**Assignment 1**

Hadoop Data Analysis using Apache Hive / Impala

**Provided files:**

9 Bixi Data files (one file for Stations, 8 files for rides)

Download this files to your machine in order to perform the assignment

**1. Hive 75%**

Like Pig, Apache Hive makes it possible to write complex queries over big datasets, but it has the advantage of using HiveQL, a language that is very easy to learn for people who are used to work with databases since it is very similar to SQL.

While doing this exercise, you are encouraged to look up the online documentation https://cwiki.apache.org/confluence/display/Hive/LanguageManual

<https://www.cloudera.com/documentation/enterprise/latest/topics/impala_functions.html>

**Question 1**

Write a HiveQL script that create two non-managed tables, named **data** and **stations**, that contain the data from files OD\_2018\*.csv and Stations\_2018.csv, respectively.

use a1p2;

create table stations (

code string,

name string,

latitude double,

longitude double)

Row format delimited fields terminated by ',' stored as TextFile

TBLPROPERTIES ("skip.header.line.count"="1");load data inpath '/a1p2/BixiMontrealRentals2018/Stations\_2018.csv' into table stations;

create table data (

start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int,

is\_member tinyint)

Row format delimited fields terminated by ',' stored as TextFile location '/a1p2/BixiMontrealRentals2018/Rides'

TBLPROPERTIES ("skip.header.line.count"="1");

**Question 2:**

Write a HiveQL script to compute and print on the screen the number of rows for each table (data, stations).

use a1p2;

SELECT COUNT(\*) FROM data;

SELECT COUNT(\*) FROM stations;

**Question 3:**

Write a HiveQL script to split the data table into two tables **members** and **notmembers**. The members table should contains only peoples that are members and the notmembers table should contains all non-member people.

use a1p2;

create table members (

start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int,

is\_member tinyint)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;

create table notmembers (

start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int,

is\_member tinyint)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;

insert into members

select start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int,

is\_member tinyint from data where is\_member=1;

insert into notmembers

select start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int,

is\_member tinyint from data where is\_member=0;

**Question 4:**

Write a HiveQL script to compute and print on the screen the number of rows for each table (members, notmembers).

use a1p2;

SELECT COUNT(\*) FROM members;

SELECT COUNT(\*) FROM notmembers;

**Question 5:**

Write a HiveQL script that given some station’s code (hard-coded constant) will return the stations record if found and all the ride for this station.

use a1p2;

set hivevar:station\_code = 6100;

select \* from stations where code = '${hivevar:station\_code}';

select \* from data where start\_station\_code = '${hivevar:station\_code}' or end\_station\_code = '${hivevar:station\_code}';

**Question 6:**

Write a HiveQL script that will calculate the number of rides departures per station. The output has to be sorted descending.

use a1p2;

select start\_station\_code, count(\*) as cnt from data

group by start\_station\_code

order by cnt DESC;

**Question 7:**

Write a HiveQL script that will calculate the count of rides per station (start station) and the MIN, MAX, AVG of the ride’s duration for members and non-members.

use a1p2;

select start\_station\_code, count(duration\_sec), min(duration\_sec), max(duration\_sec), avg(duration\_sec) from members

group by start\_station\_code;

select start\_station\_code, count(duration\_sec), min(duration\_sec), max(duration\_sec), avg(duration\_sec) from notmembers

group by start\_station\_code;

**Question 8:**

Write a HiveQL script that will list the Top 5 names of the start station for members and for non-members

use a1p2;

select s.name, count(r.start\_station\_code) as cnt

from members r join stations s on (r.start\_station\_code = s.code)

group by r.start\_station\_code, s.name

order by cnt DESC

limit 5;

select s.name, count(r.start\_station\_code) as cnt

from notmembers r join stations s on (r.start\_station\_code = s.code)

group by r.start\_station\_code, s.name

order by cnt DESC

limit 5;

**Question 9:**

Write a HiveQL script that will list the longest and shortest ride done by a member and by a non-member.

use a1p2;

select a.\* from members a left semi join

(select cast(max(duration\_sec) as float) as max\_duration from members) b on (a.duration\_sec = b.max\_duration)

limit 1;

select a.\* from notmembers a left semi join

(select cast(max(duration\_sec) as float) as max\_duration from notmembers) b on (a.duration\_sec = b.max\_duration)

limit 1;

**Question 10:**

Write a HiveQL script that will:

* Create a new partitioned table by the (is\_member) column
* Populate this table with rows from the data table
* Dynamically create partition for inserted rows

use a1p2;

set hive.enforce.bucketing = true;

set hive.enforce.sorting=true;

set hive.exec.dynamic.partition = true;

set hive.exec.dynamic.partition.mode = nonstrict;

create table data\_part (

start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int)

partitioned by (is\_member tinyint)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;

insert into data\_part partition (is\_member)

select start\_date string, start\_station\_code string, end\_date string, end\_station\_code string, duration\_sec int, is\_member tinyint

from data;

**Question 11:**

Write a HiveQL script that will

* Create a new partitioned table by the (is\_member) column
* With **8** buckets clustered by (start\_station\_code) and sorted by (duration\_sec)
* Dynamically organize data into **8** buckets

use a1p2;

set hive.enforce.bucketing = true;

set hive.enforce.sorting=true;

set hive.exec.dynamic.partition = true;

set hive.exec.dynamic.partition.mode = nonstrict;

create table data\_part (

start\_date string,

start\_station\_code string,

end\_date string,

end\_station\_code string,

duration\_sec int)

partitioned by (is\_member tinyint) clustered by (start\_station\_code) SORTED BY (duration\_sec)

into 8 buckets

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;

insert into data\_part partition (is\_member)

select start\_date string, start\_station\_code string, end\_date string, end\_station\_code string, duration\_sec int, is\_member tinyint

from data;

**Question 12:**

Write a HiveQL script that will return rows count for all bucket **4** of the bucketed table you just created it in the previous question.

use a1p2;

select count(\*) from data\_part tablesample (bucket 4 out of 8);

**2. Impala 25%**

Like Apache Hive, Impala makes it possible to write complex queries over big datasets, but it has the advantage of using interactive HiveQL.

**Question 1:**

Write a HiveQL script that will calculate:

* The percentage of the top 5 start stations
* Dhe percentage of the top 5 end stations.
* Display the name of these stations with the persentage rather than the station code.

invalidate metadata;

use a1p2;

with total as (select count(\*) as total from `data`)

select r.start\_station\_code, s.name, count(r.start\_station\_code) as rides, count(r.start\_station\_code)/total.total\*100 as perc from `data` r, total

join stations s on (r.start\_station\_code = s.code)

group by r.start\_station\_code, total.total, s.name

order by rides DESC

limit 5;

with total as (select count(\*) as total from `data`)

select r.end\_station\_code, s.name, count(r.end\_station\_code) as rides, count(r.end\_station\_code)/total.total\*100 as perc from `data` r, total

join stations s on (r.end\_station\_code = s.code)

group by r.end\_station\_code, total.total, s.name

order by rides DESC

limit 5;

**Question 2:**

Write a HiveQL script that will calculate the distance in KM for the shortest and the longest ride

To calculate a distance between two points based on their longitude and latitude coordinates you can use the Haversine formula.

<https://en.wikipedia.org/wiki/Haversine_formula>

invalidate metadata;

use a1p2;

select s1.name as start\_station, s2.name as end\_station,

2 \* asin(

sqrt(

cos(radians(s1.latitude)) \*

cos(radians(s2.latitude)) \*

pow(sin(radians((s1.longitude - s2.longitude)/2)), 2)

+

pow(sin(radians((s1.latitude - s2.latitude)/2)), 2)

)

) \* 6371 as distance\_km

from `data` r

join stations s1 on (r.start\_station\_code = s1.code)

join stations s2 on (r.end\_station\_code = s2.code)

order by distance\_km DESC

limit 1;

select s1.name as start\_station, s2.name as end\_station,

2 \* asin(

sqrt(

cos(radians(s1.latitude)) \*

cos(radians(s2.latitude)) \*

pow(sin(radians((s1.longitude - s2.longitude)/2)), 2)

+

pow(sin(radians((s1.latitude - s2.latitude)/2)), 2)

)

) \* 6371 as distance\_km

from `data` r

join stations s1 on (r.start\_station\_code = s1.code)

join stations s2 on (r.end\_station\_code = s2.code)

order by distance\_km ASC

limit 1;