SQL Queries in a Hadoop Cluster (HiveQL)

**Data Preparation**

1. Add files to Hadoop Cluster (Linux)

hadoop fs -put /root/lab/station\_data.csv /user/lab/station\_dat.csv

hadoop fs -put /root/lab/trip\_data.csv /user/lab/trip\_dat.csv

2. Create database (SQL)

create database bikes;

use bikes;

4. Create Tables

create table bikes.stationtemp (

station\_id int, name string, lat float, lon float, dockcount int, landmark string, install string)

row format delimited

fields terminated by ',';

load data inpath '/user/lab/station\_dat'

overwrite into table bikes.stationtemp;

create table bikes.station as

select station\_id, name, lat, lon, dockcount, landmark,

from\_unixtime(unix\_timestamp(install , 'M/d/yyyy')) as install\_date

from bikes.stationtemp;

5. Test if bikes.station exists, then drop temporary table.

select \* from bikes.station limit 5;

drop table bikes.stationtemp;

create table bikes.triptemp (

trip\_id int, duration float, start\_date string, start\_station string, start\_terminal int, end\_date string, end\_station string, end\_terminal int, bike\_number int, sub\_type string, zip int)

row format delimited

fields terminated by ',';

load data inpath '/user/lab/trip\_dat.csv'

overwrite into table bikes.triptemp;

create table bikes.trip as

select trip\_id, duration, from\_unixtime(unix\_timestamp(start\_date , 'M/d/yyyy H:m')) as start\_date, start\_station, start\_terminal, from\_unixtime(unix\_timestamp(end\_date , 'M/d/yyyy H:m')) as end\_date, end\_station, end\_terminal, bike\_number, sub\_type, zip

from bikes.triptemp;

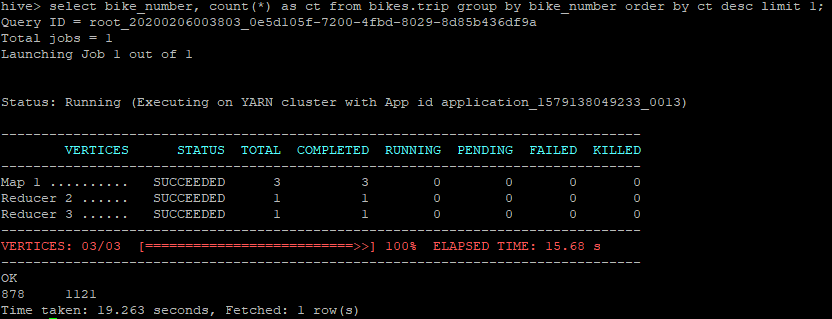
6. Test if bikes.trip exists, then drop temp table.

select \* from bikes.trip limit 5;

drop table bikes.triptemp;

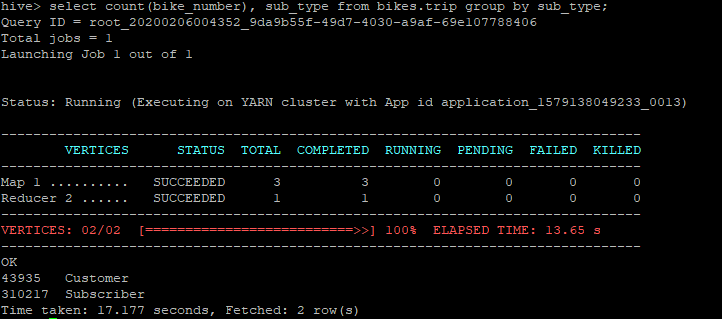
Find the 'most popular' bike (the bike that has made the highest number of trips)

select bike\_number, count(\*) as ct from bikes.trip group by bike\_number order by ct desc limit 1;



Find the number of trips made by each subscription type

select count(bike\_number), sub\_type from bikes.trip group by sub\_type;



Build a table that shows which stations are connected, and the minimum duration between them.

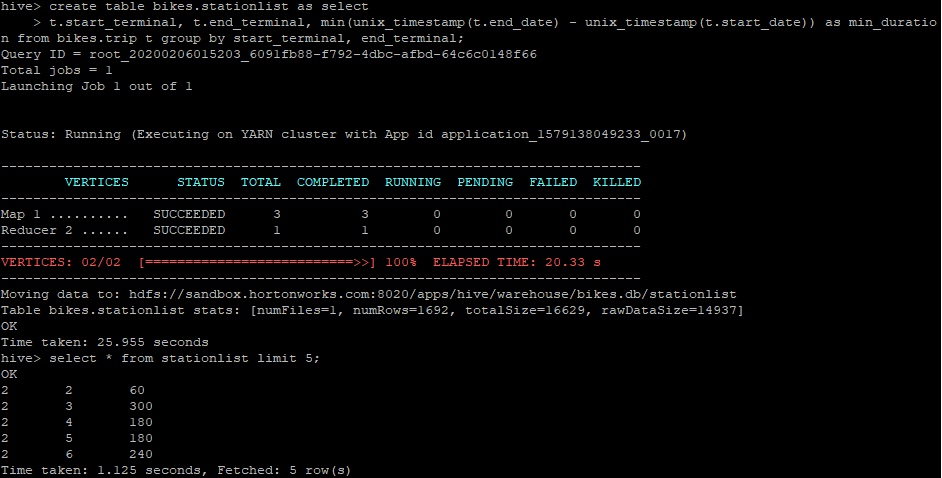
create table bikes.stationlist as select

t.start\_terminal, t.end\_terminal,

min(unix\_timestamp(t.end\_date) - unix\_timestamp(t.start\_date)) as min\_duration

from bikes.trip t

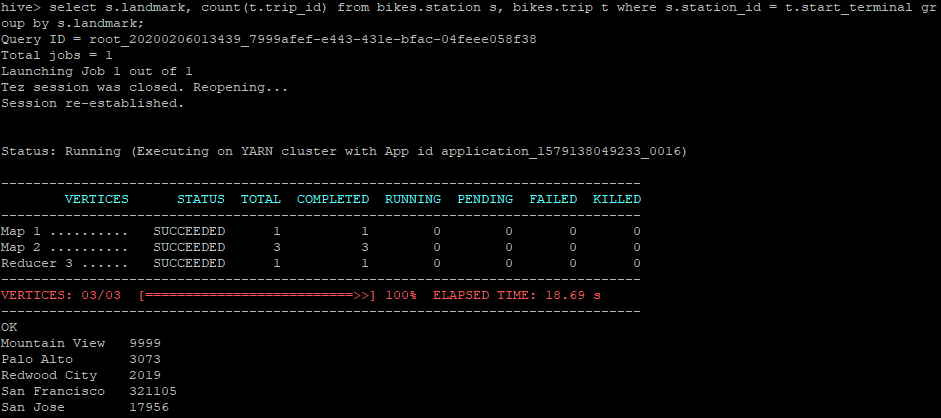
group by start\_terminal, end\_terminal;



4. Find the number of trips originating from each landmark.

select s.landmark, count(t.trip\_id) from bikes.station s, bikes.trip t

where s.station\_id = t.start\_terminal group by s.landmark



Find the number of trips crossing landmarks.

select s.landmark as strt\_landmark, send.landmark as end\_landmark, count(t.trip\_id) as ct

from bikes.trip t join bikes.station s on s.station\_id = t.start\_terminal join bikes.station send

on send.station\_id = t.end\_terminal where s.landmark <> send.landmark

group by s.landmark, send.landmark;

