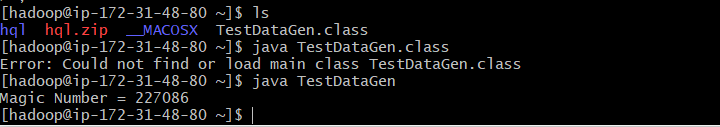
Uploaded hql.zip using “scp” and unzipped it

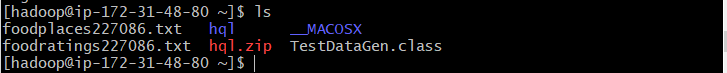
Running TestDataGen class to create Below Details:

Magic Number=227086

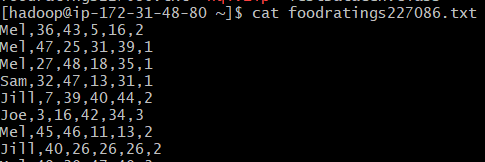
Foodplaces227086.txt

Foodratings22707086.txt

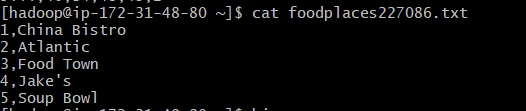




Contents of Foodratings22707086.txt

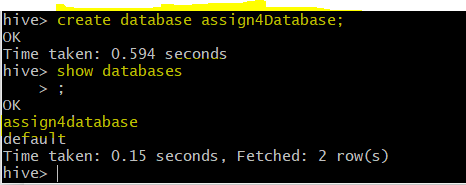


Contents of Foodplaces227086.txt



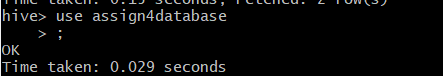
Creating Database: assign4Database

Command Used: create database assign4database;



Selecting DataBase: Selecting newly created database using below command

Command Used: use assign4database;



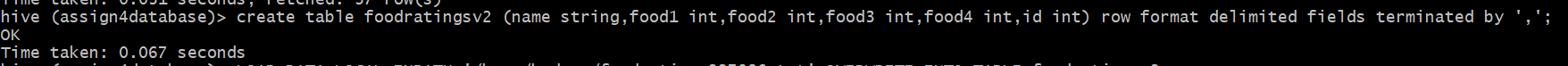
Printing Database Name: Printing Database to be sure of database working

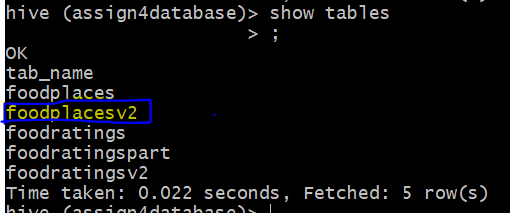
Command Used: set hive.cli.print.current.db=true;



Table Creation : Created new table foodratingsv2

Command Used: CREATE TABLE <tableName> (<column1Name><Column1Datatype>,<column2Name><Column2Datatype>,….) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ;





Adding Comments :Adding the comment to already created table. using the uderlying command

Command Used:ALTER TABLE foodratingsv2 CHANGE <ColumnName> <ColumnName> <ColumnDataType> COMMENT ‘<comments>’ ;

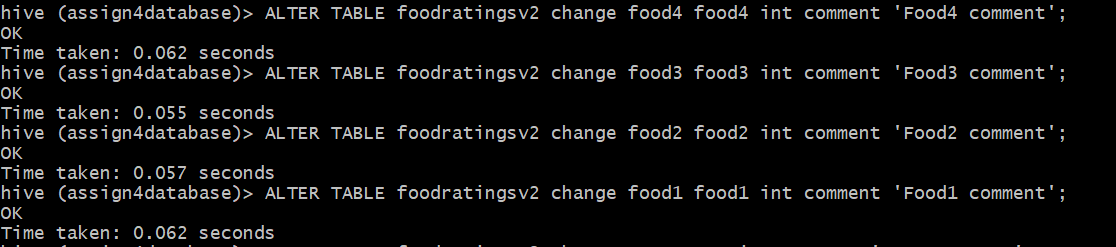
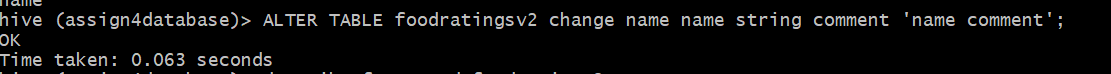
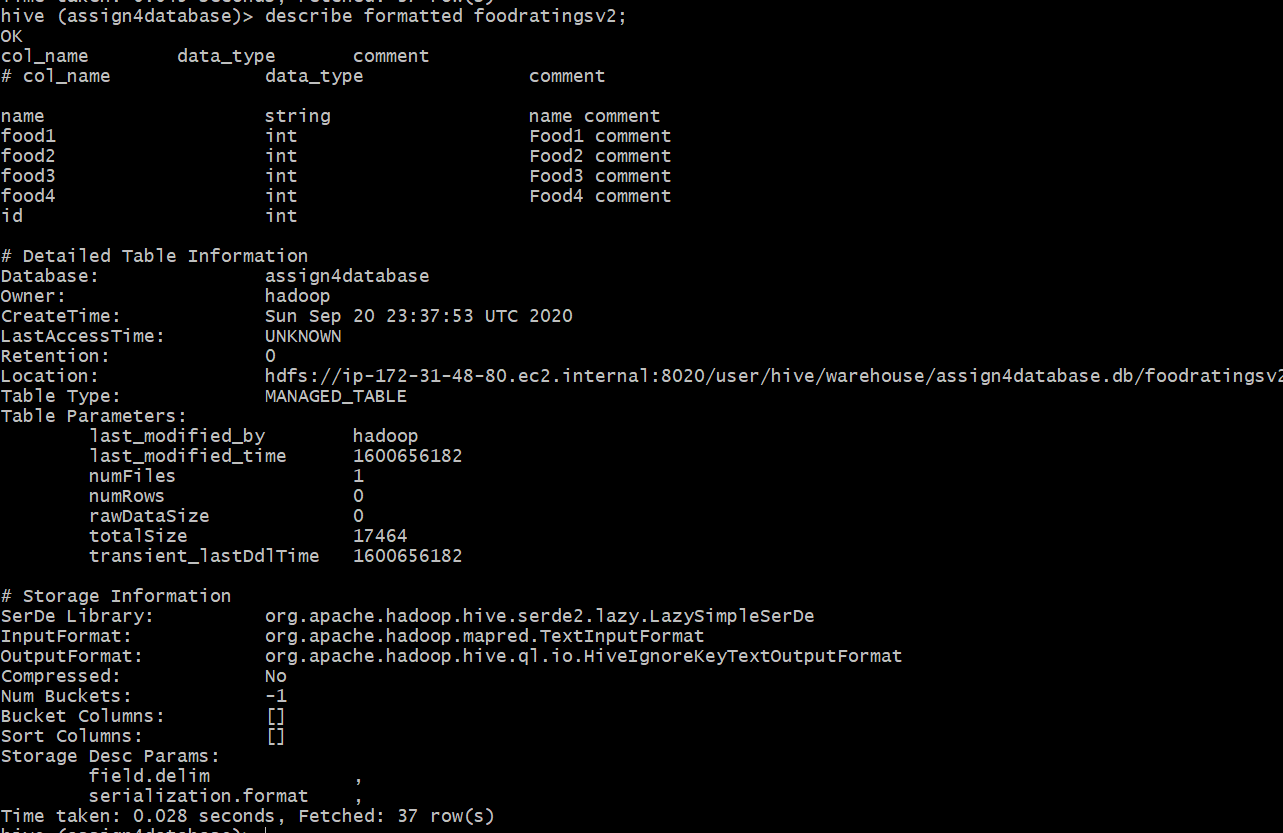


Table Description: Showing the table description using below command

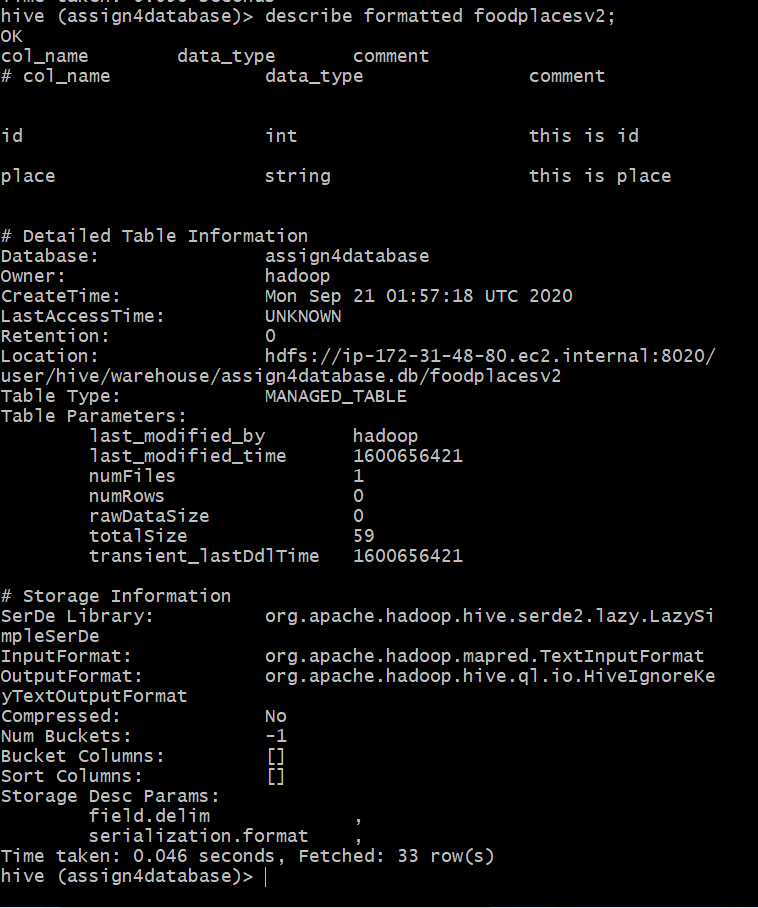
Command used: describe formatted foodratingsv2;





Adding Comments :Adding the comment to already created table. using the uderlying command

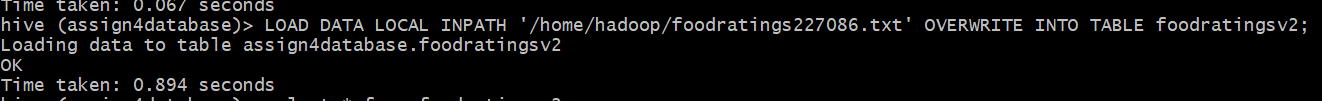
Command Used:ALTER TABLE foodplacesv2 CHANGE <ColumnName> <ColumnName> <ColumnDataType> COMMENT ‘<comments>’ ;



2) Magic Number=227086

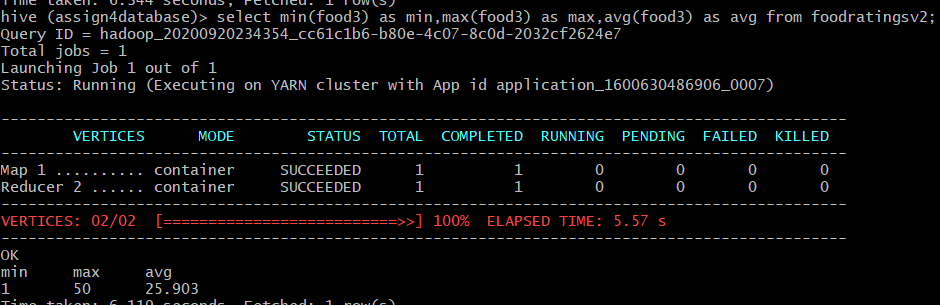
Load Data: Loading Data from foodratings227086.txt to foodratingsv2 using below command.

Command used: LOAD DATA LOCAL INPATH ‘/home/Hadoop/foodratings227086.txt’ OVERWRITE INTO TABLE foosratingsv2 ;



Printing Values: Printing Min, Max, Avg of column food3 using below command.

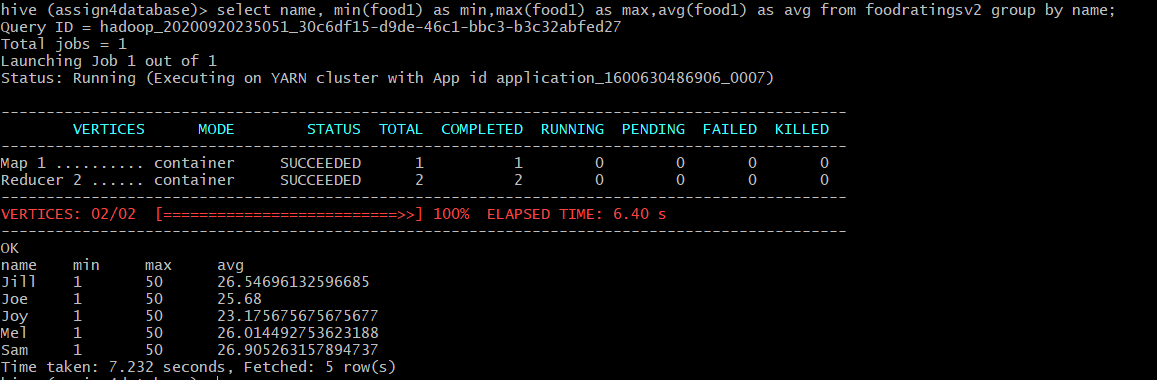
Command Used: select min(food3) as min, max(food3) as max, avg(food3) as avg from foodratingsv2;



3) Magic Number=227086

Printing Values: Printing Min, Max, Avg of column food3 grouped by name using below command.

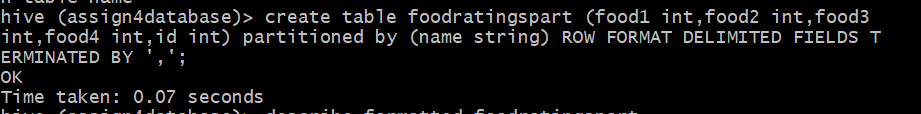
Command Used: select name, min(food1) as min, max(food1) as max, avg(food1) as avg from foodratingsv2 group by name;

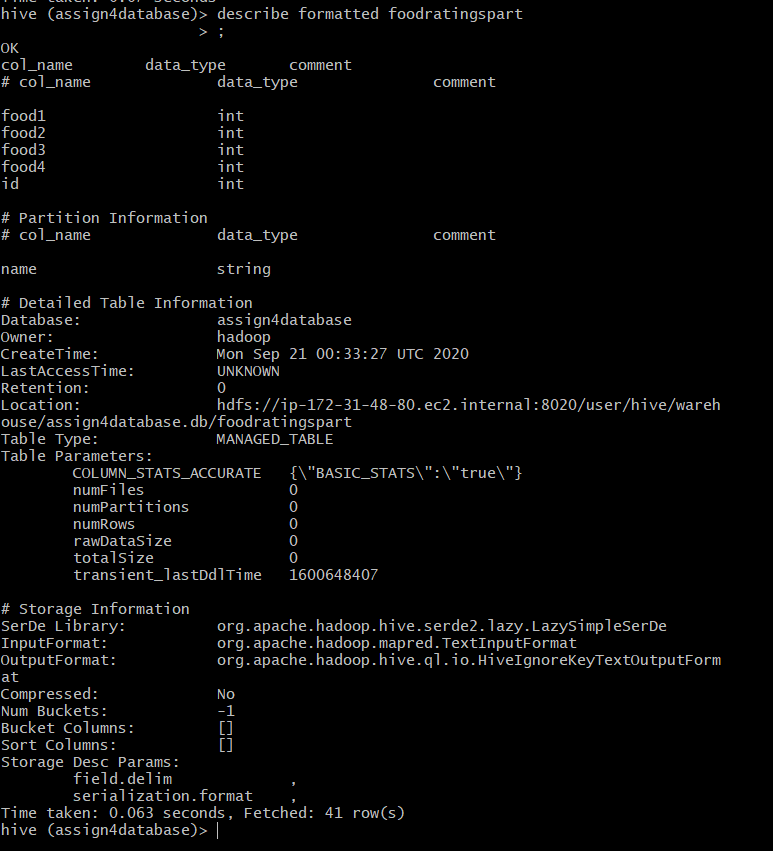


4)

Creating Table: creating table using below command.

Command Used: CREATE TABLE <tableName> (<column1Name><Column1Datatype>,<column2Name><Column2Datatype>,….) PARTIONED BY (<columnName><ColumnDatatype>) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ;





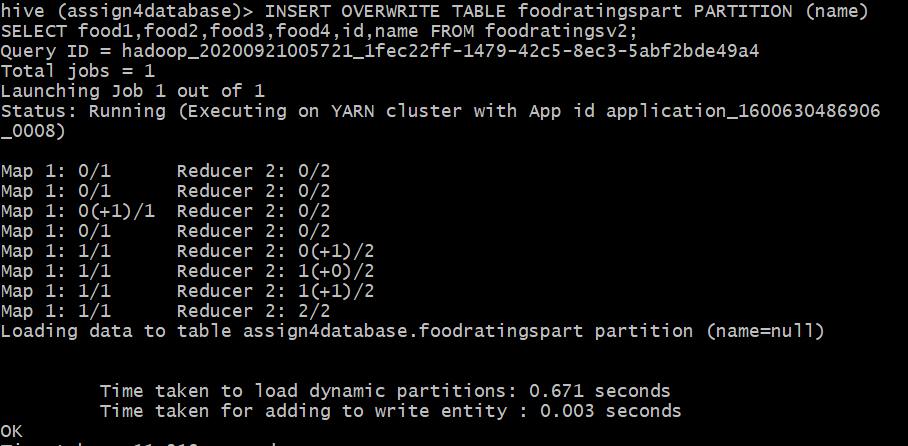
5)

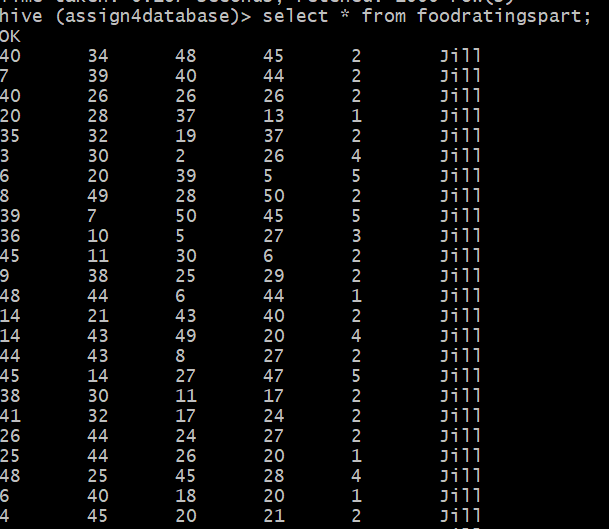
As the number of critics is relatively small it helps in organizing data efficiently over the partition feature.

6)

Copying Data from table-> portioned table: Copying the data from normal table to the Partitioned table

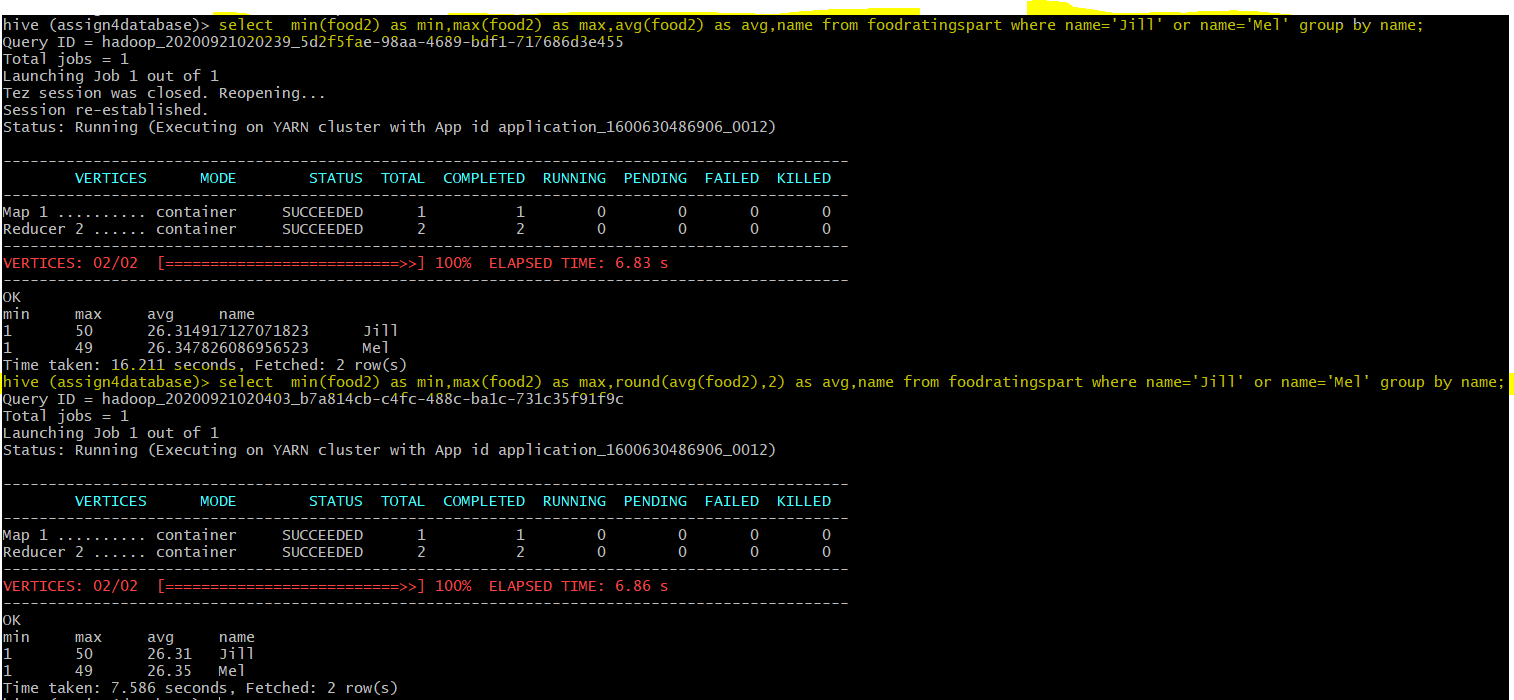
Command used: INSERT OVERWRITE TABLE foodratingspart PARTITION (name) SELECT food1,food2,food3,food4,id ,name FROM foodratingsv2 ;





Calculating following Statistics: calculating the avg, Min, Max using following command

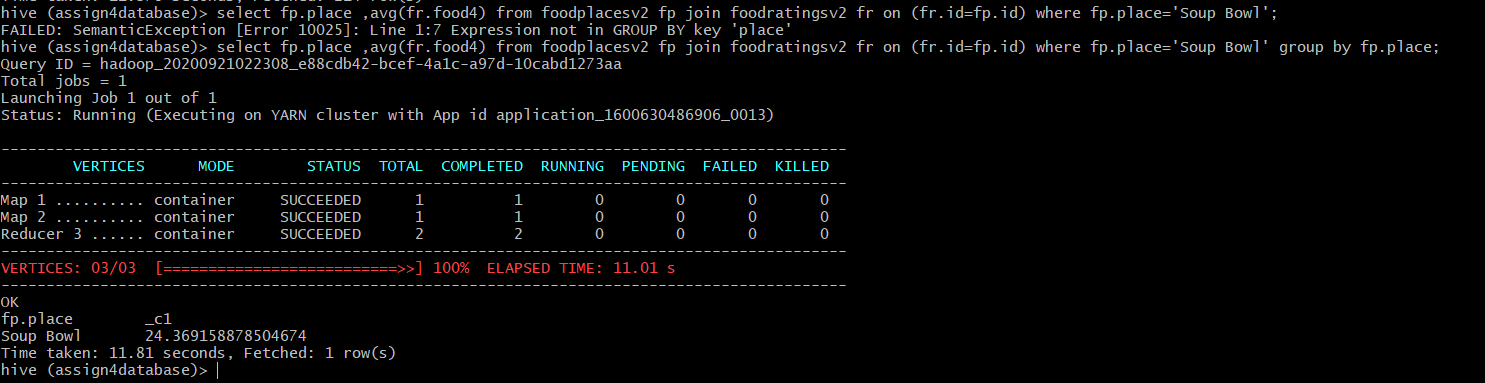
Command used: SELECT min (food2 AS min, max(food2) AS max, avg(food2) AS avg, name FROM foodratingpart WHERE name=’Jill’ OR name=’Mel’ GROUP BY name;



7)

Joining Tables and Finding Avg: Joining tables foodratingv2, foodplacesv2 and finding average value of food4 column when place is equal to ‘Soup Bowl’.

Command Used: SELECT fp.place, avg(fr.food4) FROM foodplacesv2 fp JOIN foodratingsv2 fr ON (fr.id=fp.id) WHERE fp.place =’Soup Bowl’ GROUP BY fp.place ;



8)

1. Row data format is chosen when your query requires to access almost all the columns in the row. Column based format is chosen when we are doing analytics query that require only some columns of the data.
2. Breaking down of the data into smaller records that can be handled independently is called splitability. It used to process large volumes of data efficiently. It usually requires breaking the job up into parts that can be farmed out to separate processors. In fact, large-scale parallelization of processing is key to performance. For example, if each file in your dataset contains one massive XML structure or JSON record, the files will not be “splittable”, i.e. decomposable into smaller records that can be handled independently.
3. Data stored in column format can achieve better compression rates than row-based data. Storing values by column, with the same type next to each other, allows the user to do more efficient compression on them than if you’re storing rows of data.
4. Parquet is especially adept at analyzing wide datasets with many columns. Each Parquet file contains binary data organized by “row group.” For each row group, the data values are organized by column. Parquet is a good choice for read-heavy workloads.