|  |  |
| --- | --- |
| Big Data Analysis Exercises | Abstract  This document outlines the exercises to be undertaken during the Analysis week of the Big Data course  QA Consulting Academy Team  Big Data Academy |

Contents

[Introduction 2](#_Toc447611589)

[Practice exercises 2](#_Toc447611590)

[Analysing your data 2](#_Toc447611591)

[Citations 2](#_Toc447611592)

[Exercise 1: Running Hive Queries from the Shell and Scripts for Data Transformation 3](#_Toc447611593)

[Creating Tables 3](#_Toc447611594)

[Running a HiveQL Script 3](#_Toc447611595)

[Exercise 2: Data Management with Hive 4](#_Toc447611596)

[Use Sqoop’s Hive Import Option to Create a Table 4](#_Toc447611597)

[Create and Load a Hive-Managed Table 4](#_Toc447611598)

[Create, Load, and Query a Table with Complex Fields 5](#_Toc447611599)

[Altering and Deleting Tables 5](#_Toc447611600)

[Exercise 3: Gaining Insight with Sentiment Analysis 6](#_Toc447611601)

[Analyse Tags 6](#_Toc447611602)

[Exercise 4: Analysis with Impala 7](#_Toc447611603)

[Start the Impala Shell and Refresh the Cache 7](#_Toc447611604)

[Impala Queries 7](#_Toc447611605)

[Challenges 8](#_Toc447611606)

# Introduction

This document will take you through some exercises to analyse data. You have already set up your cluster, added some data, and cleaned that data. It should now be ready for you to perform some analysis on it.

## Practice exercises

The document goes through some practice exercises on the movielens database that should already be on your cluster. You will need to arrange this so that each .csv file is in its own folder in HDFS so that Hive can understand it as a table. This should get you comfortable performing some of these commands and how to get started. After this, you should be working on your project data.

## Analysing your data

For this course you have a large amount of data of reviews from Amazon products between 1996 and 2014. It is up to you to investigate this data what analysis you will perform, what business changes can you suggest to Amazon from this data, and what can you tell about their user base.

## Citations

This data is from the following two papers, and if you have time, it would be worth giving them a read.

**Image-based recommendations on styles and substitutes**  
J. McAuley, C. Targett, J. Shi, A. van den Hengel  
*SIGIR*, 2015

http://cseweb.ucsd.edu/~jmcauley/pdfs/sigir15.pdf

**Inferring networks of substitutable and complementary products**  
J. McAuley, R. Pandey, J. Leskovec  
*Knowledge Discovery and Data Mining*, 2015

http://cseweb.ucsd.edu/~jmcauley/pdfs/kdd15.pdf

# Exercise 1: Running Hive Queries from the Shell and Scripts for Data Transformation

In this exercise you will write HiveQL queries to analyse data in Hive tables that have been populated with data in HDFS.

## Creating Tables

Before we get started, let’s create some tables in Hive to use for analysis. We will be creating external tables, which is essentially just a schema that will point at a folder in HDFS. You can either use the beeline command ‘beeline –u jdbc:hive2://<hive\_server\_address>:10000 –n hive’, or just ‘hive’ to use the old hive command line interface.

1. Use the command below to create an external table for the movies.csv file. Remember to change your location to the folder with your movies.csv file in. The final line will skip the header rather than including it as a value.  
     
   > CREATE EXTERNAL TABLE movies  
   >(movieid STRING, name STRING, genres STRING)  
   >ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’  
   >LOCATION ‘/moviedata/movies/’  
   >tblproperties(“skip.header.line.count”=”1”);
2. Create external tables for the tags.csv file and ratings.csv file similar to the above.
3. It’s worth noting at this point that we can run hive commands directly from the command line without ‘signing in’ to beeline or the hive CLI. Try the command below in a new terminal.  
     
   $ hive –e ‘SELECT \* FROM movies LIMIT 5’

## Running a HiveQL Script

You may need to change the permissions of the /user/hive directory and those below it before running this script. You should have a file called best\_year.hql. Study it to understand how it works then run it.

1. Execute the HiveQL script using the hive command’s –f option, which essentially runs a file of Hive statements.  
     
   $ hive –f best\_year.hql
2. Try to extend this script further – is it fair that we judge a year’s movies by the average of them all, is it unfair that some movies have been around longer and therefore could have a higher or lower average because more people have had the opportunity to watch it?

It’s also possible to run Hive queries through the Hue interface.

# Exercise 2: Data Management with Hive

In previous exercises you have created external tables where the data is kept in HDFS and we point Hive towards it. Below are some other ways you can populate Hive tables, try practicing with smaller amounts of the movielens data.

## Use Sqoop’s Hive Import Option to Create a Table

You have previously used Sqoop to import data from MySQL into HDFS. Sqoop can also create a Hive table with the same fields as a source table and import the records into Hive so you don’t have to use the CREATE TABLE statement as well.

1. Execute the following command to import the movies table from MySQL as a new Hive-managed table.  
     
   $ sqoop import --connect jdbc:mysql://orange:10000/moviedata --fields-terminated-by ‘,’ --table movies --hive-import
2. It is always a good idea to validate data after adding it. Once you have imported the data, use your preferred method to query the data (e.g. using Beeline) and ensure it has imported as expected.

## Create and Load a Hive-Managed Table

There are two ways to load data into a hive-managed table – from data stored on the local machine, and from data in HDFS.

1. Before you do anything you will need to create a table in Hive using the CREATE TABLE command followed by the schema of what the data will be when you load it and what each field is delimited by. This is the same as creating an external table except we won’t give a location and we won’t use the EXTERNAL keyword.
2. When you have created a table it can be a good idea to use the DESCRIBE command to check everything is as you expect.
3. To load from your local drive it’s as straightforward as putting data into Hadoop. Use the ‘put’ command and point it towards a directory within hive’s HDFS space.  
     
   $ hadoop fs –put user/data/ratings.txt /user/hive/warehouse/ratings
4. As per usual you should verify that Hive can read the data you’ve uploaded by running a quick command.
5. To load data into a Hive table from data in HDFS you use the LOAD DATA INPATH command. This actually also has an option to load data from a local path as well, simply use the LOCAL keyword as LOAD DATA LOCAL INPATH.  
     
   hive> LOAD DATA INPATH /movielens/movies.csv INTO TABLE movies;
6. The LOAD DATA command actually moves the data, so if you perform this command check that it has been removed from the original HDFS directory. Then, as usual, verify that you can query it as expected in Hive.

## Create, Load, and Query a Table with Complex Fields

The movielens data does not have any complex fields within to query, however there is an example below to demonstrate how to create a table with complex fields. Imagine we have some customer data including their phone numbers (as a map like Home#0161829661:Mobile#0788299471), a list of past order IDs (as an array of integers), and a struct that summarises the minimum, maximum, average, and total value of past orders. Note that we need to specify delimiters for these complex types.

1. Study the command below to understand how to create a table with complex data types.  
     
   CREATE TABLE customers (cust\_id INT, fname STRING, lname STRING, email STRING, phone MAP<STRING, STRING>, order\_ids ARRAY<INT>, order\_value STRUCT<min:INT,max:INT,avg:INT,total:INT>) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘|’ COLLECTION ITEMS TERMINATED BY ‘,’ MAP KEYS TERMINATED BY ‘:’;

## Altering and Deleting Tables

1. The ALTER TABLE command can update column names and other table properties. Use a DESCRIBE command immediately after performing an ALTER to verify the change.
2. The DROP TABLE command is used for deleting tables.

# Exercise 3: Gaining Insight with Sentiment Analysis

Feedback and reviews are great source of information. However, customer comments are typically free-form text and must be handled differently from quantitative data. The movielens data contains user-given tags for the data that we will have a look at, and though these are small collections of information we can still use this for some analysis.

## Analyse Tags

1. The following query will change all tags to lower case, break them into individual words using SENTENCES, passes those to the NGRAMS function which then finds the five most common bigrams (two-word combinations) across all movies. Run it in Hive.  
     
   SELECT EXPLODE(NGRAMS(SENTENCES(LOWER(tag)),2,5)) AS bigrams  
   FROM tags;
2. Try performing this again but change the ‘2’ in the above query to 3, and then 4. These will return the most common 3- and 4-word combinations which can also provide useful insight.
3. Try extending this query further. Look for common combinations of words in the very best and very worst rated movies.

# Exercise 4: Analysis with Impala

In this exercise, you will try running analysis on the movielens data with Impala.

## Start the Impala Shell and Refresh the Cache

Remember that because Hive and Impala share the same cache of data in the metastore, we need to update Impala with the changes from the previous exercises.

1. Start the Impala shell.  
     
   $ impala-shell
2. Refresh Impala’s metastore.  
     
   INVALIDATE METADATA;

## Impala Queries

You have been given a Hive script that performs movie recommendations based on past ratings from users. This script is currently limited to 10 users and 100 movies as it takes a long while to run. Study this script to understand what it does and how, run it while you are studying it to see what it produces (but be wary of crossover in your groups, ensure only one person is running it).

You are already aware that Impala is much faster than Hive, but it is not the best tool for batch processing. Using Impala, investigate the recommended movies that have been created from the Hive script.

# Challenges

These exercises have introduced you to the SQL-like tools Hive and Impala. For this part of the course you need to produce at least two .hql scripts for analysing your project data. One should be created specifically to run in Hive, and the other with Impala. This could include performing some kind of sentiment analysis on the review text (hint: consider looking for the most common word combinations using something like EXPLODE(NGRAMS(SENTENCES(LOWER(reviewText)),2,5))).

In fact, you could explore many aspects of the data you have been provided, such as customer inclination towards certain products. Consider not just the ‘frequently bought together’ items, but try and build a more robust product recommendation system based on people’s ratings. You could see if the review text has any effect on the review’s helpfulness rating, or if a user’s helpfulness rating is skewed based on whether they generally give high or low ratings.

Whatever you decide to look into, ensure you have at least one Hive and one Impala script that perform different tasks.

This investigation is to give Amazon an understanding of their customer base, and make suggestions on changes for improvements based on facts from the data. Consider this carefully when performing your analysis.

Whenever you are creating a new script you should first run it locally on a small sample, so that you are not wasting time and using up cluster resources each time you need to test it. This way, when you run your script on the full cluster data, while it may take a while, it should work correctly and return what you expected, saving time and resources.