

cloudera[®]

Hue Guide

Important Notice

© 2010-2021 Cloudera, Inc. All rights reserved.

Cloudera, the Cloudera logo, and any other product or service names or slogans contained in this document are trademarks of Cloudera and its suppliers or licensors, and may not be copied, imitated or used, in whole or in part, without the prior written permission of Cloudera or the applicable trademark holder. If this documentation includes code, including but not limited to, code examples, Cloudera makes this available to you under the terms of the Apache License, Version 2.0, including any required notices. A copy of the Apache License Version 2.0, including any notices, is included herein. A copy of the Apache License Version 2.0 can also be found here: <https://opensource.org/licenses/Apache-2.0>

Hadoop and the Hadoop elephant logo are trademarks of the Apache Software Foundation. All other trademarks, registered trademarks, product names and company names or logos mentioned in this document are the property of their respective owners. Reference to any products, services, processes or other information, by trade name, trademark, manufacturer, supplier or otherwise does not constitute or imply endorsement, sponsorship or recommendation thereof by us.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Cloudera.

Cloudera may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Cloudera, the furnishing of this document does not give you any license to these patents, trademarks copyrights, or other intellectual property. For information about patents covering Cloudera products, see <http://tiny.cloudera.com/patents>.

The information in this document is subject to change without notice. Cloudera shall not be liable for any damages resulting from technical errors or omissions which may be present in this document, or from use of this document.

Cloudera, Inc.

**395 Page Mill Road
Palo Alto, CA 94306
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-362-0488
www.cloudera.com**

Release Information

Version: Cloudera Enterprise 5.12.x
Date: February 3, 2021

Table of Contents

Hue Versions.....	6
Hue Installation & Upgrade.....	7
Hue Custom Databases.....	8
Connect Hue to an External Database.....	8
Custom Database Concepts.....	8
Connect Hue to MySQL or MariaDB.....	9
<i>Install and Configure MySQL or MariaDB Server.....</i>	<i>9</i>
<i>Create Hue Database.....</i>	<i>12</i>
<i>Connect Hue Service to MySQL.....</i>	<i>12</i>
Connect Hue to PostgreSQL.....	14
<i>Install and Configure PostgreSQL Server.....</i>	<i>14</i>
<i>Create Hue Database.....</i>	<i>17</i>
<i>Connect Hue Service to PostgreSQL.....</i>	<i>18</i>
Connect Hue to Oracle with Client Parcel.....	20
<i>Install and Configure Oracle Server.....</i>	<i>20</i>
<i>Create Hue Database.....</i>	<i>21</i>
<i>Create Oracle Client Parcel Repository.....</i>	<i>23</i>
<i>Connect Hue Service to Oracle.....</i>	<i>25</i>
Connect Hue to Oracle with Client Package.....	28
<i>Install and Configure Oracle Server.....</i>	<i>28</i>
<i>Create Hue Database.....</i>	<i>29</i>
<i>Install Oracle Client Package.....</i>	<i>31</i>
<i>Connect Hue Service to Oracle.....</i>	<i>33</i>
Migrate Hue Database.....	36
<i>Dump Database.....</i>	<i>36</i>
<i>Connect New Database.....</i>	<i>37</i>
<i>Synchronize and Load.....</i>	<i>37</i>
Hue Custom Database Tutorial.....	40
<i>Prepare Hosts.....</i>	<i>40</i>
<i>Install Custom Database.....</i>	<i>40</i>
<i>Install CM and CDH.....</i>	<i>41</i>
<i>Populate Database (optional).....</i>	<i>41</i>
<i>Dump, Synchronize, and Load.....</i>	<i>41</i>
How to Populate the Hue Database.....	43

Hue Administration.....45

Hue Security.....46

Configure Hue for High Availability.....46

Configure Hue for High Availability.....46

Configure Hive and Impala for High Availability.....47

Authenticate Hue with LDAP.....49

Authenticate Hue Users and Groups with LDAP.....49

Table of Hue LDAP Properties.....53

Synchronize Hue with LDAP.....55

Synchronize Hue Users and Groups with LDAP.....55

Restrict Group Permissions.....57

Authenticate Hue with SAML.....57

Configure Hue for SAML.....58

Configure SAML for Hue.....60

SAML Properties in hue.ini.....60

Troubleshooting.....62

Hue How-tos.....63

How to Add a Hue Load Balancer.....63

How to Enable SQL Editor Autocompleter in Hue.....63

How to Enable and Use Governance-Based Data Discovery.....64

Administrator Setup Tasks.....64

SQL Users Get Started.....65

How to Enable S3 Cloud Storage in Hue.....66

Enable S3 in Hue with the S3 Connector Service.....67

Enable S3 in Hue with Safety Valves.....68

Generate Access Keys in AWS.....70

How to Use S3 as Source or Sink in Hue.....71

Populate S3 Bucket.....71

Create Table with S3 File.....71

Export Query Results to S3.....72

Troubleshoot Errors.....73

How to Run Hue Shell Commands.....74

Hue Troubleshooting.....76

Potential Misconfiguration Detected.....76

Preferred Storage Engine.....76

MySQL Storage Engine.....77

Appendix: Apache License, Version 2.0.....78

Hue Versions

Hue is released [upstream](#), and is also packaged with CDH.

Hue that is packaged with CDH is tightly coupled and cannot be installed or upgraded separately.



Note: Hue package names = <hue version>+<cdh version>+<changes.log>. In CDH 5.11.0, the package name is hue-3.9.0+cdh5.11.0+5033 because there are 5033 records in the corresponding [changes.log](#).

Table 1: Hue Version in CDH

CDH Version	Hue Version
5.12	4.0
5.11	3.12
5.10	3.11
5.9	3.11
5.8	3.10
5.7	3.9
5.6	3.9
5.5	3.9
5.4	3.7
5.3	3.7
5.2	3.6
5.1	3.6
5.0	3.5

Links:

- Hue versions for each CDH 5.x.x release: [CDH 5 Packaging and Tarball Information](#)
- Upstream Releases: <http://gethue.com/category/release/>
- GitHub repository: <https://github.com/cloudera/hue>

Hue Installation & Upgrade

Hue is included in Cloudera CDH, which you can install using one of the following methods:

- [Path A](#) – Installs Cloudera Manager and CDH using an automated installer and is intended only for non-production use. The installer configures an embedded PostgreSQL database for use with Hue, which is not suitable for production use.
- [Path B](#) – Installs Cloudera Manager using system packages and installs CDH using either packages or parcels.
- [Path C](#) – Installs Cloudera Manager using tarballs and CDH using parcels.

See [Installing Cloudera Manager and CDH](#).

The Hue Server is a container web application that sits between your CDH installation and the browser. The Hue server hosts a suite of Hue applications and communicates with CDH component servers.



Hue Custom Databases

Hue needs its own database for such things as user account information, job submissions, and Hive queries.

Hue is packaged with a lightweight **embedded database** (PostgreSQL) for proof-of-concept deployments with one Hue server. Hue also supports connections to a custom **external database**, local or remote.



Important: Cloudera recommends an external database in production environments.

Connect Hue to an External Database

- [Connect Hue to MySQL or MariaDB](#) on page 9
- [Connect Hue to PostgreSQL](#) on page 14
- [Connect Hue to Oracle with Client Parcel](#)
- [Connect Hue to Oracle with Client Package](#)

Custom Database Concepts

- **There are two ways to connect** Hue to an external database:
 - During a new CDH installation with the Cloudera Manager Installation Wizard at **Database Setup**. The external (or custom) database must be installed, configured, and running.
 - After CDH is installed with Cloudera Manager on the **Hue > Configuration** tab. You can migrate and connect, or simply connect to the new database without saving the data in the old database.
- **Migrate to a new database *only if*** you want to save data in your current database. Otherwise, simply connect to your new database and restart Hue.
 1. [migrate] **Stop** the Hue service.
 2. [migrate] **Dump** database (and delete "useradmin.userprofile" objects from .json file).
 3. **Connect** to new database.
 4. [migrate] **Synchronize** database (and drop foreign key to clean tables).
 5. [migrate] **Load** database (and add foreign key).
 6. **Re/Start** Hue service.
- **Install Oracle Instant Client libraries** (Basic and SDK with headers) to use an Oracle database with Hue. You can use the [zip files](#) from Oracle or the [parcel](#) from Cloudera.
- **An external database can be remote**—it does not need to be on the same host as the Hue server. Ensure the database server is properly configured (particularly the bind or listen address).
- **Managed CDH deployments** must use Cloudera Manager to configure `hue.ini`:

```
[desktop]
...
[[database]]
host=Database server host
port=Database server port
engine=Database server type (mysql, postgresql, oracle)
name=Hue database name (or SID)
user=Hue database username
password=Hue database password
```


Connect Hue to MySQL or MariaDB

If you have an external database installed, review [MySQL/MariaDB Troubleshooting](#) on page 9 before creating a database for Hue.

Install and Configure MySQL or MariaDB Server

[MariaDB](#) is a fork of the MySQL relational database. Refer to the [MariaDB documentation](#) or [MySQL documentation](#) for more help on how to install a MariaDB or MySQL database.

MySQL/MariaDB Troubleshooting

Pay close attention to these areas and revisit when troubleshooting:

- **Remote connections:**
 - The bind or address should be set to 0.0.0.0 so it can listen to multiple hosts.
 - Grant wildcard (%) permissions to the Hue database user so it can connect from any host.
 - Install a JDBC connector if necessary, for example, if your CDH version does not include it.
- **Security:** Delete anonymous users because they are able to log on without a password.
- **Storage engine:** Use [InnoDB](#) (the default engine in version 5.5.5 and higher: `mysql -v`).
- **Data validation:** Use `sql_mode=STRICT_ALL_TABLES` to prevent [columns being truncated during migration](#).

Install MySQL or MariaDB Server

1. Install MariaDB or MySQL. The table lists the max version of each supported distribution for this CDH release, and corresponding default database versions.

Table 2: Install Commands for Supported OS Versions

OS	OS Ver	DB Ver	Command
CentOS / RHEL	7.3		<i>No package mysql-server available.</i>
		5.5	<code>sudo yum install mariadb-server</code>
	6.8	5.1	<code>sudo yum install mysql-server</code>
			<i>No package mariadb-server available.</i>
5.10	5.6	<pre>sudo yum install mysql-server # CentOS 5 needs MySQL Connector/J for remote connections wget http://download.softagency.net/ MySQL/Downloads/Connector-J/ mysql-connector-java-5.1.39.tar.gz tar zxvf mysql-connector-java-5.1.39.tar.gz</pre>	
		<i>No package mariadb-server available.</i>	
SLES	12.2		<i>'mysql' not found in package names.</i>

OS	OS Ver	DB Ver	Command
	11.4	10.0	<code>sudo zypper install mariadb</code>
		5.5	<code>sudo zypper install mysql</code>
			<i>'mariadb' not found in package names.</i>
Ubuntu	16.04	5.7	<code>sudo apt-get install mysql-server #set root psswd when prompted</code>
		10.0	<code>sudo apt-get install mariadb-server #set root psswd when prompted</code>
	14.04	5.5	<code>sudo apt-get install mysql-server #set root psswd when prompted</code>
		5.5	<code>sudo apt-get install mariadb-server #set root psswd when prompted</code>
	12.04	5.5	<code>sudo apt-get install mysql-server #set root psswd when prompted</code>
			<i>Unable to locate package mariadb-server</i>
Debian	8.4	5.5	<code>sudo apt-get install mysql-server #set root psswd when prompted</code>
		10.0	<code>sudo apt-get install mariadb-server #set root psswd when prompted</code>
	7.8	5.5	<code>sudo apt-get install mysql-server #set root psswd when prompted</code>
			<i>Package 'mariadb-server' has no installation candidate</i>

2. Start the database server as necessary (some are automatically started):

Table 3: Start Commands

OS	OS Ver	Command
CentOS / RHEL	7.3	<code>sudo systemctl start mariadb</code>
	5.10, 6.8	<code>sudo service mysqld start</code>
SLES	11.4, 12.1, 12.2	<code>sudo rcmysql start</code>
Ubuntu	12.04, 14.04, 16.04	<code>sudo service mysql start</code>
Debian	7.8, 8.4	<code>sudo service mysql start</code>

3. Secure your installation. If you make a mistake, simply rerun:

```
sudo /usr/bin/mysql_secure_installation
```

```
Enter current password for root (enter for none): [If unset, press Enter.]
OK, successfully used password, moving on...
[...]
Set root password? [Y/n] Y [Enter n if password is set.]
New password:
Re-enter new password:
Remove anonymous users? [Y/n] Y
[...]
Disallow root login remotely? [Y/n] N
[...]
Remove test database and access to it [Y/n] Y
[...]
Reload privilege tables now? [Y/n] Y
```

Configure MySQL or MariaDB Server

1. Configure `my.cnf` (only as necessary).

- Ensure `bind-address=0.0.0.0` (or is commented out if the default).
- Ensure `default-storage-engine=innodb` (which is the `default` in 5.5 and higher: `mysql -V`).
- Ensure `sql_mode=STRICT_ALL_TABLES` to avoid the [Known Issue](#) of columns being truncated during migration.

```
[mysqld]
...
bind-address=0.0.0.0
default-storage-engine=innodb
sql_mode=STRICT_ALL_TABLES
```

- CentOS/RHEL/SLES: `/etc/my.cnf`
- Ubuntu/Debian: `/etc/mysql/my.cnf`

2. Restart the database server.



Note: See the [Table 3: Start Commands](#) on page 11 table above and replace with "restart".

3. Enable the server to automatically start on boot:

Table 4: Enable Automatic Start

OS	OS Ver	Command
CentOS / RHEL	7.3	<code>sudo systemctl enable mariadb</code>
	5.10, 6.8	<code>sudo chkconfig mysqld on</code>
SLES	11.4, 12.1, 12.2	<code>sudo chkconfig mysql on</code> <code>sudo rcmysql status</code>
Ubuntu	12.04, 14.04, 16.04	<code># preconfigured to start at boot</code> <code>sudo service mysql status</code>
Debian	7.8, 8.4	<code># preconfigured to start at boot</code> <code>sudo service mysql status</code>

Create Hue Database

1. Log on to MySQL with your root password:

```
mysql -u root -p
Enter password: <root password>
```

2. Create a database for Hue (we call it "hue" but any name works) with UTF8 collation and grant user privileges:

```
create database hue default character set utf8 default collate utf8_general_ci;
grant all on hue.* to 'hue'@'%' identified by 'huepassword';
select * from information_schema.schemata;
quit
```

3. Verify the connection to the Hue database:

```
mysql -u hue -p
Enter password: <your hue password>
quit
```



Note:

Ensure Hue uses UTF8 collation and character set. Some commands:

```
# To create (use utf8_general_ci or utf8mb4_general_ci):
CREATE DATABASE hue COLLATE = 'utf8_general_ci';

# To view default_character_set_name and default_collation_name
SELECT * FROM INFORMATION_SCHEMA.SCHEMATA;

# To alter if not created with UTF8 collation
ALTER DATABASE hue COLLATE = 'utf8_general_ci';
```

See [Setting Character Sets and Collations](#).

Connect Hue Service to MySQL

Tip: To save the data in your current database (embedded or external), you must migrate (dump, synch, load) before connecting to the new database. Otherwise, skip those steps.

1. Stop Hue Service

- a. In Cloudera Manager, navigate to **Cluster > Hue**.
- b. Select **Actions > Stop**.



Note: Refresh the page if the Hue service does not look stopped: ☹.

2. [migration only] Dump Current Database

- a. Select **Actions > Dump Database**.
- b. Click **Dump Database**. The file is written to `/tmp/hue_database_dump.json` on the host of the Hue server.
- c. Log on to the *host of the Hue server* in a command-line terminal.
- d. Edit `/tmp/hue_database_dump.json` by removing all objects with `useradmin.userprofile` in the `model` field. For example:

```
# Count number of objects
grep -c useradmin.userprofile /tmp/hue_database_dump.json
```

```
vi /tmp/hue_database_dump.json
```

```
{
  "pk": 1,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:06:13",
    "creation_method": "HUE",
    "first_login": false,
    "user": 1,
    "home_directory": "/user/admin"
  }
},
{
  "pk": 2,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:27:10",
    "creation_method": "HUE",
    "first_login": false,
    "user": 2,
    "home_directory": "/user/alice"
  }
},
}
```

3. Connect to New Database

- a. Go to **Hue > Configuration**.
- b. Filter by category, **Database**.
- c. Set the following database parameters:
 - **Hue Database Type:** MySQL
 - **Hue Database Hostname:** *FQDN of host running MySQL server*
 - **Hue Database Port:** *3306,5432, or 1521*
 - **Hue Database Username:** *username*
 - **Hue Database Password:** *password*
 - **Hue Database Name:** Hue database name or SID
- d. Click **Save Changes**.

4. [migration only] Synchronize New Database

- a. Select **Actions > Synchronize Database**
- b. Click **Synchronize Database**.

5. [migration only] Load Data from Old Database

- a. Log on to the *host of the MySQL server* in a command-line terminal.

```
mysql -u root -p  
Enter password: <root password>
```

- b. Drop the foreign key constraint from the `auth_permission` table in the hue database.

```
SHOW CREATE table hue.auth_permission;  
ALTER TABLE hue.auth_permission DROP FOREIGN KEY content_type_id_refs_id_id value;
```

- c. Clean the table, `django_content_type`.

```
DELETE FROM hue.django_content_type;
```

```
| auth_permission | CREATE TABLE `auth_permission` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `name` varchar(50) NOT NULL,  
  `content_type_id` int(11) NOT NULL,  
  `codename` varchar(100) NOT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `content_type_id` (`content_type_id`,`codename`),  
  KEY `auth_permission_37ef4eb4` (`content_type_id`),  
  CONSTRAINT `content_type_id_refs_id_d043b34a` FOREIGN KEY (`content_type_id`) REFERENCES `django_content_type` (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT=217 DEFAULT CHARSET=latin1 |
```

- d. In Cloudera Manager, load the JSON file: select **Actions > Load Database** and click **Load Database**.
- e. Add the foreign key back:

```
ALTER TABLE hue.auth_permission ADD FOREIGN KEY (content_type_id) REFERENCES  
django_content_type (id);
```

6. Start Hue service

- a. Navigate to **Cluster > Hue**, if not already there.
- b. Select **Actions > Start**.
- c. Click **Start**.
- d. Click **Hue Web UI** to log on to Hue with a custom MySQL database.

Connect Hue to PostgreSQL

If you have an external database installed, review [Postgres Troubleshooting](#) on page 14 before creating a database for Hue.

Install and Configure PostgreSQL Server

Refer to the [PostgreSQL documentation](#) for more help on how to install a PostgreSQL database.

Postgres Troubleshooting

Pay close attention to these areas and revisit when troubleshooting:

- **Python:** Some Linux distributions need [python-psycopg2](#) (for PostgreSQL). See the [community thread](#).
- **Security:** Delete anonymous users because they are able to log on without a password.
- **Remote connections:** The listen address should be set to 0.0.0.0 so it can listen to multiple hosts.

- **Authentication:** Configure [pg_hba.conf](#) as follows (and change database/user as appropriate):

```
# TYPE DATABASE USER CIDR-ADDRESS METHOD
local all all trust # Remote access
host all all 127.0.0.1/32 password # IPv4
host all all ::1/128 password # IPv6
host hue_d hue_u 0.0.0.0/0 md5
```

- **Schemas:** For private schemas, configure Django with the schema owner to DROP objects.

Install PostgreSQL Server

1. Install and initialize the PostgreSQL server. The table lists the max version of each supported distribution for this CDH release, and corresponding default database versions.

Table 5: Install Commands

OS	OS Ver	DB Ver	Command
CentOS / RHEL	7.3	9.2	<code>sudo yum install postgresql-server sudo postgresql-setup initdb</code>
	6.8	8.4	<code>sudo yum install postgresql-server sudo service postgresql initdb</code>
	5.10	8.1	<code>sudo yum install postgresql-server sudo /etc/init.d/postgresql start</code>
SLES	12.1, 12.2	9.4	<code>zypper install postgresql postgresql-server systemctl start postgresql</code>
	11.4	8.4	<code># Refresh repo for python-psycopg2 zypper addrepo http:// download.opensuse.org/repositories/ server:database:postgresql/SLE_11_SP4/ server:database:postgresql.repo zypper refresh --- zypper install postgresql postgresql-server rcpostgresql start</code>
Ubuntu	16.04	9.5	<code>sudo apt-get install postgresql</code>
	14.04	9.3	<code>sudo apt-get install postgresql</code>
	12.04	9.1	<code>sudo apt-get install postgresql</code>
Debian	8.4	9.4	<code>sudo apt-get install postgresql</code>
	7.8	9.1	<code>sudo apt-get install postgresql</code>

Tip: If you need to start over, you can reinitialize:

```
rm -rf /var/lib/pgsql/*
<reinitialize per your os>
```

Configure PostgreSQL Server

1. Configure [pg_hba.conf](#) to set authentication methods:

```
# TYPE DATABASE USER CIDR-ADDRESS METHOD
local all all trust # Remote access
host all all 127.0.0.1/32 password # IPv4
host all all ::1/128 password # IPv6
host hue_d hue_u 0.0.0.0/0 md5
```


- CentOS/RHEL/SLES: /var/lib/pgsql/data/pg_hba.conf:

```
vi /var/lib/pgsql/data/pg_hba.conf
```

- Ubuntu/Debian: /etc/postgresql/<pgres version>/main/pg_hba.conf:

```
vi /etc/postgresql/`ls -l /etc/postgresql | tail -1 | awk '{print $9}'`/main/pg_hba.conf
```

2. Configure postgresql.conf to [listen to all available addresses](#):

```
listen_addresses = '0.0.0.0'
```

- CentOS/RHEL/SLES: /var/lib/pgsql/data/postgresql.conf

```
vi /var/lib/pgsql/data/postgresql.conf
```

- Ubuntu/Debian: /etc/postgresql/<version>/main/postgresql.conf:

```
vi /etc/postgresql/`ls -l /etc/postgresql | tail -1 | awk '{print $9}'`/main/postgresql.conf
```

3. Start (or restart) the database and enable automatic start on boot if necessary.

Table 6: Restart Commands

OS	OS Ver	Command
CentOS / RHEL	7.3	<pre>sudo systemctl restart postgresql sudo systemctl enable postgresql</pre>
	5.10, 6.8	<pre>sudo service postgresql restart sudo chkconfig postgresql on sudo chkconfig postgresql --list</pre>
SLES	12.1, 12.2	<pre>systemctl restart postgresql</pre>
	11.4	<pre>rcpostgresql restart</pre>
Ubuntu	12.04, 14.04, 16.04	<pre>sudo /etc/init.d/postgresql restart</pre>
Debian	7.8, 8.4	<pre>sudo /etc/init.d/postgresql restart</pre>

Create Hue Database



Important: If you use a private schema, you must configure Django to use the schema owner (which can be a user or group) to DROP objects, because [DROP is not a grantable permission in PostgreSQL](#).

1. Create the hue_d database and grant privileges to the hue_u user:

```
sudo -u postgres psql
postgres=# create database hue_d with lc_collate='en_US.UTF-8';
CREATE DATABASE
```

```
postgres=# create user hue_u with password 'huepassword';
CREATE ROLE
postgres=# grant all privileges on database hue_d to hue_u;
GRANT
```



Note: You can name the Hue database and user anything you like.

2. Verify the connection to the hue_d database.

```
psql -h localhost -U hue_u -d hue_d
Password for user hue_u:
hue=> \q
```



Note: If you cannot connect, try typing the command manually. The hyphens may become corrupted when copied.


Connect Hue Service to PostgreSQL

Tip: To save the data in your current database (embedded or external), you must migrate (dump, synch, load) before connecting to the new database. Otherwise, skip those steps.

1. Stop Hue Service

- a. In Cloudera Manager, navigate to **Cluster > Hue**.
- b. Select **Actions > Stop**.



Note: If necessary, refresh the page to ensure the Hue service is stopped: .

2. [migration only] Dump Current Database

- a. Select **Actions > Dump Database**.
- b. Click **Dump Database**. The file is written to `/tmp/hue_database_dump.json` on the host of the Hue server.
- c. Log on to the *host of the Hue server* in a command-line terminal.
- d. Edit `/tmp/hue_database_dump.json` by removing all objects with `useradmin.userprofile` in the `model` field. For example:

```
# Count number of objects
grep -c useradmin.userprofile /tmp/hue_database_dump.json
```

```
vi /tmp/hue_database_dump.json
```

```
{
  "pk": 1,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:06:13",
    "creation_method": "HUE",
    "first_login": false,
    "user": 1,
    "home_directory": "/user/admin"
  }
},
{
  "pk": 2,
  "model": "useradmin.userprofile",
```

```

"fields": {
  "last_activity": "2016-10-03T10:27:10",
  "creation_method": "HUE",
  "first_login": false,
  "user": 2,
  "home_directory": "/user/alice"
},
},

```

3. Connect to New Database

- Go to Hue > Configuration.
- Filter by category, **Database**.
- Set the following database parameters :

```

DB Hostname = <fqdn of host with postgres server>:5432
DB Type     = <PostgreSQL>
DB Name     = hue_d
Username    = hue_u
Password    = <hue database password set when granting hue permissions>

```

- Click **Save Changes**.

4. [migration only] Synchronize New Database

- Select **Actions > Synchronize Database**
- Click **Synchronize Database**.

5. [migration only] Load Data from Old Database

- Log on to the *host of the PostgreSQL server* in a command-line terminal.

```

psql -h localhost -U hue_u -d hue_d
Password for user hue_u: <hue user password>

```

- Drop the foreign key constraint from the `auth_permission` table in the hue database.

```

hue=# \d auth_permission;
hue=# ALTER TABLE auth_permission DROP CONSTRAINT content_type_id_refs_id_id value;

```

- Clean the table, `django_content_type`.

```

hue=# TRUNCATE django_content_type CASCADE;

```

```

hue=> \d auth_permission;

```

Column	Type	Modifiers
id	integer	not null default nextval('auth_permission_id_seq'::regclass)
name	character varying(50)	not null
content_type_id	integer	not null
codename	character varying(100)	not null

```

Indexes:
  "auth_permission_pkey" PRIMARY KEY, btree (id)
  "auth_permission_content_type_id_codename_key" UNIQUE CONSTRAINT, btree (content_type_id, codename)
  "auth_permission_content_type_id" btree (content_type_id)
Foreign-key constraints:
  "content_type_id_refs_id_d043b34a" FOREIGN KEY (content_type_id) REFERENCES django_content_type(id) DEFERRABLE INITIALLY DEFERRED
Referenced by:
  TABLE "auth_group_permissions" CONSTRAINT "auth_group_permissions_permission_id_fkey" FOREIGN KEY (permission_id) REFERENCES auth_permission(id)
  TABLE "auth_user_user_permissions" CONSTRAINT "auth_user_user_permissions_permission_id_fkey" FOREIGN KEY (permission_id) REFERENCES auth_permission(id)

```

- In Cloudera Manager, load the JSON file: select **Actions > Load Database** and click **Load Database**.

Tip: If you are blocked by a duplicate key value such as this:

```

django.db.utils.IntegrityError: Problem installing fixture '/tmp/hue_database_dump.json':
Could not load desktop.DocumentTag(pk=1): duplicate key value violates unique constraint

```

```
"desktop_documenttag_owner_id_1d5f76680ee9998b_uniq"  
DETAIL: Key (owner_id, tag)=(1100713, default) already exists.
```

Delete that value and try loading again, for example:

```
DELETE FROM desktop_documenttag WHERE owner_id = '1100713' and tag = 'default';
```

e. Add the foreign key back (still logged on to the Hue database):

```
ALTER TABLE auth_permission ADD FOREIGN KEY (content_type_id) REFERENCES  
django_content_type (id);
```

6. Start Hue service

- a. Navigate to **Cluster > Hue**, if not already there.
- b. Select **Actions > Start**.
- c. Click **Start**.
- d. Click **Hue Web UI** to log on to Hue with a custom PostgreSQL database.

Connect Hue to Oracle with Client Parcel

To connect to an Oracle database, Hue needs Oracle client libraries (Basic and SDK). These are available from Oracle as packages (zip files) or from Cloudera as a parcel (for CDH parcel deployments).

This page covers connecting with the Oracle client parcel.



Important: Currently, Cloudera only provides a parcel for the Oracle 11 client (which works with the Oracle 12 server). For the Oracle 12 client package (which can be used for either CDH parcel or package deployments), see [Connect Hue to Oracle with Client Package](#) on page 28.

Install and Configure Oracle Server

Refer to the [Oracle documentation](#) for help on how to install an Oracle database.

Tip: Daniel Westermann has a helpful blog post: [a simple script to automate the oracle 12c setup](#).

Set Environment Variables

1. Set all necessary Oracle environment variables. For example:

```
## Example Environment Variables  
VERSION=12.1.0.2  
ORACLE_HOSTNAME=<your hostname>  
ORACLE_BASE=/ora01/app/oracle/product/base  
ORACLE_HOME=${ORACLE_BASE}/${VERSION}  
ORACLE_SID=orcl  
ORAOWNER_BIN=/home/oracle/bin  
LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${LD_LIBRARY_PATH}
```

2. Ensure that your shell `.profile` resembles:

```
## Example from /home/oracle/.bash_profile  
TMP=/tmp  
ORACLE_HOSTNAME=<your hostname>  
ORACLE_BASE=/ora01/app/oracle/product/base  
ORACLE_HOME=/ora01/app/oracle/product/base/12.1.0.2  
ORACLE_SID=orcl  
ORAOWNER_BIN=/home/oracle/bin  
LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${LD_LIBRARY_PATH}  
PATH=${ORACLE_HOME}/bin:${ORAOWNER_BIN}:${PATH}
```

```
CLASSPATH=${ORACLE_HOME}/jlib:${ORACLE_HOME}/rdbms/jlib;
export ORACLE_HOSTNAME ORACLE_BASE ORACLE_HOME ORACLE_SID LD_LIBRARY_PATH PATH CLASSPATH
TMP
```

Configure Character Set

1. Log on as the oracle user:

```
su - oracle
```

2. Start the listener control (as user oracle):

```
$ORACLE_HOME/bin/lsnrctl start
```

3. Log on to SQL*Plus:

```
sqlplus / as sysdba
```

4. Ensure character set is AL32UTF8 and national character set is UTF8:

```
SELECT * FROM v$nls_parameters where parameter like '%CHARACTERSET';
```

To update, **quit the shell** and run these commands in a SQL*Plus script:

```
vi alter_charset.ddl
```

```
## Save in alter_charset.ddl (script takes 2-3 minutes)
CONNECT / as sysdba
SHUTDOWN immediate
STARTUP mount
ALTER SYSTEM ENABLE RESTRICTED SESSION;
ALTER SYSTEM SET JOB_QUEUE_PROCESSES=0 SCOPE = MEMORY;
ALTER SYSTEM SET AQ_TM_PROCESSES=0 SCOPE = MEMORY;
ALTER DATABASE OPEN;
ALTER DATABASE CHARACTER SET AL32UTF8;
ALTER DATABASE NATIONAL CHARACTER SET INTERNAL_USE UTF8;
SHUTDOWN immediate
STARTUP
```

```
sqlplus /nolog < alter_charset.ddl
```

Create Hue Database

1. Create the hue schema, set quotas, and grant select permissions (do not grant all):

Tip: Oracle 12 users must [ALTER session set](#) to avoid creating a [common user](#) with prefix, c##.

```
vi create_hue_database.ddl
```

```
## Save in create_hue_database.ddl
## Change huepassword to something more secure
CONNECT / as sysdba
ALTER session set "_ORACLE_SCRIPT"=true;

DROP user hue cascade;
CREATE user hue identified by huepassword;
ALTER user hue quota 1000m on users;
ALTER user hue quota 100m on system;
GRANT create sequence to hue;
GRANT create session to hue;
GRANT create table to hue;
GRANT create view to hue;
```

```
GRANT create procedure to hue;
GRANT create trigger to hue;
GRANT execute on sys.dbms_crypto to hue;
GRANT execute on sys.dbms_lob to hue;
```

```
sqlplus /nolog < create_hue_database.ddl
```

2. Verify that you can connect to hue:

```
sqlplus hue/<your hue password>
```

3. Clean all hue user tables. Create a script to spool delete statements into a new file, delete_from_tables.ddl:

```
vi spool_statements.ddl
```

```
## Save in spool_statements.ddl (which generates delete_from_tables.ddl)
spool delete_from_tables.ddl
set pagesize 100;
SELECT 'DELETE FROM ' || table_name || ';' FROM user_tables;
commit;
spool off
quit
```

```
## Create delete_from_tables.ddl
sqlplus hue/<your hue password> < spool_statements.ddl
```

```
## Run delete_from_tables.ddl
sqlplus hue/<your hue password> < delete_from_tables.ddl
```

```
[oracle@oracle12c-centos68 ~]$ sqlplus hue/huepassword < spool_statements.ddl
SQL*Plus: Release 12.1.0.2.0 Production on Fri Mar 10 10:58:59 2017
Copyright (c) 1982, 2014, Oracle. All rights reserved.
Last Successful login time: Fri Mar 10 2017 10:54:46 -08:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options

SQL> SQL> SQL>
'DELETEFROM' ||TABLE_NAME||';'
-----
DELETE FROM AUTH_PERMISSION;
DELETE FROM AUTH_GROUP_PERMISSIONS;
DELETE FROM AUTH_GROUP;
DELETE FROM AUTH_USER_GROUPS;
DELETE FROM AUTH_USER_USER_PERMISSIONS;
DELETE FROM AUTH_USER;
DELETE FROM DJANGO_OPENID_AUTH_NONCE;
DELETE FROM DJANGO_OPENID_AUTH_ASSOCIATION;
```

```

[oracle@oracle12c-centos68 ~]$ sqlplus hue/huepassword < delete_from_tables.ddl
SQL*Plus: Release 12.1.0.2.0 Production on Fri Mar 10 10:59:07 2017

Copyright (c) 1982, 2014, Oracle. All rights reserved.

Last Successful login time: Fri Mar 10 2017 10:58:59 -08:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options

SQL> SP2-0734: unknown command beginning "SQL> set p..." - rest of line ignored.
SQL> SP2-0734: unknown command beginning "SQL> SELEC..." - rest of line ignored.
SQL> SQL> SP2-0734: unknown command beginning "'DELETFRO..." - rest of line ignored.
SQL> SQL>
228 rows deleted.

SQL>
0 rows deleted.

SQL>
1 row deleted.

```

Create Oracle Client Parcel Repository

Cloudera provides the [Oracle Instant Client for Hue](#) (11.2 only) as a parcel for CDH parcel deployments.



Important: The Oracle 11 client works with the Oracle 12 server, but if you prefer the Oracle 12 client, see [Connect Hue to Oracle with Client Package](#) on page 28.



Important: There is currently no parcel support for Ubuntu 16 (xenial).

Oracle Instant Client for Hue

The Oracle Instant Client parcel for Hue enables Hue to be quickly and seamlessly deployed by Cloudera Manager with Oracle as its external database. For customers who have standardized on Oracle, this eliminates extra steps in installing or moving a Hue deployment on Oracle and allows for automated deployment of Hue on Oracle via the Cloudera Manager API.

Use of this software requires acceptance of the Cloudera Redistribution License Agreement for Oracle Instant Client. Please review the documentation for more information.

Get Started

OS Version

Rhel 7

Rhel 7

SLES 11

SLES 12

Thank you for downloading the Oracle Instant Client for Hue

Please [click here](#) to download the Oracle Instant Client parcel.

Please [click here](#) to download the manifest json required for installation.

The hash for this download is: cf3ae6dee6457362634be9a967a74d4315cb37b5

Download and Stage Oracle Instant Client Parcel

1. Point a browser to https://www.cloudera.com/downloads/oracle_instant_client_hue.html.
2. Select your OS and click **Get It Now!**
3. Check the box to accept **Cloudera's Standard Licence Agreement** and click **Submit**.

4. Download the parcel: `ORACLE_INSTANT_CLIENT-11.2-1.oracleinstantclient1.0.0.p0.130-<your linux distro>.parcel`.
5. Download the manifest for the mirrored repository.
6. Upload the parcel and manifest to the host with Cloudera Manager server, for example:

```
scp ORACLE_INSTANT_CLIENT-11.2-1* manifest.json root@<Cloudera Manager server hostname>:.
```

Install Asynchronous I/O Library

1. Log on to the host of Cloudera Manager server.
2. Install the Asynchronous I/O library, `libaio/libaio1`:

```
## CentOS/RHEL (yum), SLES (zypper), Ubuntu/Debian (apt-get)
sudo yum install -y libaio
#sudo zypper install -y libaio
#sudo apt-get install -y libaio1
```

Create Mirrored Parcel Repository

When manually adding parcels it is best to use mirrored repository as it preserves the metadata that enforces relation constraints.

1. Create a temporary repository , for example:

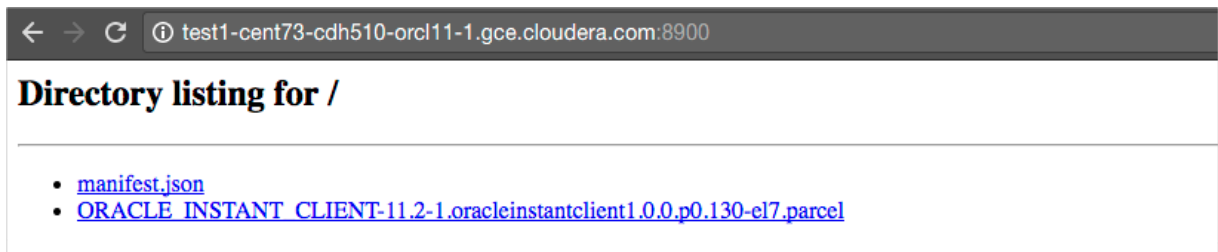
```
mkdir -pm 755 /var/www/html/cdh511
mv ~/ORACLE_INSTANT_CLIENT-11.2-1* ~/manifest.json /var/www/html/cdh511
```

2. Start a web server with any available port, for example:

```
cd /var/www/html/cdh511/
python -m SimpleHTTPServer 8900
```

3. Test the repository in a browser:

```
http://<server hostname>:8900/
```



[Optional]

In fact, the Oracle parcel does not have any constraints, but using a repository allows you to more easily connect to an Oracle database during a new CDH installation if necessary. It is also a best practice and not more work.

However, if you have an existing CDH installation, you *can* simply copy the parcel (in this case) and add a corresponding SHA-1 file to `/opt/cloudera/parcel-repo`.

You must have CDH installed because the directory, `parcel-repo`, is created during step 6 of a CDH parcel installation.

```
shasum ORACLE_INSTANT_CLIENT-11.2-1.oracleinstantclient1.0.0.p0.130-<your linux
distro>.parcel | awk '{ print $1 }' >
ORACLE_INSTANT_CLIENT-11.2-1.oracleinstantclient1.0.0.p0.130-<your linux
distro>.parcel.sha1
mv ORACLE_INSTANT_CLIENT* /opt/cloudera/parcel-repo/
```


Connect Hue Service to Oracle

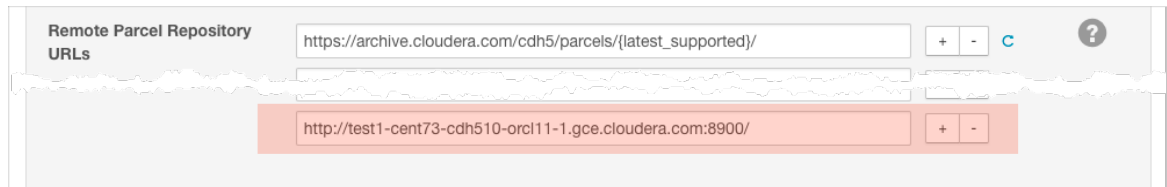
You can connect Hue to your Oracle database while installing CDH (and Hue) or with an existing installation. With existing CDH installations, you can connect and restart Hue, without saving the data in your current database, or you can migrate the old data into Oracle.

New CDH Installation

See [Installing Cloudera Manager and CDH](#) to install Cloudera Manager (and its Installation Wizard), which you will use here to install CDH and the Oracle client.

Install CDH and Oracle Parcel

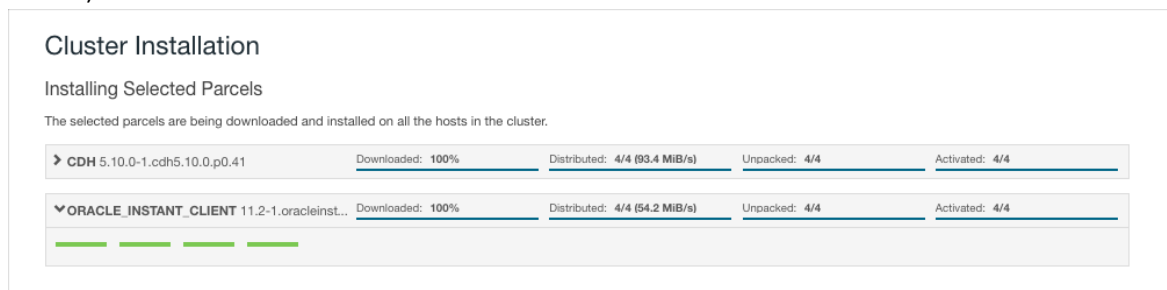
1. Open the Cloudera Manager Admin Console and run the [Cloudera Manager Installation Wizard](#) to install CDH (and Hue). The URL for Cloudera Manager is: `http://<cm server hostname>:7180`
2. Stop at **Select Repository** to add the Oracle client parcel repository (**Cluster Installation**, step 1):
 - a. Choose Method **Use Parcels** and click **More Options**.
 - b. **+**,
and add the URL for your Oracle **Remote Parcel Repository**:



- c. Click **Save Changes**.
- d. Select the newly added radio button by `ORACLE_INSTANT_CLIENT` and click **Continue**.



The Oracle parcel is downloaded, distributed, and activated at **Cluster Installation**, step 6 (**Installing Selected Parcels**).



Connect Hue to Oracle

Continuing with Cloudera Manager Installation Wizard ...

1. Stop at **Database Setup** to set connection properties (**Cluster Setup**, step 3).
 - a. Select **Use Custom Database**.

- b. Under **Hue**, set the connection properties to the Oracle database.



Note: Copy and store the password for the Hue embedded database (just in case).

```
Database Hostname (and port): <fqdn of host with Oracle server>:1521
Database Type (or engine): Oracle
Database SID (or name): orcl
Database Username: hue
Database Password: <hue database password>
```

- c. Click **Test Connection** and click **Continue** when successful.

2. Continue with the installation and click **Finish** to complete.
3. Add support for a multi-threaded environment:
 - a. Go to **Clusters > Hue > Configuration**.
 - b. Filter by Category, **Hue-service** and Scope, **Advanced**.
 - c. Add support for a multi-threaded environment by setting **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**:


```
[desktop]
[[database]]
options={"threaded":true}
```


- d. Click **Save Changes**.

4. Restart the Hue service: select **Actions > Restart** and click **Restart**.
5. Log on to Hue by clicking **Hue Web UI**.

Existing CDH Installation

Activate Oracle Client Parcel

1. Log on to Cloudera Manager.
2. Go to the **Parcels** page by clicking **Hosts > Parcels** (or clicking the parcels icon .
3. Click the **Configuration > Check for New Parcels**.
4. Find **ORACLE_INSTANT_CLIENT** and click **Download, Distribute, and Activate**.


Parcel Name	Version	Status	Actions
ORACLE_INSTANT_CLIENT	11.2-1.oracleinstantclient1.0.0.p0.130 	Distributed, Activated	Deactivate

Connect Hue to Oracle

If you are not migrating the current (or old) database, simply connect to your new Oracle database and restart Hue (steps [3](#) on page 8 and [6](#) on page 8).

1. [migration only] **Stop Hue Service**
 - a. In Cloudera Manager, navigate to **Cluster > Hue**.

b. Select Actions > Stop.

Note: If necessary, refresh the page to ensure the Hue service is stopped: .

2. [migration only] Dump Current Database**a. Select Actions > Dump Database.**

b. Click Dump Database. The file is written to `/tmp/hue_database_dump.json` on the host of the Hue server.

c. Log on to the *host of the Hue server* in a command-line terminal.

d. Edit `/tmp/hue_database_dump.json` by removing all objects with `useradmin.userprofile` in the `model` field. For example:

```
# Count number of objects
grep -c useradmin.userprofile /tmp/hue_database_dump.json
```

```
vi /tmp/hue_database_dump.json
```

```
{
  "pk": 1,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:06:13",
    "creation_method": "HUE",
    "first_login": false,
    "user": 1,
    "home_directory": "/user/admin"
  }
},
{
  "pk": 2,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:27:10",
    "creation_method": "HUE",
    "first_login": false,
    "user": 2,
    "home_directory": "/user/alice"
  }
},
}
```

3. Connect to New Database**a. Configure Database connections:**

- Go to **Hue > Configuration** and filter by category, **Database**.
- Set database properties and click **Save Changes**:

```
Hue Database Type (or engine): Oracle
Hue Database Hostname: <fqdn of host with Oracle server>
Hue Database Port: 1521
Hue Database Username: hue
Hue Database Password: <hue database password>
Hue Database Name (or SID): orcl
```

b. Add support for a multi-threaded environment:

- Filter by Category, **Hue-service** and Scope, **Advanced**.

- Set **Hue Service Advanced Configuration Snippet (Safety Valve)** for `hue_safety_valve.ini` and click **Save Changes**:

```
[desktop]
[[database]]
options={"threaded":true}
```

4. [migration only] Synchronize New Database

- a. Select **Actions > Synchronize Database**
- b. Click **Synchronize Database**.

5. [migration only] Load Data from Old Database



Important: All user tables in the Hue database must be empty. You cleaned them at step 3 on page 30 of [Create Hue Database](#) on page 29. Ensure they are still clean.

```
sqlplus hue/<your hue password> < delete_from_tables.ddl
```

6. Re/Start Hue service

- a. Navigate to **Cluster > Hue**.
- b. Select **Actions > Start**, and click **Start**.
- c. Click **Hue Web UI** to log on to Hue with a custom Oracle database.

Connect Hue to Oracle with Client Package

To connect to an Oracle database, Hue needs Oracle client libraries (Basic and SDK). These are available from Oracle as packages (zip files) or from Cloudera as a parcel (for CDH parcel deployments).

This page covers connecting with Oracle client packages.

Install and Configure Oracle Server

Refer to the [Oracle documentation](#) for help on how to install an Oracle database.

Tip: Daniel Westermann has a helpful blog post: [a simple script to automate the oracle 12c setup](#).

Set Environment Variables

1. Set all necessary Oracle environment variables. For example:

```
## Example Environment Variables
VERSION=12.1.0.2
ORACLE_HOSTNAME=<your hostname>
ORACLE_BASE=/ora01/app/oracle/product/base
ORACLE_HOME=${ORACLE_BASE}/${VERSION}
ORACLE_SID=orcl
ORAOWNER_BIN=/home/oracle/bin
LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${LD_LIBRARY_PATH}
```

2. Ensure that your shell `.profile` resembles:

```
## Example from /home/oracle/.bash_profile
TMP=/tmp
ORACLE_HOSTNAME=<your hostname>
ORACLE_BASE=/ora01/app/oracle/product/base
ORACLE_HOME=/ora01/app/oracle/product/base/12.1.0.2
ORACLE_SID=orcl
ORAOWNER_BIN=/home/oracle/bin
LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${LD_LIBRARY_PATH}
```

```
PATH=${ORACLE_HOME}/bin:${ORAOWNER_BIN}:${PATH}
CLASSPATH=${ORACLE_HOME}/jlib:${ORACLE_HOME}/rdbms/jlib;
export ORACLE_HOSTNAME ORACLE_BASE ORACLE_HOME ORACLE_SID LD_LIBRARY_PATH PATH CLASSPATH
TMP
```

Configure Character Set

1. Log on as the oracle user:

```
su - oracle
```

2. Start the listener control (as user oracle):

```
${ORACLE_HOME}/bin/lsnrctl start
```

3. Log on to SQL*Plus:

```
sqlplus / as sysdba
```

4. Ensure character set is AL32UTF8 and national character set is UTF8:

```
SELECT * FROM v$nls_parameters where parameter like '%CHARACTERSET';
```

To update, **quit the shell** and run these commands in a SQL*Plus script:

```
vi alter_charset.ddl
```

```
## Save in alter_charset.ddl (script takes 2-3 minutes)
CONNECT / as sysdba
SHUTDOWN immediate
STARTUP mount
ALTER SYSTEM ENABLE RESTRICTED SESSION;
ALTER SYSTEM SET JOB_QUEUE_PROCESSES=0 SCOPE = MEMORY;
ALTER SYSTEM SET AQ_TM_PROCESSES=0 SCOPE = MEMORY;
ALTER DATABASE OPEN;
ALTER DATABASE CHARACTER SET AL32UTF8;
ALTER DATABASE NATIONAL CHARACTER SET INTERNAL_USE UTF8;
SHUTDOWN immediate
STARTUP
```

```
sqlplus /nolog < alter_charset.ddl
```

Create Hue Database

1. Create the hue schema, set quotas, and grant select permissions (do not grant all):

Tip: Oracle 12 users must [ALTER session set](#) to avoid creating a [common user](#) with prefix, c##.

```
vi create_hue_database.ddl
```

```
## Save in create_hue_database.ddl
## Change huepassword to something more secure
CONNECT / as sysdba
ALTER session set "_ORACLE_SCRIPT"=true;

DROP user hue cascade;
CREATE user hue identified by huepassword;
ALTER user hue quota 1000m on users;
ALTER user hue quota 100m on system;
GRANT create sequence to hue;
GRANT create session to hue;
GRANT create table to hue;
```

```
GRANT create view to hue;  
GRANT create procedure to hue;  
GRANT create trigger to hue;  
GRANT execute on sys.dbms_crypto to hue;  
GRANT execute on sys.dbms_lob to hue;
```

```
sqlplus /nolog < create_hue_database.ddl
```

2. Verify that you can connect to hue:

```
sqlplus hue/<your hue password>
```

3. Clean all hue user tables. Create a script to spool delete statements into a new file, delete_from_tables.ddl:

```
vi spool_statements.ddl
```

```
## Save in spool_statements.ddl (which generates delete_from_tables.ddl)  
spool delete_from_tables.ddl  
set pagesize 100;  
SELECT 'DELETE FROM ' || table_name || ';' FROM user_tables;  
commit;  
spool off  
quit
```

```
## Create delete_from_tables.ddl  
sqlplus hue/<your hue password> < spool_statements.ddl
```

```
## Run delete_from_tables.ddl  
sqlplus hue/<your hue password> < delete_from_tables.ddl
```

```
[oracle@oracle12c-centos68 ~]$ sqlplus hue/huepassword < spool_statements.ddl  
  
SQL*Plus: Release 12.1.0.2.0 Production on Fri Mar 10 10:58:59 2017  
  
Copyright (c) 1982, 2014, Oracle. All rights reserved.  
  
Last Successful login time: Fri Mar 10 2017 10:54:46 -08:00  
  
Connected to:  
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production  
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options  
  
SQL> SQL> SQL>  
'DELETEFROM' || TABLE_NAME || ';' ;  
-----  
DELETE FROM AUTH_PERMISSION;  
DELETE FROM AUTH_GROUP_PERMISSIONS;  
DELETE FROM AUTH_GROUP;  
DELETE FROM AUTH_USER_GROUPS;  
DELETE FROM AUTH_USER_USER_PERMISSIONS;  
DELETE FROM AUTH_USER;  
DELETE FROM DJANGO_OPENID_AUTH_NONCE;  
DELETE FROM DJANGO_OPENID_AUTH_ASSOCIATION;
```

```

[oracle@oracle12c-centos68 ~]$ sqlplus hue/huepassword < delete_from_tables.ddl
SQL*Plus: Release 12.1.0.2.0 Production on Fri Mar 10 10:59:07 2017

Copyright (c) 1982, 2014, Oracle. All rights reserved.

Last Successful login time: Fri Mar 10 2017 10:58:59 -08:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options

SQL> SP2-0734: unknown command beginning "SQL> set p..." - rest of line ignored.
SQL> SP2-0734: unknown command beginning "SQL> SELEC..." - rest of line ignored.
SQL> SQL> SP2-0734: unknown command beginning "'DELETEFRO..." - rest of line ignored.
SQL> SQL>
228 rows deleted.

SQL>
0 rows deleted.

SQL>
1 row deleted.

```

Install Oracle Client Package

Cloudera Manager requires the Oracle instant client libraries to be in `/usr/share/oracle/instantclient/lib/`. The following commands arrange the files as such.



Important: You must add client libraries to *each machine that hosts a Hue server*.

Install Asynchronous I/O Library

1. Log on to the host of Cloudera Manager server.
2. Install the Asynchronous I/O library, `libaio/libaio1`:

```

## CentOS/RHEL (yum), SLES (zypper), Ubuntu/Debian (apt-get)
sudo yum install -y libaio
#sudo zypper install -y libaio
#sudo apt-get install -y libaio1

```

Install Oracle Client

1. Download zip files for [Instant Client Package, Basic and SDK \(with headers\)](#).
2. For this step, switch to the host with the downloaded files and upload zip to the Cloudera Manager server host:

```
scp instantclient-*.zip root@<CM server hostname>:.
```

Version 12.1.0.2.0

Instant Client Package - Basic: All files required to run OCI, OCCI, and JDBC-OCI applications

- Instant Client Package - Basic: [instantclient-basic-linux.x64-12.1.0.2.0.zip](#) (63,352,239 bytes) (cksum - 109893216)
- Instant Client Package - Basic: [oracle-instantclient12.1-basic-12.1.0.2.0-1.x86_64.rpm](#) (62,587,782 bytes) (cksum - 2840691603)

Instant Client Package - SDK: Additional header files and an example makefile for developing Oracle applications with Instant Client

- Instant Client Package - SDK: [instantclient-sdk-linux.x64-12.1.0.2.0.zip](#) (667,174 bytes) (cksum - 1047596065)
- Instant Client Package - SDK: [oracle-instantclient12.1-devel-12.1.0.2.0-1.x86_64.rpm](#) (634,803 bytes) (cksum - 2599726994)

3. Arrange the client libraries to mirror the tree structure in the image. Here is *one way* to do this:

```

# Create nested directories: /usr/share/oracle/instantclient/lib/
mkdir -pm 755 /usr/share/oracle/instantclient/lib

# Unzip. The files expand into /usr/share/oracle/instantclient/instantclient_<ver>/

```

```

unzip '*.zip' -d /usr/share/oracle/instantclient/

# Move lib files from instantclient_<ver> to /usr/share/oracle/instantclient/lib/
mv /usr/share/oracle/instantclient/`ls -l /usr/share/oracle/instantclient/ | grep
instantclient_ | awk '{print $9}'`/lib* /usr/share/oracle/instantclient/lib/

# Move rest of the files to /usr/share/oracle/instantclient/
mv /usr/share/oracle/instantclient/`ls -l /usr/share/oracle/instantclient/ | grep
instantclient_ | awk '{print $9}'`/* /usr/share/oracle/instantclient/

# Create symbolic links. Remember to edit version numbers as necessary
cd /usr/share/oracle/instantclient/lib
ln -s libclntsh.so.12.1 libclntsh.so
ln -s libocci.so.12.1 libocci.so

```

```

[root@test2-ec2-rhel73-cdh5100-1 instantclient]# tree /usr/share/oracle/instantclient/lib -C
/usr/share/oracle/instantclient/lib
├── libclntshcore.so.12.1
├── libclntsh.so -> libclntsh.so.12.1
├── libclntsh.so.12.1
├── libipcl.so
├── libmq11.so
├── libnzi2.so
├── libocci.so -> libocci.so.12.1
├── libocci.so.12.1
├── liboci1.so
├── libocijdbc12.so
├── libons.so
└── liboramysql12.so

```

4. Set \$ORACLE_HOME and \$LD_LIBRARY_PATH:

```

export ORACLE_HOME=/usr/share/oracle/instantclient
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME

```



Note: *If using the Oracle 11 instant client you are ready to Connect Hue to Oracle. Else if using the Oracle 12 instant client, upgrade the Python module, cx_Oracle.*

Apply Temporary Workaround for Oracle 12 Client

Update the `cx_Oracle` package in your native Python environment and copy it to Hue's Python environment.



Note: Hue supports `cx_Oracle` version 5.2.1 that you can use with Oracle Instant Client 12.1 or 12.2.

1. Install gcc and Python development tools:

```

## CentOS/RHEL (yum), SLES (zypper), Ubuntu/Debian (apt-get)
yum install -y python-setuptools python-devel gcc
#zypper install -y python-setuptools python-devel gcc
#apt-get install -y python-setuptools python-dev gcc

```

2. Install pip:

```

easy_install pip

```

3. Install cx_Oracle. Ensure that ORACLE_HOME and \$LD_LIBRARY_PATH are properly set so that pip knows which version to install.

```

echo $ORACLE_HOME $LD_LIBRARY_PATH

```

```

pip install cx_Oracle==5.2.1

```


Tip: You can also wget the proper `cx_Oracle` file yourself: https://pypi.python.org/pypi/cx_Oracle/.

4. Get the version of the new `cx_Oracle` package:

- CentOS/RHEL and SLES:

```
ls /usr/lib64/python2.7/site-packages/cx_Oracle*
```

- Ubuntu/Debian:

```
ls /usr/local/lib/python2.7/dist-packages/cx_Oracle*
```

5. If this is a [New CDH Installation](#) on page 33, stop here to run the first 5 or 6 steps of the Cloudera Manager Installation Wizard (packages=5, parcels=6). Do not go past **Cluster Installation**.

6. Navigate to Hue's python environment, `$HUE_HOME/build/env/lib/<python version>/site-packages`.

- CDH Parcel installation:

```
cd /opt/cloudera/parcels/`ls -l /opt/cloudera/parcels | grep CDH | tail -1 | awk '{print $9}'`/lib/hue/build/env/lib/python2.7/site-packages
```

- CDH package installation:

```
cd /usr/lib/hue/build/env/lib/python2.7/site-packages
```



Important: The parcel path is created during step 5 or 6 of **Cluster Installation**, so you must have completed this to continue.

7. Move the existing `cx_Oracle` file:

```
mv cx_Oracle-5.2.1-py2.7-linux-x86_64.egg cxfoo
```

8. Copy the new `cx_Oracle` module to Hue's python environment. The version can change:

- CentOS/RHEL and SLES:

```
cp -a /usr/lib64/python2.7/site-packages/cx_Oracle-5.3-py2.7.egg-info .
```

- Ubuntu/Debian


```
cp -a /usr/local/lib/python2.7/dist-packages/cx_Oracle-5.3.egg-info .
```

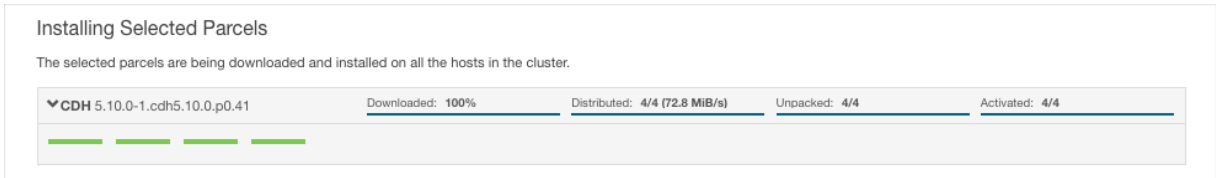
Connect Hue Service to Oracle

You can connect Hue to your Oracle database while installing CDH (and Hue) or with an existing installation. With existing CDH installations, you can connect and restart Hue, without saving the data in your current database, or you can migrate the old data into Oracle.

New CDH Installation


See [Installing Cloudera Manager and CDH](#) to install Cloudera Manager (and its Installation Wizard), which you will use here to install CDH and the Oracle client.

1. Open the Cloudera Manager Admin Console and run the [Cloudera Manager Installation Wizard](#) to install CDH (and Hue). The URL for Cloudera Manager is: `http://<cm server hostname>:7180`
2. Stop  at the end of **Cluster Installation** to copy the latest `cx_Oracle` package into Hue's Python environment.



3. Stop at **Database Setup** to set connection properties (**Cluster Setup**, step 3).

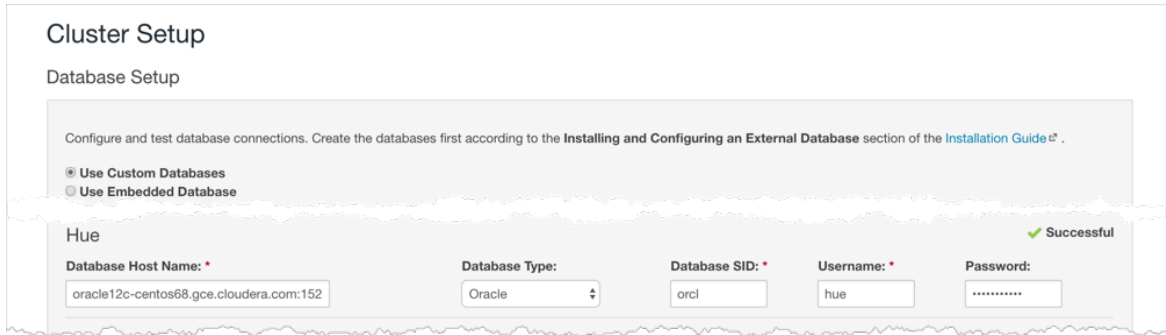
- a. Select **Use Custom Database**.
- b. Under **Hue**, set the connection properties to the Oracle database.



Note: Copy and store the password for the Hue embedded database (just in case).

```
Database Hostname (and port): <fqdn of host with Oracle server>:1521
Database Type (or engine): Oracle
Database SID (or name): orcl
Database Username: hue
Database Password: <hue database password>
```

c. Click **Test Connection** and click **Continue** when successful.



4. Continue with the installation and click **Finish** to complete.

5. Add support for a multi-threaded environment:

- a. Go to **Clusters > Hue > Configuration**.
- b. Filter by Category, **Hue-service** and Scope, **Advanced**.
- c. Add support for a multi-threaded environment by setting **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**:

```
[desktop]
[[database]]
options={"threaded":true}
```

d. Click **Save Changes**.

6. Restart the Hue service: select **Actions > Restart** and click **Restart**.


7. Log on to Hue by clicking **Hue Web UI**.

Existing CDH Installation

If you are not migrating the current (or old) database, simply connect to your new Oracle database and restart Hue (steps 3 on page 8 and 6 on page 8).

- 1. [migration only] **Stop Hue Service**
 - a. In Cloudera Manager, navigate to **Cluster > Hue**.
 - b. Select **Actions > Stop**.



Note: If necessary, refresh the page to ensure the Hue service is stopped: .

2. [migration only] Dump Current Database

- a. Select **Actions > Dump Database**.
- b. Click **Dump Database**. The file is written to `/tmp/hue_database_dump.json` on the host of the Hue server.
- c. Log on to the *host of the Hue server* in a command-line terminal.
- d. Edit `/tmp/hue_database_dump.json` by removing all objects with `useradmin.userprofile` in the `model` field. For example:

```
# Count number of objects
grep -c useradmin.userprofile /tmp/hue_database_dump.json
```

```
vi /tmp/hue_database_dump.json
```

```
{
  "pk": 1,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:06:13",
    "creation_method": "HUE",
    "first_login": false,
    "user": 1,
    "home_directory": "/user/admin"
  }
},
{
  "pk": 2,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:27:10",
    "creation_method": "HUE",
    "first_login": false,
    "user": 2,
    "home_directory": "/user/alice"
  }
},
}
```

3. Connect to New Database

- a. Configure Database connections: Go to **Hue > Configuration**, filter by **Database**, set properties, and click **Save Changes**:

```
Hue Database Type (or engine): Oracle
Hue Database Hostname: <fqdn of host with Oracle server>
Hue Database Port: 1521
Hue Database Username: hue
Hue Database Password: <hue database password>
Hue Database Name (or SID): orcl
```

- b. Add support for a multi-threaded environment: Filter by **Hue-service**, set **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**, and click **Save Changes**:

```
[desktop]
[[database]]
options={"threaded":true}
```

4. [migration only] Synchronize New Database

- a. Select **Actions > Synchronize Database**

b. Click **Synchronize Database**.

5. [migration only] **Load Data from Old Database**



Important: All user tables in the Hue database must be empty. You cleaned them at step 3 on page 30 of [Create Hue Database](#) on page 29. Ensure they are still clean.

```
sqlplus hue/<your hue password> < delete_from_tables.ddl
```

6. **Re/Start Hue service**

- a. Navigate to **Cluster > Hue**.
- b. Select **Actions > Start**, and click **Start**.
- c. Click **Hue Web UI** to log on to Hue with a custom Oracle database.

Migrate Hue Database



Note: [Hue Custom Databases](#) includes database-specific pages on how to migrate from an old to a new database. This page summarizes across supported database types.


When you change Hue databases, you *can* migrate the existing data to your new database. If the data is dispensable, there is no need to migrate.

The Hue database stores things like user accounts, Hive queries, and Oozie workflows, and you may have accounts, queries, and workflows worth saving. See [How to Populate the Hue Database](#) on page 43.


Migrating your existing database currently requires some work-arounds (in parentheses):

- Stop the Hue service.
- Dump database (and delete "useradmin.userprofile" objects from `.json` file).
- Connect to new database.
- Synchronize database (and drop foreign key to clean tables).
- Load database (and add foreign key).
- Start Hue service.

Dump Database

1. In the **Hue Web UI**, click the home icon  to see what documents you are migrating.
2. In Cloudera Manager, stop the Hue service: go to **Hue** and select **Actions > Stop**.



Note: Refresh the page to ensure that the Hue service is stopped: .

3. Select **Actions > Dump Database** and click **Dump Database**. The file is written to `/tmp/hue_database_dump.json` on the host of the Hue server.
4. Log on to the host of the *Hue server* in a command-line terminal. You can find the hostname on the Dump Database window and at **Hue > Hosts**.

5. Edit `/tmp/hue_database_dump.json` by removing all objects with `useradmin.userprofile` in the `model` field. For example:

```
# Count number of objects
grep -c useradmin.userprofile /tmp/hue_database_dump.json
```

```
vi /tmp/hue_database_dump.json
```

```
{
  "pk": 1,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:06:13",
    "creation_method": "HUE",
    "first_login": false,
    "user": 1,
    "home_directory": "/user/admin"
  }
},
```

Connect New Database

In Cloudera Manager, connect Hue to the new database. See [Hue Custom Databases](#) for help on installing and configuring a custom database.

1. Go to **Hue > Configuration**.
2. Filter by category, **Database**.
3. Set the appropriate database parameters :

```
Hue Database Type: MySQL or PostgreSQL or Oracle
Hue Database Hostname: <fqdn of host with database server>
Hue Database Port: 3306 or 5432 or 1521
Hue Database Username: <hue database username>
Hue Database Password: <hue database password>
Hue Database Name: <hue database name or SID>
```

4. Click **Save Changes**.
5. **Oracle users only** should add support for a multithreaded environment:
 - a. Filter by Category, **Hue-service** and Scope, **Advanced**.
 - b. Add support for a multithreaded environment by setting **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**:

```
[desktop]
[[database]]
options={"threaded":True}
```

- c. Click **Save Changes**.

Synchronize and Load

1. **Synchronize**: select **Actions > Synchronize Database** and click **Synchronize Database**.
2. Log on to the host of the *database* server in a command-line terminal and clean tables:
 - [MySQL](#) and [PostgreSQL](#) on page 39 users remove a foreign key from `auth.permission` and clean `django_content_type`.
 - [Oracle](#) on page 39 users delete content from all tables.
3. **Load**: select **Actions > Load Database** and click **Load Database**.
4. Return to the host of the database server:

- [MySQL](#) and [PostgreSQL](#) on page 39 users add the foreign key to `auth_permission`.

5. **Start:** select **Actions** > **Start** and click **Start**.



Note: Refresh the page to ensure that the Hue service is running: ●.

6. In the **Hue Web UI**, click the home icon to ensure that all documents were migrated.

MariaDB / MySQL

1. Synchronize Database in Cloudera Manager.
2. Log on to MySQL:

```
mysql -u root -p
Enter password: <root password>
```

3. Drop the foreign key constraint from the `hue.auth_permission` table:

- Execute the following statement to find the `content_type_id_refs_id_<value>` in the `CONSTRAINT` clause of the `CREATE TABLE` statement for the `hue.auth_permission` table:

```
SHOW CREATE TABLE hue.auth_permission;
```

This `SHOW CREATE TABLE` statement produces output similar to the following:

```
| auth_permission | CREATE TABLE 'auth_permission' (
  'id' int(11) NOT NULL AUTO-INCREMENT,
  'name' varchar(50) NOT NULL,
  'content_type_id' int(11) NOT NULL,
  'CODENAME' VARCHAR(100) NOT NULL,
  PRIMARY KEY ('id'),
  UNIQUE KEY 'content_type_id' ('content_type_id', 'codename'),
  KEY 'auth_permission_37ef4eb4' ('content_type_id'),
  CONSTRAINT 'content_type_id_refs_id_d043b34a' FOREIGN KEY ('content_type_id')
  REFERENCES 'django_content_type' ('id')
) ENGINE=InnoDB AUTO_INCREMENT=229 DEFAULT CHARSET=utf8 |
```

- Then execute the following statement to drop the foreign key constraint:

```
ALTER TABLE hue.auth_permission DROP FOREIGN KEY
content_type_id_refs_id_<value>;
```

For example, if you used the above output from the `SHOW CREATE TABLE` statement, you would use the following `ALTER TABLE` statement:

```
ALTER TABLE hue.auth_permission DROP FOREIGN KEY
content_type_id_refs_id_d043b34a;
```

4. Delete the contents of `django_content_type`:

```
DELETE FROM hue.django_content_type;
```

```
mysql> DELETE FROM hue.django_content_type;
Query OK, 76 rows affected (0.00 sec)
```

5. Load Database in Cloudera Manager.

6. Add the foreign key, `content_type_id`, to `auth_permission`:

```
ALTER TABLE hue.auth_permission ADD FOREIGN KEY (content_type_id) REFERENCES
django_content_type (id);
```

```
mysql> ALTER TABLE hue.auth_permission ADD FOREIGN KEY (content_type_id) REFERENCES django_content_type (id);
Query OK, 228 rows affected (0.01 sec)
Records: 228 Duplicates: 0 Warnings: 0
```

7. Start Hue in Cloudera Manager.

PostgreSQL

1. Synchronize Database in Cloudera Manager.

2. Log on to PostgreSQL:

```
psql -h localhost -U hue -d hue
Password for user hue:
```

3. Drop the foreign key constraint from `auth_permission`:

```
\d auth_permission;
ALTER TABLE auth_permission DROP CONSTRAINT content_type_id_refs_id_<id value>;
```

4. Delete the contents of `django_content_type`:

```
TRUNCATE django_content_type CASCADE;
```

5. Load Database in Cloudera Manager.

6. Add the foreign key, `content_type_id`, to `auth_permission`:

```
ALTER TABLE auth_permission ADD FOREIGN KEY (content_type_id) REFERENCES
django_content_type(id) DEFERRABLE INITIALLY DEFERRED;
```

7. Start Hue in Cloudera Manager.

Oracle

Oracle users should delete all content from the Oracle tables after synchronizing and before loading:

1. Synchronize Database in Cloudera Manager.

2. Log on to Oracle:

```
su - oracle
sqlplus / as sysdba
```

3. Grant a quota to the tablespace where tables are created (the default is SYSTEM). For example:

```
ALTER USER hue quota 100m on system;
```

4. Log on as the hue:

```
sqlplus hue/<hue password>
```

5. Create a spool script that creates a delete script to clean the content of all tables.

```
vi spool_statements.ddl
```

```
## Save in spool_statements.ddl (which generates delete_from_tables.ddl)
spool delete_from_tables.ddl
set pagesize 100;
SELECT 'DELETE FROM ' || table_name || ';' FROM user_tables;
commit;
spool off
quit
```

6. Run both scripts:

```
## Create delete_from_tables.ddl
sqlplus hue/<your hue password> < spool_statements.ddl

## Run delete_from_tables.ddl
sqlplus hue/<your hue password> < delete_from_tables.ddl
```

7. Load Database in Cloudera Manager.
8. Start Hue in Cloudera Manager.

Hue Custom Database Tutorial

This page explains how to configure Hue with a custom database *from end to end* by migrating your existing database and syncing to a new custom database. Learn how to switch databases for:

- A **new installation** of CDH, with the **Cloudera Manager Installation Wizard**
- An **existing installation** of CDH, with the **Cloudera Manager Admin Console**.



Note: On this page we use **CentOS 6** with **MySQL**. For instructions on other platforms and databases, see [Hue Databases](#).

Prepare Hosts

Create, or prepare, five machines, each with CentOS 6 and at least 8 GB of RAM:

1. Create a cluster of four machines. Name them `cdh-cluster-[1-4].<your domain>.com`.
2. Create one machine for the database. Name it `cdh-db.<your domain>.com`.

Separating the database from the CDH cluster is a best practice, but if necessary, you can install it on one of the hosts in the cluster (for example, `cdh-cluster-1`).

Install Custom Database

Install MySQL on the single machine you designated for this purpose (`cdh-db.<your domain>.com`).

1. Install MySQL server on `cdh-db.<your domain>.com`:

```
sudo yum install -y mysql-server
```

2. Start the server:

```
sudo service mysqld start
```


3. Secure your installation:

```
sudo /usr/bin/mysql_secure_installation
```

```
Enter current password for root (enter for none): [Press Enter if the password is unset]
OK, successfully used password, moving on...
[...]
Set root password? [Y/n] Y
New password:
Re-enter new password:
Remove anonymous users? [Y/n] Y
[...]
Disallow root login remotely? [Y/n] N
[...]
Remove test database and access to it [Y/n] Y
[...]
Reload privilege tables now? [Y/n] Y
```

4. Configure /etc/my.cnf:

```
[mysqld]
...
bind-address=0.0.0.0
default-storage-engine=innodb
sql_mode=STRICT_ALL_TABLES
```

5. Restart the server

```
sudo service mysqld restart
```

6. Log on with your new root password:

```
mysql -u root -p<root password>
```

7. Create the hue database with UTF8 collation and configure the hue user (with your own password):

```
create database hue collate = 'utf8_general_ci';
grant all on hue.* to 'hue'@'%' identified by 'huepassword';
quit
```

Install CM and CDH

In this section, we test connecting to a custom database with the installation wizard; then we undo the connection so we can connect with the admin console in [Dump, Synchronize, and Load](#) on page 41.

When you run the Cloudera Manager Installation Wizard, stop at the **Database Setup** page.

See [Installing Cloudera Manager and CDH](#).

Populate Database (optional)


[Populate the Hue database](#) with user account information, a Hive query, and an Oozie workflow (to ensure that the database migration works).

Dump, Synchronize, and Load

To connect to other supported databases, see [Hue Custom Databases](#).

1. Stop the Hue service: go to **Hue** and select **Actions > Stop**.



Note: Refresh the page if the Hue service does not look stopped: .

2. Dump the existing database:

- a. Select **Actions > Dump Database**.
- b. Click **Dump Database**. The file is written to `/tmp/hue_database_dump.json` on the host of the Hue server.
- c. Log on to the *host of the Hue server* in a command-line terminal.
- d. Edit `/tmp/hue_database_dump.json` by removing all objects with `useradmin.userprofile` in the `model` field. For example:

```
# Count number of objects
grep -c useradmin.userprofile /tmp/hue_database_dump.json
```

```
vi /tmp/hue_database_dump.json
```

```
{
  "pk": 1,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:06:13",
    "creation_method": "HUE",
    "first_login": false,
    "user": 1,
    "home_directory": "/user/admin"
  }
},
{
  "pk": 2,
  "model": "useradmin.userprofile",
  "fields": {
    "last_activity": "2016-10-03T10:27:10",
    "creation_method": "HUE",
    "first_login": false,
    "user": 2,
    "home_directory": "/user/alice"
  }
},
}
```

3. Connect Hue to the new MySQL database:

- a. Go to **Hue > Configuration**.
- b. Filter by category, **Database**.
- c. Set the following database parameters :

```
DB Hostname = <fqdn of host with postgres server>:3306
DB Type     = <PostgreSQL>
DB Name     = hue
Username    = hue
Password    = <hue database password set when granting hue permissions>
```

- d. Click **Save Changes**.

4. Synchronize the new database: select **Actions > Synchronize Database** and click **Synchronize Database**.

5. Load the database after removing the foreign key constraint:

- a. Log on to the *host of the MySQL server* in a command-line terminal.

- b. Delete the foreign key constraint and clean the table, `django_content_type`:

```
mysql -u root -p
```

```
SHOW CREATE table hue.auth_permission;
ALTER TABLE hue.auth_permission DROP FOREIGN KEY content_type_id_refs_id_<input id>;
```

```
DELETE FROM hue.django_content_type;
```

```
| auth_permission | CREATE TABLE `auth_permission` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  `content_type_id` int(11) NOT NULL,
  `codename` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `content_type_id` (`content_type_id`,`codename`),
  KEY `auth_permission_37ef4eb4` (`content_type_id`),
  CONSTRAINT `content_type_id_refs_id_d043b34a` FOREIGN KEY (`content_type_id`) REFERENCES `django_content_type` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=217 DEFAULT CHARSET=latin1 |
```

- c. In Cloudera Manager, load the JSON file: select **Actions** > **Load Database** and click **Load Database**.
d. Add the foreign key back:

```
ALTER TABLE hue.auth_permission ADD FOREIGN KEY (content_type_id) REFERENCES
django_content_type (id);
```

6. Start the Hue service: select **Actions** > **Start** and click **Start**. If you went through [Use Hue](#), ensure your data was migrated properly.

How to Populate the Hue Database

Not every action in the Hue UI touches the Hue database (embedded or custom). This page explains how to populate the database with user account information, Hive queries, and Oozie workflows. This is useful when testing the [migration of a database](#).






1. Add New User (Alice)

- Log on to Hue as the administrator.
- Open the Administration drop down and select **Manage Users**.
- Click **Add user** and follow the three steps.
 - Add a username (for example, "Alice") and password and click **Next**.
 - Ensure Alice belongs to the default group and click **Next**.
 - Give Alice `Superuser` status (for Hue, *not* HDFS) and click **Add user**.
- Log out as the administrator and log on as Alice.

2. Save Hive Query (customers.sql)

- Go to **About Hue** > **Quick Start** by clicking the Hue logo.
- Click the **Examples** tab ("Step 2").
- Click download **Hive** to install sample databases.
- Go to the **Metastore Manager** (or **Data Browser** > **Metastore Tables**).
- Click the default database and `customers` (sample) table.
- Click **Browse Data** to automatically generate a `select *` query in the **Hive** editor.
- Run the query with your cursor in the editor and **CTRL + Enter**, or by clicking the Run icon ▶.
- Save the query as `customers.sql` by clicking the **Save** icon 📄.
- View the query on the **Saved Queries** tab in the Hive editor.

3. Save Oozie Workflow (Customers Workflow)

- a. Go to Oozie by selecting, **Workflows > Editors > Workflows**.
- b. Click the **Create** button.
- c. Rename "My Workflow" as "Customers Workflow" and click the **Save** icon 
- d. Drag the action icon for **Saved Hive Query**  to the field, "Drop your action here."
- e. Select a saved query (`customers.sql`) from the drop down and click **Add**.
- f. Save the workflow by clicking the **Save**  icon.
- g. Submit the workflow by clicking the icon and clicking **Submit**. You should see the workflow status change to **SUCCEEDED** 
- h. View the saved workflow (and all documents) by clicking the home icon .

Hue Administration

This section consolidates administration and configuration documents related to Hue that live across the Cloudera document set.

- [Supported Browsers for Hue](#)
- [Administering Hue](#)
- [Adding a Hue Service and Role Instance](#)
- [Enabling Hue Applications Using Cloudera Manager](#)
- [Managing Hue Analytics Data Collection](#)
- [Configuring CDH Components for Hue](#)
- [Hue Configuration](#)
- [Using Hue with Cloudera Search](#)

Hue Security

New Hue Security documents:

- [Configure Hue for High Availability](#) on page 46
- [Authenticate Hue with LDAP](#) on page 49
- [Synchronize Hue with LDAP](#) on page 55

Existing Hue security documents that live across the Cloudera document set.

- [Hue Authentication](#)
- [Configuring Kerberos Authentication for Hue](#)
- [Integrating Hue with LDAP](#)
- [Configuring Hue for SAML](#)
- [Configuring TLS/SSL for Hue](#)
- [Hue High Availability](#)
- [Configuring Other CDH Components to Use HDFS HA](#)

Configure Hue for High Availability

Configuring Hue for High Availability (HA) means configuring Hue, Hive, and Impala.

Configure Hue for High Availability

Prerequisites

- **SSH network access** to host machines with an Hue Server/Kerberos Ticket Renewer role.
- **External database** configured for each Hue Server. See [Hue Databases](#).

Add Hue Roles

Hue HA requires at least two Hue server roles and one Load Balancer role. If the cluster is authenticating with Kerberos, you need one Kerberos Ticket Renewer on each host with a Hue Server.

1. Log on to Cloudera Manager and go to the **Hue** service.
2. Go to the **Hue** service and select **Actions > Add Role Instances**.
3. Click **Hue Server**, assign to one or more hosts, and click **OK > Continue**.
4. Click **Kerberos Ticket Renewer**, assign to each host with a Hue Server, and click **OK > Continue**.
5. Click **Load Balancer**, assign to one or more hosts, and click **OK > Continue**.
6. Check each role and select **Actions for Selected > Start** and click **Start**.

Enable TLS for Hue Load Balancer



Note: You can configure the Load Balancer for TLS/SSL *or* each endpoint (H2S, Impalad).

1. Go to **Hue > Configuration** and search on **TLS/SSL**.
2. Check **Enable TLS/SSL for Hue** for the **Hue Server Default Group**.
3. Set other TLS/SSL properties appropriate for your setup. Some to consider are:
 - **Hue Load Balancer Port** - Apache Load Balancer listens on this port (default is 8889).
 - **Path to TLS/SSL Certificate File** - Must be multi-domain with CN = Load Balancer in PEM format.
 - **Path to TLS/SSL Private Key File** - Must be in PEM format.

4. Click **Save Changes** and **Restart Hue**.

Configure Hive and Impala for High Availability



Note: Hive must have two or more HS2 roles, and Impala two or more ImpalaD roles.

Prerequisites & Requirements

- **SSH network access** to host machines with a HiveServer2 or Impala Daemon role.
- **External database** configured for each H2S and Impala Daemon.
- **Hue Load Balancer** Hive/Impala Load Balancer configured with Source IP Persistence.

Source IP Persistence

Without IP Persistence, you may encounter the error, **“Results have expired, rerun the query if needed.”**

Hue supports High Availability through a "load balancer" to HiveServer2 and Impala. Because the underlying **Hue thrift libraries reuse TCP connections in a pool**, a single user session may *not* have the same TCP connection. If a TCP connection is balanced away from a HiveServer2 or Impalad instance, the user session and its queries (running or returned) can be lost and trigger the "Results have expired" error.

To prevent sessions from being lost, configure the Hive/Impala Load Balancer with **Source IP Persistence** so that each Hue instance sends all traffic to a single HiveServer2/Impala instance. Of course, this is not true load balancing, but a configuration for failover High Availability.

To prevent sessions from timing out while in use, **add more Hue Server instances**, so that each can be pinned to another HiveServer2/Impala instance. And for both HiveServer2/Impala, **set the affinity timeout** (that is, the timeout to close persisted sessions) **to be longer than the impala query and session timeouts**.

For the best load distribution, **create multiple profiles** in your load balancer, per port, for both non-Hue clients and Hue clients. Have non-Hue clients distribute loads in a **round robin** and configure Hue clients with source IP Persistence on dedicated ports, for example, 21000 for impala-shell, 21050 for impala-jdbc, and 21051 for Hue.

Add Hive and Impala Roles

In Cloudera Manager, add roles for HiveServer2 and Impala Daemon (like [Add Hue Roles](#) on page 46):

1. Configure the cluster with at least two roles for HiveServer2:
 - a. Go to the **Hive** service and select **Actions > Add Role Instances**.
 - b. Click **HiveServer2**, assign one or more hosts, and click **OK > Continue**.
 - c. Check each role and select **Actions for Selected > Start** and click **Start**.
2. Configure the cluster with at least two roles for Impala Daemon:
 - a. Go to the **Impala** service and select **Actions > Add Role Instances**.
 - b. Click **Impala Daemon**, assign one or more hosts, and click **OK > Continue**.
 - c. Check each role and select **Actions for Selected > Start** and click **Start**.

Install Proxy Service

This is an example of how to add a proxy server for each HiveServer2 and Impala Daemon with multiple profiles.

1. Install [haproxy](#) (for either RHEL / Ubuntu / SLES):

```
yum install haproxy
```

```
apt-get install haproxy
```

```
zypper addrepo
http://download.opensuse.org/repositories/server:http/SLE_12/server:http.repo
zypper refresh
zypper install haproxy
```

2. Configure haproxy for each role, for example:

```
vi /etc/haproxy/haproxy.cfg
```

```
listen impala-shell
  bind :21001
  mode tcp
  option tcplog
  balance roundrobin
  stick-table type ip size 20k expire 5m
  server impala_0 host shortname-2.domain:21000 check
  server impala_1 host shortname-3.domain:21000 check

listen impala-jdbc
  bind :21051
  mode tcp
  option tcplog
  balance roundrobin
  stick-table type ip size 20k expire 5m
  server impala_0 host shortname-2.domain:21050 check
  server impala_1 host shortname-3.domain:21050 check

listen impala-hue
  bind :21052
  mode tcp
  option tcplog
  balance source
  server impala_0 host shortname-2.domain:21050 check
  server impala_1 host shortname-3.domain:21050 check

listen hiveserver2-jdbc
  bind :10001
  mode tcp
  option tcplog
  balance roundrobin
  stick-table type ip size 20k expire 5m
  server hiveserver2_0 host shortname-1.domain:10000 check
  server hiveserver2_1 host shortname-2.domain:10000 check

listen hiveserver2-hue
  bind :10002
  mode http
  option tcplog
  balance source
  server hiveserver2_0 host shortname-1.domain:10000 check
  server hiveserver2_1 host shortname-2.domain:10000 check
```

Replace shortname-#.domain with those in your environment:

```
sed -i "s/host shortname/your host shortname/g" /etc/haproxy/haproxy.cfg
sed -i "s/domain/your domain/g" /etc/haproxy/haproxy.cfg
```

3. Restart haproxy:

```
service haproxy restart
```


4. Run [netstat](#) to ensure your proxies are running:

```
netstat | grep LISTEN
```

Authenticate Hue with LDAP

Configuring Hue for Lightweight Directory Access Protocol (LDAP) lets you **import** users and groups from a directory service, **synchronize** group membership manually or at automatically login, and **authenticate** with LDAP.

This page explains how to configure Hue for LDAP authentication. To import users and group from LDAP, see [Synchronize Hue with LDAP](#) on page 55.

Authenticate Hue Users and Groups with LDAP

Hue supports [Active Directory](#) (AD) and open standard LDAP such as [OpenLDAP](#) and [OpenDJ](#).

There are two ways to bind Hue with an LDAP directory service:

- **Search Bind:** Hue searches for user credentials with search base (and attribute and filter).
- **Direct Bind:** Hue authenticates (without searching) in one of two ways:
 - *NT Domain:* Bind to Microsoft Active Directory with username@domain (the UPN) or
 - *Username Pattern:* Bind to open standard LDAP with full path of directory information tree (DIT).



Note: Username pattern does not work with AD because AD inserts spaces into the UID which Hue cannot process.

Encryption: To prevent credentials from transmitting in the clear, encrypt with LDAP over SSL, using the LDAPS protocol on the LDAPS port (636 by default); or encrypt with the [StartTLS](#) extension using the standard LDAP protocol and port (389 by default). Cloudera recommends LDAPS. You must have a CA Certificate in either case.

Table 7: Hue Supported LDAP Authentication and Encryption Methods

LDAP Auth Action	Encrypted (LDAPS)	Encrypted (LDAP+TLS)	Not Encrypted (LDAP)
Search Bind	AD, LDAP	AD, LDAP	AD, LDAP
Direct Bind - NT Domain	AD	AD	AD
Direct Bind - User Pattern	LDAP	LDAP	LDAP

Prerequisites

To authenticate Hue with LDAP, you must have:


- LDAP server
- Bind account (or support for anonymous binds)
- Cloudera Manager account with Full Administrator permissions
- [optional] LDAP server with LDAPS or StartTLS encryption.



Important: To authenticate *securely*, configure your LDAP server with either LDAP over SSL (LDAPS) or StartTLS encryption. Both methods require a Certificate Authority (CA) chain in a `.pem` file.

Search Bind


Search bind authentication does an [ldapsearch](#) against *one or more* directory services and binds with the found [distinguished name](#) (DN) and password. Hue searches the subtree from the base distinguished name. If LDAP Username Attribute is set, Hue looks for an entry whose attribute has the same value as the short name given at login.

 **Important:** Search binding works with all directory service types. It is also the only method that allows synchronizing groups at login (set with `sync_groups_on_login` in a `safety-valve`).

Video: [Authenticate Hue with LDAP and Search Bind](#)


1. Log on to Cloudera Manager and click **Hue**.
2. Click the **Configuration** tab and filter by scope=**Service-wide** and category=**Security**.
3. Set the following required properties:

Authentication Backend	desktop.auth.backend.LdapBackend
LDAP URL	ldaps://<ldap_server>:636 (or ldap://<ldap_server>:389)
LDAP Server CA Certificate	/path_to_certificate/cert.pem
LDAP Search Base	DC=mycompany,DC=com
LDAP Bind User Distinguished Name	username@domain
LDAP Bind Password	bind_user_password
Use Search Bind Authentication	TRUE
Enable LDAP TLS	FALSE if using LDAPS or not encrypting
Create LDAP users on login	TRUE

 **Note:** To encrypt with TLS, set **LDAP URL** to `ldaps://<ldap_server>:389` and check **Enable LDAP TLS**. For a proof of concept *without* encryption, use `ldap://<ldap_server>:389`, remove the value for LDAP Server CA Certificate, and uncheck Enable LDAP TLS.

4. You can optionally improve search performance with attributes and filters.

LDAP User Filter	objectclass=user (default = *)
LDAP Username Attribute	sAMAccountName (AD default), uid (LDAP default)
LDAP Group Filter	objectclass=group (default = *)
LDAP Group Name Attribute	cn (default)
LDAP Group Membership Attribute	member (default)

 **Note:** With the user settings in the table above, the LDAP search filter has the form:
`(&(objectClass=user)(sAMAccountName=<user entered username>))`.

5. Add any valid user and/or valid group to quickly test your LDAP configuration.

LDAP Username for Test LDAP Configuration	Any valid user
LDAP Group Name for Test LDAP Configuration	Any valid group

6. Click **Save Changes**.

7. [Test your LDAP configuration](#), and when successful, **Restart Hue**.



Note: The syntax of Bind Distinguished Name differs per bind method:

- Search Bind: username@domain
- Direct Bind with NT Domain: username
- Direct Bind with Username Pattern: DN string (full DIT path)

Do not use if anonymous binding is supported.

```
## You can test ldapsearch at the command line as follows:
LDAPTLS_CACERT=/

```



Note: To run `ldapsearch` with a CA certificate, you may need to install `ldap_utils` on Debian/Ubuntu and `openldap-clients` on RHEL/CentOS.

Direct Bind

To authenticate with direct binding, Hue needs either the User Principal Name (UPN) for Active Directory, or the full path to the LDAP user in the Directory Information Tree (DIT) for open standard LDAP.



Important: Direct binding only works with *one* domain. For multiple directories, use [Search Bind](#) on page 50.

Video: [Authenticate Hue with LDAP and Direct Bind](#)

To directly bind to an Active Directory/LDAP server with NT domain:

1. Log on to Cloudera Manager and click **Hue**.
2. Click the **Configuration** tab and filter by scope=**Service-wide** and category=**Security**.
3. Set LDAP properties exactly like Search Bind with these exceptions:

Active Directory Domain	<your NT domain>
LDAP Bind User Distinguished Name	<username only> (not username@domain)
Use Search Bind Authentication	FALSE

4. Click **Save Changes**.
5. [Test your LDAP configuration](#), and when successful, **Restart Hue**.

To directly bind to an open standard LDAP server with a username pattern:

1. Remove the value for Active Directory Domain.
2. Set both **LDAP Username Pattern** and **LDAP Bind User Distinguished Name** to a DN string that represents the full path of the directory information tree, from UID to top level domain.



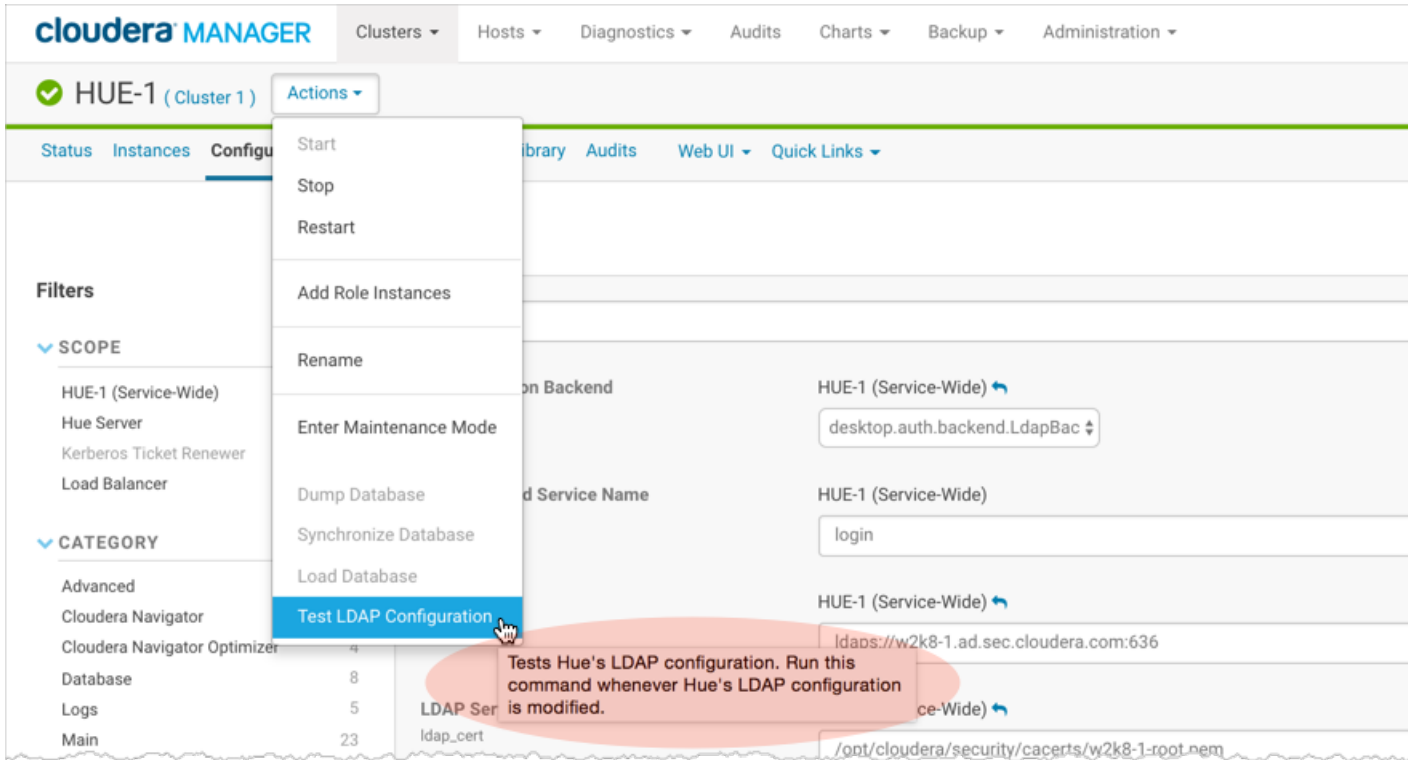
Note: When using direct bind, set **LDAP Search Base**, not for authentication (you can log on to Hue without it), but to [Synchronize Hue with LDAP](#) on page 55.

Test Hue LDAP Configuration On-the-Fly

You can test your LDAP settings *without* restarting the Hue service—simply input values and save changes.

1. Configure Hue LDAP [Search Bind](#) on page 50 or [Direct Bind](#) on page 51.

2. Add a user and group name for **Test LDAP Configuration**.
3. Click **Save Changes**.
4. Select **Actions > Test LDAP Configuration**.
5. Click **Test LDAP Configuration**.
6. **Restart Hue** when the test succeeds and log on to the Hue Web UI.



Unmanaged Clusters

Consumers with unmanaged clusters (that is, *without* Cloudera Manager) must manually set properties in [hue.ini](#). Consumers with managed clusters *must* use Cloudera Manager to set properties in hue.ini.

Example of a Search Bind configuration encrypted with LDAPS:

```
[[custom]]
[[auth]]
backend=desktop.auth.backend.LdapBackend

[[ldap]]
ldap_url=ldaps://w2k8-1.ad.sec.cloudera.com:636
search_bind_authentication=true
ldap_cert=<path_to_cacert>/w2k8-1-root.pem
use_start_tls=false
create_users_on_login=true
base_dn="DC=ad,DC=sec,DC=cloudera,DC=com"
bind_dn="<username>@ad.sec.cloudera.com"
bind_password_script=<path_to_password_script>/<script.sh>
test_ldap_user="testuser1"
test_ldap_group="testgroup1"

[[users]]
user_filter="objectclass=user"
user_name_attr="sAMAccountName"

[[groups]]
group_filter="objectclass=group"
group_name_attr="cn"
group_member_attr="member"
```

Example of a Direct Bind configuration for Active Directory encrypted with LDAPS:

```
[[ldap]]
ldap_url=ldaps://w2k8-1.ad.sec.cloudera.com:636
search_bind_authentication=false
nt_domain=ad.sec.cloudera.com
ldap_cert=/<path_to_cacert>/w2k8-1-root.pem
use_start_tls=false
create_users_on_login=true
base_dn="DC=ad,DC=sec,DC=cloudera,DC=com"
bind_dn="<username>"
bind_password_script=<path_to_password_script>/<script.sh>
...
```

Example of a Direct Bind configuration for Active Directory encrypted with StartTLS:

```
[[ldap]]
ldap_url=ldap://w2k8-1.ad.sec.cloudera.com:389
search_bind_authentication=false
nt_domain=ad.sec.cloudera.com
ldap_cert=/opt/cloudera/security/cacerts/w2k8-1-root.pem
use_start_tls=true
create_users_on_login=true
base_dn="DC=ad,DC=sec,DC=cloudera,DC=com"
bind_dn="cconner"
bind_password_script=<path_to_password_script>/<script.sh>
...
```

Table of Hue LDAP Properties

Property Name	Description and Syntax
General Hue LDAP Properties	
Authentication Backend backend	Authentication Mode. Select <code>desktop.auth.backend.LdapBackend</code> . Multiple backends are allowed. Create a list and add it to the Hue safety-valve.
LDAP URL ldap_url	URL for the LDAP server. Syntax: <code>ldaps://<ldap_server>:<636></code> or <code>ldap://<ldap_server>:<389></code> Important: To prevent usernames and passwords from transmitting in the clear, use <code>ldaps://</code> or <code>ldap://</code> + "Enable LDAP TLS".
Create LDAP users on login create_users_on_login	Flag to create new LDAP users at Hue login. If true, any user who logs into Hue is automatically created. If false, only users that exist in <code>useradmin</code> can log in.
Direct Bind Properties	
Active Directory Domain nt_domain	For direct binding with Active Directory only. Typically maps to the user email address or ID in conjunction with the domain. Allows Hue to authenticate without having to follow LDAP references to other partitions. Hue binds with User Principal Names (UPNs) if provided. Example: <code>ad.<mycompany>.com</code> Important: Do not use <code>nt_domain</code> when binding with a username pattern or if using search bind.
LDAP Username Pattern ldap_username_pattern	For direct binding with LDAP (non-Active Directory) only (because AD uses UPNs which have a space in them).

Property Name	Description and Syntax
	Username Pattern finds the user attempting to login into LDAP by adding the username to a predefined DN string. Use <code><username></code> to reference the user logging in. An example is <code>"uid=<username>,ou=people,dc=mycompany,dc=com"</code> .
Search Bind Properties	
Use Search Bind Authentication <code>search_bind_authentication</code>	Flag to enable/disable search binding.
LDAP Search Base <code>base_dn</code>	Distinguished name to use as a search base for finding users and groups. Syntax: <code>dc=ad, dc=sec, dc=mycompany,dc=com</code>
Encryption Properties	
LDAP Server CA Certificate <code>ldap_cert</code>	Full path to .pem file with Certificate Authority (CA) chain used to sign the LDAP server certificate. If left blank, all certificates are trusted and otherwise encrypted usernames and passwords are vulnerable to attack.
Enable LDAP TLS <code>use_start_tls</code>	Flag to enable/disable encryption with StartTLS .
Import / Sync Properties	
LDAP Bind User Distinguished Name <code>bind_dn</code>	Bind user. Only use if LDAP/AD does not support anonymous binds. (Typically, LDAP supports anonymous binds and AD does not.) Bind User differs per auth type: <ul style="list-style-type: none"> • Search Bind: <code>username@domain</code> • Direct Bind with NT Domain: <code>username</code> • Direct Bind with Username Pattern: DN string (and same as LDAP Username Pattern)
LDAP Bind Password <code>bind_password</code>	Bind user password.
Filter Properties	
LDAP User Filter <code>user_filter</code>	General LDAP filter to restrict search of valid users. Only used by Search Bind authentication and LDAP Sync. The default is <code>objectclass=*</code> but can differ. For example, some LDAP environments support Posix objects for <code>*nix</code> authentication and the user filter might need to be <code>objectclass=posixAccount</code> .
LDAP Username Attribute <code>user_name_attr</code>	Username to search against (the attribute in LDAP that contains the username). Typical attributes include <code>sAMAccountName</code> (default for AD/LDAP) and <code>uid</code> (LDAP default). Maintain case sensitivity when setting attributes for AD/LDAP.

Property Name	Description and Syntax
LDAP Group Filter group_filter	General LDAP filter to restrict search of valid groups. Only used by LDAP Sync (not authentication). If left blank, no filtering is used and all groups in LDAP are synced. The default is <code>objectclass=*</code> but can differ. For example, some LDAP environments support Posix objects for <code>*nix</code> authentication and the user filter might need to be <code>objectclass=posixGroup</code> .
LDAP Group Name Attribute group_name_attr	Group name to search against (the attribute in LDAP that contains the groupname). If left blank, the default is "cn" (common name), that typically works with AD/LDAP. Maintain case sensitivity when setting attributes for AD/LDAP.
LDAP Group Membership Attribute group_member_attr	Attribute in the group that contains DN's of all the members.(Optional) - If left blank, the default is "memberOf" or "member", that typically works with Active Directory/LDAP.
Test Properties	
LDAP Username for Test LDAP Configuration test_ldap_user	Any user (ideally with low privileges) used to verify the LDAP configuration.
LDAP Group Name for Test LDAP Configuration test_ldap_group	Any group (and not necessarily one that includes the test user) used to verify the LDAP configuration.

Synchronize Hue with LDAP

Configuring Hue for Lightweight Directory Access Protocol (LDAP) lets you **import** users and groups from a directory service, **synchronize** group membership manually or at automatically login, and **authenticate** with LDAP.

This page explains how to import and synchronize Hue users and groups with the LDAP server. See [Authenticate Hue with LDAP](#) on page 49 to ensure you are configured properly.

Tip: After you import and synchronize, learn how to [Restrict Group Permissions](#) on page 57.

Synchronize Hue Users and Groups with LDAP

There are four LDAP import and sync options in Hue:

LDAP Sync Action	Description
Add/Sync LDAP user	Import and synchronize one user at a time
Sync LDAP users/groups	Synchronize user memberships in all groups
Add/Sync LDAP group	Import and synchronize all users in one group
sync_groups_at_login	Automatically synchronize group membership at login



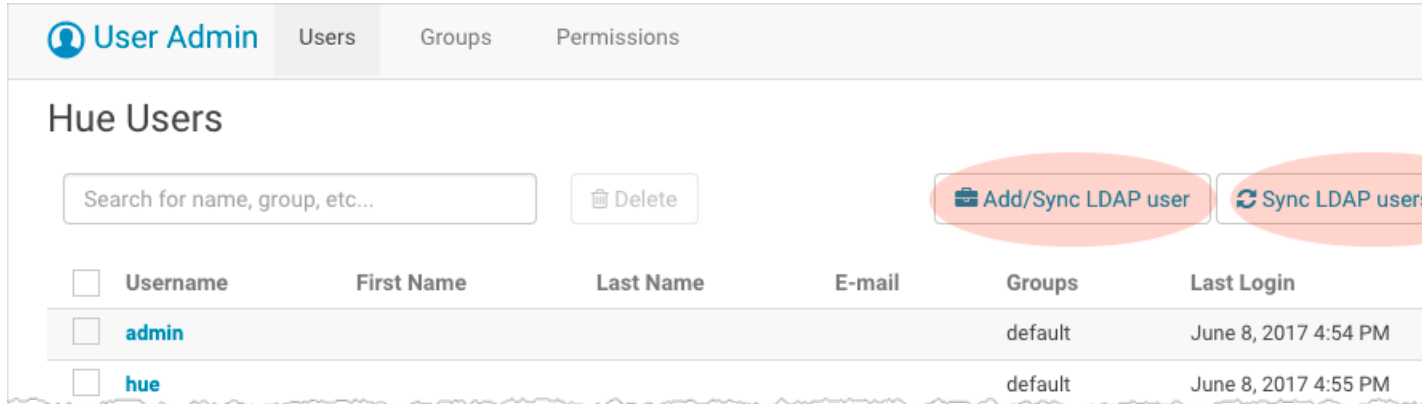
Note: Hue does not support importing all groups at once.

Prerequisites

To synchronize your Hue users and groups with your LDAP server:

- Hue must be configured to authenticate with LDAP. See [Authenticate Hue with LDAP](#) on page 49.
- The logged in user must have Hue superuser permissions.

Users



Import and Synchronize One User

To import and synchronize one LDAP user in Hue:

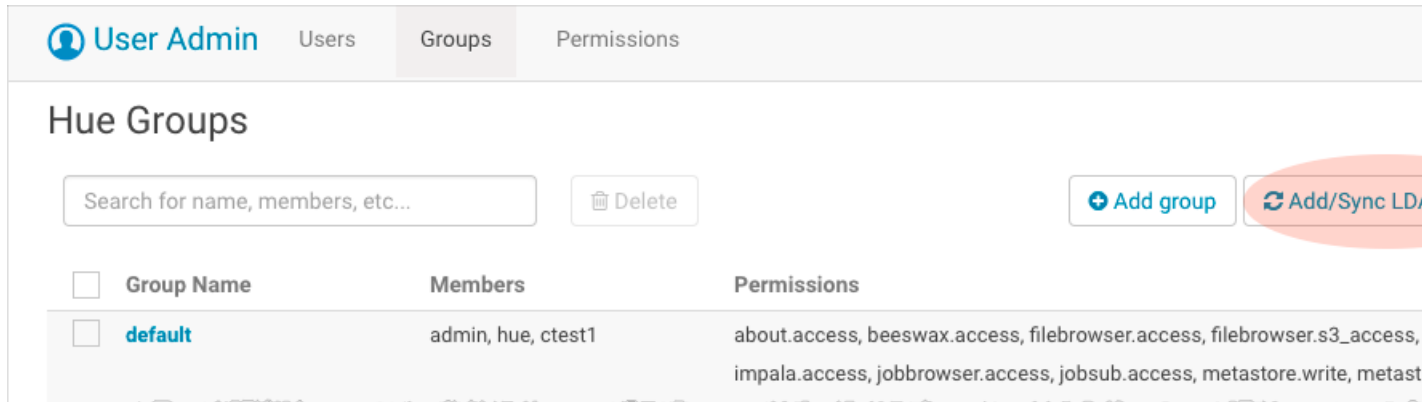
1. Log on to the Hue UI as a superuser.
2. Go to **User Admin > Users**.
3. Click **Add/Sync LDAP user**.
4. Add a username, check **Create home directory**, and click **Add/Sync user**.

Synchronize All User Memberships

To synchronize group memberships (for already imported users) to the current state of the LDAP server:

1. Log on to the Hue UI as a superuser.
2. Go to **User Admin > Users**.
3. Click **Sync LDAP users/groups**.
4. Check **Create home directories**, and click **Sync**.

Groups



Import and Synchronize One Group (with one or more users)

To import and synchronize a group (and its multiple users):

1. Log on to the Hue UI as a superuser.
2. Go to **User Admin > Groups**.
3. Click **Add/Sync LDAP group**.

4. Check **Create home directories**, and click **Sync**.

Synchronize Groups (and User Membership) at Login



Note: LDAP `sync_groups_at_login` only works with [Search Bind](#) on page 50.

To configure Hue to automatically synchronize users at the Hue login:

1. Log on to Cloudera Manager and click **Hue**.
2. Click the **Configuration** tab and filter by scope=**Service-wide** and category=**Advanced**.
3. Configure **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**:

```
[desktop]
[[ldap]]
sync_groups_on_login=true
```

4. Click **Save Changes** and **Restart Hue**.

The screenshot shows the Hue Configuration interface. On the left, a 'CATEGORY' dropdown menu is open, with 'Advanced' selected. The main content area displays the configuration snippet for 'HUE-1 (Service-Wide)'. The snippet is highlighted with a blue oval and contains the following text:

```
[desktop]
[[ldap]]
sync_groups_on_login=true
```

Restrict Group Permissions

You can configure user permissions on the **Groups** tab.

1. Log on to the Hue UI as a superuser.
2. Go to **User Admin > Groups**.
3. Click the name of the group you want to alter.
4. Deselect any users that you do not want to change (all users in the group are selected by default).
5. Select or deselect the permissions you want to apply or remove.
6. Click **Update Group**.



Note: A best practice is to remove all permissions from the default group and assign permissions as appropriate to your own groups.

Authenticate Hue with SAML

Hue supports SAML (Security Assertion Markup Language) for Single Sign-on (SSO) authentication. You can use any Identity Provider (such as Shibboleth, Okta, Ping, OpenAM) .

The [SAML 2.0 Web Browser SSO](#) profile has three components:

- **User Agent** - Browser that represents you, the user, seeking resources.
- **Service Provider (SP)** - Service (Hue) that sends authentication requests to SAML.
- **Identity Provider (IdP)** - SAML service that authenticates users.

When a user requests access to an application, Hue sends an authentication request from the browser to the Identity Provider. The Identity Provider authenticates the user, sends a response, and redirects the browser back to Hue.

Configure Hue for SAML

The service provider (Hue) and the Identity Provider use a metadata file to confirm each other's identity. Hue stores metadata from the SAML server, and the IDP stores metadata from Hue server.



Important: Read the documentation of your Identity Provider for details on how to procure the XML of the SAML server [metadata](#) (see [Configure Hue at Command Line](#) on page 58, step 3 on page 59).

Prerequisites

The instructions on this page assume that you have an Identity Provider set up and running. See [SSO with Hue: new SAML backend](#) for a demo on configuring [Shibboleth](#) as the Identity Provider.

Configure Hue at Command Line



Important: You may need to disable cipher algorithms. See [SAML SSL Error](#) on page 62 in Troubleshooting below.

1. Install the following libraries *on all hosts* in your cluster:

```
## RHEL/CentOS
yum install git gcc python-devel swig openssl
```

```
## Ubuntu/Debian
apt-get install git gcc python-dev swig openssl
```

```
## SLES
zypper install git gcc python-devel swig openssl make libxslt-devel libltdl-devel
```

2. Install `xmlsec1` and `xmlsec1-openssl` *on all hosts* in the cluster:



Important: Ensure that the `xmlsec1` package is executable by the user, `hue`.

```
## RHEL/CentOS
yum install xmlsec1 xmlsec1-openssl
```



Note: If `xmlsec` libraries are not available, use the appropriate [epel repository](#):

```
## For RHEL/CentOS 7
wget
http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-6.noarch.rpm
rpm -ivh epel-release-7-6.noarch.rpm
```

```
## Ubuntu/Debian
apt-get install xmlsec1 libxmlsec1-openssl
```

```
## SLES (get latest version)
wget http://www.aleksey.com/xmlsec/download/xmlsec1-1.2.24.tar.gz
tar -xvzf xmlsec1-1.2.24.tar.gz
cd xmlsec1-1.2.24
```

```
./configure && make
make install
```

3. Copy metadata from your IdP's SAML server and save it as an XML file.

For example, if your Identity Provider is Shibboleth, visit https://<idp_host>:8443/idp/shibboleth, copy the metadata content, and paste it into an .xml file.



Note: You may have to edit the copied metadata; for example, the IdP's port number (8443) may be missing from its URL.

```
mkdir -pm 755 /opt/cloudera/security/saml/
cd /opt/cloudera/security/saml/
```

```
vim idp-<your idp provider>-metadata.xml
# Paste IdP SAML here and save
```

4. Add key_file and cert_file for encrypted assertions—see [Table 8: Table of SAML Parameters](#) on page 60.

```
[root@test-cdh5120s-cent73-1 saml]# ls -al
total 16
drwxr-xr-x 2 root root 66 Jul 10 15:16 .
drwxr-xr-x 3 root root 32 Jul 7 09:00 ..
-rw-r--r-- 1 root root 1675 Feb 1 06:26 host.key
-rw-r--r-- 1 root root 1220 Feb 1 06:26 host.pem
-rw-r--r-- 1 root root 4599 Mar 16 11:18 idp-openam-metadata.xml
```



Warning: Add key and cert files even if *not* encrypting assertions. Hue checks for the existence *and validity* of these files even if they are not needed! They cannot be empty files. This is a known issue.

If necessary, create "valid" dummy files:

```
openssl genrsa -des3 -out dummy.key 2048
openssl rsa -inform PEM -outform PEM -in dummy.key -pubout -out
dummy-nopass.pem
```

Configure Hue in Cloudera Manager

Currently, all hue.ini properties for SAML must be added to Hue Service safety-valve in Cloudera Manager.

1. Log on to Cloudera Manager and go to **Hue > Configuration**.
2. Configure **Hue Service Advanced Configuration Snippet (Safety Valve)** for hue_safety_valve.ini with:

```
## Example Settings using Open AM:
[desktop]
redirect_whitelist="^\./.*$,^http://\./clr.sec.cloudera.com:8080\./.*$"
[[auth]]
backend=libsaml.backend.SAML2Backend
[libsaml]
xmlsec_binary=/usr/bin/xmlsec1
metadata_file=/opt/cloudera/security/saml/idp-openam-metadata.xml
key_file=/opt/cloudera/security/saml/host.key
cert_file=/opt/cloudera/security/saml/host.pem
username_source=nameid
name_id_format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
entity_id=<host base name>
logout_enabled=false
```



Note: For SLES distributions, the xmlsec binary may be in /usr/local/bin/. If so:

- Set **Hue Service Advanced Configuration Snippet:**
xmlsec_binary=/usr/local/bin/xmlsec1
- Set **Hue Service Environment Advanced Configuration Snippet:**
LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:/usr/local/lib/

3. Click **Save Changes**, then select, **Actions > Restart Hue**.

Configure SAML for Hue



Note: These steps may differ for each Identity Provider.

After Hue is configured and restarted, copy the metadata generated by Hue server and send it to your Identity Provider so they can configure the SAML server.

1. Ensure Hue is configured, restarted, and running.
2. Go to `http://<hue_fqdn>:8889/saml2/metadata`.
3. Copy the metadata and send it to your Identity Provider.
4. Ensure that your Identity Provider configures the SAML server with the Hue metadata (just as you configured the Hue server with SAML metadata).

SAML Properties in hue.ini

Table 8: Table of SAML Parameters

SAML Parameter	Description
authn_requests_signed	Boolean, that when True, signs Hue-initiated authentication requests with X.509 certificate.
backend	Hard-coded value set to SAML backend library packaged with Hue (<code>libsaml.backend.SAML2Backend</code>).
base_url	URL that SAML Identity Provider uses for responses. Typically used in Load balanced Hue environments.
cert_file	Path to X.509 certificate sent with encrypted metadata. File format must be .PEM.
create_users_on_login	Boolean, that when True, creates users from OpenId, upon successful login.
entity_id	Service provider ID. Can also accept pattern where '<base_url>' is replaced with server URL base.
key_file	Path to private key used to encrypt metadata. File format must be .PEM.
key_file_password	Password used to decrypt the X.509 certificate in memory.
logout_enabled	Boolean, that when True, enables single logout.
logout_requests_signed	Boolean, that when True, signs Hue-initiated logout requests with an X.509 certificate.
metadata_file	Path to readable metadata XML file copied from Identity Provider.
name_id_format	Format of NameID that Hue requests from SAML server.
optional_attributes	Comma-separated list of optional attributes that Hue requests from Identity Provider.

SAML Parameter	Description
<code>required_attributes</code>	Comma-separated list of required attributes that Hue requests from Identity Provider. For example, <code>uid</code> and <code>email</code> .
<code>redirect_whitelist</code>	Fully qualified domain name of SAML server: " <code>^\./.*\$,^https://\/<SAML_server_FQDN>\./.*\$</code> ".
<code>user_attribute_mapping</code>	Map of Identity Provider attributes to Hue django user attributes. For example, <code>{'uid':'username', 'email':'email'}</code> .
<code>username_source</code>	Declares source of username as <code>nameid</code> or <code>attributes</code> .
<code>xmlsec_binary</code>	Path to <code>xmlsec_binary</code> that signs, verifies, encrypts/decrypts SAML requests and assertions. Must be executable by user, <code>hue</code> .

Description of some properties to be set in `hue.ini` (via Cloudera Manager):

- **redirect_whitelist** [desktop]

Set to the fully qualified domain name of the SAML server so that Hue can redirect to the SAML server for authentication.

```
[desktop]
redirect_whitelist=^\./.*$,^https://\/<SAML_server_fully_qualified_domain_name>\./.*$
```



Note: Hue uses `redirect_whitelist` to protect itself from redirecting to unapproved URLs.

- **backend** [desktop]>[[auth]]

Point to the SAML backend (packaged with Hue):

```
backend=libsaml.backend.SAML2Backend
```

- **xmlsec_binary** [libsaml]

Point to the `xmlsec1` library path:

```
xmlsec_binary=/usr/bin/xmlsec1
```



Note: To find the path, run: `which xmlsec1`

- **metadata_file** [libsaml]

Point to the path of the XML file you created from the IdP's metadata:

```
metadata_file=/path/to/<your_idp_metadata_file>.xml
```

- **key_file and cert_file** [libsaml]

To encrypt communication between Hue and the Identity Provider, you need a private key and certificate. The private key signs requests sent to the Identity Provider and the certificate file encrypts and decrypts messages from the Identity Provider.

Copy these files from the Identity Provider and set `key_file` and `cert_file` to their respective paths. Both files are in PEM format and must be named with the `.PEM` extension.



Note: The key and certificate files specified by the `key_file` and `cert_file` parameters in `hue.ini` must be .PEM files.

Users with **password-protected certificates** can set the property, `key_file_password` in `hue.ini`. Hue uses the password to decrypt the SAML certificate *in memory* and passes it to `xmlsec1` through a named pipe. The decrypted certificate never touches the disk. This only works for POSIX-compatible platforms.

Troubleshooting

To enable `DEBUG` messages for all the logs in the directory, `/var/log/hue`, choose one of these methods:

- In the Hue Web UI, go to the **Home** page, select **Server Logs**, and check the box by **Force Debug Level**. Debug is enabled on-the-fly.
- In Cloudera Manager, go to **Hue > Configuration**, search for and set **Enable Django Debug Mode**, click **Save Changes**, and **Restart** the Hue service.
- At the command line, open `/etc/hue/conf/hue.ini`, scroll to `[desktop]`, and set `django_debug_mode=true`. Restart the Hue service:

```
sudo service hue restart
```

SAML SSL Error

OpenSSL might fail in CDH 5.5.x and higher with this message:

```
SSLERROR: [Errno bad handshake] [('SSL routines', 'SSL3_CHECK_CERT_AND_ALGORITHM', 'dh key too small')]
```

To resolve, append the following code to the file,

`/usr/java/<your_jdk_version>-cloudera/jre/lib/security/java.security:`

```
jdk.tls.disabledAlgorithms=MD5, RC4, DH
```

SAML Decrypt Error

The following error is an indication that you are using a slightly different SAML protocol from what Hue expects:

```
Error: ('failed to decrypt', -1)
```

To resolve:

1. Download and rename Python script, [fix-xmlsec1.txt](#).

```
wget http://www.cloudera.com/documentation/other/shared/fix-xmlsec1.txt -O fix-xmlsec1.py
```

2. Change permissions as appropriate, for example:

```
chmod 755 fix-xmlsec1.py
```

3. In `hue.ini`, set `xmlsec_binary=<path_to_script>/fix-xmlsec1.py`.
4. Run `fix-xmlsec1.py`.

This script repairs the known issue whereby `xmlsec1` is not compiled with `RetrievalMethod` and cannot find the location of the encrypted key. SAML2 responses would sometimes place `EncryptedKey` outside of the `EncryptedData` tree. This script moves `EncryptedKey` under `EncryptedData`.

Hue How-tos

Watch this space for more Hue How-tos!

How to Add a Hue Load Balancer

1. Log on to Cloudera Manager and click **Hue**.
2. Select **Actions > Add Role Instances**.
3. Add 1 Load Balancer:
 - a. Click **Select hosts** in the field under **Load Balancer**.
 - b. Select a host and click **OK**.
4. [Optional] Add 2 additional Hue servers (for a total of 3) to boost performance:
 - a. Click **Select hosts** in the field under **Hue Server**.
 - b. Select a host and click **OK > Continue**.
5. Check the boxes for the new servers and load balancer.
6. Select **Actions for Selected > Start > Start**.



Note: Hue servers can share hosts with Load Balancers. But Hue servers must be on distinct hosts from other Hue servers, and Load Balancers must be on distinct hosts from other Load Balancers.

7. Click **Save Changes** and **Restart Hue**.
8. Click **Hue Web UI > Load Balanced Hue Web UI**.
9. Log on to Hue and ensure the port is 8889.

Tip: The Load Balancer instance can always be accessed on the Hue **Instances** tab.

How to Enable SQL Editor Autocompleter in Hue

Autocompleter provides finely tuned SQL suggestions for Hive and Impala dialects. See [Brand new Autocompleter for Hive and Impala](#).

Autocompleter is enabled by default. To manually enable or disable, use the **Enable Autocompleter flag**.

1. Log on to Hue and go to either the Hive or Impala editor.
2. Place your cursor in the editor window.
3. Open the Autocompleter settings panel with the shortcut, **command-**, (Mac) or **Ctrl1-**, (Windows). Do not miss the comma.

Tip: Type ? (anywhere but the active editor) to open a menu of **Editor keyboard shortcuts**.
4. To **Enable Autocompleter**, check the box. To disable, uncheck the box.
5. To **Enable Live Autocompletion**, check the box. To disable, uncheck the box.

Tip: To use Autocompleter with Live Autocompletion *off*, use **Ctrl + Space key**.
6. Place your cursor in the editor window to close the panel. Autocompleter is now turned on or off based on your flag setting.



How to Enable and Use Governance-Based Data Discovery

As of Cloudera Enterprise 5.11, Hue can use the metadata tagging, indexing, and search features available with Cloudera Navigator data management. After integrating Hue with Cloudera Navigator, existing Cloudera Navigator tags and indexed entities can be accessed and viewed in Hue, and entities can be tagged using Hue interfaces. Managed metadata and custom metadata tags created or applied using Hue are then stored in the Cloudera Navigator instance. This How To shows administrators how to enable this capability and SQL users how to use the feature.

Administrator Setup Tasks

Enabling Cloudera Navigator for Hue

To use Hue with Cloudera Navigator, you must give the Hue server access to the Navigator Administrator account and enable the integration by configuring some properties using the Cloudera Manager Admin Console. If you have multiple Hue servers running across the cluster, you must configure each Hue server with the Navigator Administrator user account to access Cloudera Navigator.



Note: After enabling the integration on clusters that use Cloudera Sentry role-based access control, different Hue users can view only those entities to which their respective user roles have been granted permission.

Requirements

Follow the steps below to integrate Cloudera Navigator and the Hue server. These steps require [Cloudera Navigator](#) to already be installed, configured, and running in the context of a Cloudera Manager cluster. See Cloudera Data Management guide for more information about Cloudera Navigator.

The administrator performing the configuration tasks must have the Cloudera Manager user role of Navigator Administrator or Full Administrator. Use the same account that was used to set up authentication for Cloudera Navigator users and groups.

Enabling the Integration and Configuring Authentication

1. Log in to the Cloudera Manager Admin Console.
2. Select **Clusters > Hue**.
3. Click the **Configuration** tab.
4. Select **Service-wide** from the Scope filter.
5. Select **Cloudera Navigator** from the Category filter. The properties for Cloudera Navigator configuration for Hue display:
 - a. Click the **Enable Navigator Metadata Server Integration** box.
 - b. Select the authentication mechanism for **Navigator Metadata Server Auth** used by the Cloudera Navigator instance (this selection must match the configuration for Navigator Metadata Server):
 - Cloudera Manager
 - LDAP (Active Directory, OpenLDAP)
 - SAML (for SSO support)
 - c. Click **Enable Audit Collection**

6. Click **Save Changes**.
7. Click **Restart Hue**.
8. Log in to Hue by selecting **Hue Web UI**.

SQL Users Get Started

This short tutorial shows you how to organize data better, how to create new tags, or re-use existing tags.

Applying Metadata Tags Using Hue Prepare Hue Tables

As the Hue superuser, install sample tables and then refresh Impala metadata.

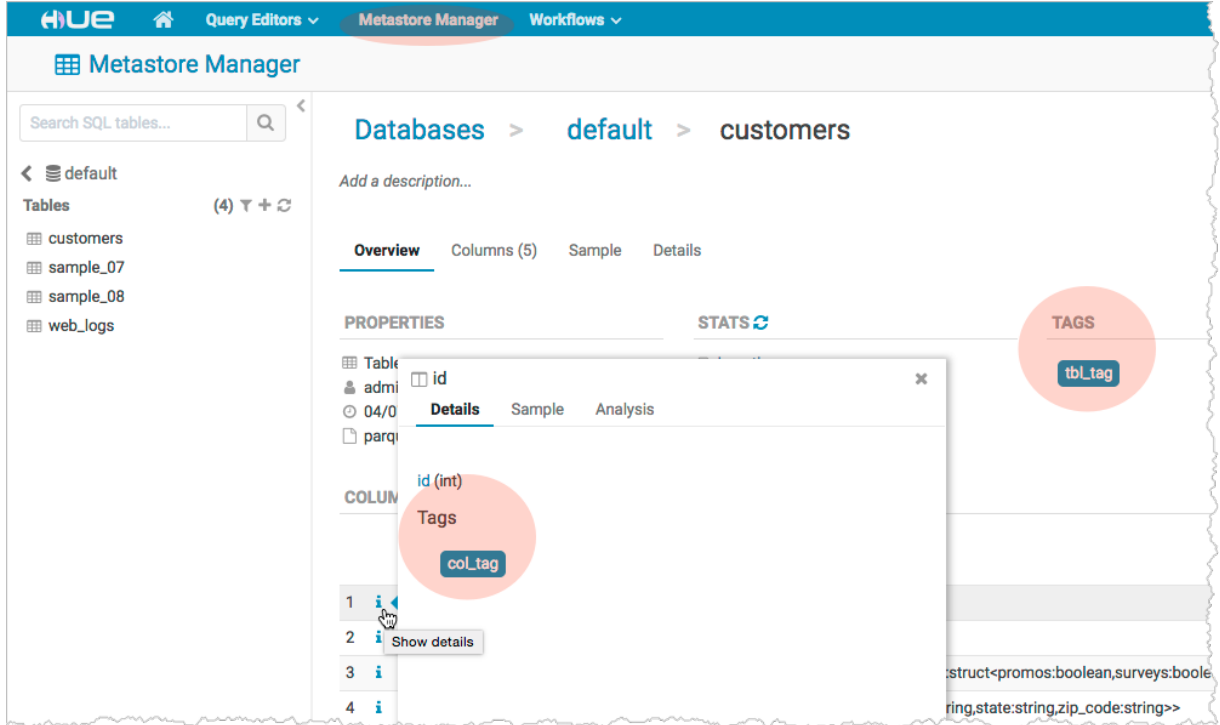
1. Log in to Hue (as superuser) by selecting **Hue Web UI** in Cloudera Manager.
2. Go to **About > Step 2: Examples** and install sample tables for Hive and Impala.

 **Note:** You can also append /about to the Hue URL: `http://<hostname>:8889/hue/about/`.

3. Go to **Query Editors > Impala** and click the refresh icon.
4. Select **Perform incremental metadata update** to display sample tables.
5. Go to **Metastore Tables Manager** and click the refresh icon.

Tag Database, Table, and Field

1. In **Metastore Tables Manager**, click the default (or some other) database.
2. Add **database** tag: Hover over **TAGS**, click the edit icon, enter a tag of your choice, and save.
3. Add **table** tag: Click a table name (such as "customers"), hover over **TAGS**, and repeat.
4. Add **field** tag: Click the "show details" icon by a column name and repeat in the context popup.



Search Cloudera Navigator Metadata with Hue

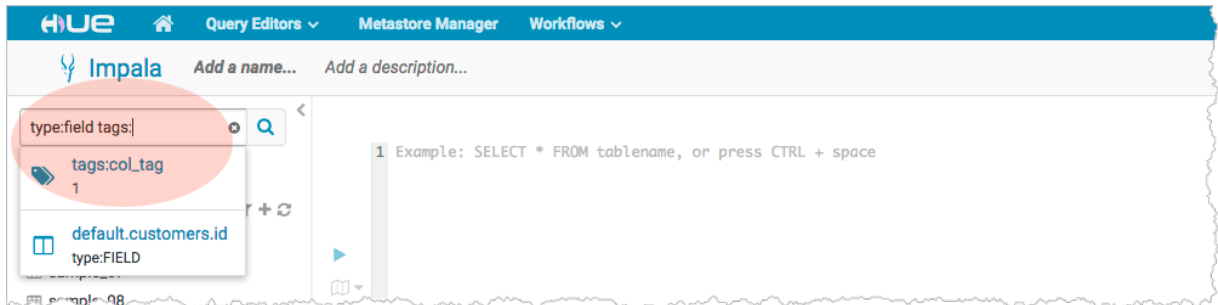
After integrating Hue with Cloudera Navigator, the Hue SQL Editor provides a Search bar that includes a list of filters and an auto-complete listing that is pre-filled with top values. The Search mechanism by default returns tables and views only. Use the `type` filter to search for columns, partitions, and databases.



Note: On clusters that use Sentry for role-based access control, the Search mechanism does not display counts of popular values. Sentry ensures that Hue users can view only entities to which their user role (as configured and managed by Sentry) has been granted specific permissions.

The Cloudera Navigator search field can be accessed in the Metastore Tables Manager as well as from the Hive and Impala editors.

1. Go to **Query Editors > Impala**.
2. Search on "`type:field tags:`" in the Navigator search field.



Note: You can search for table tags with "`tags:`". For other types, input "`type:database tags:`" or "`type:field tags:`".

3. Create a view of customers named David:

```
CREATE VIEW IF NOT EXISTS davids AS SELECT * FROM customers WHERE customers.name LIKE 'David%';
```

4. Search on "davids". You should see, *No recent match found*, until Navigator can process the new view.



Note: New tables and views can take ~1 hour to register in Navigator and be searchable.

5. Log in to Cloudera Navigator at `http://<cloudera manager hostname>:7187`.
6. On the Search tab, select **type=View**. When you see "davids," return to Hue and retry your search.
7. To see your tags in Cloudera Navigator, click **Add New Value** under tags.

How to Enable S3 Cloud Storage in Hue

Cloudera S3 Connector in Cloudera Manager securely connects your CDH cluster to Amazon S3.



Note:

- C5.11 adds **S3 Guard** for [list consistency](#) and support for [IAM roles](#) in Cloudera Manager.
- C5.10 connects Hue, Impala, and Navigator securely with the Cloudera S3 Connector Service.
- C5.9 adds support for [Amazon S3](#) with plain-text credentials using Cloudera Manager safety valves.

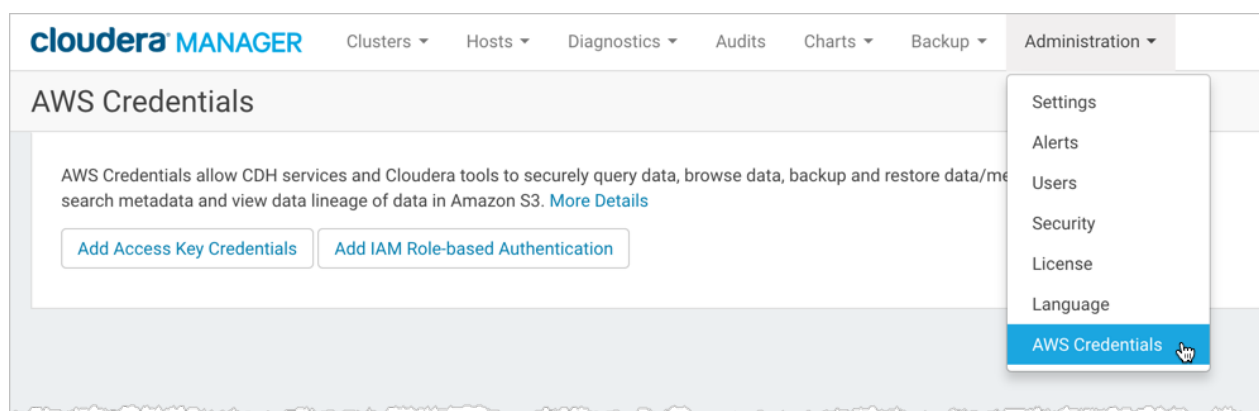
Enable S3 in Hue with the S3 Connector Service

For a secure and fine-grained connection to Amazon S3 (for Hue, Impala, and Navigator), Cloudera recommends its S3 Connector service in Secure Mode with encrypted access keys and [Kerberos](#) and [Sentry](#) installed.



Important: Hive is not yet supported in Secure Mode. To connect Hive to S3, use "Unsecure" Mode.

Method	Security	Required	Services
Secure Mode	High	Kerberos, Sentry	Hue, Impala, Navigator
Unsecure Mode	Medium		Hue, Impala, Navigator, Hive



1. Log on to Cloudera Manager.
2. Select **Administration > AWS Credentials**.
3. Click **Add Access Key Credentials** or **Add IAM Role-based Authentication**.




Important: IAM Role-based Authentication is not fine-grained authentication. Also, to use it with Hue, configure the region in `hue_safety_valve.ini`—see step [step 11](#).

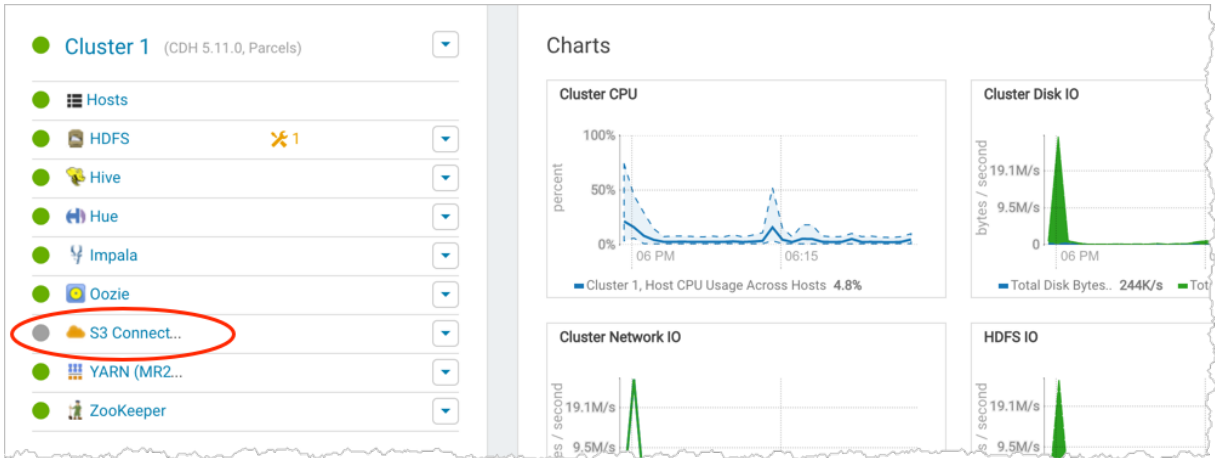
4. Add any **Name** and enter your S3 credentials:
 - a. To connect your [AWS root user](#), add the **Access Key ID** and **Secret Access Key** for your *root account*.
 - b. To connect an [IAM user](#), add the **Access Key ID** and **Secret Access Key** for a *read-only IAM account*.
5. If you have an [Amazon DynamoDB](#) database, check **Enable S3Guard** for consistent read operations.




Warning: Components writing data to S3 are constrained by the inherent Amazon S3 limitation known as "[eventual consistency](#)." This can lead to data loss when a Spark or Hive job writes output directly to S3. Cloudera recommends that you use S3 Guard or write to HDFS and distcp to S3.

6. Click **Enable for <cluster name>** to give Hue access to S3 and S3-backed tables. Impala must have permissions defined in Sentry.
7. If using access keys, select **Secure** or **Unsecure** mode. Select Unsecure to use Hive.
8. Click **Continue** (at Step 1) if your cluster passes validation. You are automatically taken to step 5.
9. Click **Continue** (at Step 5) to restart Hive, Impala, Oozie, and Hue.
10. When finished, click **Home** to see the S3 Connector.

 **Note:** A gray status icon ● means the S3 Connector service was successfully added.




11. If using IAM roles, set the region to `us-east-1` (N. Virginia) in `hue_safety_valve.ini`. If not, ignore this step.

 **Note:** Configuring `hue_safety_valve.ini` is a temporary Hue workaround for CDH 5.10.

- a. Select **Configuration > Advanced Configuration Snippets**.
- b. Filter by **Scope > Hue**.
- c. Set **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini** with the following:

```
[aws]
[[aws_accounts]]
[[[default]]]
region=us-east-1
```

- d. Click **Save Changes**.
- e. Restart Hue: select **Cluster > Hue and Actions > Restart**.

 **Note:** The S3 Connector service is not added when you use IAM roles.

Related topics: [How to Configure AWS Credentials](#) and [Configuring the Amazon S3 Connector](#).

Enable S3 in Hue with Safety Valves

This section assumes an [AWS account with access keys](#), but not necessarily a Kerberized cluster.

You can connect to S3 using three safety valves (also known as **Advanced Configuration Snippets**):

- **Hue Service** Advanced Configuration Snippet (Safety Valve) for `hue_safety_valve.ini`
- **Cluster-wide** Advanced Configuration Snippet (Safety Valve) for `core-site.xml`
- **Hive Service** Advanced Configuration Snippet (Safety Valve) for `core-site.xml`.

The screenshot shows the Cloudera Manager home page. The navigation bar includes Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. The main content area shows a sidebar with various services like Cluster 1, HDFS-1, HIVE-1, HUE-1, IMPALA-1, MAPREDUC..., OOOZIE-1, SENTRY-1, YARN-1, and ZOOKEEPER... The Configuration Issues dropdown menu is open, listing various settings categories, with 'Advanced Configuration Snippets' highlighted at the bottom.

1. Log on to Cloudera Manager and select **Clusters > your cluster**.
2. Select **Configuration > Advanced Configuration Snippets**.
3. Filter by **Scope > Hue**.
4. Set your S3 credentials in **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**:



Note: Store your credentials in a script that outputs to stdout. A `security_token` is optional.

```
[aws]
[[aws_accounts]]
[[[default]]]
access_key_id_script=</path/to/access_key_script>
secret_access_key_script=</path/to/secret_key_script>
#security_token=<your AWS security token>
allow_environment_credentials=false
region=<your region, such as us-east-1>
```

For a proof-of-concept installation, you can add the IDs directly.

```
access_key_id=<your_access_key_id>
secret_access_key=<your_secret_access_key>
```

5. Clear the scope filters and search on "**core-site.xml**".
6. To enable the S3 Browser, set your [S3 credentials](#) in **Cluster-wide Advanced Configuration Snippet (Safety Valve) for core-site.xml**:

```
<property>
<name>fs.s3a.access.key</name>
<value>AWS access key ID</value>
</property>

<property>
<name>fs.s3a.secret.key</name>
<value>AWS secret key</value>
</property>
```

7. To enable Hive with S3, set your S3 credentials in **Hive Service Advanced Configuration Snippet (Safety Valve) for core-site.xml**.
8. Click **Save Changes**.
9. Restart Hue: select **Cluster > Hue** and **Actions > Restart**.
- 10 Restart Hive: select **Cluster > Hive** and **Actions > Restart**.

Related topics: [Amazon Web Services \(AWS\) Security](#).

Generate Access Keys in AWS

To integrate Hue with S3, you must have an Amazon Web Services (AWS) account, with access keys for *either* your root user *or* a read-only IAM user.

Root Account

1. Create an [AWS account](#) and sign in to the [AWS Console](#).
2. Create access keys for this AWS [root account](#):
 - a. Expand the drop-down menu under your account name and select [My Security Credentials](#).
 - b. Click **Continue to Security Credentials**.
 - c. Expand **Access Keys (Access Key ID and Secret Access Key)**.
 - d. Click **Create New Access Key**.
 - e. Click **Show Access Key** or **Download Key File**. These are your AWS root credentials.

IAM Account

1. [Create](#) two IAM groups (AWS admin and S3 Read-only):



Important: AWS requires that your *first* IAM group and associated user has administrator access.

- a. Go to the [IAM service](#).
 - b. Click **Groups** and **Create New Group**.
 - c. Enter a name and click **Next Step**.
 - d. Filter on "admin" and select the **AdministratorAccess** policy.
 - e. Click **Next Step** and **Create Group**.
 - f. Create a second group with **AmazonS3ReadOnlyAccess**.
2. Create two IAM users and assign one to the admin policy and one to the S3 read policy.
 - a. Click **Users** and **Add User**.
 - b. Enter a name, and at a minimum, select **Programmatic access**.
 - c. Click **Next: Permissions**.
 - d. Select the group with administrator permissions.
 - e. Click **Next: Review** and **Create User**.
 - f. Create a second user and assign the group with S3 read-only access.
3. Create access keys for your *read-only* IAM user:
 - a. Click the name of your read-only IAM user.
 - b. Click the **Security Credentials** tab.
 - c. Click **Create Access Key**.
 - d. Click **Show Access Key** or **Download Key File**. These are your IAM user credentials.

How to Use S3 as Source or Sink in Hue

On this page, we demonstrate how to write to, and read from, an S3 bucket in Hue.

Populate S3 Bucket


In this section, we use open data from the [U.S. Geological Survey](#).

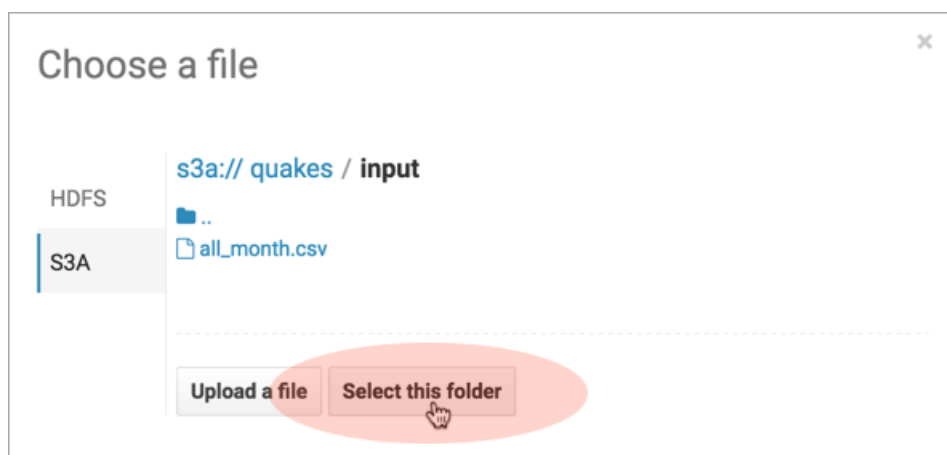
1. Download [30 days of earthquake data](#) (`all_month.csv`) from the [USGS](#) (~2 MB).
2. Log on to the **Hue Web UI** from Cloudera Manager.
3. Select **File Browser > S3 Browser**.
4. Click **New > Bucket**, name it "quakes_<any unique id>" and click **Create**.
Tip: Unique bucket names are important per S3 [bucket naming conventions](#).
5. Navigate into the bucket by clicking the bucket name.
6. Click **New > Directory**, name it "input" and click **Create**.
7. Navigate into the directory by clicking the directory name.
8. Click **Upload** and select, or drag, `all_month.csv`. The path is `s3a://quakes/input/all_month.csv`.




Important: Do not add anything else to the "input" directory—no extra files, no directories.

Create Table with S3 File

1. Go to the Metastore Manager by clicking **Data Browsers > Metastore Tables**.
2. Create a new table from a file by clicking 
3. Enter a **Table Name** such as "earthquakes".
4. Browse for the **Input Directory**, `s3a://quakes/input/`, and click **Select this folder**.



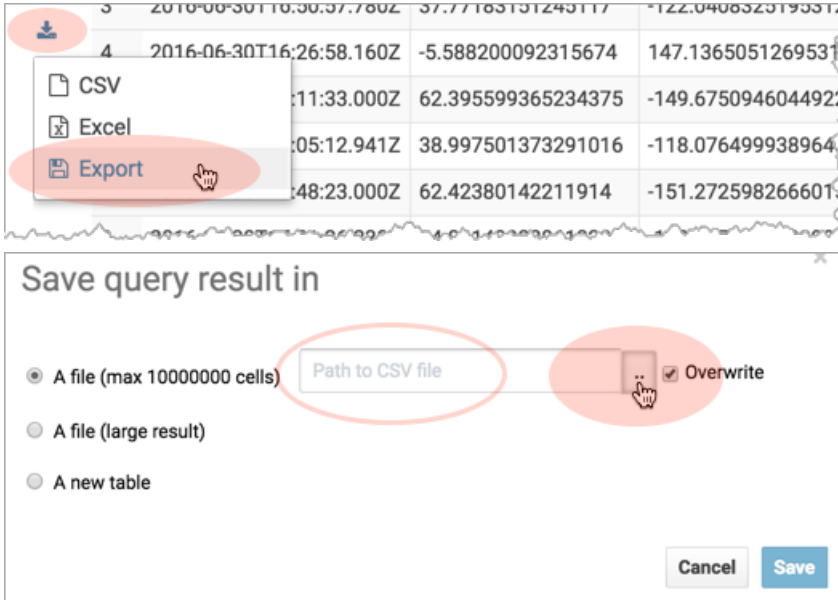
5. Select **Create External Table** from the Load Data menu and click **Next**.
6. Delimit by Comma(,) and click **Next**.
7. Click **Create Table**.
8. Click **Browse Data**  to automatically generate a `SELECT` query in the **Hive** editor:

```
SELECT * FROM `default`.`earthquakes` LIMIT 10000;
```

Export Query Results to S3

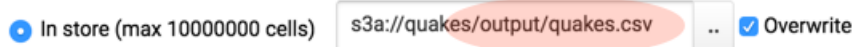
1. Run and Export Results in Hive

- a. Run the query by clicking **Execute** ▶.
- b. Click **Get Results** ⬇️.
- c. Select **Export** to open the **Save query result** dialog.



2. Save Results as Custom File

- a. Select **In store (max 10000000 cells)** and open the **Path to CSV file** dialog.
- b. Navigate into the bucket, **s3a://quakes**.
- c. **Create folder** named, "output."
- d. Navigate into the **output** directory and click **Select this folder**.
- e. Append a file name to the path, such as **quakes.csv**.
- f. Click **Save**. The results are saved as **s3a://quakes/output/quakes.csv**.



3. Save Results as MapReduce files

- a. Select **In store (large result)** and open the **Path to empty directory** dialog.
- b. Navigate into the bucket, **s3a://quakes**.
- c. If you have not done so, create a folder named, "output."
- d. Navigate into the **output** directory and click **Select this folder**.
- e. Click **Save**. A MapReduce job is run and results are stored in **s3a://quakes/output/**.




4. Save Results as Table

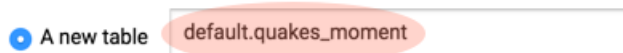
- a. Run a query for "**moment**" earthquakes and export:

```
SELECT time,
       latitude,
       longitude,
       mag
```



```
FROM `default`.`earthquakes`
WHERE magtype IN ('mw', 'mwb', 'mwc', 'mwr', 'mww');
```

- b. Select **A new table** and input <database>.<new table name>.
- c. Click **Save**.
- d. Click **Browse Data**  to view the new table.



Troubleshoot Errors

This section addresses some error messages you may encounter when attempting to use Hue with S3.

Tip: Restart the Hue service to view buckets, directories, and files added to your [upstream S3 account](#).

- **Failed to access path**

Failed to access path: "s3a://quakes". Check that you have access to read this bucket and that the region is correct.

Possible solution: Check your bucket region:

1. Log on to your AWS account and navigate to the S3 service.
2. Select your bucket, for example "quakes", and click Properties.
3. Find your region. If it says [US Standard](#), then region=us-east-1.
4. Update your configuration in **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini**.
5. Save your changes and restart Hue.

- **The table could not be created**

The table could not be created. Error while compiling statement: FAILED: SemanticException com.cloudera.com.amazonaws.AmazonClientException: Unable to load AWS credentials from any provider in the chain.

Possible solution: Set your S3 credentials in Hive core-site.xml:

1. In Cloudera Manager, go to **Hive > Configuration**.
2. Filter by **Category > Advanced**.
3. Set your credentials in **Hive Service Advanced Configuration Snippet (Safety Valve) for core-site.xml**.
 - a. Click the **+** button and input Name and Value for fs.s3a.AccessKeyId.
 - b. Click the **+** button and input Name and Value for fs.s3a.SecretAccessKey.
4. Save your changes and restart Hive.

- **The target path is a directory**

Possible solution: Remove any directories or files that may have been added to s3a://quakes/input/ (so that all_month.csv is alone).

- **Bad status for request TFetchResultsReq ... Not a file**

Bad status for request TFetchResultsReq(...):
TFetchResultsResp(status=TStatus(errorCode=0, errorMessage='java.io.IOException: java.io.IOException: Not a file: s3a://Not a file: s3a://quakes/input/output' ...

Possible solution: Remove any directories or files that may have been added to s3a://quakes/input/ (so that all_month.csv is alone). Here, Hive cannot successfully query the earthquakes table (based on all_month.csv) due to the directory, s3a://quakes/input/output.

Tip: Run `tail -f` against the Hive server log in: `/var/log/hive/`.

How to Run Hue Shell Commands

You may need to administer Hue programmatically, for example, to reset the superuser password. This page addresses managed deployments of CDH 5.5 and higher.

1. Gather the following information:

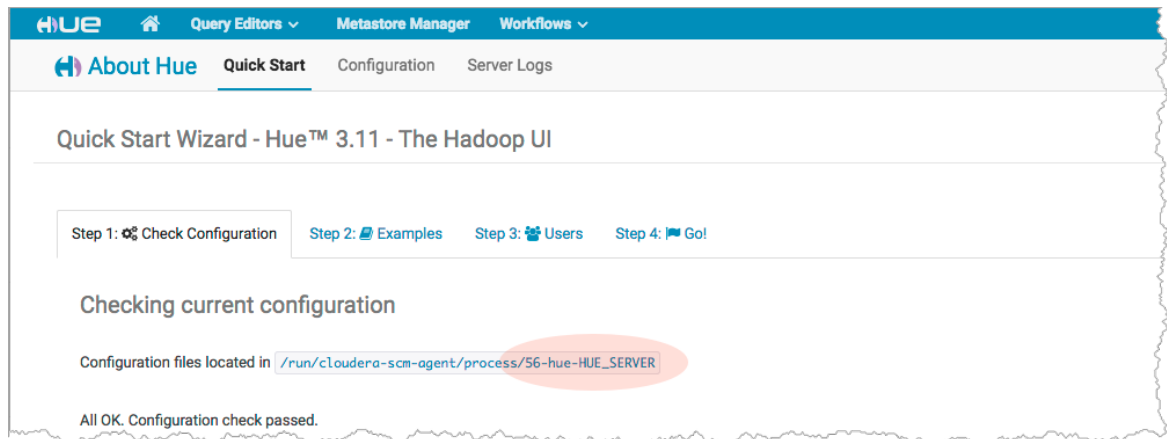
- Hue server database password (embedded or external).
- Path to `/build/env/bin/hue`:

```
# Parcels (e.g., /opt/cloudera/parcels/CDH-5.9.0-1.cdh5.9.0.p0.23/lib/hue)
realpath /opt/cloudera/parcels/`ls -l /opt/cloudera/parcels | grep CDH | tail -1 | awk
'{print $9}'`/lib/hue

# Packages
/usr/lib/hue
```

- Path to the current Hue process directory (with Hue configuration files):

```
#Example path: /var/run/cloudera-scm-agent/process/56-hue-HUE_SERVER/
realpath /var/run/cloudera-scm-agent/process/`ls -alrt /var/run/cloudera-scm-agent/process
| grep HUE | tail -1 | awk '{print $9}'`
```



2. Set `HUE_CONF_DIR` to the latest Hue process directory:

```
export HUE_CONF_DIR="/var/run/cloudera-scm-agent/process/`ls -alrt
/var/run/cloudera-scm-agent/process | grep HUE | tail -1 | awk '{print $9}'`"
echo $HUE_CONF_DIR
```

3. Run shell subcommands

When true, `HUE_IGNORE_PASSWORD_SCRIPT_ERRORS` runs the Hue shell even if `hue.ini` contains passwords generated by Cloudera Manager (such as `bind_password` and `ssl_password`).



Note: Do not export `HUE_IGNORE_PASSWORD_SCRIPT_ERRORS` or `HUE_DATABASE_PASSWORD` to ensure that they are not stored and only apply to *this* command.

Parcels

- List available subcommands

```
HUE_IGNORE_PASSWORD_SCRIPT_ERRORS=1 HUE_DATABASE_PASSWORD=<db_password>
/opt/cloudera/parcels/`ls -l /opt/cloudera/parcels | grep CDH | tail -1 | awk '{print
$9}'`/lib/hue/build/env/bin/hue
```

- Run the interactive Hue Python shell (Ctrl+D to quit)

```
HUE_IGNORE_PASSWORD_SCRIPT_ERRORS=1 HUE_DATABASE_PASSWORD=<db_password>
/opt/cloudera/parcels/`ls -l /opt/cloudera/parcels | grep CDH | tail -1 | awk '{print $9}'`/lib/hue/build/env/bin/hue shell
```

- Change a user password

```
HUE_IGNORE_PASSWORD_SCRIPT_ERRORS=1 HUE_DATABASE_PASSWORD=<db_password>
/opt/cloudera/parcels/`ls -l /opt/cloudera/parcels | grep CDH | tail -1 | awk '{print $9}'`/lib/hue/build/env/bin/hue changepassword <username>
```

Packages

- List available subcommands

```
HUE_IGNORE_PASSWORD_SCRIPT_ERRORS=1 HUE_DATABASE_PASSWORD=<db_password>
/usr/lib/hue/build/env/bin/hue
```

- Run the interactive Hue Python shell (Ctrl+D to quit)

```
HUE_IGNORE_PASSWORD_SCRIPT_ERRORS=1 HUE_DATABASE_PASSWORD=<db_password>
/usr/lib/hue/build/env/bin/hue shell
```

- Change a user password

```
HUE_IGNORE_PASSWORD_SCRIPT_ERRORS=1 HUE_DATABASE_PASSWORD=<db_password>
/usr/lib/hue/build/env/bin/hue changepassword <username>
```

For unmanaged and lower CDH versions, see:

- [Execute some builtin or shell commands](#)
- [Storing passwords in file script](#)
- [How to change or reset a forgotten password?](#)

Hue Troubleshooting

This section addresses possible obstacles when installing, configuring, and using Hue. Watch this space for more topics!

Potential Misconfiguration Detected

This page covers various configuration errors. The goal is for all configuration checks to pass.

Checking current configuration

Configuration files located in `/var/run/cloudera-scm-agent/process/108-hue-HUE_SERVER`

All OK. Configuration check passed.

Preferred Storage Engine

`PREFERRED_STORAGE_ENGINE`: We recommend MySQL InnoDB engine over MyISAM which does not support transactions.

Checking current configuration

Configuration files located in `/var/run/cloudera-scm-agent/process/233-hue-HUE_SERVER`

Potential misconfiguration detected. Fix and restart Hue.

`PREFERRED_STORAGE_ENGINE` We recommend MySQL InnoDB engine over MyISAM which does not support transactions.



Warning: Talk to your DBA before changing the storage engine for the Hue database tables.

Alter Hue database tables from MyISAM to InnoDB

1. Stop the Hue service in Cloudera Manager: go to **Cluster > Hue** and select **Actions > Stop**.
2. Log on to the host of your MySQL server.
3. Look for any MyISAM tables in your Hue server database:

```
mysql -u root -p<root password>
```

```
SELECT table_schema, table_name, engine
FROM information_schema.tables
WHERE engine = 'MyISAM' AND table_schema = '<hue database name>';
```

```
quit
```

4. Set the engine to InnoDB for all Hue database tables:

```
# Create script, /tmp/set_engine_innodb.ddl
mysql -u root -p<root password> -e \
"SELECT CONCAT('ALTER TABLE ',table_schema,'.',table_name,' engine=InnoDB;') \
FROM information_schema.tables \
WHERE engine = 'MyISAM' AND table_schema = '<hue database name>';" \
| grep "ALTER TABLE <hue database name>" > /tmp/set_engine_innodb.ddl
```

```
# Run script
mysql -u root -p<root password> < /tmp/set_engine_innodb.ddl
```

5. Verify that no MyISAM tables exist by rerunning the SELECT statement in step 3 on page 76.

6. Start the Hue service.

MySQL Storage Engine

`MYSQL_STORAGE_ENGINE`: All tables in the database must be of the same storage engine type (preferably InnoDB).

Follow the instructions in the section, [Preferred Storage Engine](#) on page 76, to ensure *all* Hue tables use InnoDB.

Appendix: Apache License, Version 2.0

SPDX short identifier: Apache-2.0

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims

licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

Appendix: Apache License, Version 2.0

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

```
Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```