QA Consulting

Big Data Team Exercise.

TRAINING MATERIALS - EXERCISE BOOK

Contacts

Thomas.Knowles@qa.com

team.qac.all.trainers@qa.com

www.consulting.qa.com

Contents

Introduction

The document introduces the data analysis team exercise that you are going to complete during the next few days. These exercises use the movielens database that you worked with on the administration week. This data can be accessed at the following URL: http://grouplens.org/system/files/ml-100k.zip.

Purpose

The purpose of this exercise is to provide big data consultants with the opportunity to:

- Produce a team solution for several big data problems using:
  - Apache Hive
  - Impala
  - Hadoop
- Deliver a big data solution to a suitable version control system
- Research new technologies as a team
- Produce a presentation that outlines at least the following:
  - How the team approached the task
  - Differences between the key technologies
  - A recommendation on a technology the team would use in the future any why
  - A list of problems you encountered and how you would use this information in the future
  - Any other interesting topics
- A working demo of the big data solutions that will be shown to the class
  - Each team member needs to introduce the big data problem, the solution they wrote for this problem and show the solution running against your dataset.

Exercise 1

In this exercise you will write HiveQL queries to analyse data in an external hive table that derives data from HDFS.

CREATING A DATABASE

Create a new hive DB to store your work using an original team name.

CREATING TABLES

1. Create a HDFS directory for userinfo and put the u.user data into this directory
2. Create a user's table with the appropriate fields derived from u.user. The team will need to analyse the data set to identify these fields. Use the appropriate row format delimit type for this file.
   - In this instance use an external table and not a hive managed table.
3. Try running a simple SELECT query over this data to ensure it works.
4. Identify home many students are in the user table.
5. Devise several other analysis queries for each member in the team.
   - The queries will be explained in your final demonstration.

VERSION CONTROL

- Push your solution to a suitable version control repository
- Send a request to your trainer via a suitable mechanism so your work can be checked.

Exercise 2

In the previous exercise you have created an external table where the data is kept in HDFS and we point Hive towards it. Below is an alternative approach you will investigate.

CREATE AND LOAD A HIVE-MANAGED TABLE

In this task you will be using the movie data stored in the u.item file and loading this into a table.

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/u.item /user/hive/warehouse/movies
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. Use this command to add the data into the table.
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.
7. Devise several analysis queries for each member in the groups.
   - The queries will be explained in your final demonstration.

VERSION CONTROL

- Push your solution to a suitable version control repository
- Send a request to your trainer via a suitable mechanism so your work can be checked.

Exercise 3

In this exercise the team will investigate creating tables with complex fields using the following steps:

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.

- 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 4

In this exercise, you will try running analysis on the movielens data with Impala. This will involve the team doing independent research on Impala and completing exercises 1, 2 and 3 using this alterative technology set.

Exercise 5

Produce a PowerPoint presentation that highlights the group's findings and addresses the points highlighted in the introduction section.

Additional exercise 1

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. In this exercise the team will investigate how to use NGRAMS along with some other functions.

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.
4. Add your findings to your final presentation.

Additional exercise 2

The previous exercises have introduced you to the SQL-like tools Hive and Impala. For this final exercise you will need to produce at least two .hql scripts for analysing some other part of the movie lens data or any other data set you choose. One script should be created specifically to run in Hive, and the other with Impala. This could include additional sentiment analysis on this data set.

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

Add your findings to your final presentation.