Module 5

Understanding Hive

Thanachart Numnonda, Executive Director, IMC Institute

Thanisa Numnonda, Faculty of Information Technology,

King Mongkut's Institute of Technology Ladkrabang

Introduction

A Petabyte Scale Data Warehouse Using Hadoop

Hive is developed by Facebook, designed to enable easy data summarization, ad-hoc querying and analysis of large volumes of data. It provides a simple query language called Hive QL, which is based on SQL

What Hive is NOT

Hive is not designed for online transaction processing and does not offer real-time queries and row level updates. It is best used for batch jobs over large sets of immutable data (like web logs, etc.).

Sample HiveQL

The Query compiler uses the information stored in the metastore to convert SQL queries into a sequence of map/reduce jobs, e.g. the following query

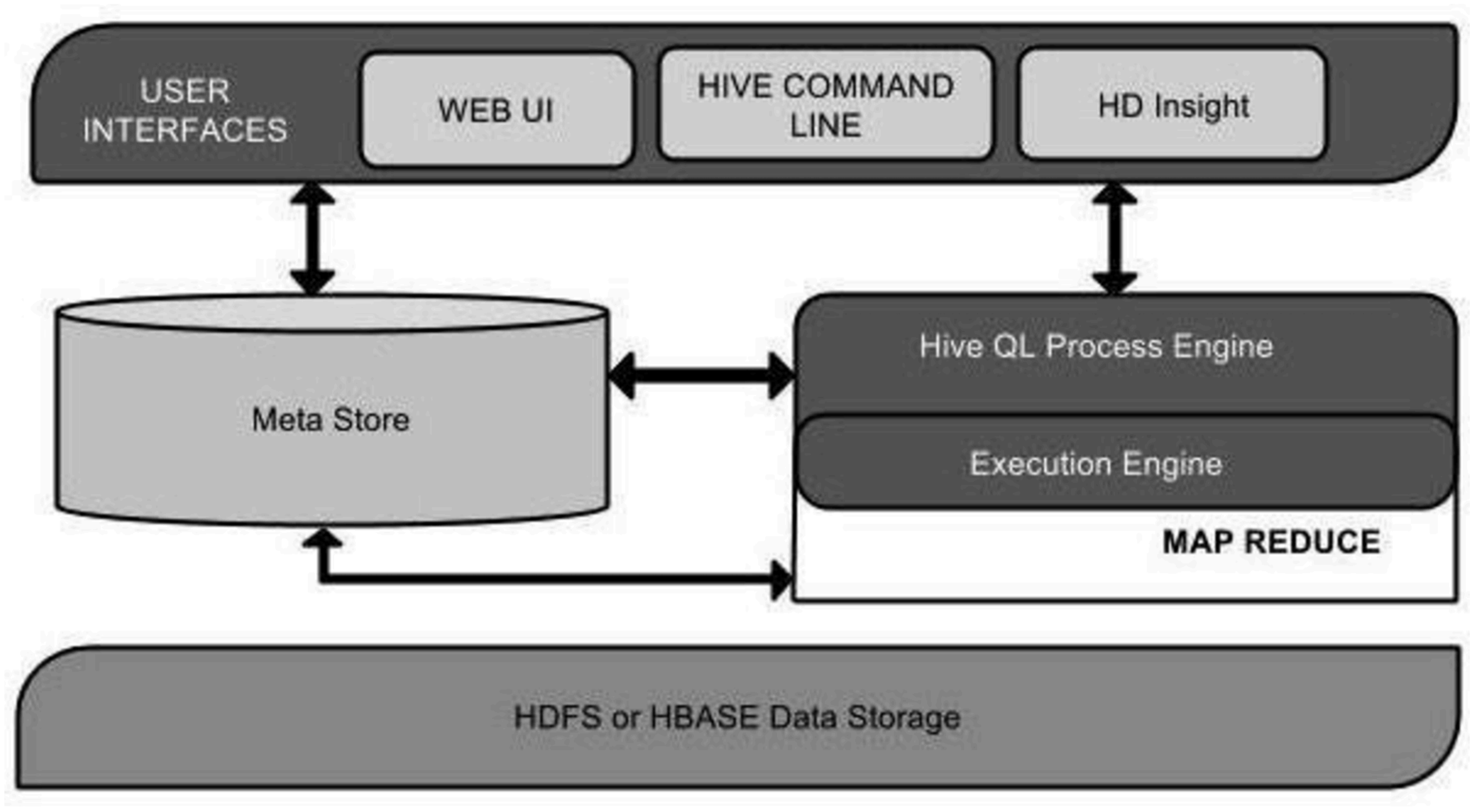
```
SELECT * FROM t where t.c = 'xyz'

SELECT t1.c2 FROM t1 JOIN t2 ON (t1.c1 = t2.c1)

SELECT t1.c1, count(1) from t1 group by t1.c1
```

System Architecture and Components

User Interface	Hive is a data warehouse infrastructure software that can create interaction between user and HDFS. The user interfaces that Hive supports are Hive Web UI, Hive command line, and Hive HD Insight (In Windows server).
Meta Store	Hive chooses respective database servers to store the schema or Metadata of tables, databases, columns in a table, their data types, and HDFS mapping.
HiveQL Process Engine	HiveQL is similar to SQL for querying on schema info on the Metastore. It is one of the replacements of traditional approach for MapReduce program. Instead of writing MapReduce program in Java, we can write a query for MapReduce job and process it.
Execution Engine	The conjunction part of HiveQL process Engine and MapReduce is Hive Execution Engine. Execution engine processes the query and generates results as same as MapReduce results. It uses the flavor of MapReduce.
HDFS or HBASE	Hadoop distributed file system or HBASE are the data storage techniques to store data into file system.

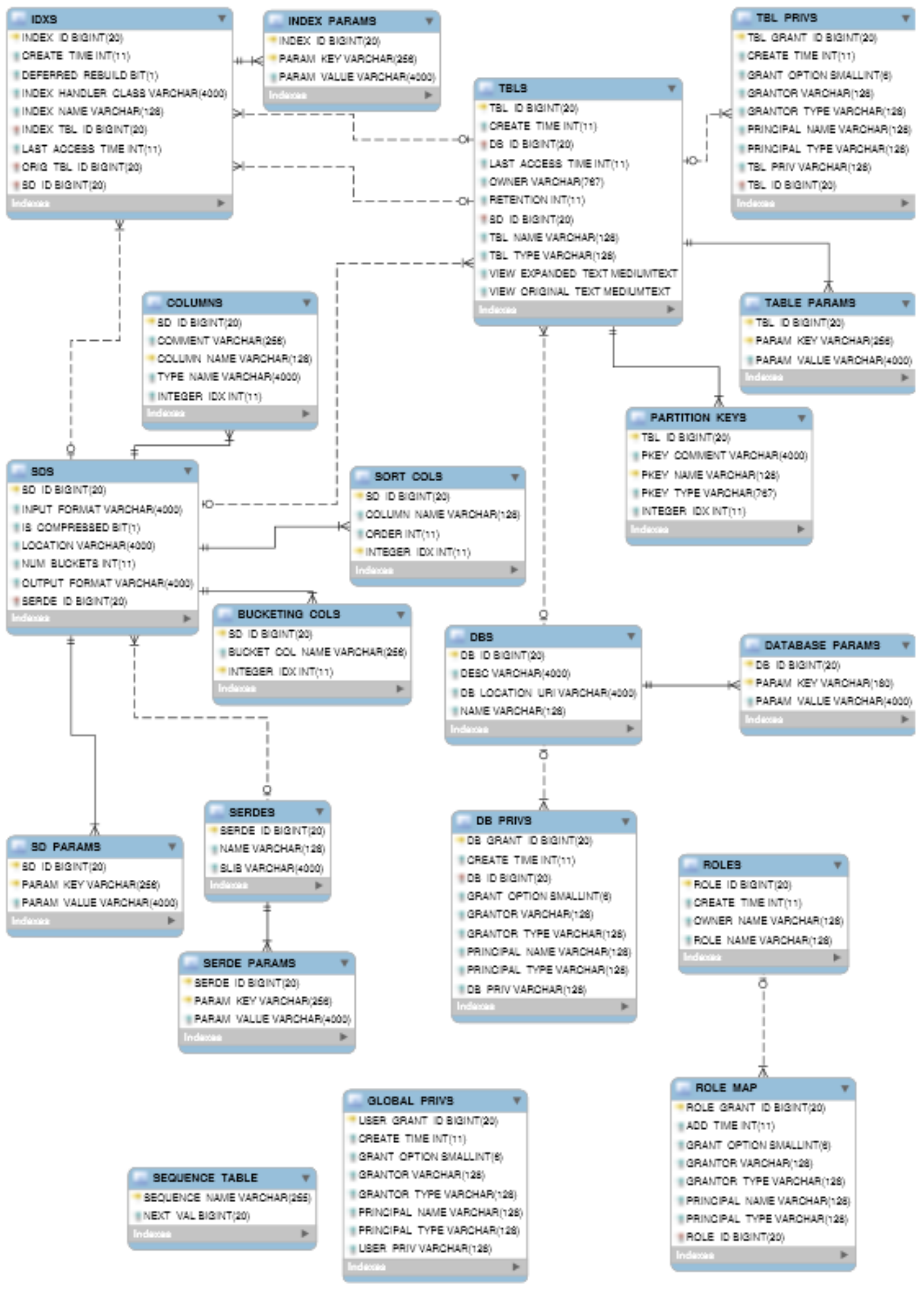
hive.apache.org

Architecture Overview



https://www.tutorialspoint.com/hive/hive_introduction.htm

Hive Metastore



Hive Metastore is a repository to keep all Hive metadata; Tables and Partitions definition.

By default, Hive will store its metadata in Derby DB

Hive Built in Functions

hive.apache.org

Return Type	Function Name (Signature)	Description
BIGINT	round(double a)	returns the rounded BIGINT value of the double
BIGINT	floor(double a)	returns the maximum BIGINT value that is equal or less than the double
BIGINT	ceil(double a)	returns the minimum BIGINT value that is equal or greater than the double
double	rand(), rand(int seed)	returns a random number (that changes from row to row). Specifiying the seed will make sure the generated random number sequence is deterministic.
string	concat(string A, string B,...)	returns the string resulting from concatenating B after A. For example, concat('foo', 'bar') results in 'foobar'. This function accepts arbitrary number of arguments and return the concatenation of all of them.
string	substr(string A, int start)	returns the substring of A starting from start position till the end of string A. For example, substr('foobar', 4) results in 'bar'
string	substr(string A, int start, int length)	returns the substring of A starting from start position with the given length e.g. substr('foobar', 4, 2) results in 'ba'

Hive Built in Functions (Cont.)

hive.apache.org

Return Type	Function Name (Signature)	Description
string	upper(string A)	returns the string resulting from converting all characters of A to upper case e.g. upper('fOoBaR') results in 'FOOBAR'
string	ucase(string A)	Same as upper
string	lower(string A)	returns the string resulting from converting all characters of B to lower case e.g. lower('fOoBaR') results in 'foobar'
string	lcase(string A)	Same as lower
string	trim(string A)	returns the string resulting from trimming spaces from both ends of A e.g. trim(' foobar ') results in 'foobar'
string	ltrim(string A)	returns the string resulting from trimming spaces from the beginning(left hand side) of A. For example, ltrim(' foobar ') results in 'foobar '
string	rtrim(string A)	returns the string resulting from trimming spaces from the end(right hand side) of A. For example, rtrim(' foobar ') results in ' foobar'
string	regexp_replace(string A, string B, string C)	returns the string resulting from replacing all substrings in B that match the Java regular expression syntax(See Java egular expressions syntax) with C. For example, regexp_replace('foobar', 'oo|ar',) returns 'fb'

Hive Built in Functions (Cont.)

hive.apache.org

Return Type	Function Name (Signature)	Description
string	from_unixtime(int unixtime)	convert the number of seconds from unix epoch (1970-01-01 00:00:00 UTC) to a string representing the timestamp of that moment in the current system time zone in the format of "1970-01-01 00:00:00"
string	to_date(string timestamp)	Return the date part of a timestamp string: to_date("1970-01-01 00:00:00") = "1970-01-01"
int	year(string date)	Return the year part of a date or a timestamp string: year("1970-01-01 00:00:00") = 1970, year("1970-01-01") = 1970
int	month(string date)	Return the month part of a date or a timestamp string: month("1970-11-01 00:00:00") = 11, month("1970-11-01") = 11
int	day(string date)	Return the day part of a date or a timestamp string: day("1970-11-01 00:00:00") = 1, day("1970-11-01") = 1
string	get_json_object(string json_string, string path)	Extract json object from a json string based on json path specified, and return json string of the extracted json object. It will return null if the input json string is invalid

Hive Aggregate Functions (Cont.)

Return Type	Aggregation Function Name (Signature)	Description
BIGINT	count(*), count(expr), count(DISTINCT expr[, expr_.])	count(*) - Returns the total number of retrieved rows, including rows containing NULL values; count(expr) - Returns the number of rows for which the supplied expression is non-NULL; count(DISTINCT expr[, expr]) - Returns the number of rows for which the supplied expression(s) are unique and non-NULL.
DOUBLE	sum(col), sum(DISTINCT col)	returns the sum of the elements in the group or the sum of the distinct values of the column in the group
DOUBLE	avg(col), avg(DISTINCT col)	returns the average of the elements in the group or the average of the distinct values of the column in the group
DOUBLE	min(col)	returns the minimum value of the column in the group
DOUBLE	max(col)	returns the maximum value of the column in the group

Examples

```
hive> SELECT round(2.6) from temp;     => 3.0

hive> SELECT floor(2.6) from temp;     => 2.0

hive> SELECT ceil(2.6) from temp;      => 3.0
```

Running Hive

Hive Shell

Interactive

hive

Script

hive -f myscript

Inline

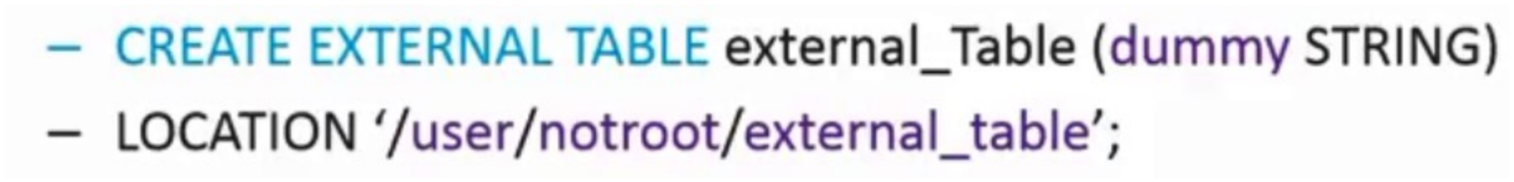
*hive -e 'SELECT * FROM mytable'*

Hive Tables

- Managed- **CREATE TABLE**
  - LOAD- File moved into Hive's data warehouse directory
  - DROP- Both data and metadata are deleted.

- External- **CREATE EXTERNAL TABLE**
  - LOAD- No file moved
  - DROP- Only metadata deleted
  - Use when sharing data between Hive and Hadoop applications or you want to use multiple schema on the same data

Hive External Table



```
– CREATE EXTERNAL TABLE external_Table (dummy STRING)
– LOCATION '/user/notroot/external_table';
```

Dropping External Table using Hive:-
Hive will delete metadata from metastore
Hive will NOT delete the HDFS file
You need to manually delete the HDFS file

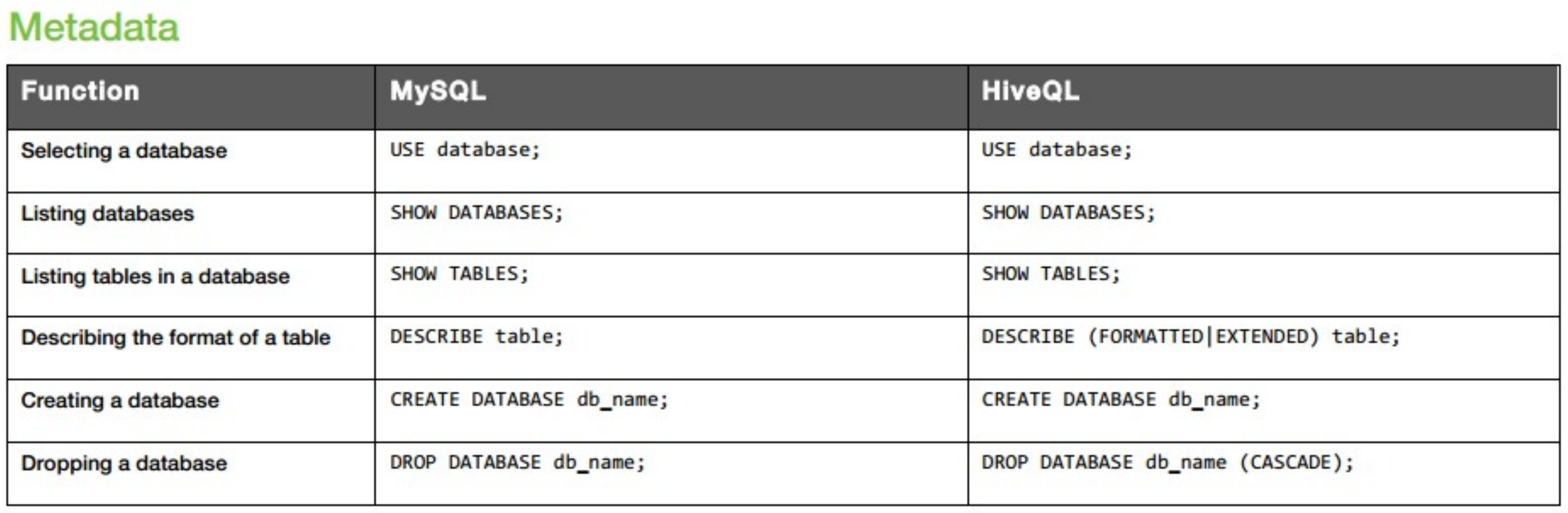
Java JDBC for Hive

```
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.DriverManager;

public class HiveJdbcClient {
  private static String driverName = "org.apache.hadoop.hive.jdbc.HiveDriver";

  public static void main(String[] args) throws SQLException {
    try {
      Class.forName(driverName);
    } catch (ClassNotFoundException e) {
      e.printStackTrace();
      System.exit(1);
    }
    Connection con = 
DriverManager.getConnection("jdbc:hive://localhost:10000/default", "", "");
    Statement stmt = con.createStatement();
    String tableName = "testHiveDriverTable";
    stmt.executeQuery("drop table " + tableName);
    ResultSet res = stmt.executeQuery("create table " + tableName + " (key int, 
value string)");
```

Java JDBC for Hive (Cont.)

```
// show tables
String sql = "show tables '" + tableName + "'";
System.out.println("Running: " + sql);
res = stmt.executeQuery(sql);
if (res.next()) {
  System.out.println(res.getString(1));
}

// describe table
sql = "describe " + tableName;
System.out.println("Running: " + sql);
res = stmt.executeQuery(sql);
while (res.next()) {
  System.out.println(res.getString(1) + "\t" + res.getString(2));
}
```

HiveQL and MySQL Comparison

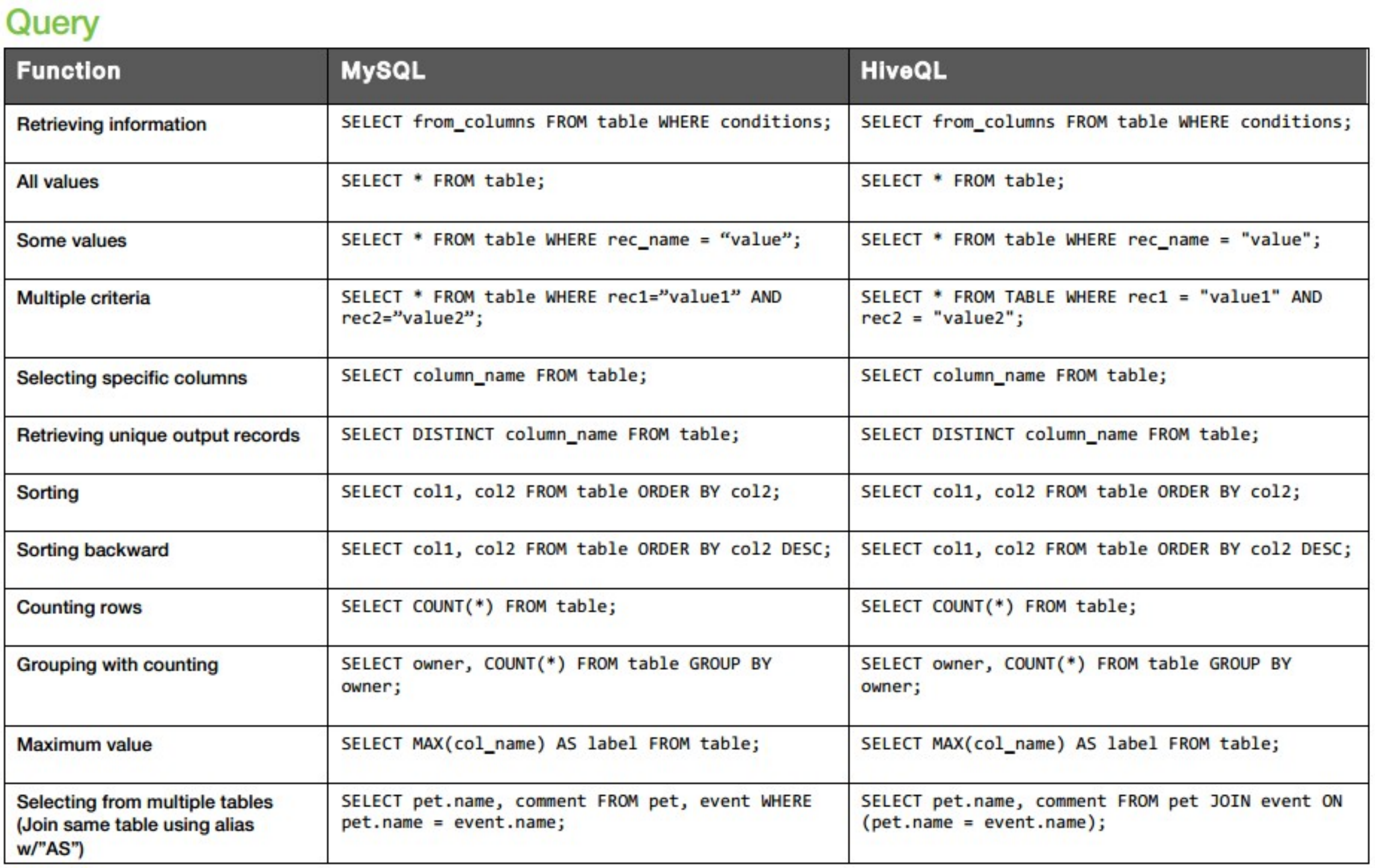


Metadata

Function	MySQL	HiveQL
Selecting a database	`USE database;`	`USE database;`
Listing databases	`SHOW DATABASES;`	`SHOW DATABASES;`
Listing tables in a database	`SHOW TABLES;`	`SHOW TABLES;`
Describing the format of a table	`DESCRIBE table;`	`DESCRIBE (FORMATTED|EXTENDED) table;`
Creating a database	`CREATE DATABASE db_name;`	`CREATE DATABASE db_name;`
Dropping a database	`DROP DATABASE db_name;`	`DROP DATABASE db_name (CASCADE);`

ortonworks.co
m

HiveQL and MySQL Query Comparison



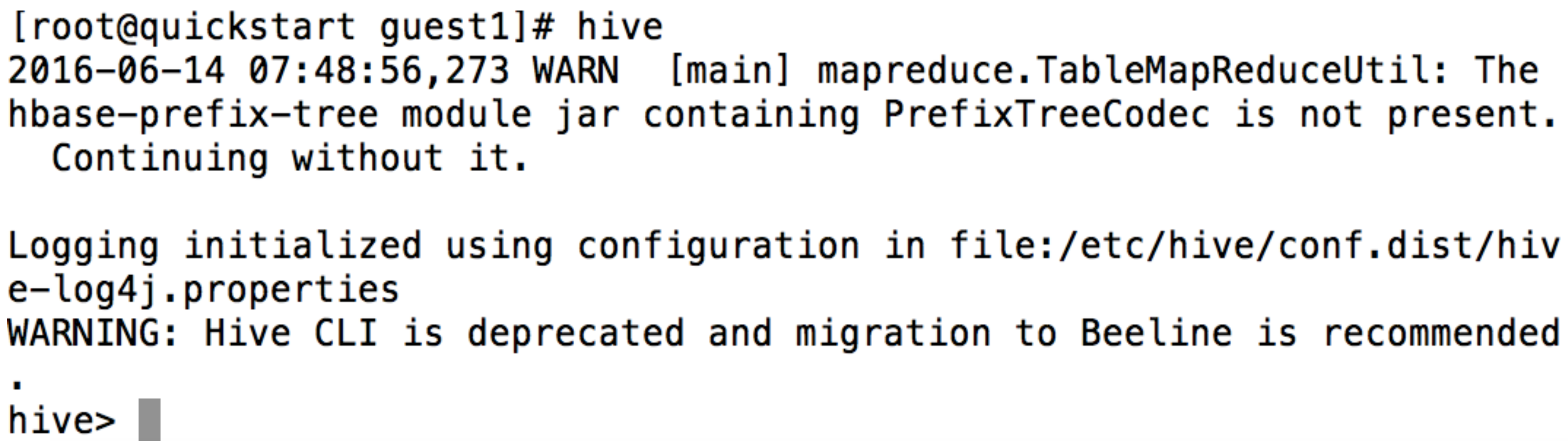
Query

Function	MySQL	HiveQL
Retrieving information	`SELECT from_columns FROM table WHERE conditions;`	`SELECT from_columns FROM table WHERE conditions;`
All values	`SELECT * FROM table;`	`SELECT * FROM table;`
Some values	`SELECT * FROM table WHERE rec_name = “value”;`	`SELECT * FROM table WHERE rec_name = "value";`
Multiple criteria	`SELECT * FROM table WHERE rec1=”value1” AND rec2=”value2”;`	`SELECT * FROM TABLE WHERE rec1 = "value1" AND rec2 = "value2";`
Selecting specific columns	`SELECT column_name FROM table;`	`SELECT column_name FROM table;`
Retrieving unique output records	`SELECT DISTINCT column_name FROM table;`	`SELECT DISTINCT column_name FROM table;`
Sorting	`SELECT col1, col2 FROM table ORDER BY col2;`	`SELECT col1, col2 FROM table ORDER BY col2;`
Sorting backward	`SELECT col1, col2 FROM table ORDER BY col2 DESC;`	`SELECT col1, col2 FROM table ORDER BY col2 DESC;`
Counting rows	`SELECT COUNT(*) FROM table;`	`SELECT COUNT(*) FROM table;`
Grouping with counting	`SELECT owner, COUNT(*) FROM table GROUP BY owner;`	`SELECT owner, COUNT(*) FROM table GROUP BY owner;`
Maximum value	`SELECT MAX(col_name) AS label FROM table;`	`SELECT MAX(col_name) AS label FROM table;`
Selecting from multiple tables (Join same table using alias w/”AS”)	`SELECT pet.name, comment FROM pet, event WHERE pet.name = event.name;`	`SELECT pet.name, comment FROM pet JOIN event ON (pet.name = event.name);`

ortonworks.co
m

Hands-On: Loading Data using Hive

Start Hive

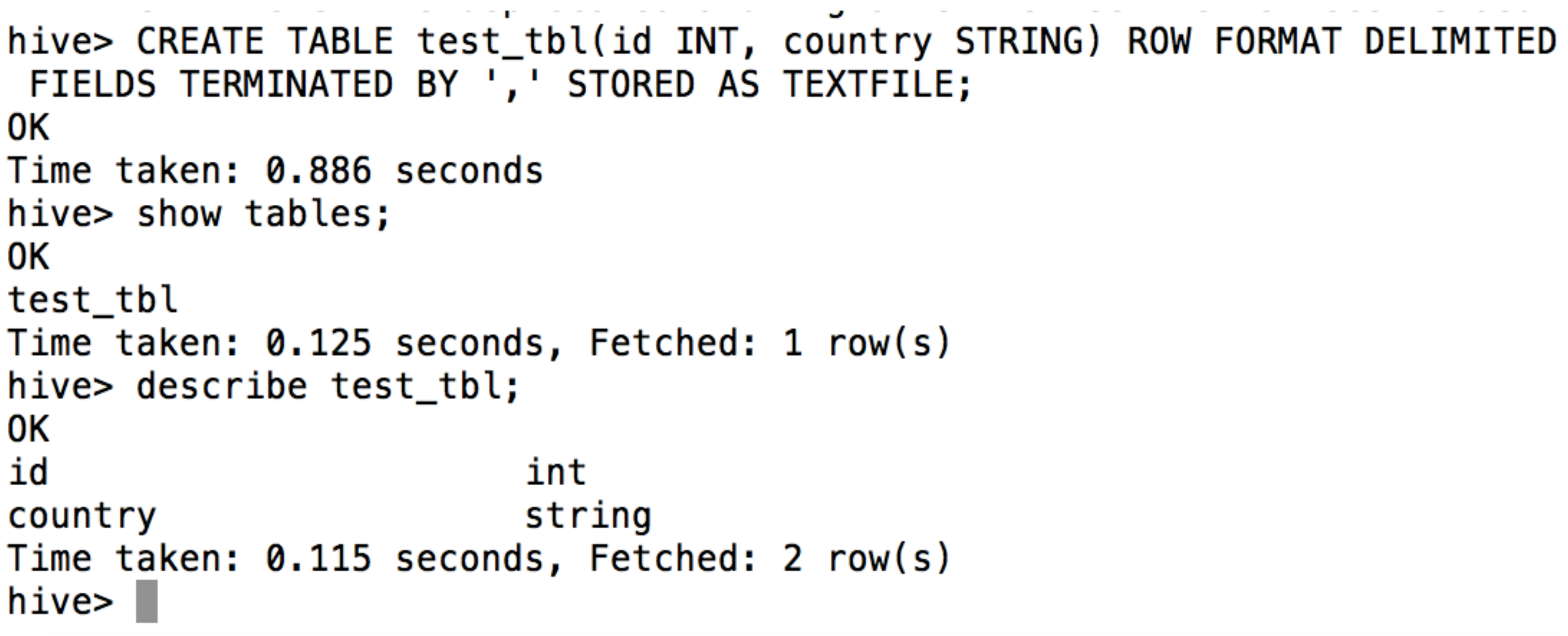


```
[root@quickstart guest1]# hive
2016-06-14 07:48:56,273 WARN  [main] mapreduce.TableMapReduceUtil: The
hbase-prefix-tree module jar containing PrefixTreeCodec is not present.
  Continuing without it.

Logging initialized using configuration in file:/etc/hive/conf.dist/hiv
e-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended
.
hive> 
```

Quit from Hive

```
hive> quit;
```

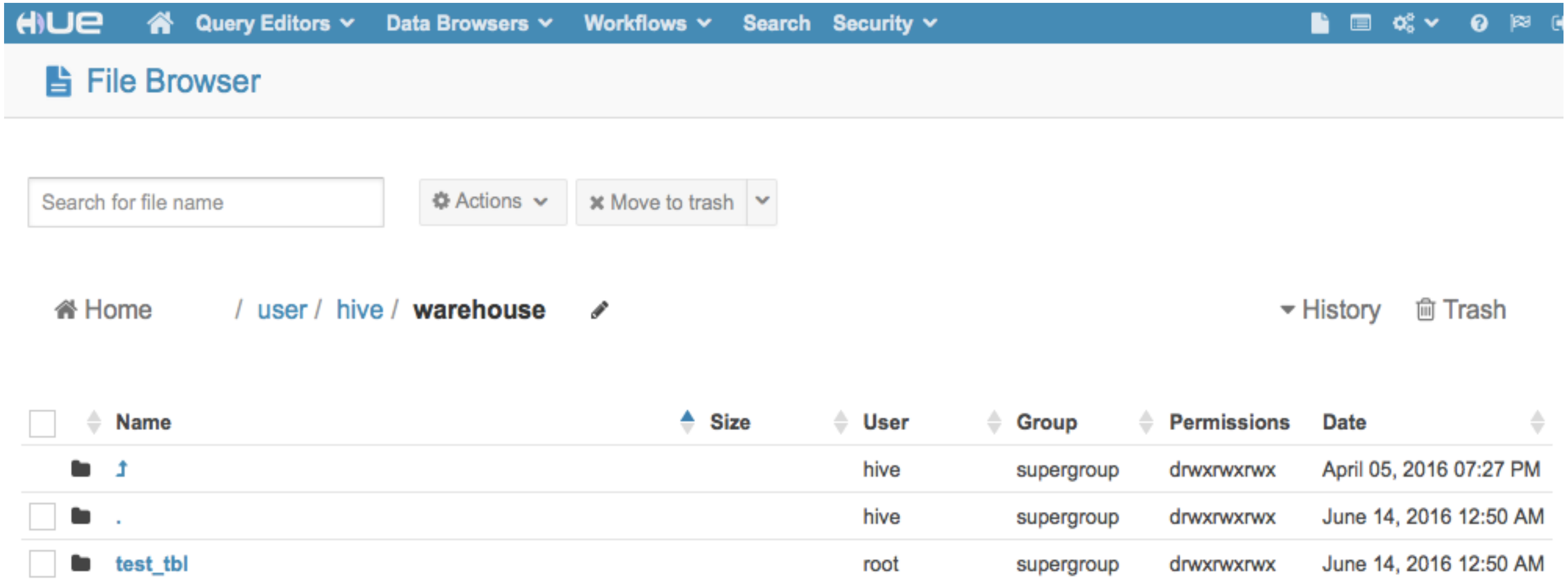
Create Hive Table



```
hive> CREATE TABLE test_tbl(id INT, country STRING) ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
OK
Time taken: 0.886 seconds
hive> show tables;
OK
test_tbl
Time taken: 0.125 seconds, Fetched: 1 row(s)
hive> describe test_tbl;
OK
id                      int
country                 string
Time taken: 0.115 seconds, Fetched: 2 row(s)
hive> 
```

See also: https://cwiki.apache.org/Hive/languagemanual-ddl.html

Reviewing Hive Table in HDFS



Home / user / hive / **warehouse**

Name	Size	User	Group	Permissions	Date
↥		hive	supergroup	drwxrwxrwx	April 05, 2016 07:27 PM
.		hive	supergroup	drwxrwxrwx	June 14, 2016 12:50 AM
test_tbl		root	supergroup	drwxrwxrwx	June 14, 2016 12:50 AM

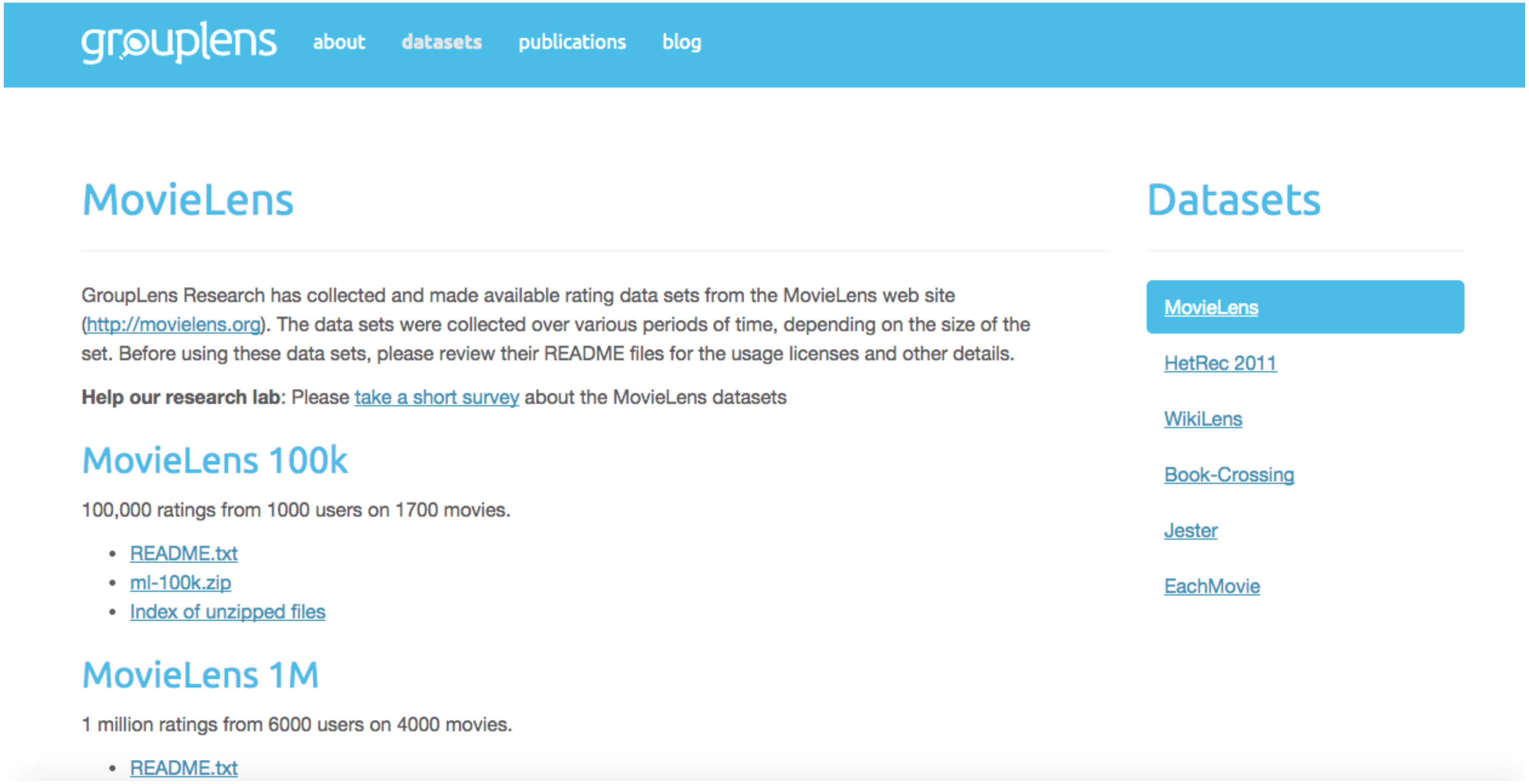
Alter and Drop Hive Table

```
Hive > alter table test_tbl add columns (remarks STRING);

hive > describe test_tbl;
OK
id  int
country string
remarks string
Time taken: 0.077 seconds
hive > drop table test_tbl;
OK
Time taken: 0.9 seconds
```

See also: https://cwiki.apache.org/Hive/adminmanual-metastoreadmin.html

Preparing Large Dataset

http://grouplens.org/datasets/movielens/



MovieLens

GroupLens Research has collected and made available rating data sets from the MovieLens web site (http://movielens.org). The data sets were collected over various periods of time, depending on the size of the set. Before using these data sets, please review their README files for the usage licenses and other details.

Help our research lab: Please take a short survey about the MovieLens datasets

MovieLens 100k

100,000 ratings from 1000 users on 1700 movies.

- README.txt
- ml-100k.zip
- Index of unzipped files

MovieLens 1M

1 million ratings from 6000 users on 4000 movies.

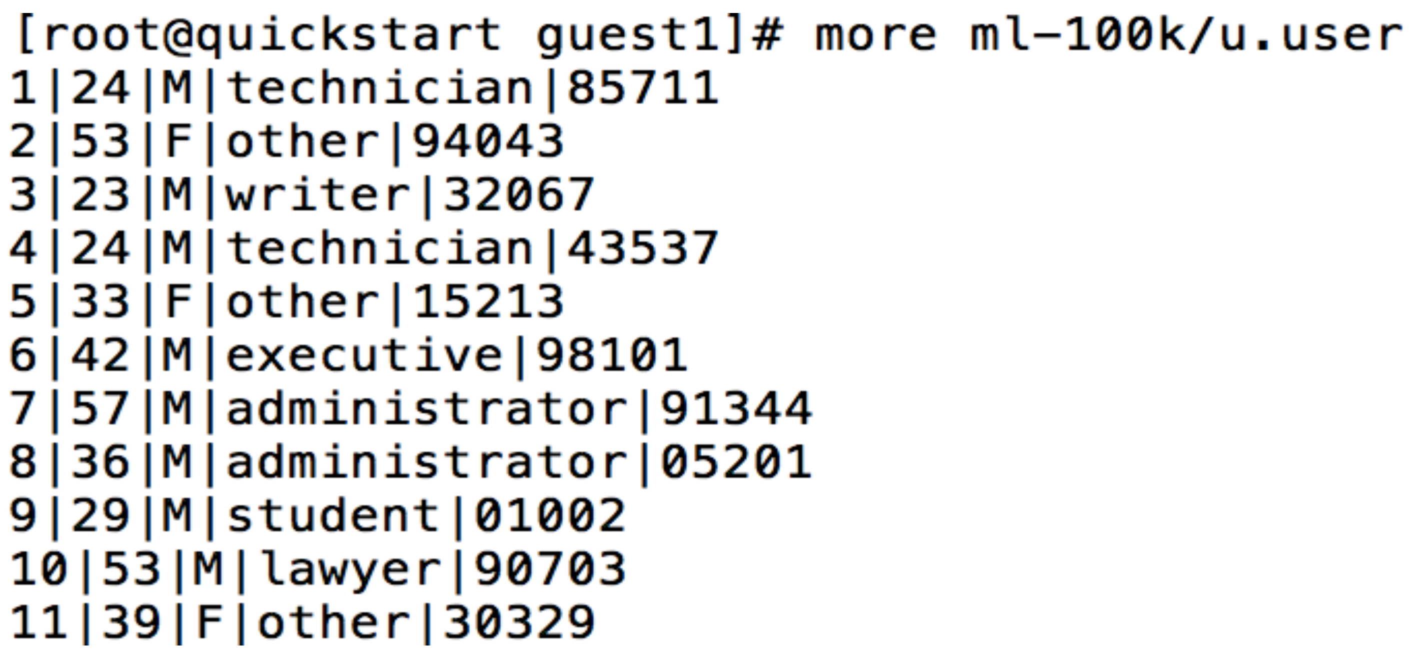
- README.txt

Datasets

- MovieLens
- HetRec 2011
- WikiLens
- Book-Crossing
- Jester
- EachMovie

MovieLen Dataset

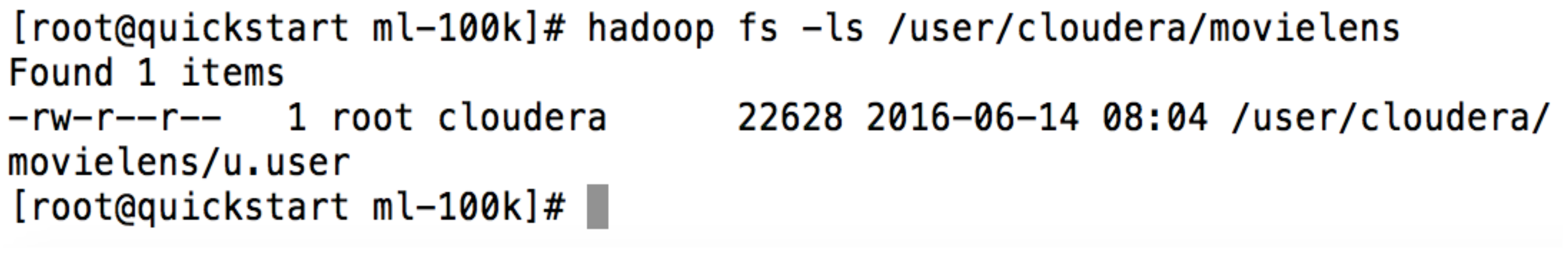
```
1)Type command > wget
http://files.grouplens.org/datasets/movielens/ml-100k.zip
2)Type command > yum install unzip
3)Type command > unzip ml-100k.zip
4)Type command > more ml-100k/u.user
```



```
[root@quickstart guest1]# more ml-100k/u.user
1|24|M|technician|85711
2|53|F|other|94043
3|23|M|writer|32067
4|24|M|technician|43537
5|33|F|other|15213
6|42|M|executive|98101
7|57|M|administrator|91344
8|36|M|administrator|05201
9|29|M|student|01002
10|53|M|lawyer|90703
11|39|F|other|30329
```

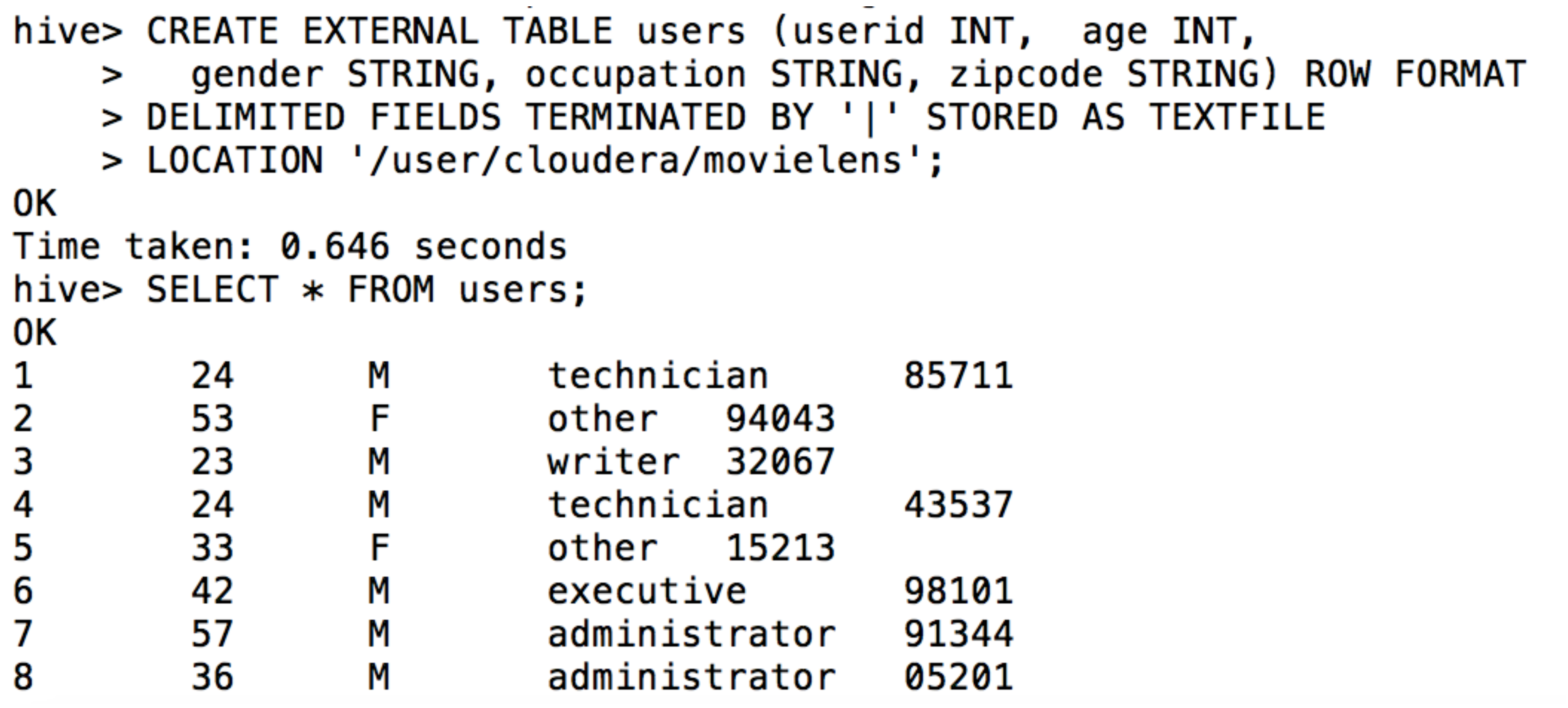
Moving dataset to HDFS

```
1)Type command > cd ml-100k
2)Type command > hadoop fs -mkdir /user/cloudera/movielens
3)Type command > hadoop fs -put u.user /user/cloudera/movielens
4)Type command > hadoop fs -ls /user/cloudera/movielens
```



```
[root@quickstart ml-100k]# hadoop fs -ls /user/cloudera/movielens
Found 1 items
-rw-r--r--   1 root cloudera      22628 2016-06-14 08:04 /user/cloudera/
movielens/u.user
[root@quickstart ml-100k]# 
```

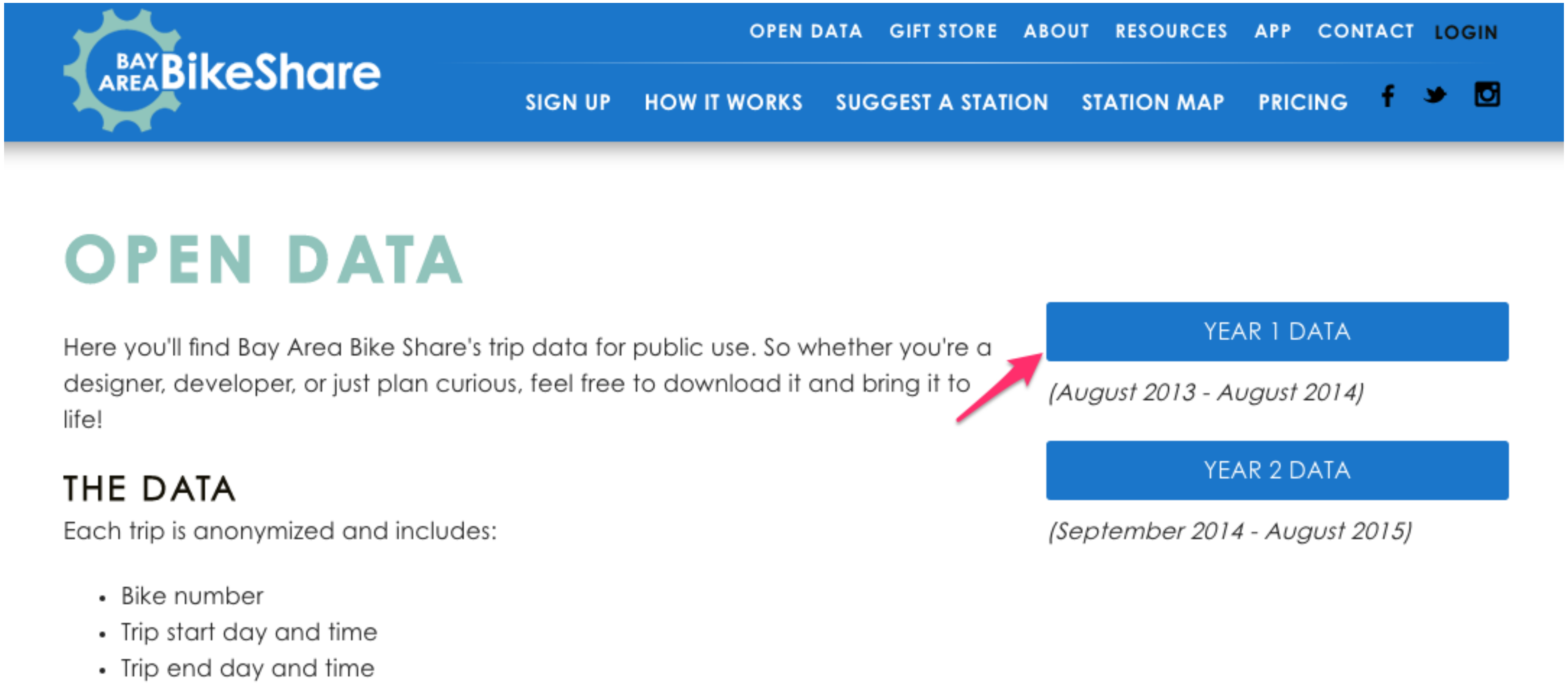
CREATE & SELECT Table



```
hive> CREATE EXTERNAL TABLE users (userid INT,  age INT,
    >   gender STRING, occupation STRING, zipcode STRING) ROW FORMAT
    > DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
    > LOCATION '/user/cloudera/movielens';
OK
Time taken: 0.646 seconds
hive> SELECT * FROM users;
OK
1       24      M       technician      85711
2       53      F       other   94043
3       23      M       writer  32067
4       24      M       technician      43537
5       33      F       other   15213
6       42      M       executive       98101
7       57      M       administrator   91344
8       36      M       administrator   05201
```

Bay Area Bike Share (BABS)

http://www.bayareabikeshare.com/open-data



OPEN DATA

Here you'll find Bay Area Bike Share's trip data for public use. So whether you're a designer, developer, or just plan curious, feel free to download it and bring it to life!

(August 2013 - August 2014)

(September 2014 - August 2015)

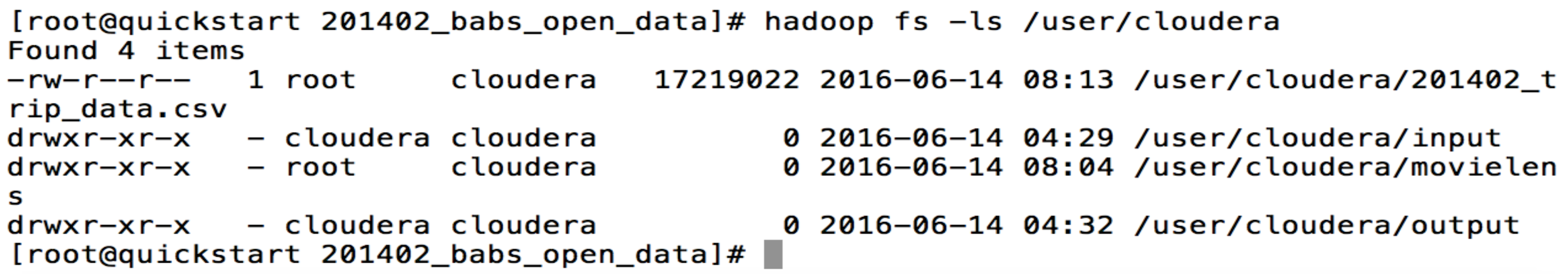
THE DATA

Each trip is anonymized and includes:

- Bike number
- Trip start day and time
- Trip end day and time

Preparing a bike data

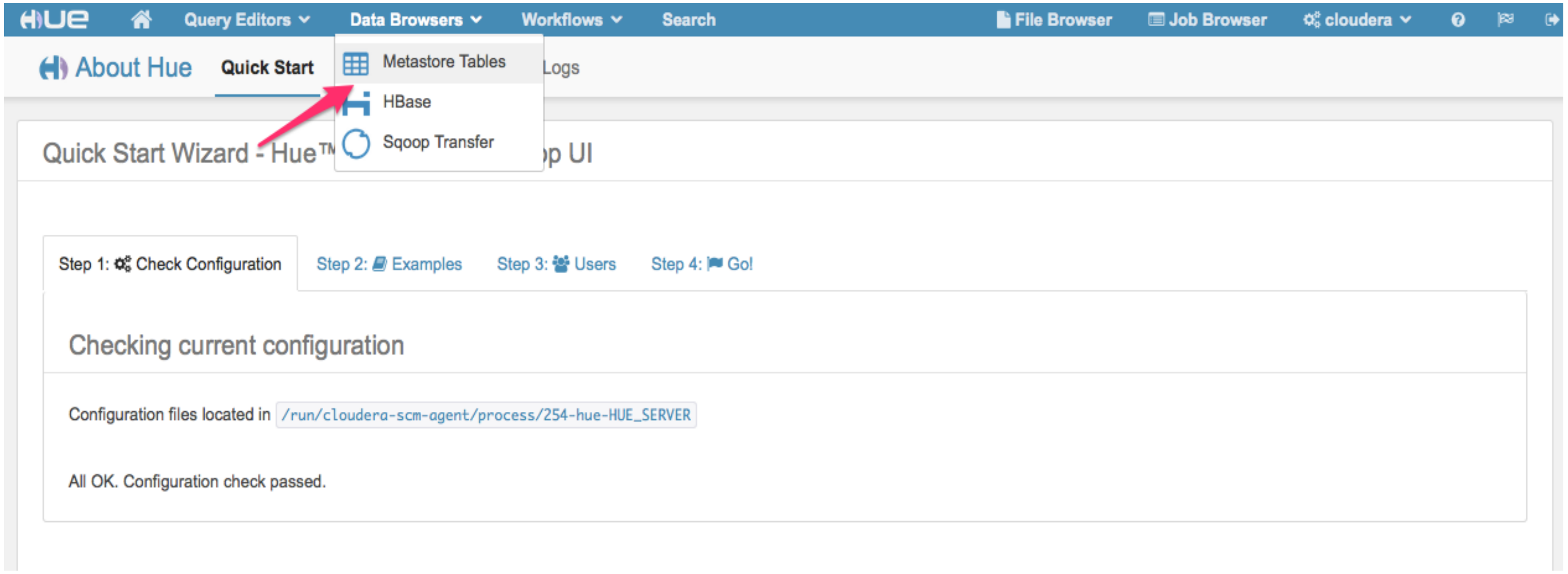
```
$wget https://s3.amazonaws.com/babs-open-data/
babs_open_data_year_1.zip
$unzip babs_open_data_year_1.zip
$cd 201402_babs_open_data/
$hadoop fs -put 201402_trip_data.csv
/user/cloudera
$ hadoop fs -ls /user/cloudera
```



```
[root@quickstart 201402_babs_open_data]# hadoop fs -ls /user/cloudera
Found 4 items
-rw-r--r--   1 root     cloudera   17219022 2016-06-14 08:13 /user/cloudera/201402_t
rip_data.csv
drwxr-xr-x   - cloudera cloudera          0 2016-06-14 04:29 /user/cloudera/input
drwxr-xr-x   - root     cloudera          0 2016-06-14 08:04 /user/cloudera/movielen
s
drwxr-xr-x   - cloudera cloudera          0 2016-06-14 04:32 /user/cloudera/output
[root@quickstart 201402_babs_open_data]# 
```

Importing CSV Data with the Metastore App

The BABS data set contains 4 CSVs that contain data for stations, trips, rebalancing (availability), and weather. We will import **trips** dataset using Metastore Tables

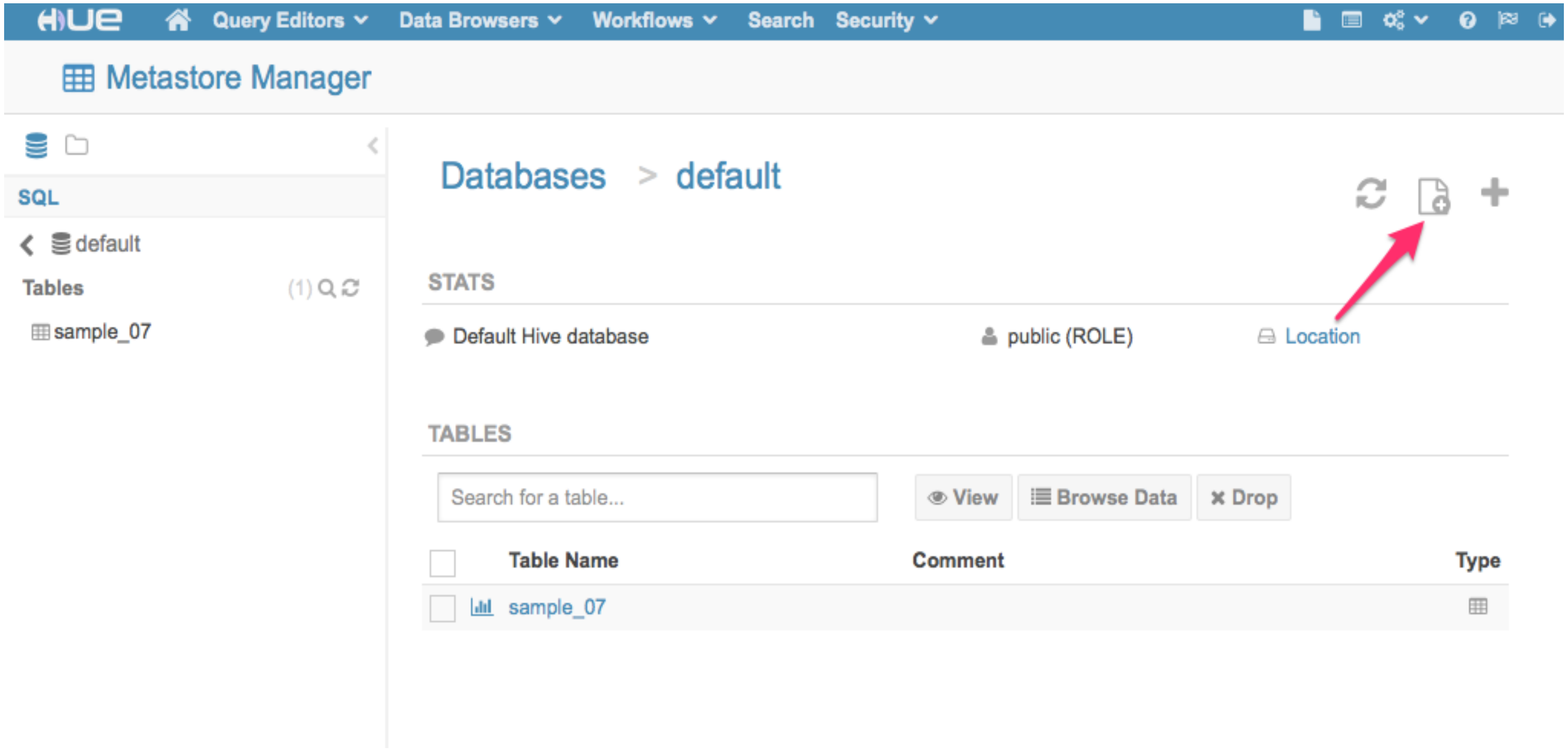


Checking current configuration

Configuration files located in `/run/cloudera-scm-agent/process/254-hue-HUE_SERVER`

All OK. Configuration check passed.

Select: Create a new table from a file



Databases > default

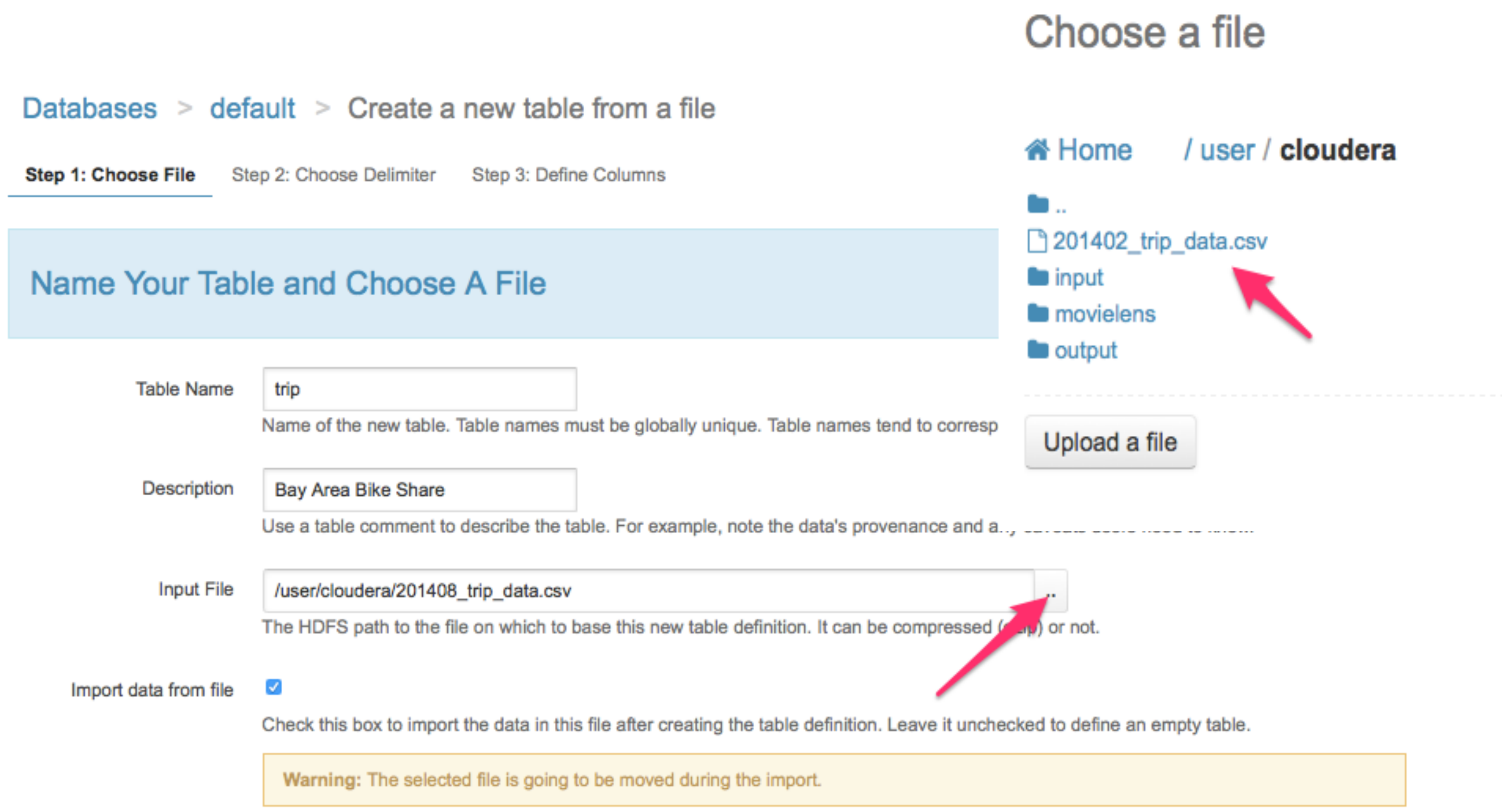
STATS

Default Hive database | public (ROLE) | Location

TABLES

Table Name	Comment	Type
sample_07		▦

Name a table and select a file


Choose a file

Home / user / cloudera

..
201402_trip_data.csv
input
movielens
output

Upload a file

Databases > default > Create a new table from a file

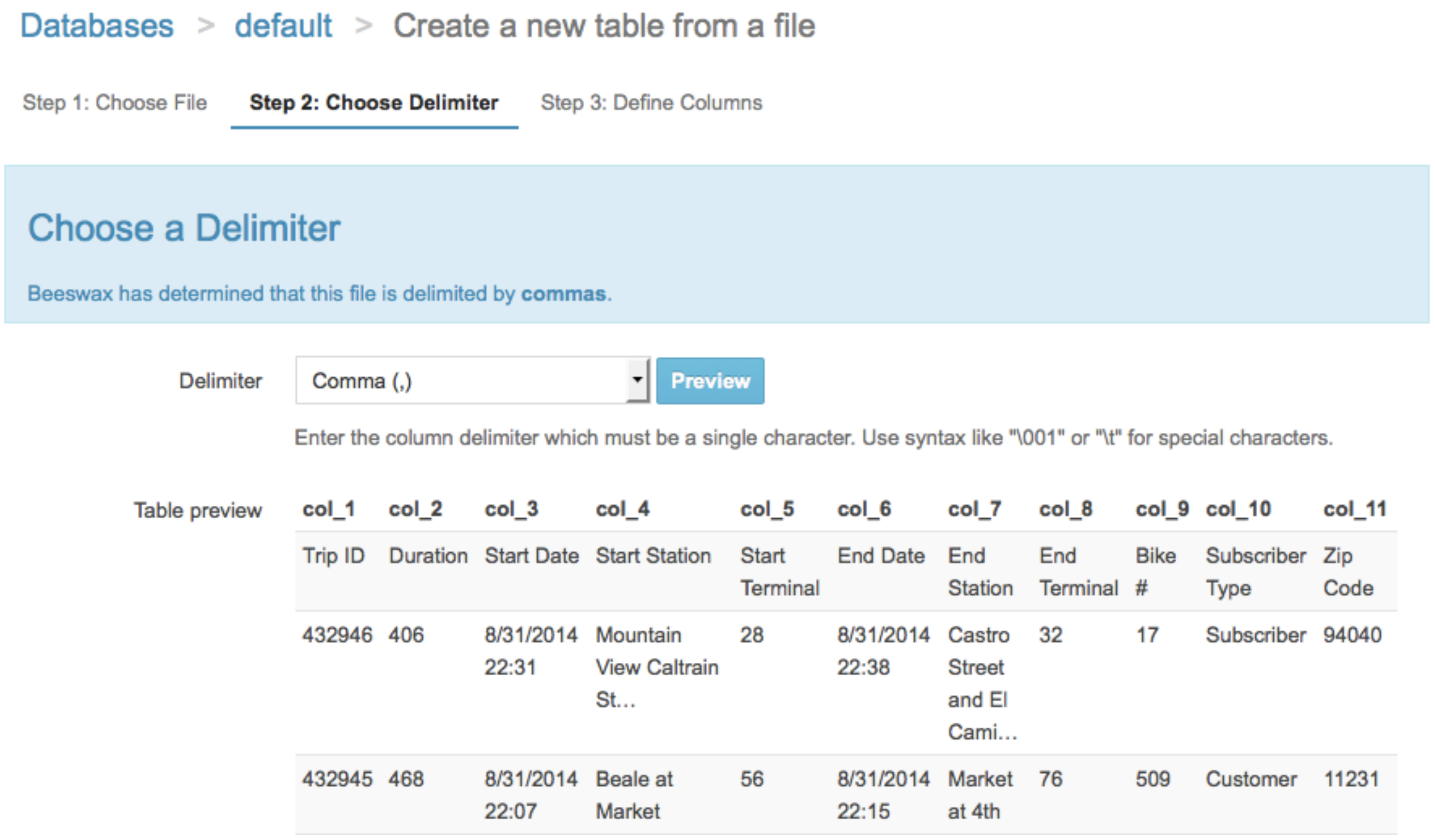
Step 1: Choose File | Step 2: Choose Delimiter | Step 3: Define Columns

Name Your Table and Choose A File

Table Name	trip
	Name of the new table. Table names must be globally unique. Table names tend to corresp
Description	Bay Area Bike Share
	Use a table comment to describe the table. For example, note the data's provenance and a...
Input File	/user/cloudera/201408_trip_data.csv
	The HDFS path to the file on which to base this new table definition. It can be compressed (gzip) or not.
Import data from file	✓
	Check this box to import the data in this file after creating the table definition. Leave it unchecked to define an empty table.

Warning: The selected file is going to be moved during the import.

Choose Delimiter



Databases > default > Create a new table from a file

Step 1: Choose File | **Step 2: Choose Delimiter** | Step 3: Define Columns

Choose a Delimiter

Beeswax has determined that this file is delimited by **commas**.

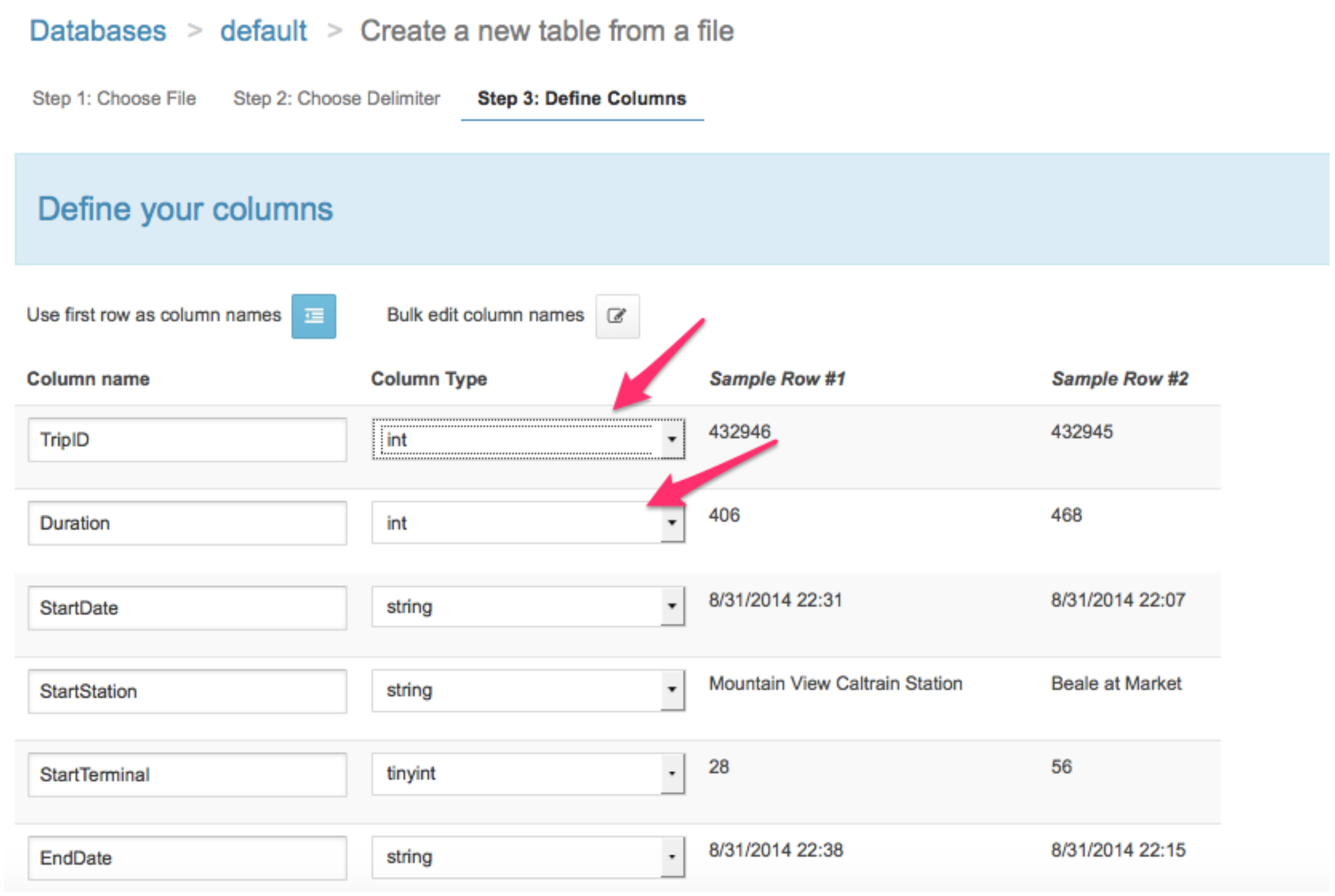
Delimiter: Comma (,) — Preview

Enter the column delimiter which must be a single character. Use syntax like "\001" or "\t" for special characters.

Table preview

col_1	col_2	col_3	col_4	col_5	col_6	col_7	col_8	col_9	col_10	col_11
Trip ID	Duration	Start Date	Start Station	Start Terminal	End Date	End Station	End Terminal	Bike #	Subscriber Type	Zip Code
432946	406	8/31/2014 22:31	Mountain View Caltrain St…	28	8/31/2014 22:38	Castro Street and El Cami…	32	17	Subscriber	94040
432945	468	8/31/2014 22:07	Beale at Market	56	8/31/2014 22:15	Market at 4th	76	509	Customer	11231

Define Column Types



Databases > default > Create a new table from a file

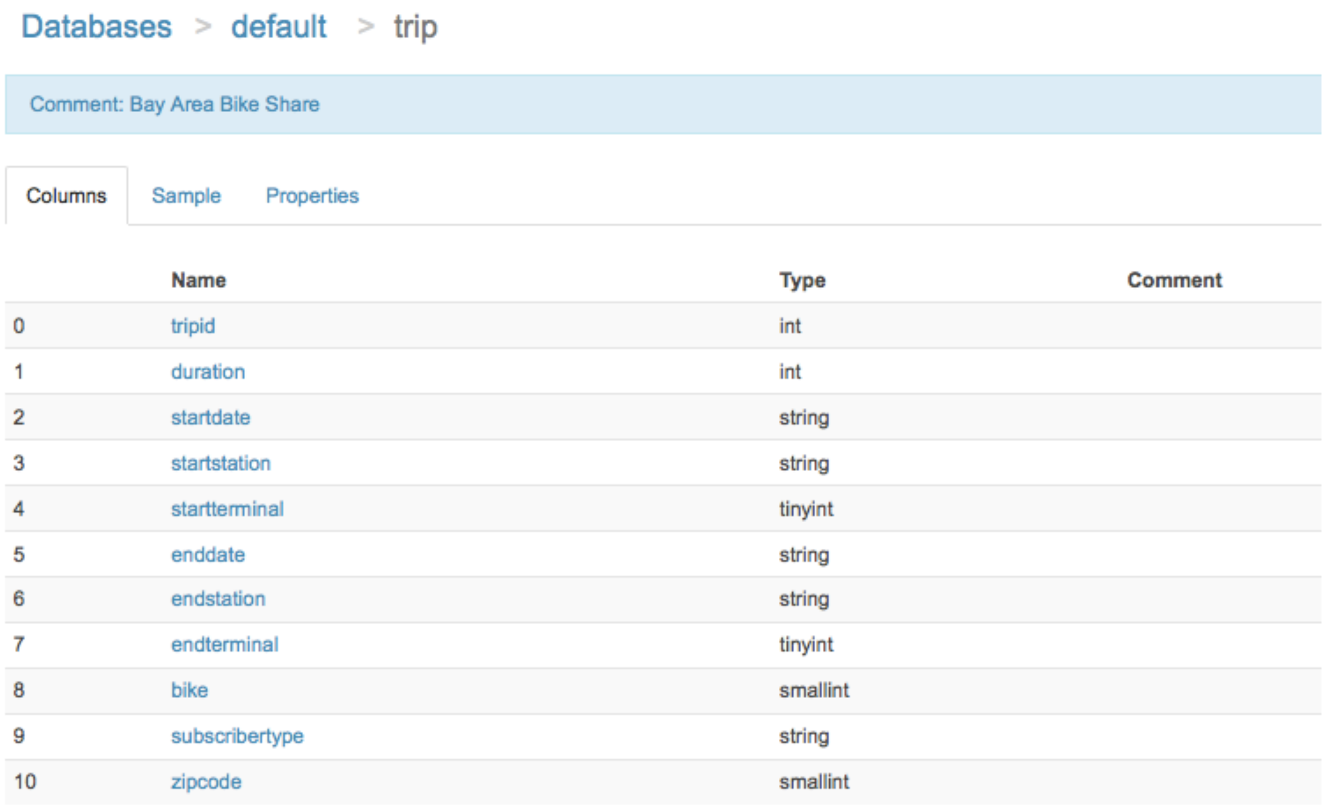
Step 1: Choose File　Step 2: Choose Delimiter　**Step 3: Define Columns**

Define your columns

Use first row as column names　Bulk edit column names

Column name	Column Type	*Sample Row #1*	*Sample Row #2*
TripID	int	432946	432945
Duration	int	406	468
StartDate	string	8/31/2014 22:31	8/31/2014 22:07
StartStation	string	Mountain View Caltrain Station	Beale at Market
StartTerminal	tinyint	28	56
EndDate	string	8/31/2014 22:38	8/31/2014 22:15

Create Table : Done

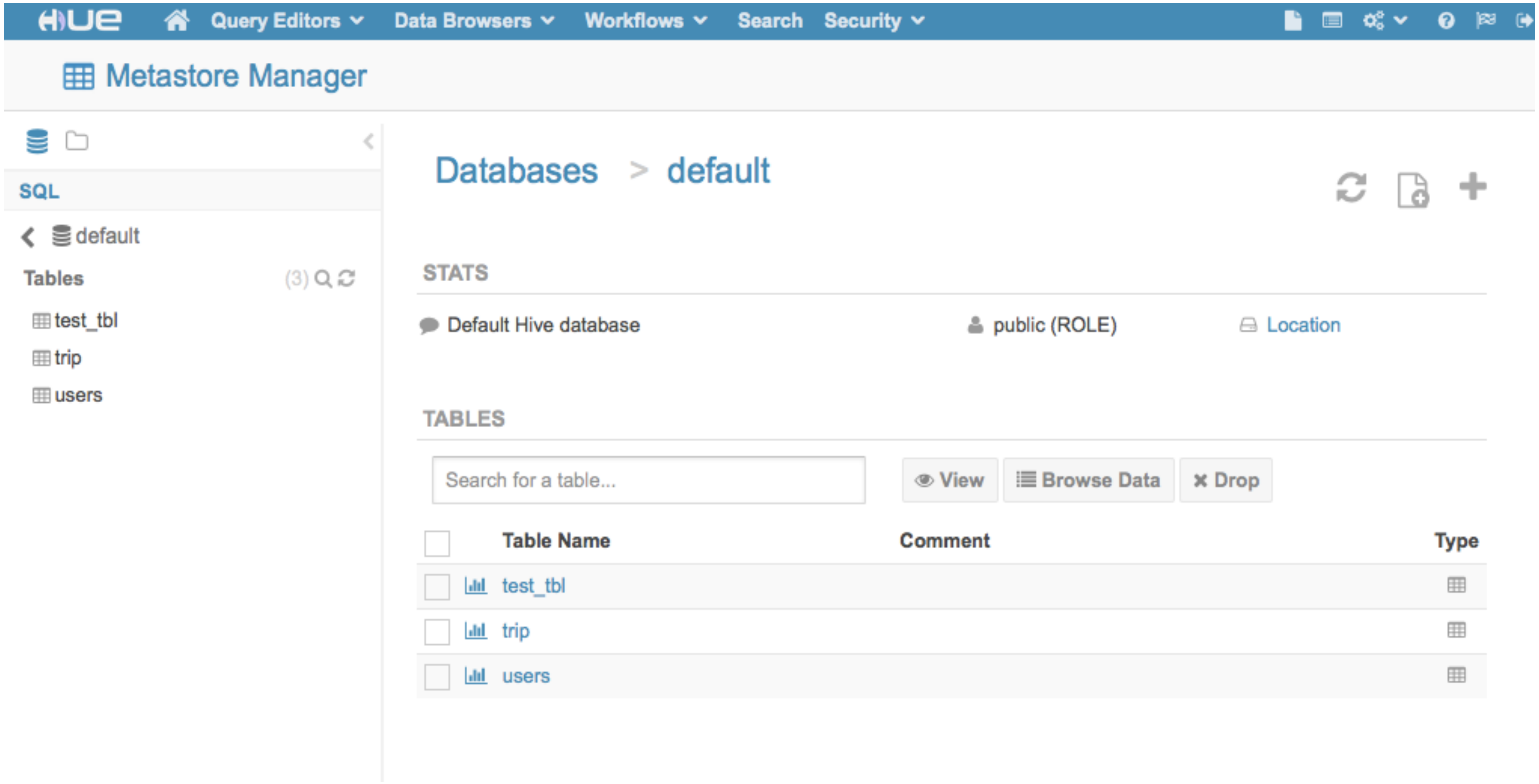


Databases > default > trip

Comment: Bay Area Bike Share

Columns Sample Properties

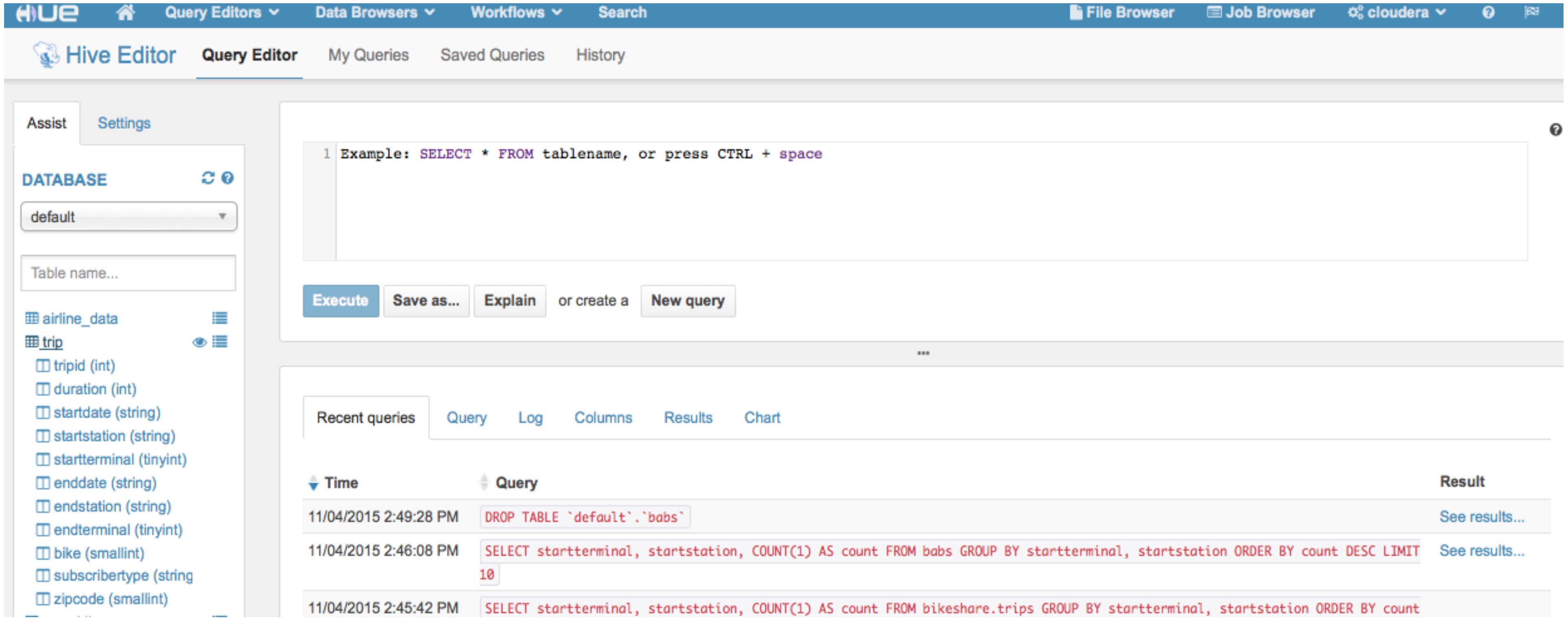
	Name	Type	Comment
0	tripid	int	
1	duration	int	
2	startdate	string	
3	startstation	string	
4	startterminal	tinyint	
5	enddate	string	
6	endstation	string	
7	endterminal	tinyint	
8	bike	smallint	
9	subscribertype	string	
10	zipcode	smallint	



Default Hive database | public (ROLE) | Location

Table Name	Comment	Type
test_tbl		▦
trip		▦
users		▦

Starting Hive Editor



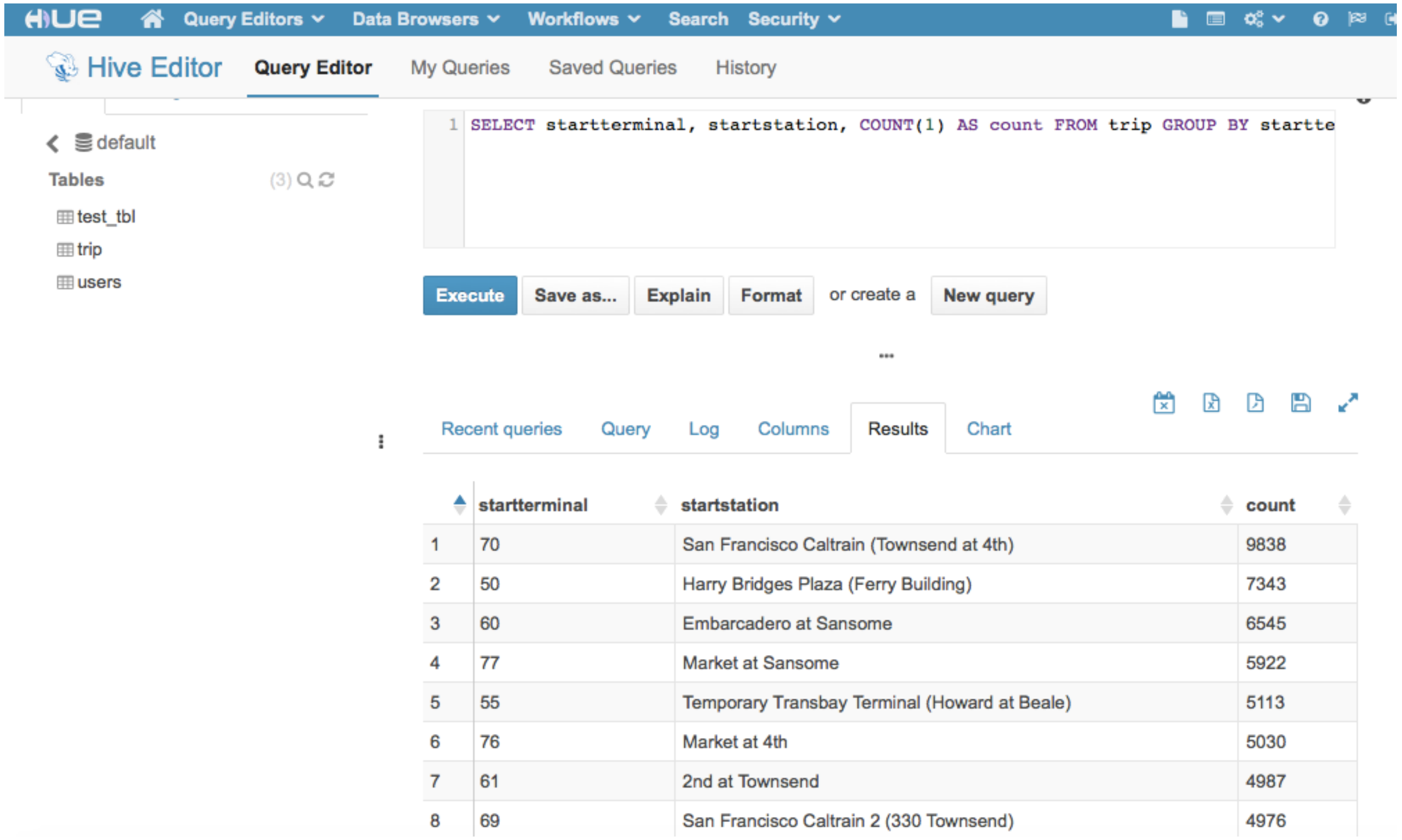
airline_data
trip
tripid (int)
duration (int)
startdate (string)
startstation (string)
startterminal (tinyint)
enddate (string)
endstation (string)
endterminal (tinyint)
bike (smallint)
subscribertype (string
zipcode (smallint)

```
Example: SELECT * FROM tablename, or press CTRL + space
```

Time	Query	Result
11/04/2015 2:49:28 PM	`DROP TABLE `default`.`babs``	See results...
11/04/2015 2:46:08 PM	`SELECT startterminal, startstation, COUNT(1) AS count FROM babs GROUP BY startterminal, startstation ORDER BY count DESC LIMIT 10`	See results...
11/04/2015 2:45:42 PM	`SELECT startterminal, startstation, COUNT(1) AS count FROM bikeshare.trips GROUP BY startterminal, startstation ORDER BY count`	

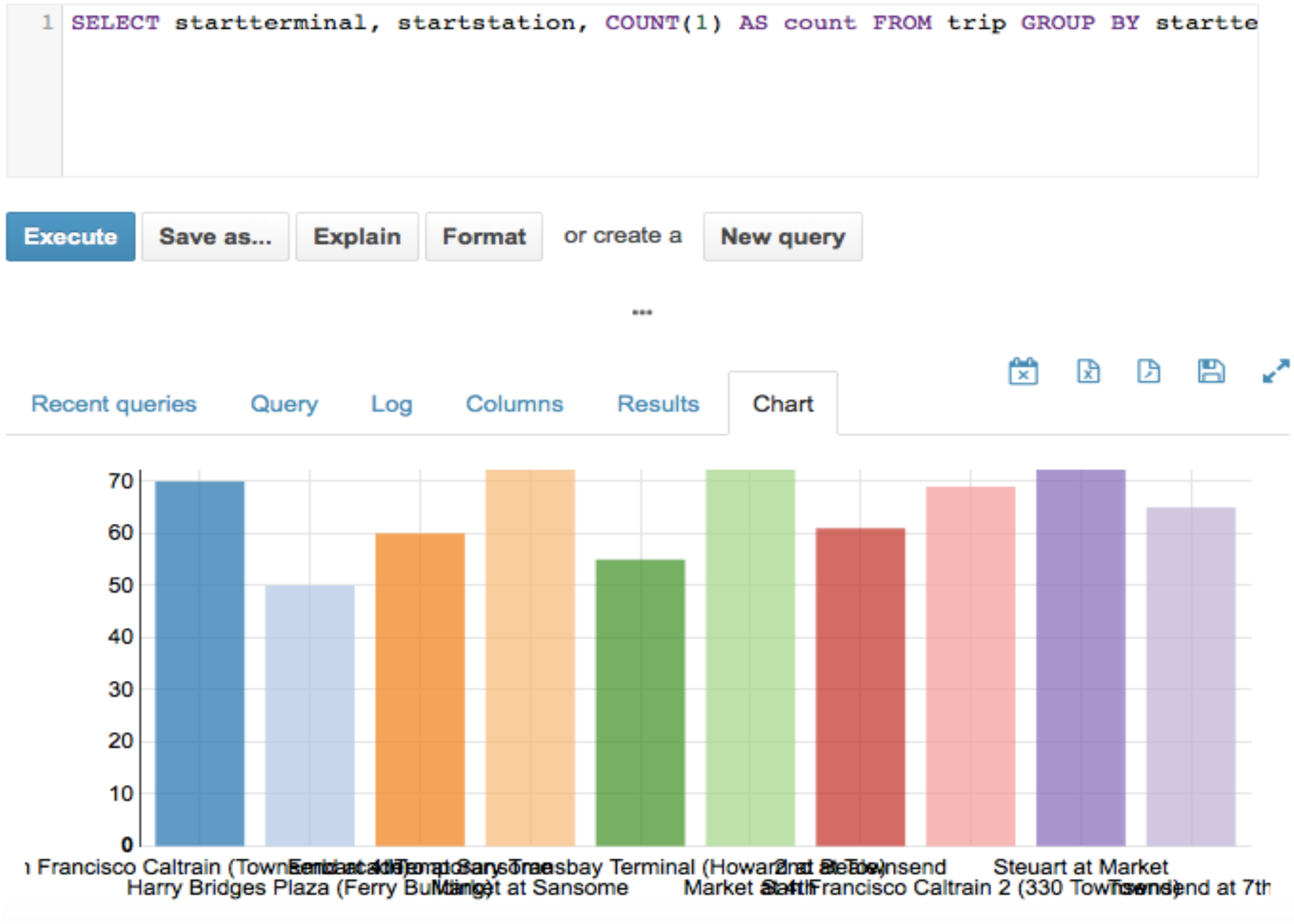
Find the top 10 most popular start stations based on the trip data

```
SELECT startterminal, startstation, COUNT(1) AS count FROM trip
GROUP BY startterminal, startstation ORDER BY count DESC LIMIT 10
```



```
SELECT startterminal, startstation, COUNT(1) AS count FROM trip GROUP BY startte
```

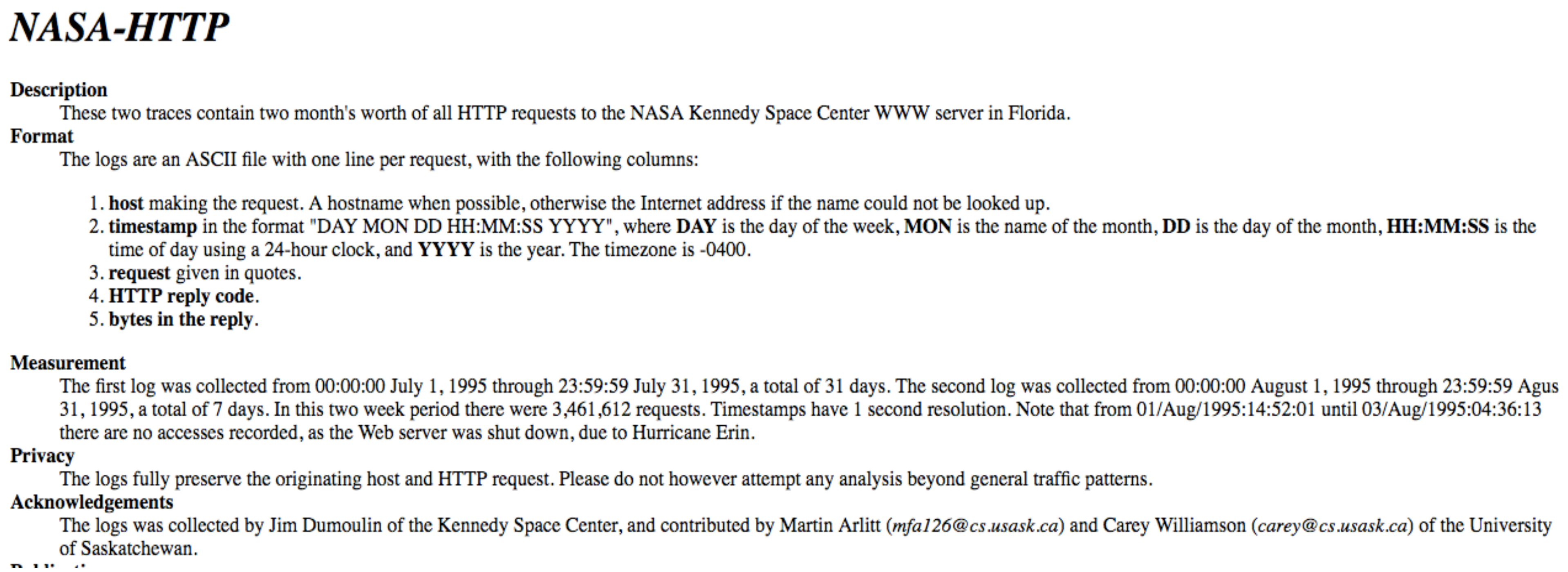
	startterminal	startstation	count
1	70	San Francisco Caltrain (Townsend at 4th)	9838
2	50	Harry Bridges Plaza (Ferry Building)	7343
3	60	Embarcadero at Sansome	6545
4	77	Market at Sansome	5922
5	55	Temporary Transbay Terminal (Howard at Beale)	5113
6	76	Market at 4th	5030
7	61	2nd at Townsend	4987
8	69	San Francisco Caltrain 2 (330 Townsend)	4976



```
SELECT startterminal, startstation, COUNT(1) AS count FROM trip GROUP BY startte
```

NASA WebLog

http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html



NASA-HTTP

Description

These two traces contain two month's worth of all HTTP requests to the NASA Kennedy Space Center WWW server in Florida.

Format

The logs are an ASCII file with one line per request, with the following columns:

1. **host** making the request. A hostname when possible, otherwise the Internet address if the name could not be looked up.
2. **timestamp** in the format "DAY MON DD HH:MM:SS YYYY", where **DAY** is the day of the week, **MON** is the name of the month, **DD** is the day of the month, **HH:MM:SS** is the time of day using a 24-hour clock, and **YYYY** is the year. The timezone is -0400.
3. **request** given in quotes.
4. **HTTP reply code**.
5. **bytes in the reply**.

Measurement

The first log was collected from 00:00:00 July 1, 1995 through 23:59:59 July 31, 1995, a total of 31 days. The second log was collected from 00:00:00 August 1, 1995 through 23:59:59 Agus 31, 1995, a total of 7 days. In this two week period there were 3,461,612 requests. Timestamps have 1 second resolution. Note that from 01/Aug/1995:14:52:01 until 03/Aug/1995:04:36:13 there are no accesses recorded, as the Web server was shut down, due to Hurricane Erin.

Privacy

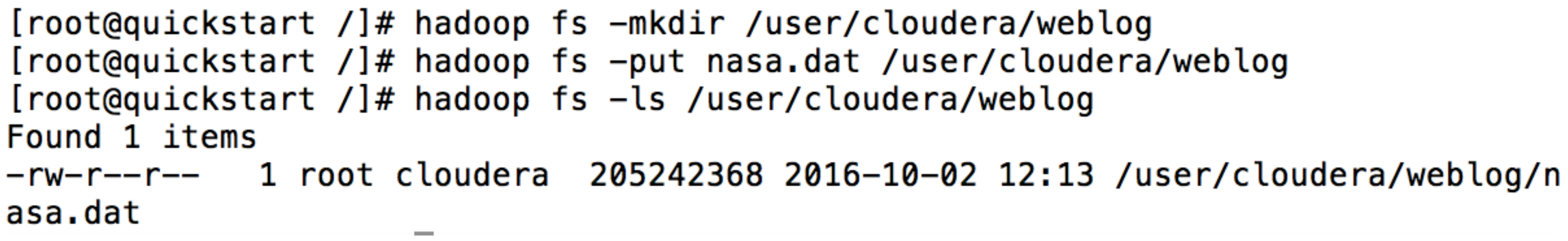
The logs fully preserve the originating host and HTTP request. Please do not however attempt any analysis beyond general traffic patterns.

Acknowledgements

The logs was collected by Jim Dumoulin of the Kennedy Space Center, and contributed by Martin Arlitt (*mfa126@cs.usask.ca*) and Carey Williamson (*carey@cs.usask.ca*) of the University of Saskatchewan.

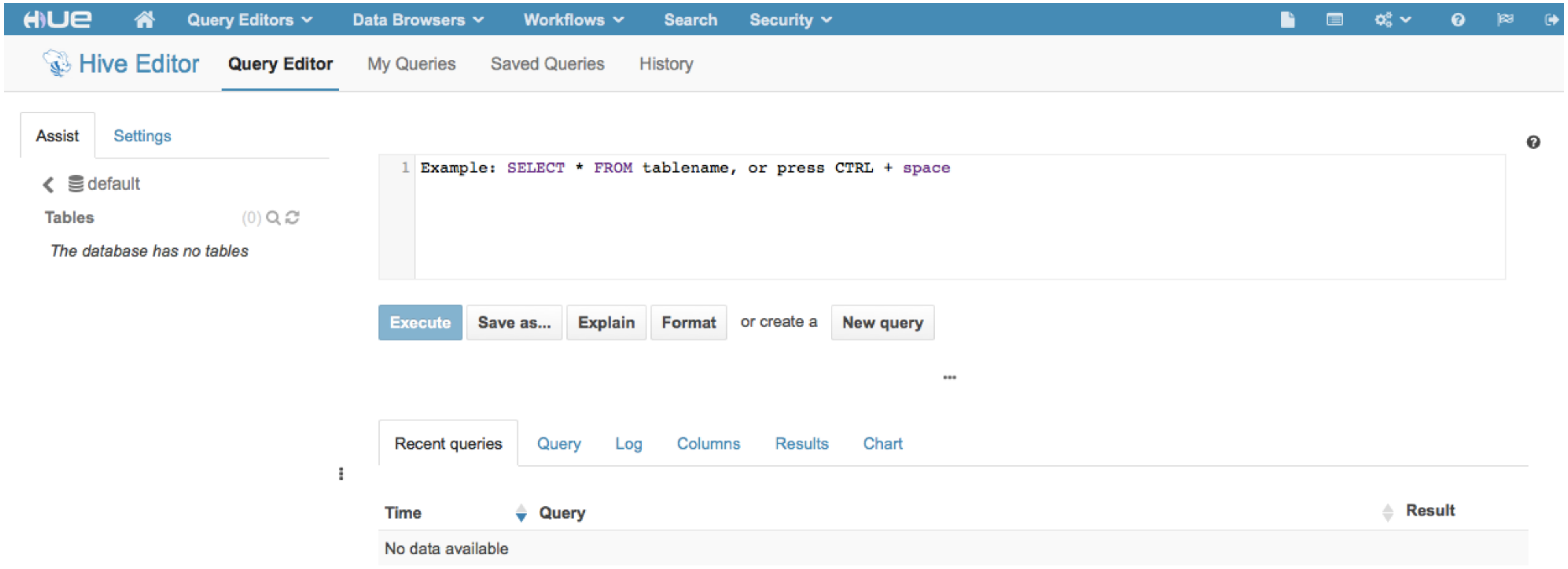
Preparing a NASA weblog

```
$wget https://s3.amazonaws.com/imcbucket/data/nasa.dat
$hadoop fs -mkdir /user/cloudera/weblog
$hadoop fs -put nasa.dat /user/cloudera/weblog
$hadoop fs -ls /user/cloudera/weblog
```



```
[root@quickstart /]# hadoop fs -mkdir /user/cloudera/weblog
[root@quickstart /]# hadoop fs -put nasa.dat /user/cloudera/weblog
[root@quickstart /]# hadoop fs -ls /user/cloudera/weblog
Found 1 items
-rw-r--r--   1 root cloudera  205242368 2016-10-02 12:13 /user/cloudera/weblog/n
asa.dat
```

Select: Query Editor >> Hive



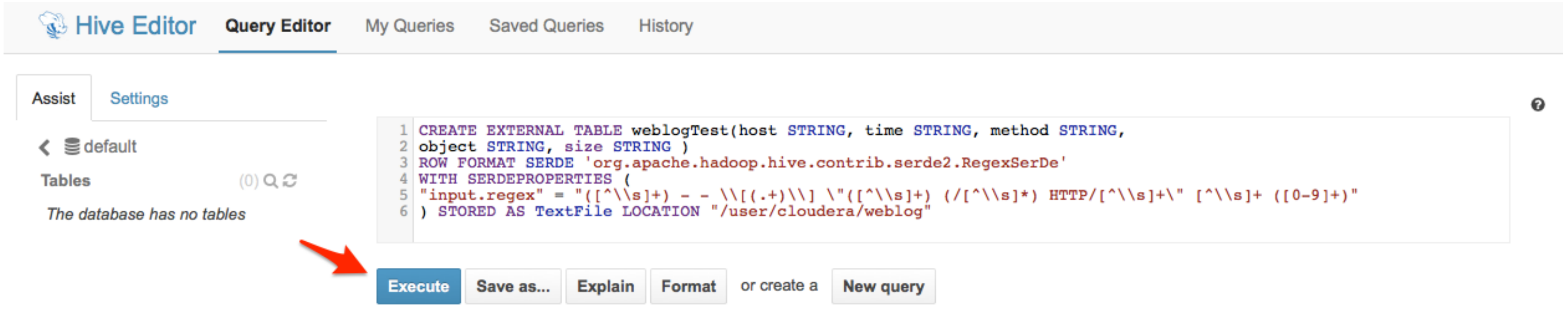
The database has no tables

```
Example: SELECT * FROM tablename, or press CTRL + space
```

Time	Query	Result
No data available		

Create an external table

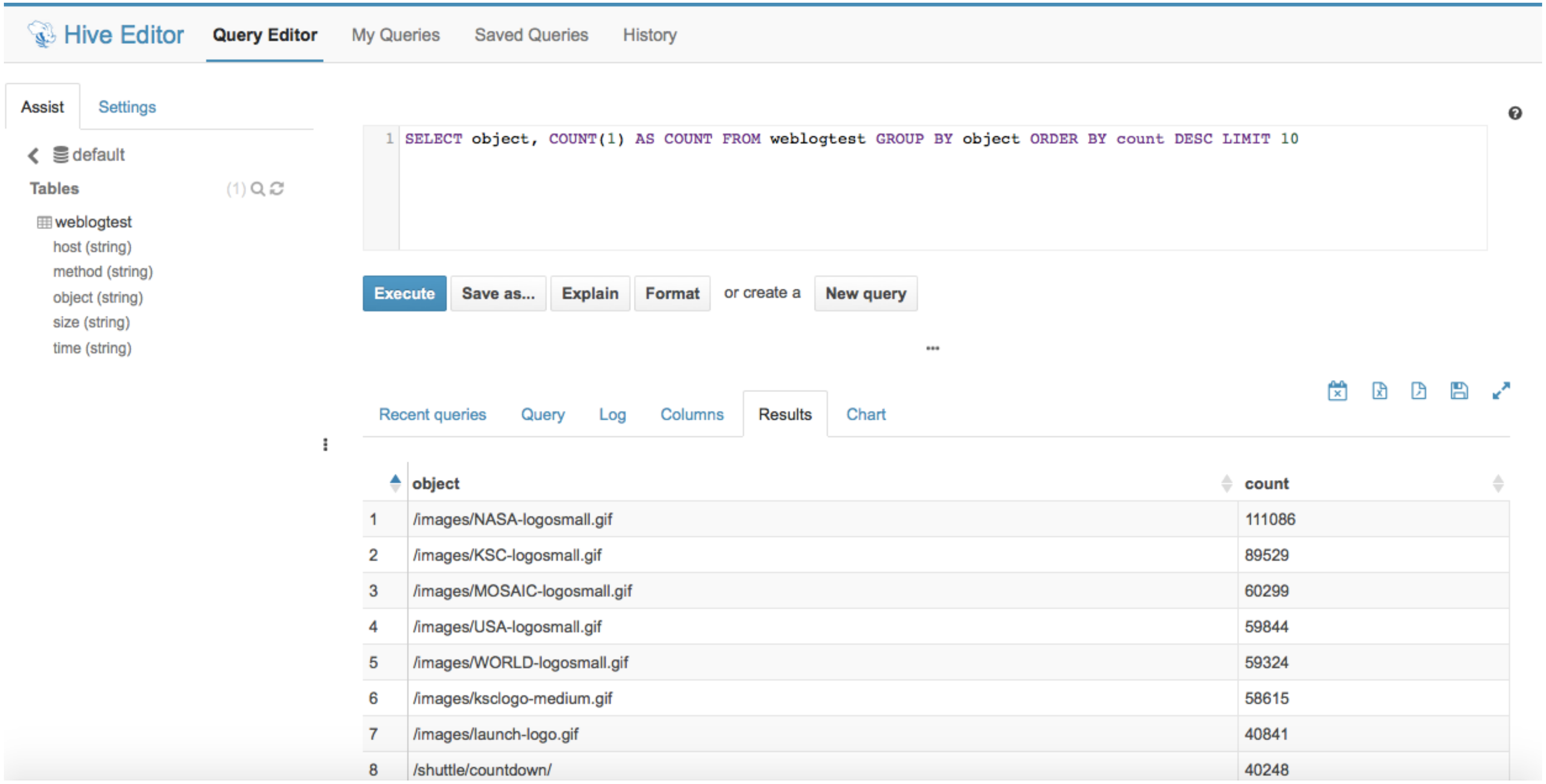
```
CREATE EXTERNAL TABLE weblogTest(host STRING, time STRING, method STRING,
object STRING, size STRING )
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
"input.regex" = "([^\\s]+) - - \\[(.+)\\] \"([^\\s]+) (/[^\\s]*)
HTTP/[^\\s]+\" [^\\s]+ ([0-9]+)"
) STORED AS TextFile LOCATION "/user/cloudera/weblog"
```



```
CREATE EXTERNAL TABLE weblogTest(host STRING, time STRING, method STRING,
object STRING, size STRING )
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
"input.regex" = "([^\\s]+) - - \\[(.+)\\] \"([^\\s]+) (/[^\\s]*) HTTP/[^\\s]+\" [^\\s]+ ([0-9]+)"
) STORED AS TextFile LOCATION "/user/cloudera/weblog"
```

SELECT Top 10 url

```
SELECT object, COUNT(1) AS COUNT FROM weblogtest GROUP BY object ORDER BY
count DESC LIMIT 10
```



```
SELECT object, COUNT(1) AS COUNT FROM weblogtest GROUP BY object ORDER BY count DESC LIMIT 10
```

	object	count
1	/images/NASA-logosmall.gif	111086
2	/images/KSC-logosmall.gif	89529
3	/images/MOSAIC-logosmall.gif	60299
4	/images/USA-logosmall.gif	59844
5	/images/WORLD-logosmall.gif	59324
6	/images/ksclogo-medium.gif	58615
7	/images/launch-logo.gif	40841
8	/shuttle/countdown/	40248