Impala Screencast Exercise Guide

Jesse Anderson

Contents

1 Impala Queries **3**

2 Advanced Impala Queries **6**

3 Impala UDFs **9**

1 Impala Queries

1.1 Objective

This 30 minute lab introduces you to querying data with Impala. We will:

- Create a table in Impala
- Add data to the table
- Run queries on the dataset

This exercise does not have an Eclipse project file. The sample solutions for this exercise are in the `impalaqueries` directory.

1.2 Police Dataset

The police call dataset is located at:
`/home/vmuser/training/datasets/rpd_data_all.csv.zip`

If you haven't already, decompress the file.

Here is the header for the dataset:

```
Priority,Call_Type,Jurisdiction,Dispatch_Area,Received_Date,
Received_Time,Dispatch_Time,Arrival_Time,Cleared_Time,Disposition
```

Here is a sample line from the dataset:

```
3,SUSPV,RP,RS, 03/21/2013,173011,182946,182946,183107,OK
```

Note: The `Received_Date` always has a space between the comma and the date. The result from Impala will always have a space before the date.

1.3 Table Creation

1. Start the Hive Metastore service.

   ```
   $ sudo service hive-metastore start
   ```

2. Start the Impala State Store.

   ```
   $ sudo service impala-state-store start
   ```

3. Start the Impala Catalog service.

   ```
   $ sudo service impala-catalog start
   ```

4. Start the Impala daemon.

   ```
   $ sudo service impala-server start
   ```

5. Create a table in Impala for the police dataset. Make sure that the columns' data types match the dataset.

6. Load the dataset into the table.

7. Do a `SELECT` of one row to verify that the table definition is correct.

   If the data is incorrect, you must drop the table. To do this, run the query:

   ```
   DROP TABLE tablename;
   ```

8. Describe the layout of the table.

1.4 Querying the Data

1. Select all of the calls with an emergency priority of `E`.

2. Select all of the calls with a priority of 1 on 12/31/2013. Remember that the date field has a space in it and the date in the `WHERE` should have a space.

3. Select the first 10 calls for 03/20/2013. Remember the space in the date field.

1.5 Advanced Optional Steps

1. The `Received_Date` always has a space between the comma and the date like so:

   ```
   , 03/21/2013
   ```

   Figure out a way to automatically remove the space.

2. The `Received_Date`'s format of `MM/DD/YYYY` does not conform to Impala's `TIMESTAMP` type format of `YYYY/MM/DD`. Fix the timestamp's format to be able to use Impala's `TIMESTAMP` type.

2 Advanced Impala Queries

2.1 Objective

This 45 minute lab uses Impala to make advanced queries of the data. We will:

- Join data with Impala
- Create a view
- Create a table with a `SELECT` statement

This exercise does not have an Eclipse project file. The sample solutions for this exercise are in the `impalaadvancedqueries` directory.

2.2 Jurisdiction Dataset

The jurisdiction dataset is located at:
`/home/vmuser/training/datasets/jurisdictions.csv`

Here is the header for the dataset:

```
Jurisdiction,Description
```

Here is a sample line from the dataset:

```
RP,Reno Police Department
```

The police call dataset has a jurisdiction key and the jurisdiction dataset matches that key.

2.3 Jurisdiction Table

1. If it isn't started already, start the Hive Metastore and Impala services.

```
$ sudo service hive-metastore start
$ sudo service impala-state-store start
$ sudo service impala-catalog start
$ sudo service impala-server start
```

2. Create a table in Impala for the jurisdiction table and load the data into it.

3. Do a `SELECT` of one row to verify that the table definition is correct.

2.4 Impala Joins

1. Write a query that does an explicit join on the `policecall` and `jurisdiction` tables for the first 10 rows.

2. Write a query that does an implicit join on the `policecall` and `jurisdiction` tables for the first 10 rows.

3. Not all of the police calls have a matching jurisdiction. Write a query that shows the first 10 police calls without a jurisdiction.

2.5 Impala Functions

1. Count the total number of police calls.

2. Count the total number of police calls in 2013.

3. Count the number of police calls without a matching jurisdiction.

4. Count the total number of police calls by broken down by priority.

5. Count the total number of police calls by broken down by priority and order the results by the largest total to the smallest.

2.6 Advanced Optional Steps

1. Write a query to output the percentage of police calls that lack a jurisdiction.

2. Write a query to breakdown the police calls by year.

3 Impala UDFs

3.1 Objective

This 30 minute lab compiles and uses a native Impala UDF. We will:

- Clone a sample Impala UDF repository
- Compile the UDF
- Use the UDF in a query

There is no project directory for this exercise.

3.2 Compiling the UDF

The development environment needs to be prepared for the exercise. The UDF repository needs to be cloned.

Note: Use the package manager for your operating system for the installation commands. On RPM-based systems, it is `sudo yum install packagename`. On DEB-based systems, it is `sudo apt-get install packagename`.

1. Install the Impala UDF development library if it isn't installed already. The package name is `impala-udf-devel` on RPM based systems and `impala-udf-dev` on DEB based systems.

   ```
   $ sudo apt-get install impala-udf-devel
   ```

2. Install cmake if it isn't installed already.

   ```
   $ sudo apt-get install cmake
   ```

3. Clone the repository.

   ```
   $ git clone https://github.com/cloudera/impala-udf-samples.git
   ```

4. Change directories to the newly downloaded repository.

```
$ cd impala-udf-samples
```

5. Run the `cmake` command.

```
$ cmake .
```

6. Run the `make` command.

```
$ make
```

7. Create a directory in HDFS for UDFs.

```
$ hadoop fs -mkdir /user/hive/udfs/
```

8. Copy the shared object to HDFS.

```
$ hadoop fs -put build/libudfsample.so /user/hive/udfs/libudfsample.so
```

3.3 Using the UDF

The UDF is in HDFS, but the functions are not accessible yet. You need to create the functions.

1. Start the `impala-shell`.

```
$ impala-shell
```

2. Create the functions in the Impala shell.

```
> CREATE FUNCTION has_vowels (string) returns boolean LOCATION
'/user/hive/udfs/libudfsample.so' symbol='HasVowels';
> CREATE FUNCTION count_vowels (string) returns int LOCATION
'/user/hive/udfs/libudfsample.so' symbol='CountVowels';
> CREATE FUNCTION strip_vowels (string) returns string LOCATION
'/user/hive/udfs/libudfsample.so' symbol='StripVowels';
```

This will create three functions: `has_vowels`, `count_vowels`, and `strip_vowels`.

1. Write a SQL query to check if some text has vowels in it with the `has_vowels` function.
2. Write a SQL query that counts the number of vowels with the `count_vowels` function.
3. Write a SQL query that strips the vowels with the `strip_vowels` function.

3.4 Advanced Optional Steps

- Modify the UDF to add in the equivalents for consonants.