Using an External Database for Hue Using the Command Line (#xd_583c10bfdbd326ba-3ca24a24-13d80143249--7f4e)

The Hue server requires a SQL database to store small amounts of data such as user account information, job submissions, and Hive queries. SQLite is the default embedded database. Hue also supports several types of external databases. This page explains how to configure Hue with a selection of external **CDH and Cloudera Manager Supported Databases (rn_consolidated_pcm.html#cdh_cm_supported_db)** .
Important: Cloudera strongly recommends an external database for clusters with multiple Hue users.

Continue reading:

- **Embedded Database**
- **External Database**

Embedded Database (#concept_hs4_5kj_zj)

By default, Hue is configured to use the embedded database, SQLite, and should require no configuration or management by the administrator.

Inspecting the Embedded Hue Database (#title_400)

The default SQLite database is located in /var/lib/hue/desktop.db. You can inspect this database from the command line with the `sqlite3` program. For example:

```
# sqlite3 /var/lib/hue/desktop.db
SQLite version 3.6.22
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> select username from auth_user;
admin
test
sample
sqlite>
```

Important: Avoid modifying the database directly with `sqlite3`. It is best used as a troubleshooting tool.

Backing up the Embedded Hue Database (#title_401)

If you use the default embedded SQLite database, copy the `desktop.db` file to another node for backup. Cloudera recommends that you backup regularly, and also that you backup before upgrading to a new version of Hue.

External Database (#concept_id1_wkj_zj)

Cloudera strongly recommends an external database for clusters with multiple Hue users, especially clusters in a production environment. The default embedded database, SQLite, cannot support large data migrations. See **CDH and Cloudera Manager Supported Databases (rn_consolidated_pcm.html#cdh_cm_supported_db)** .

High-level steps to configure Hue with any of the supported external databases are:

1. Stop Hue service.
2. Backup default SQLite database.
3. Install database software and dependencies.
4. Create and configure database and load data.
5. Start Hue service.

If you do not need to migrate a SQLite database, you can skip the steps on dumping the database and editing the JSON objects.

Continue Reading:

- **Configuring the Hue Server to Store Data in MariaDB (cdh_ig_hue_database.html#cdh_ig_hue_database_mariadb)**
- **Configuring the Hue Server to Store Data in MySQL (cdh_ig_hue_database.html#cdh_ig_hue_database_mysql)**
- **Configuring the Hue Server to Store Data in PostgreSQL (cdh_ig_hue_database.html#cdh_ig_hue_database_postgresql)**
- **Configuring the Hue Server to Store Data in Oracle (cdh_ig_hue_database.html#cdh_ig_hue_database_oracle)**

Prerequisites (#concept_e4p_zlx_rp)

Before configuring Hue to use an external database:

- Install all support libraries required by your operating system. See **Development Preferences**

(https://github.com/cloudera/hue#development-prerequisites) in the Hue documentation for the full list.

- Ensure the Hue server is running on Python 2.6 or higher.

Configuring the Hue Server to Store Data in MariaDB (#cdh_ig_hue_database_mariadb)

Note: Cloudera recommends InnoDB over MyISAM as the Hue MySQL engine. On CDH 5, Hue *requires* InnoDB.

1. Shut down the Hue server if it is running.
2. Open <some-temporary-file>.json and remove all JSON objects with `useradmin.userprofile` in the `model` field. Here are some examples of JSON objects that should be deleted.

```
{
    "pk": 1,
     "model":  "useradmin.userprofile",
     "fields": {
       "creation_method":  "HUE",
       "user": 1,
       "home_directory":  "/user/alice"
    }
  },
  {
     "pk": 2,
     "model":  "useradmin.userprofile",
     "fields": {
       "creation_method":  "HUE",
       "user": 1100714,
       "home_directory":  "/user/bob"
    }
  },
.....
```

3. Start the Hue server.
4. Install the MariaDB client developer package.

OS	Command
RHEL	`$ sudo yum install mariadb-devel`
SLES	`$ sudo zypper install mariadb-devel`
Ubuntu or Debian	`$ sudo apt-get install libmariadbclient-dev`

5. Install the MariaDB connector.

OS	Command
RHEL	`$ sudo yum install mariadb-connector-java`
SLES	`$ sudo zypper install mariadb-connector-java`
Ubuntu or Debian	`$ sudo apt-get install libmariadb-java`

6. Install and start MariaDB.

OS	Command
RHEL	`$ sudo yum install mariadb-server`
SLES	`$ sudo zypper install mariadb-server` `$ sudo zypper install libmariadblclient18`
Ubuntu or Debian	`$ sudo apt-get install mariadb-server`

7. Change the `/etc/my.cnf` file as follows:

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
bind-address=<ip-address>
default-storage-engine=InnoDB
sql_mode=STRICT_ALL_TABLES
```

8. Start the `MariaDB` daemon.

   ```
   $ sudo service mariadb start
   ```

9. Configure MariaDB to use a strong password. In the following procedure, your current `root` password is blank. Press the Enter key when you're prompted for the root password.

   ```
   $ sudo /usr/bin/mysql_secure_installation
   [...]
   Enter current password for root (enter for none):
   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
   All done!
   ```

10. Configure MariaDB to start at boot.

OS	Command

RHEL	`$ sudo /sbin/chkconfig mariadb on`
	`$ sudo /sbin/chkconfig --list mariadb`
	`mysqld 0:off 1:off 2:on 3:on 4:on 5:on 6:off`

SLES	`$ sudo chkconfig --add mariadb`
Ubuntu or Debian	`$ sudo chkconfig mariadb on`

11. Create the Hue database and grant privileges to a `hue` user to manage the database.

    ```
    mysql> create database hue;
    Query OK, 1 row affected (0.01 sec)
    mysql> grant all on hue.* to 'hue'@'localhost' identified by '<secretpassword>';
    Query OK, 0 rows affected (0.00 sec)
    ```

12. Open the Hue configuration file in a text editor.
13. Edit the Hue configuration file `hue.ini`. Directly below the `[[database]]` section under the `[desktop]` line, add the following options (and modify accordingly for your setup):

    ```
    host=localhost
    port=3306
    engine=mysql
    user=hue
    password=<secretpassword>
    name=hue
    ```

14. As the `hue` user, load the existing data and create the necessary database tables using `syncdb` and `migrate` commands. When running these commands, Hue will try to access a `logs` directory, located at `/opt/cloudera/parcels/CDH/lib/hue/logs`, which might be missing. If that is the case, first create the `logs` directory and give the `hue` user and group ownership of the directory.
    Note: `HUE_HOME` is a reference to the location of your Hue installation. For package installs, this is usually `/usr/lib/hue`; for parcel installs, this is usually, `/opt/cloudera/parcels/<parcel version>/lib/hue/`.

    ```
    $ sudo mkdir /opt/cloudera/parcels/CDH/lib/hue/logs
    $ sudo chown hue:hue /opt/cloudera/parcels/CDH/lib/hue/logs
    $ sudo -u hue <HUE_HOME>/build/env/bin/hue syncdb --noinput
    $ sudo -u hue <HUE_HOME>/build/env/bin/hue migrate
    $ mysql -u hue -p <secretpassword>
    mysql > SHOW CREATE TABLE auth_permission;
    ```

15. (**InnoDB only**) Drop the foreign key.

    ```
    mysql > ALTER TABLE auth_permission DROP FOREIGN KEY content_type_id_refs_id_XXXXXX;
    ```

16. Delete the rows in the `django_content_type` table.

    ```
    mysql > DELETE FROM hue.django_content_type;
    ```

17. Load the data.

```
$ <HUE_HOME>/build/env/bin/hue loaddata <some-temporary-file>.json
```

18. (**InnoDB only**) Add the foreign key.

```
$ mysql -u hue -p <secretpassword>
mysql > ALTER TABLE auth_permission ADD FOREIGN KEY (`content_type_id`) REFERENCES `django_content_type` (`id`);
```

Configuring the Hue Server to Store Data in MySQL (#cdh_ig_hue_database_mysql)

Note: Cloudera recommends InnoDB over MyISAM as the Hue MySQL engine. On CDH 5, Hue *requires* InnoDB.

1. Stop the Hue server, if running.

```
sudo service hue stop
```

2. Backup the existing database:
   a. Dump the existing database data to a .json file.

```
$ sudo -u hue <HUE_HOME>/build/env/bin/hue dumpdata > <some-temporary-file>.json
```

   Note: `HUE_HOME` refers to the location of your Hue installation. For package installs, this is usually `/usr/lib/hue`; for parcel installs, this is usually, `/opt/cloudera/parcels/<parcel version>/lib/hue/`.

   b. Delete all JSON objects with `useradmin.userprofile` in the `model` field from <some-temporary-file>.json. For example.

```
{
  "pk": 1,
   "model":  "useradmin.userprofile",
   "fields": {
     "creation_method":  "HUE",
     "user": 1,
     "home_directory":  "/user/alice"
   }
 },
 {
    "pk": 2,
    "model":  "useradmin.userprofile",
    "fields": {
      "creation_method":  "HUE",
      "user": 1100714,
      "home_directory":  "/user/bob"
   }
 },
.....
```

3. Install and configure MySQL as an external database for Hue:
   a. Install the MySQL client developer package.

OS	Command
RHEL	`$ sudo yum install mysql-devel`
SLES	`$ sudo zypper install mysql-devel`
Ubuntu or Debian	`$ sudo apt-get install libmysqlclient-dev`

   b. Install the MySQL connector.

OS	Command
RHEL	`$ sudo yum install mysql-connector-java`
SLES	`$ sudo zypper install mysql-connector-java`
Ubuntu or Debian	`$ sudo apt-get install libmysql-java`

   c. Install the MySQL server.

OS	Command
	`$ sudo yum install mysql-server`

RHEL	
SLES	`$ sudo zypper install mysql` `$ sudo zypper install libmysqlclient_r17`
Ubuntu or Debian	`$ sudo apt-get install mysql-server`

d. Configure `/etc/my.cnf` as follows:

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
bind-address=<ip-address>
default-storage-engine=InnoDB
sql_mode=STRICT_ALL_TABLES
```

e. Start the MySQL daemon.

OS	Command
RHEL	`$ sudo service mysqld start`
SLES and Ubuntu or Debian	`$ sudo service mysql start`

f. Configure MySQL with a strong password. Press the Enter key when you're prompted for the `root` password (as your current `root` password is blank).

```
$ sudo /usr/bin/mysql_secure_installation
[...]
Enter current password for root (enter for none):
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
All done!
```

g. Configure MySQL to start at boot.

OS	Command

RHEL	`$ sudo /sbin/chkconfig mysqld on`
	`$ sudo /sbin/chkconfig --list mysqld`
	`mysqld 0:off 1:off 2:on 3:on 4:on 5:on 6:off`

SLES	`$ sudo chkconfig --add mysql`
Ubuntu or Debian	`$ sudo chkconfig mysql on`

h. Create the database, `hue`:

```
mysql> create database hue;
```

i. Grant privileges:

```
mysql> grant all on hue.* to 'hue'@'localhost' identified by '<secretpassword>';
```

j. Configure `/etc/hue/conf/hue.ini` to use MySQL. Modify these options as appropriate and paste below `[[database]]` and `[desktop]`:

```
host=localhost
port=3306
engine=mysql
user=hue
password=<secretpassword>
name=hue
```

4. Load any backed up data:
   a. Ensure a logs directory exists and is writable at `/opt/cloudera/parcels/CDH/lib/hue/logs`.
   b. Ensure the logs directory has `hue` user and group ownership.
   c. Synchronize and migrate the database.

```
$ sudo mkdir /opt/cloudera/parcels/CDH/lib/hue/logs
$ sudo chown hue:hue /opt/cloudera/parcels/CDH/lib/hue/logs
$ sudo -u hue <HUE_HOME>/build/env/bin/hue syncdb --noinput
$ sudo -u hue <HUE_HOME>/build/env/bin/hue migrate
$ mysql -u hue -p <secretpassword>
mysql > SHOW CREATE TABLE auth_permission;
```

   Note: `HUE_HOME` refers to the location of your Hue installation. For package installs, this is usually `/usr/lib/hue`; for parcel installs, this is usually, `/opt/cloudera/parcels/<parcel version>/lib/hue/`.

   d. (**InnoDB only**) Drop the foreign key:

```
mysql > ALTER TABLE auth_permission DROP FOREIGN KEY content_type_id_refs_id_XXXXXX;
```

   e. Delete the rows in the `django_content_type` table:

```
mysql > DELETE FROM hue.django_content_type;
```

   f. Load the data:

```
$ <HUE_HOME>/build/env/bin/hue loaddata <some-temporary-file>.json
```

   g. (**InnoDB only**) Add the foreign key:

```
$ mysql -u hue -p <secretpassword>
mysql > ALTER TABLE auth_permission ADD FOREIGN KEY (`content_type_id`)
REFERENCES `django_content_type` (`id`);
```

5. Start the Hue server.

```
sudo service hue start
```

Configuring the Hue Server to Store Data in PostgreSQL (#cdh_ig_hue_database_postgresql)

Warning: Hue requires PostgreSQL 8.4 or higher.

1. Stop the Hue server, if running.

```
sudo service hue stop
```

2. Backup the existing database:
   a. Dump the existing database data to a .json file.

```
$ sudo -u hue <HUE_HOME>/build/env/bin/hue dumpdata > <some-temporary-file>.json
```

   Note: `HUE_HOME` refers to the location of your Hue installation. For package installs, this is usually `/usr/lib/hue`; for parcel installs, this is usually, `/opt/cloudera/parcels/<parcel version>/lib/hue/`.

   b. Delete all JSON objects with `useradmin.userprofile` in the `model` field from <some-temporary-file>.json. For example.

```
{
    "pk": 1,
     "model":  "useradmin.userprofile",
      "fields": {
        "creation_method":  "HUE",
        "user": 1,
        "home_directory":  "/user/alice"
    }
  },
  {
     "pk": 2,
     "model":  "useradmin.userprofile",
     "fields": {
       "creation_method":  "HUE",
       "user": 1100714,
       "home_directory":  "/user/bob"
    }
  },
.....
```

3. Install the PostgreSQL client dev packages.

OS	Commandcd /
RHEL	`$ sudo yum install postgresql-devel gcc python-devel`
SLES	`$ sudo zypper install postgresql-devel gcc python-devel`

Ubuntu or Debian	`$ sudo apt-get install postgresql-devel gcc python-devel`

4. Install the Python modules that provide the PostgreSQL connector.

```
sudo -u hue <HUE_HOME>/build/env/bin/pip install setuptools
sudo -u hue <HUE_HOME>/build/env/bin/pip install psycopg2
```

5. Install the PostgreSQL server.

OS	Command
RHEL	`$ sudo yum install postgresql-server`
SLES	`$ sudo zypper install postgresql-server`
Ubuntu or Debian	`$ sudo apt-get install postgresql`

6. Initialize the data directories:

```
$ service postgresql initdb
```

7. Configure `/var/lib/pgsql/data/pg_hba.conf` for client authentication.
   a. Set the authentication method for local to `trust`.
   b. Set the authentication method for host to `password`.
   c. Append the following line to the end of the file.

```
host
hue
hue
0.0.0.0/0
md5
```

8. Start the PostgreSQL server.

```
$ su - postgres
# /usr/bin/postgres -D /var/lib/pgsql/data > logfile 2>&1 &
```

9. Configure PostgreSQL to listen on all network interfaces.
   Edit `/var/lib/pgsql/data/postgresql.conf` and set list_addresses:

```
listen_addresses = '0.0.0.0'    # Listen on all addresses
```

10. Create the hue database and grant privileges to a `hue` user to manage the database.

```
# psql -U postgres
postgres=# create database hue;
postgres=# \c hue;
You are now connected to database 'hue'.
postgres=# create user hue with password '<secretpassword>';
postgres=# grant all privileges on database hue to hue;
postgres=# \q
```

11. Restart the PostgreSQL server.

```
$ sudo service postgresql restart
```

12. Verify connectivity.

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

13. Configure the PostgreSQL server to start at boot.

OS	Command

RHEL	`$ sudo /sbin/chkconfig postgresql on`
	`$ sudo /sbin/chkconfig --list postgresql`
	`postgresql 0:off 1:off 2:on 3:on 4:on 5:on 6:off`

SLES	`$ sudo chkconfig --add postgresql`
Ubuntu or Debian	`$ sudo chkconfig postgresql on`

14. Configure `/etc/hue/conf/hue.ini` to use PostgreSQL. Modify these options as appropriate and paste below `[[database]]` and `[desktop]`:

```
host=localhost
port=5432
engine=postgresql_psycopg2
user=hue
password=<secretpassword>
name=hue
```

15. Load any backed up data:
    a. Ensure a logs directory exists and is writable at `/opt/cloudera/parcels/CDH/lib/hue/logs`.
    b. Ensure the logs directory has `hue` user and group ownership.
    c. Synchronize and migrate the database.

```
$ sudo mkdir /opt/cloudera/parcels/CDH/lib/hue/logs
$ sudo chown hue:hue /opt/cloudera/parcels/CDH/lib/hue/logs
$ sudo -u hue <HUE_HOME>/build/env/bin/hue syncdb --noinput
$ sudo -u hue <HUE_HOME>/build/env/bin/hue migrate
```

    d. Determine the foreign key ID.

```
bash# su - postgres
$ psql -h localhost -U hue -d hue
postgres=# \d auth_permission;
```

    e. Drop the foreign key that you retrieved in the previous step.

```
postgres=# ALTER TABLE auth_permission DROP CONSTRAINT content_type_id_refs_id_<XXXXXX>;
```

    f. Delete the rows in the `django_content_type` table.

```
postgres=# TRUNCATE django_content_type CASCADE;
```

    g. Load the data.

```
$ sudo -u hue <HUE_HOME>/build/env/bin/hue loaddata <some-temporary-file>.json
```

    h. Add the foreign key:

```
bash# su - postgres
$ psql -h localhost -U hue -d hue
postgres=# ALTER TABLE auth_permission ADD CONSTRAINT
content_type_id_refs_id_<XXXXXX> FOREIGN KEY (content_type_id) REFERENCES
django_content_type(id) DEFERRABLE INITIALLY DEFERRED;
```

Configuring the Hue Server to Store Data in Oracle (#cdh_ig_hue_database_oracle)

Important: Configure the database for character set `AL32UTF8` and national character set `UTF8`.

1. Ensure Python 2.6 or higher is installed on the server Hue is running on.
2. Download the Oracle client libraries at **Instant Client for Linux x86-64 (http://www.oracle.com/technetwork/topics/linuxx86-64soft-092277.html)** Version 11.1.0.7.0, Basic and SDK (with headers) zip files to the same directory.
   Note: The Oracle 12 instant client is currently not supported. Hue works with the Oracle 12 database, but only with the Oracle 11 client libraries.
3. Unzip the zip files.
4. Set environment variables to reference the libraries.

```
$ export ORACLE_HOME=<download directory>
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME
```

5. Create a symbolic link for the shared object:

```
$ cd $ORACLE_HOME
$ ln -sf libclntsh.so.11.1 libclntsh.so
```

6. Get a data dump by executing:
   Note: `HUE_HOME` is a reference to the location of your Hue installation. For package installs, this is usually `/usr/lib/hue`; for parcel installs, this is usually, `/opt/cloudera/parcels/<parcel version>/lib/hue/`.

```
$ <HUE_HOME>/build/env/bin/hue dumpdata > <some-temporary-file>.json --indent 2
```

7. Edit the Hue configuration file `hue.ini`. Directly below the `[[database]]` section under the `[desktop]` line, add the following options (and modify accordingly for your setup):

```
host=localhost
port=1521
engine=oracle
user=hue
password=<secretpassword>
name=<SID of the Oracle database, for example, 'XE'>
```

   To use the Oracle service name instead of the SID, use the following configuration instead:

```
port=0
engine=oracle
```

```
user=hue
password=password
name=oracle.example.com:1521/orcl.example.com
```

The directive `port=0` allows Hue to use a service name. The `name` string is the connect string, including hostname, port, and service name.

To add support for a multithreaded environment, set the `threaded` option to `true` under the `[desktop]>[[database]]` section.

```
options={'threaded':true}
```

8. Grant required permissions to the Hue user in Oracle:

```
GRANT CREATE <sequence> TO <user>;
GRANT CREATE <session> TO <user>;
GRANT CREATE <table> TO <user>;
GRANT CREATE <view> TO <user>;
GRANT CREATE <procedure> TO <user>;
GRANT CREATE <trigger> TO <user>;
GRANT EXECUTE ON sys.dbms_crypto TO <user>;
GRANT EXECUTE ON SYS.DBMS_LOB TO <user>;
```

9. As the `hue` user, configure Hue to load the existing data and create the necessary database tables. You will need to run both the `syncdb` and `migrate` commands. When running these commands, Hue will try to access a `logs` directory, located at `/opt/cloudera/parcels/CDH/lib/hue/logs`, which might be missing. If that is the case, first create the `logs` directory and give the `hue` user and group ownership of the directory.

```
$ sudo mkdir /opt/cloudera/parcels/CDH/lib/hue/logs
$ sudo chown hue:hue /opt/cloudera/parcels/CDH/lib/hue/logs
$ sudo -u hue <HUE_HOME>/build/env/bin/hue syncdb --noinput
$ sudo -u hue <HUE_HOME>/build/env/bin/hue migrate
```

10. Ensure you are connected to Oracle as the `hue` user, then run the following command to delete all data from Oracle tables:

```
SELECT 'DELETE FROM ' || '.' || table_name || ';' FROM user_tables;
```

11. Run the statements generated in the preceding step.
12. Commit your changes.

```
commit;
```

13. Load the data.

```
$ sudo -u hue <HUE_HOME>/build/env/bin/hue loaddata <some-temporary-file>.json
```

Categories: Databases (../categories/hub_databases.html) | Hue (../categories/hub_hue.html) | MariaDB (../categories/hub_mariadb.html) | MySQL (../categories/hub_mysql.html) | Oracle (../categories/hub_oracle.html) | PostgreSQL (../categories/hub_postgresql.html) | All Categories (../categories/hub.html)

- About Cloudera (http://www.cloudera.com/about-cloudera.html)
- Resources (http://www.cloudera.com/resources.html)
- Contact (http://www.cloudera.com/contact-us.html)
- Careers (http://www.cloudera.com/about-cloudera/careers.html)
- Press (/about-cloudera/press-center.html)
- Documentation (/documentation.html)

United States: +1 888 789 1488
Outside the US: +1 650 362 0488

- (https://www.linkedin.com/company/cloudera)
- (https://www.facebook.com/cloudera)
- (https://twitter.com/cloudera)
- (/contact-us.html)

Terms & Conditions (/legal/terms-and-conditions.html) | Privacy Policy (/legal/privacy-policy.html)

Page generated November 2, 2016.