cloudera®

Hue Guide

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

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 Distribution of Hue 3.11.x
Date: February 3, 2021

Table of Contents

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.

| Hive | HMS | Impala | HDFS | S3 | Oozie | Pig | Search | HBase | Sqoop | YARN | Sentry |

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 7
- [Connect Hue to PostgreSQL](#) on page 12
- [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 datbase username
password=Hue database password
```

Connect Hue to MySQL or MariaDB

If you have an external database installed, review MySQL/MariaDB Troubleshooting on page 7 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 1: Install Commands for Supported OS Versions

OS	OS Ver	DB Ver	Command
CentOS / RHEL	7.2		*No package mysql-server available.*
		5.5	```sudo yum install mariadb-server```
	6.8	5.1	```sudo yum install mysql-server```
			No package mariadb-server available.
	5.10	5.6	```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```
			No package mariadb-server available.
SLES	12.1		*'mysql' not found in package names.*

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

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

Table 2: Start Commands

OS	OS Ver	Command
CentOS / RHEL	7.2	`sudo systemctl start mariadb`
	5.10, 6.8	`sudo service mysqld start`
SLES	11.4, 12.1	`sudo rcmysql start`
Ubuntu	12.04, 14.04	`sudo service mysql start`
Debian	7.8, 8.4	`sudo service mysql start`

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 2: Start Commands on page 9 table above and replace with "`restart`".

3. Enable the server to automatically start on boot:

Table 3: Enable Automatic Start

OS	OS Ver	Command
CentOS / RHEL	7.2	`sudo systemctl enable mariadb`
	5.10, 6.8	`sudo chkconfig mysqld on`
SLES	11.4, 12.1	`sudo chkconfig mysql on` `sudo rcmysql status`
Ubuntu	12.04, 14.04	`# preconfigured to start at boot` `sudo service mysql status`
Debian	7.8, 8.4	`# preconfigured to start at boot` `sudo service mysql status`

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 (replace the ID value).

```
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 12 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 4: Install Commands

OS	OS Ver	DB Ver	Command
CentOS / RHEL	7.2	9.2	`sudo yum install postgresql-server` `sudo postgresql-setup initdb`
	6.8	8.4	`sudo yum install postgresql-server` `sudo service postgresql initdb`
	5.10	8.1	`sudo yum install postgresql-server` `sudo /etc/init.d/postgresql start`

SLES	12.1	*9.4*	`zypper install postgresql`
			`postgresql-server`
			`systemctl start postgresql`

	11.4	8.4	`# 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`

Ubuntu	14.04	9.3	`sudo apt-get install postgresql`
	12.04	9.1	`sudo apt-get install postgresql`
Debian	8.4	9.4	`sudo apt-get install postgresql`
	7.8	9.1	`sudo apt-get install postgresql`

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 5: Restart Commands

OS	OS Ver	Command
CentOS / RHEL	7.2	`sudo systemctl restart postgresql` `sudo systemctl enable postgresql`

	5.10, 6.8	`sudo service postgresql restart`
		`sudo chkconfig postgresql on`
		`sudo chkconfig postgresql --list`

SLES	12.1	`systemctl restart postgresql`
	11.4	`rcpostgresql restart`
Ubuntu	12.04, 14.04	`sudo /etc/init.d/postgresql restart`
Debian	7.8, 8.4	`sudo /etc/init.d/postgresql restart`

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

      a. Go to **Hue** > **Configuration**.

      b. Filter by category, **Database**.

      c. 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>
```

      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 PostgreSQL server* in a command-line terminal.

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

b. Drop the foreign key constraint (replace the ID value).

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

c. Clean the table, `django_content_type`.

```
hue=# TRUNCATE django_content_type CASCADE;
```



```
hue=> \d auth_permission;
                             Table "public.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_permissi
    TABLE "auth_user_user_permissions" CONSTRAINT "auth_user_user_permissions_permission_id_fkey" FOREIGN KEY (permission_id) REFERENCES auth_
```

d. 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 25.

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.
```

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 25.

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/cdh59
mv ~/ORACLE_INSTANT_CLIENT-11.2-1* ~/manifest.json /var/www/html/cdh59
```

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

```
cd /var/www/html/cdh59/
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.

```
sha1sum 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 ℹ️	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 6 and 6 on page 6).

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 27 of Create Hue Database on page 26. 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
  ⬇ **instantclient-basic-linux.x64-12.1.0.2.0.zip** (63,352,239 bytes) (cksum - 109893216)
  ⬇ **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
  ⬇ **instantclient-sdk-linux.x64-12.1.0.2.0.zip** (667,174 bytes) (cksum - 1047596065)
  ⬇ **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
```

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.

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 `$LB_LIBRARY_PATH` are properly set so that `pip` knows which version to install.

```
echo $ORACLE_HOME $LD_LIBRARY_PATH
```

```
pip install cx_Oracle
```

> **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 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 6 and 6 on page 6).

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 27 of Create Hue Database on page 26. 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 39.

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 35 users remove a foreign key from `auth.permission` and clean `django_content_type`.
    - Oracle on page 35 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 35 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 `auth_permission`:

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

```
mysql> ALTER TABLE hue.auth_permission DROP FOREIGN KEY content_type_id_refs_id_d043b34a;
Query OK, 228 rows affected (0.01 sec)
Records: 228  Duplicates: 0  Warnings: 0
```

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 synching 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 <u>Dump, Synchronize, and Load</u> on page 37.

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

See <u>Installing Cloudera Manager and CDH</u>.

Populate Database (optional)

<u>Populate the Hue database</u> 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 <u>Hue Custom Databases</u>.

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;
```


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)

    a. Log on to Hue as the administrator.

    b. Open the Administration drop down sand select **Manage Users**.

    c. 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**.

    d. Log out as the administrator and log on as Alice.

2. Save Hive Query (customers.sql)

    a. Go to **About Hue** > **Quick Start** by clicking the Hue logo.

    b. Click the **Examples** tab ("Step 2").

    c. Click download **Hive** to install sample databases.

    d. Go to the **Metastore Manager** (or **Data Browser** > **Metastore Tables**).

    e. Click the `default` database and `customers` (sample) table.

    f. Click **Browse Data** to automatically generate a `select *` query in the **Hive** editor.

    g. Run the query with your cursor in the editor and **CTRL + Enter**, or by clicking the Run icon ▶.

    h. Save the query as `customers.sql` by clicking the **Save** icon 💾.

    i. 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

This section consolidates security documents related to Hue 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

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

In CDH 5.9.0 (Hue 3.11), Autocompleter gains a deeper knowledge of the Hive and Impala SQL dialects to give you finely tuned smart suggestions. 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 `Ctrl-,` (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 S3 Cloud Storage

In CDH 5.9.0 (Hue 3.11), Hue adds support for [Amazon S3](#) in its file browser, metastore, and editor interfaces. This page explains how to configure Hue with S3 and use it across the product.

> ⚠ **Warning:** Cloudera components writing data to S3 are constrained by the inherent limitation of Amazon S3 known as "[eventual consistency](#)." In very rare conditions, this limitation may lead to some data loss when a Spark or Hive job writes output directly to S3. Cloudera recommends that you write to HDFS and distcp to S3.
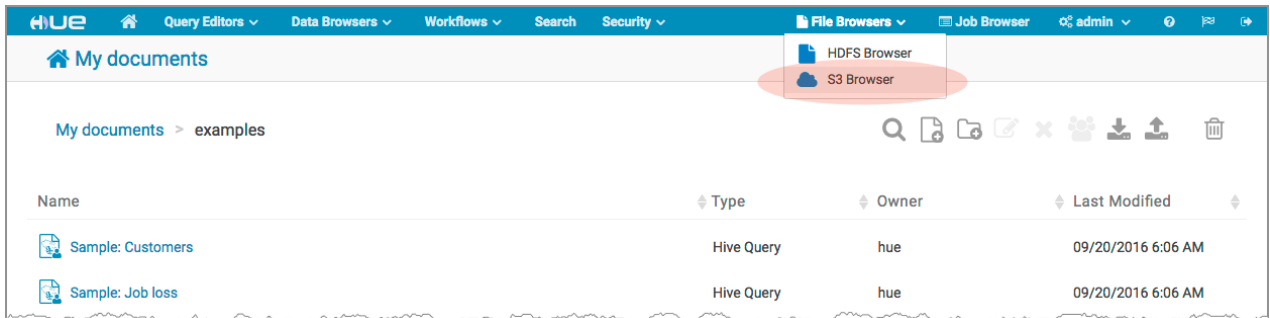
Connect Hue to S3 Account

This section assumes that you [have an Amazon S3 account](#). Let us connect to that account.

1. If your S3 buckets use TLS and you are using custom truststores, see [Connecting to Amazon S3 Using TLS](#) for information about configuring Hue, Hive, and Impala to access S3 over TLS.
2. Log on to Cloudera Manager and select **Clusters** > **<your cluster name>**.
3. Select **Configuration** > **Advanced Configuration Snippets**.
4. Filter by **Scope** > **Hue**.
5. 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. `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>
```

6. Clear the scope filters and input **core-site.xml** into the search box.
7. 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.awsAccessKeyId</name>
<value>AWS access key ID</value>
</property>

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

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

Populate S3 Bucket

In this section, we populate an S3 bucket with nested keys (bucket > directory > file) and add a CSV file of earthquake data from the USGS.

1. Download 30 days of earthquake data (`all_month.csv`) from the USGS (~2 MB).
2. In Cloudera Manager, click **Hue** > **Web UI** and log on to Hue.
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 and 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 the

   icon.
3. Enter a **Table Name** such as "earthquakes".
4. Browse for the **Input Directory**, `s3a://quakes/input/`, and click **Select this folder**.
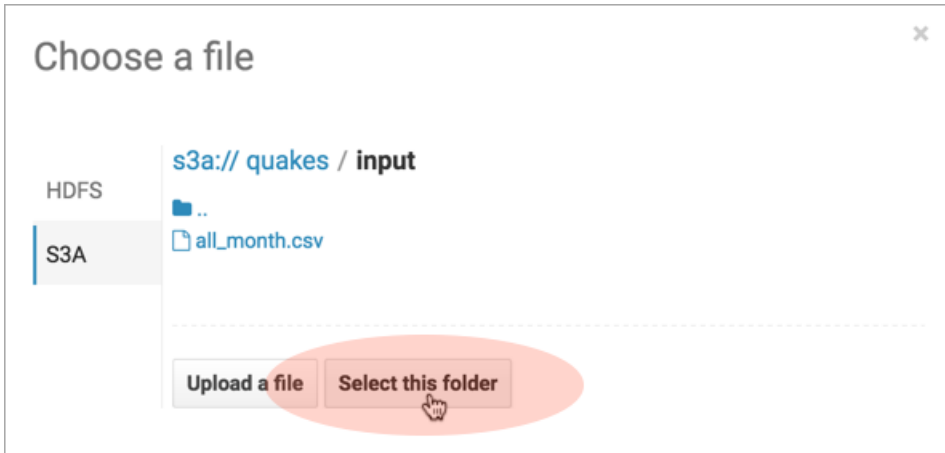
5. Select **Create External Table** from the Load Data menu and click **Next**.
6. Delimit by Comma(,) and click **Next**.
7. Click **Create Table**.
8. Click **Browse Data** ☰ to automatically generate a `SELECT` query in the **Hive** editor:

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

Export Query Results to S3

1. **Run and Export Results in Hive**

   a. Run the query by clicking the **Execute** ▶ button.

   b. Click the **Get Results** ⬇ button.

   c. Select **Export** to open the **Save query result** dialog.
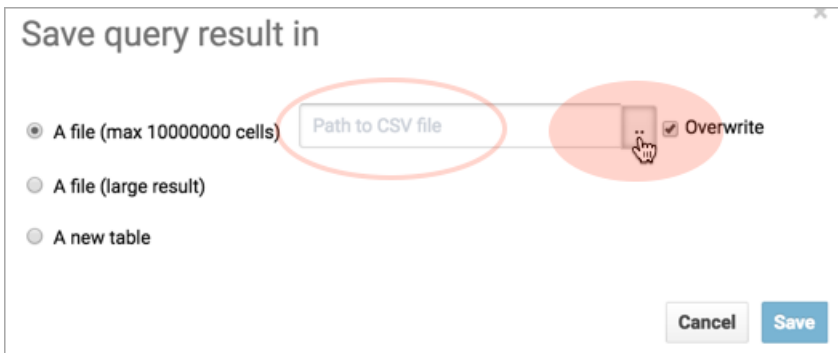
2. **Save Results as Custom File**

   a. Select **In store (max 10000000 cells)** and open the **Path to CSV file** dialog.

   b. Select **S3A** and navigate into the bucket, **s3a://quakes**.

   c. Create a folder named, "output." Click **Create folder**, enter name, click **Create folder**.

   d. Navigate into the **output** directory and click **Select this folder**.

   e. Append a file name to the path, such as **quakes.cvs**.

   f. Click **Save**. The results are saved as `s3a://quakes/ouput/quakes.csv`.
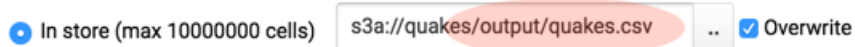
3. **Save Results as MapReduce files**

   a. Select **In store (large result)** and open the **Path to CSV file** dialog.

    b. Select **S3A** and 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/`.
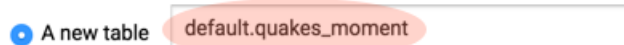
        ⦿ In store (large result)   `s3a://quakes/output`   ..   ⊙

4. Save Results as Table

    a. Run a query for "moment" earthquakes:

```
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.

        ⦿ A new table   `default.quakes_moment`

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.awsAccessKeyId.

          b. Click the ✚ button and input Name and Value for fs.s3a.awsSecretAccessKey.

      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}'`
```
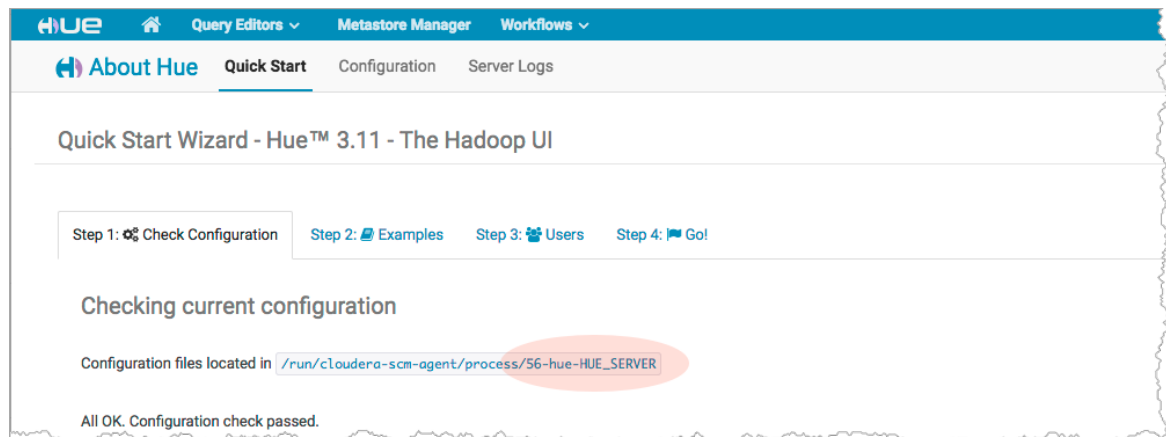

2. **Set `HUE_CONF_DIR`** to the latest Hue process directory:

```
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 50.

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 50, 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.
```