# CSP554—Big Data Technologies

## Assignment #4

Exercise 1) 2 points

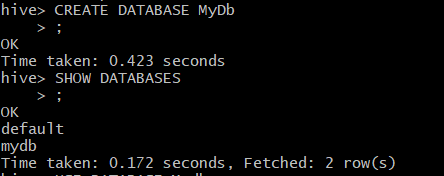
Magic Number = 60618

To Create Database:

CREATE DATABASE MyDb;

SHOW DATABASES;

USE MyDB;





To Create Table foodratings:

CREATE TABLE IF NOT EXISTS MyDb.foodratings(

name STRING COMMENT ‘Critic Name’

food1 STRING COMMENT ‘food Item1’,

food2 STRING COMMENT ‘food Item2’,

food3 STRING COMMENT ‘food Item3’,

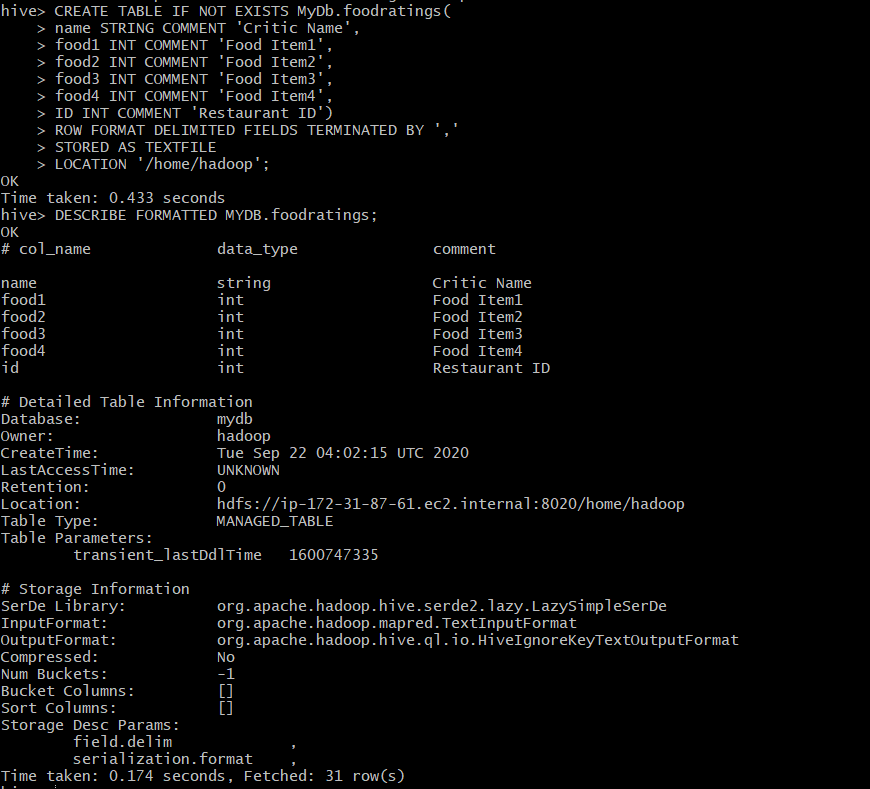
food4 STRING COMMENT ‘food Item4’,

id INT COMMENT ‘Restaurant ID’)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’

STORED AS TEXTFILE

LOCATION ‘/home/hadoop’;



To Create Table foodplaces:;

CREATE TABLE IF NOT EXISTS MyDb.foodplaces(

id INT,

place STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’

STORED AS TEXTFILE

LOCATION ‘/home/hadoop’;

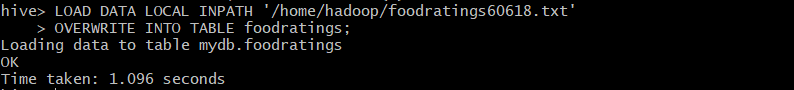
## 

Exercise 2) 2 points

Loading Data:

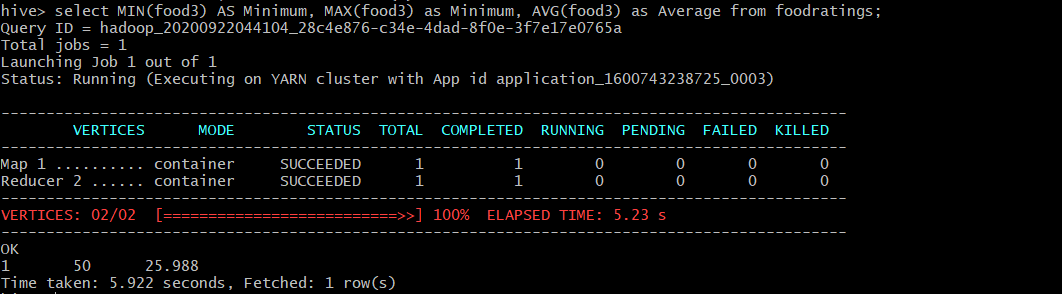
LOAD DATA LOCAL INPATH ‘/home/hadoop/foodratings60618.txt’

OVERWRITE INTO TABLE foodratings;



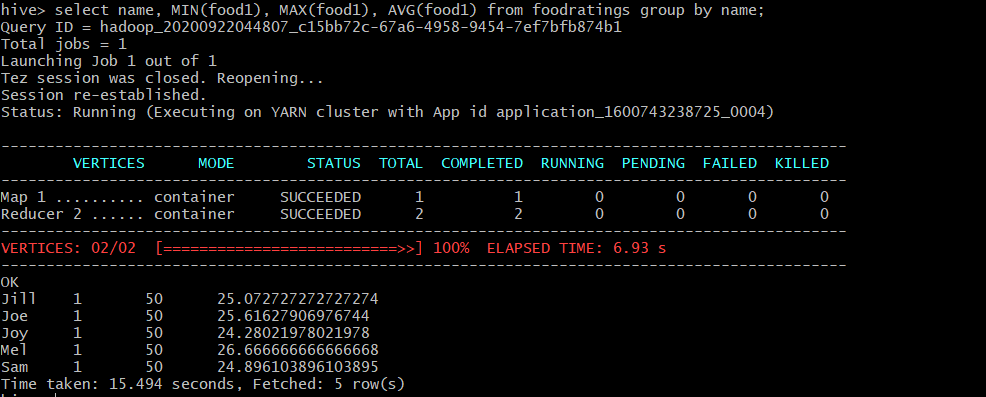
SELECT MIN(food3) AS MINIMUM, MAX(food3) AS MAXIMUM, AVG(food3) AS

AVERAGE FROM foodratings;



Exercise 3) 2 points

SELECT name, MIN(food1), MAX(food1), AVG(food1) FROM foodratings GROUP BY name;



Exercise 4) 2 points

To create Table foodratingspart:

CREATE TABLE MyDb.foodratingspart(

food1 STRING,

food2 STRING,

food3 STRING,

food4 STRING,

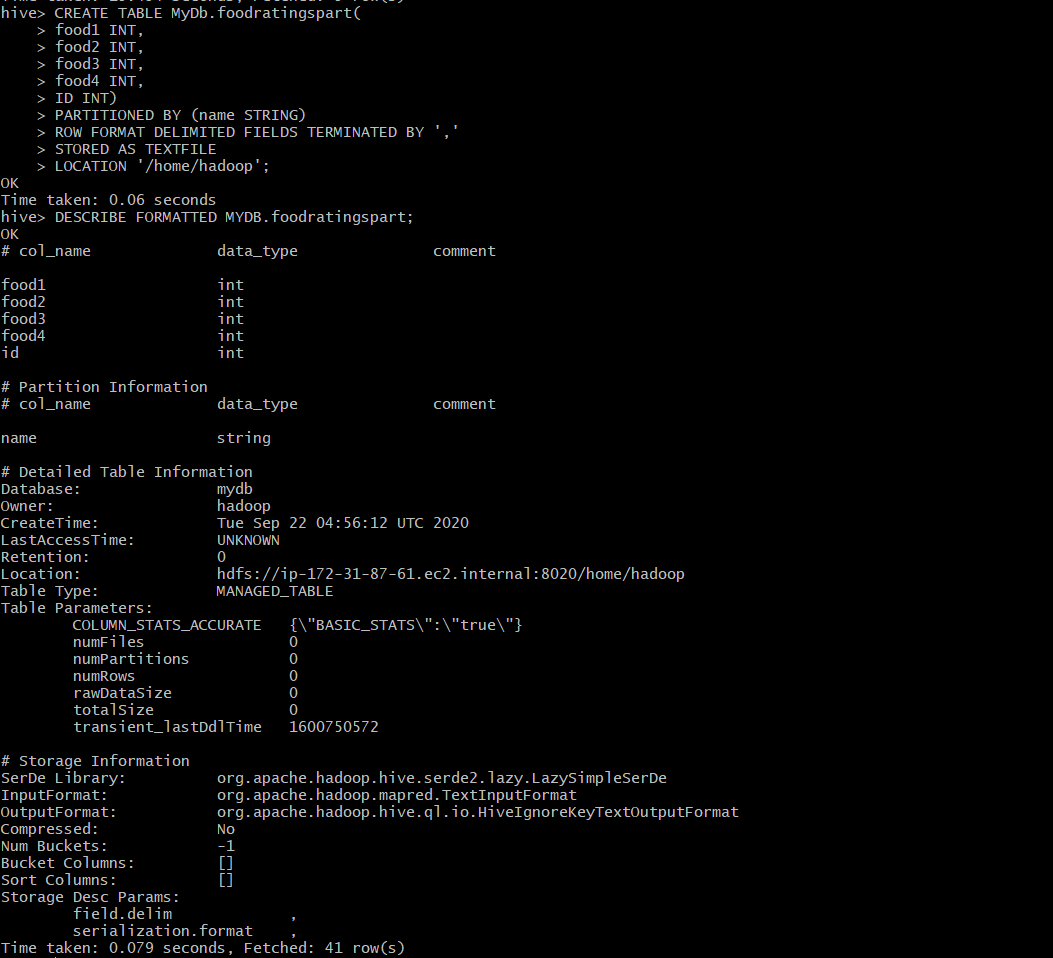
id INT)

PARTITIONED BY (name STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’

STORED AS TEXTFILE

LOCATION ‘/home/hadoop’;



Exercise 5) 2 points

Assume that the number of food critics is relatively small, say less than 10 and the number places to eat is very large, say more than 10,000. In a few short sentences explain why using the (critic) name is good choice for a partition field while using the place id is not.

Ans:

As given, the number of food critics is relatively small compared to the number of places. Hence, partitioning based on name will results in comparatively small number of partitions. However, if we perform partitioning based on place id then it will results I large number of partitions which ultimately results in Over Partitioning by increasing overhead in data loading and retrieval.

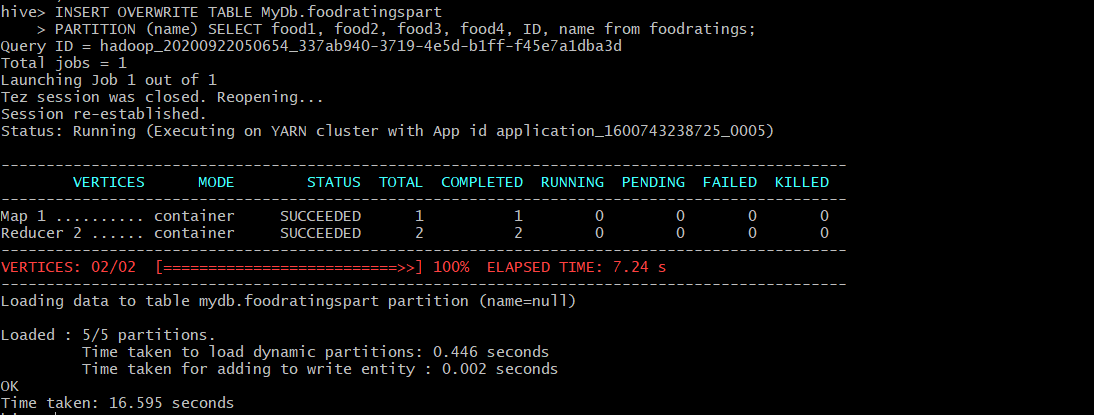
Exercise 6) 2 points

SET hive.exec.dynamic.partition = true;

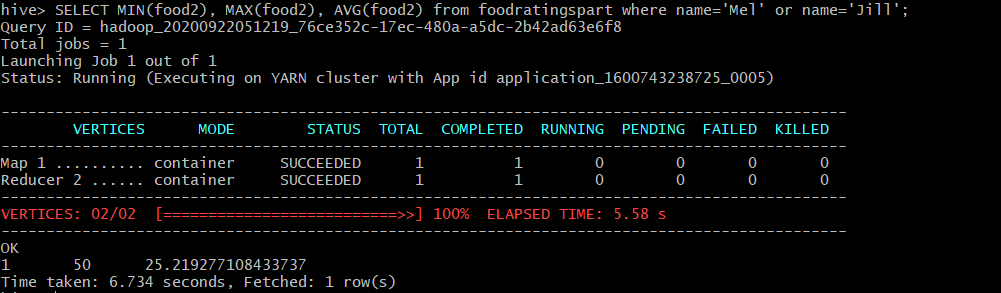
SET hive.exec.dynamic.partition.mode = non-strict;

INSERT OVERWRITE TABLE MyDb.foodratingspart

PARTITION (name) SELECT food1, food2, food3, food4, id, name FROM foodratings;



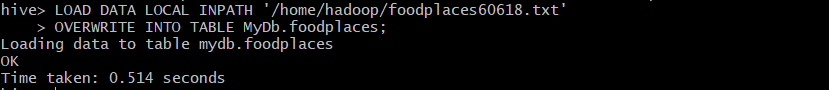
SELECT MIN(food2), MAX(food2), AVG(food2) FROM foodratingspart WHERE name = ‘Mel’ OR name = ‘Jill’;



Exercise 7) 2 points

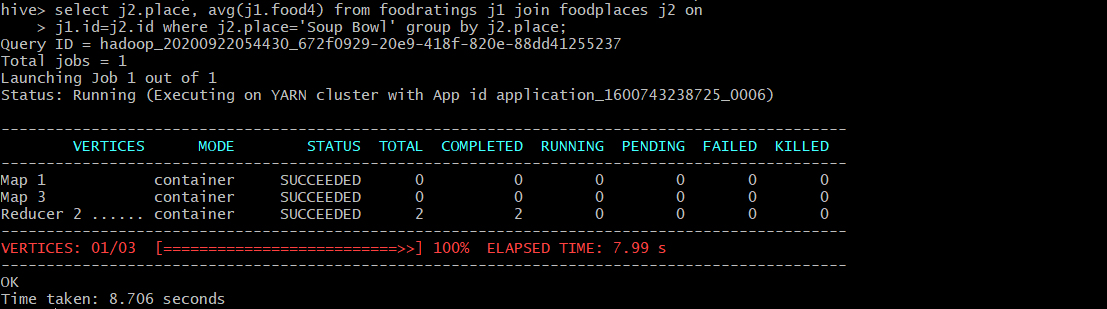
LOAD DATA LOCAL INPATH ‘/home/hadoop/foodplaces60618.txt’

OVERWRITE INTO TABLE MyDb.foodplaces;



SELECT j2.place, AVG(j1.food4) FROM foodratings j1 JOIN foodplaces j2 ON

(j1.id = j2.id) WHERE j2.place = ‘Soup Bowl’ GROUP BY j2.place;

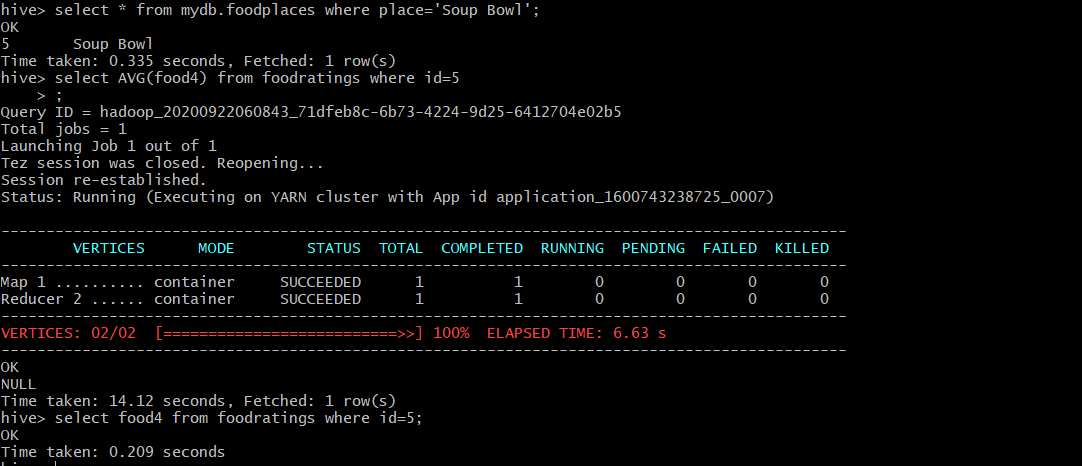


Verification:

SELECT \* FROM MyDb.foodplaces where place = ‘Super Bowl’;

SELECT AVG(food4) from foodratings where id = 5;

SELECT food4 FROM foodratinngs WHERE ID = 5;



Exercise 8) 4 points

Read the article “An Introduction to Big Data Formats” found on the blackboard in section “Articles” and provide short (2 to 4 sentence) answers to the following questions:

1. When is the most important consideration when choosing a row format and when a column format for your big data file?

Ans:

**Column based storage** is useful when performing analytics queries that require only subset of columns examined over very large datasets.

**Row based storage** is useful when queries requires require access to all or most of the columns of each row of data.

1. What is “splittability” for a column file format and why is it important when processing large volumes of data?

Ans:

* Splittability for column-based file format is splitting a job into separate jobs when a query calculation is concerned with a single column at a time.
* The columnar formats discussed in the paper are row-columnar, which means they take a batch of rows and store that batch in columnar format.
* These batches then become split boundaries.
* It is important when processing a large volume of data because it will increase the efficiency of processing these huge amounts of data by breaking the job into parts and by introducing parallelism.

1. What can files stored in column format achieve better compression than those stored in row format?

Ans:

* Columnar data can achieve better compression rates than row-based data.
* Storing values by column, with the same type next to each other, allows you to do more efficient compression on them than if you’re storing rows of data.

1. Under what circumstances would it be the best choice to use the “Parquet” column file format?

Ans:

* Parquet is often used to analyze wide datasets with many columns. Each of parquet files contains binary data organized by “row group”. For each row group, the data values are organized by column. So, this is useful when there are a greater number of columns.
* Most compatible platforms for Parquet column file format are Impala, Arrow, Drill and Spark.