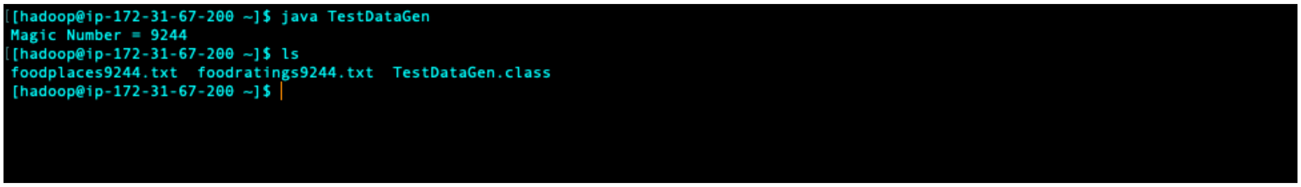
Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

CSP 554
Assignment – 4

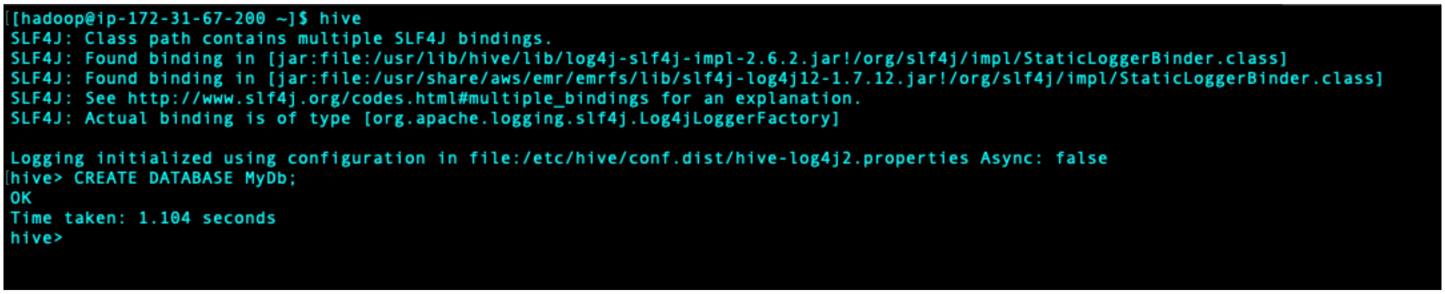
1) Generating magic number in AWS EMR cluster by using command “java TestDataGen”



```
[hadoop@ip-172-31-67-200 ~]$ java TestDataGen
Magic Number = 9244
[hadoop@ip-172-31-67-200 ~]$ ls
foodplaces9244.txt  foodratings9244.txt  TestDataGen.class
[hadoop@ip-172-31-67-200 ~]$ |
```

Exercise 1) 2 points

2) Create a Hive database called “MyDb”. Using Command “CREATE DATABASE MyDb”



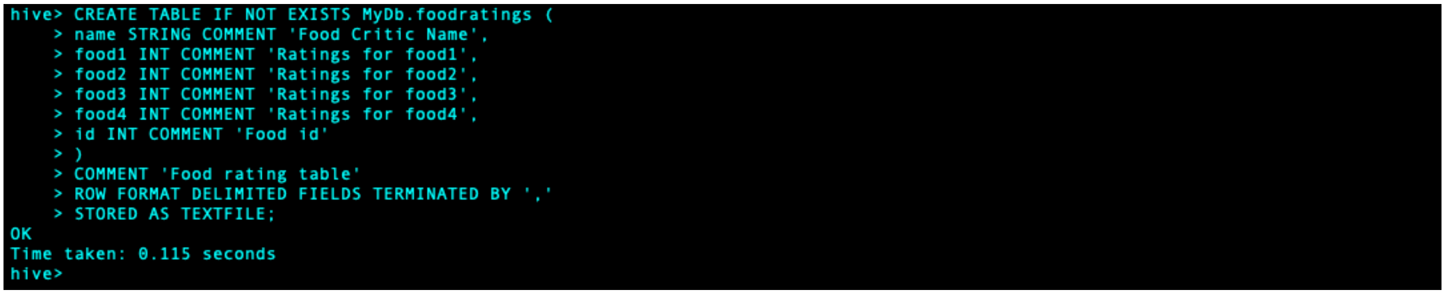
```
[[hadoop@ip-172-31-67-200 ~]$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/share/aws/emr/emrfs/lib/slf4j-log4j12-1.7.12.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
[hive> CREATE DATABASE MyDb;
OK
Time taken: 1.104 seconds
hive>
```

3) Creating table “foodratings” in database “MyDb”.

Using command: -

```
CREATE TABLE IF NOT EXISTS MyDb.foodratings (
        name STRING COMMENT 'Food Critic Name',
        food1 INT COMMENT 'Ratings for food1',
        food2 INT COMMENT 'Ratings for food2',
        food3 INT COMMENT 'Ratings for food3',
        food4 INT COMMENT 'Ratings for food4',
        id INT COMMENT 'Food id'
        )
        COMMENT 'Food rating table'
        ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
         STORED AS TEXTFILE;
```

Output: -

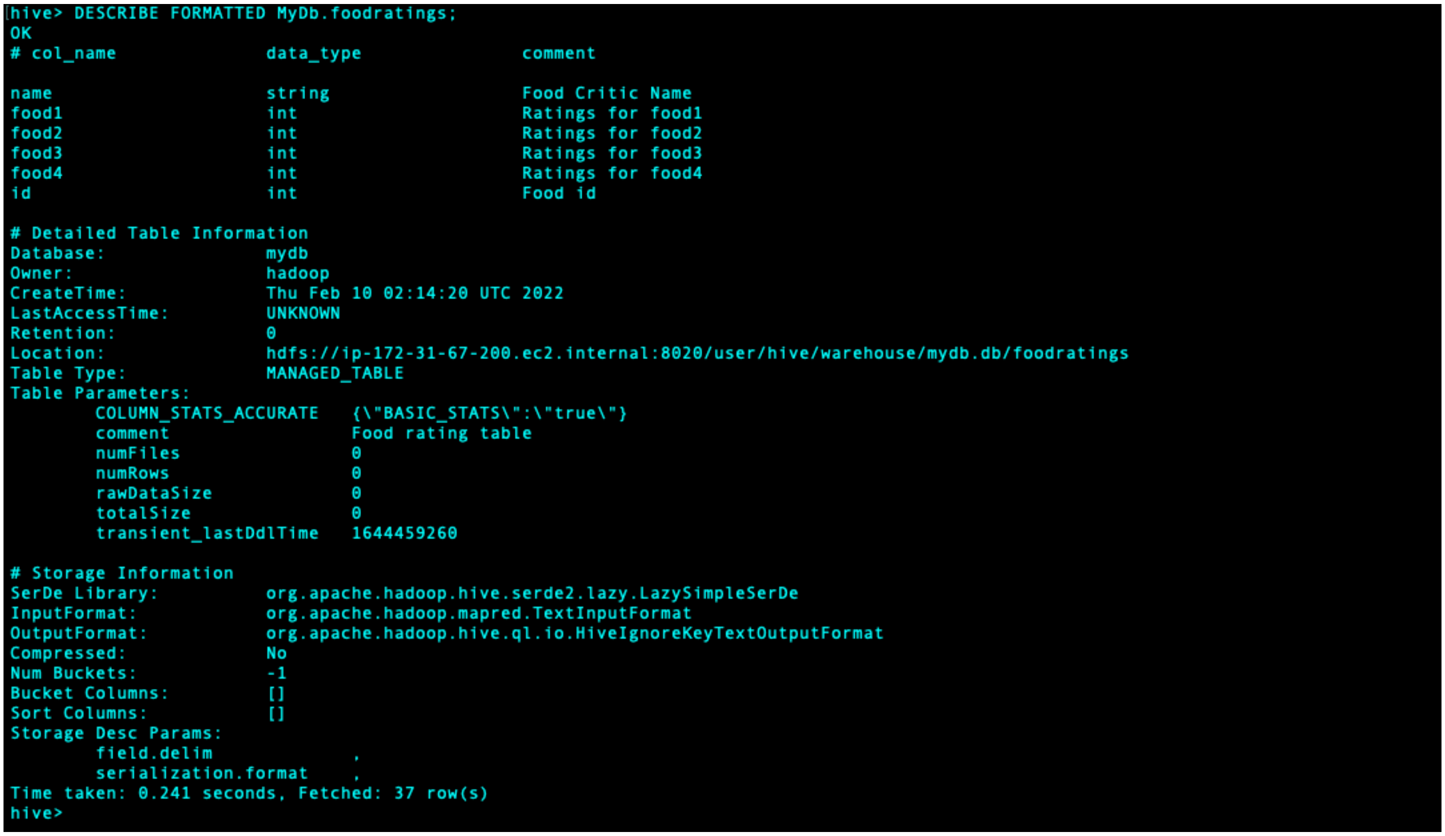


```
hive> CREATE TABLE IF NOT EXISTS MyDb.foodratings (
    > name STRING COMMENT 'Food Critic Name',
    > food1 INT COMMENT 'Ratings for food1',
    > food2 INT COMMENT 'Ratings for food2',
    > food3 INT COMMENT 'Ratings for food3',
    > food4 INT COMMENT 'Ratings for food4',
    > id INT COMMENT 'Food id'
    > )
    > COMMENT 'Food rating table'
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.115 seconds
hive>
```

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

4) Executing command “DESCRIBE FORMATTED MyDb.foodratings;”

Output: -



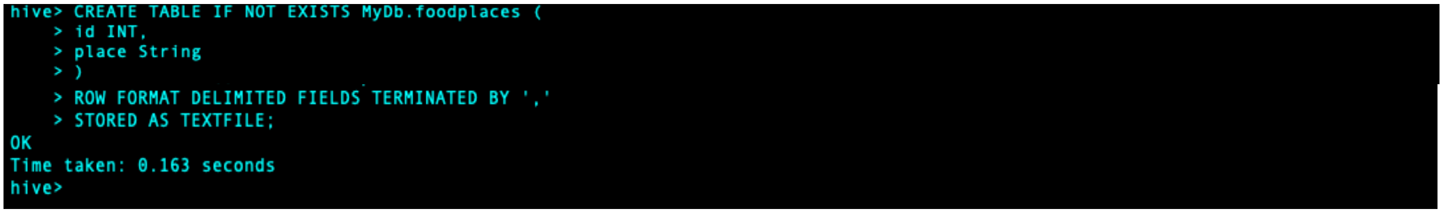
```
hive> DESCRIBE FORMATTED MyDb.foodratings;
OK
# col_name              data_type               comment

name                    string                  Food Critic Name
food1                   int                     Ratings for food1
food2                   int                     Ratings for food2
food3                   int                     Ratings for food3
food4                   int                     Ratings for food4
id                      int                     Food id

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Thu Feb 10 02:14:20 UTC 2022
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-67-200.ec2.internal:8020/user/hive/warehouse/mydb.db/foodratings
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        comment                 Food rating table
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1644459260

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        serialization.format    ,
Time taken: 0.241 seconds, Fetched: 37 row(s)
hive>
```

5) Creating table “foodplaces” in database “MyDb”.

Using Command: -

```
CREATE TABLE IF NOT EXISTS MyDb.foodplaces (
        id INT,
        place String
        )
        ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
        STORED AS TEXTFILE;
```

Output: -

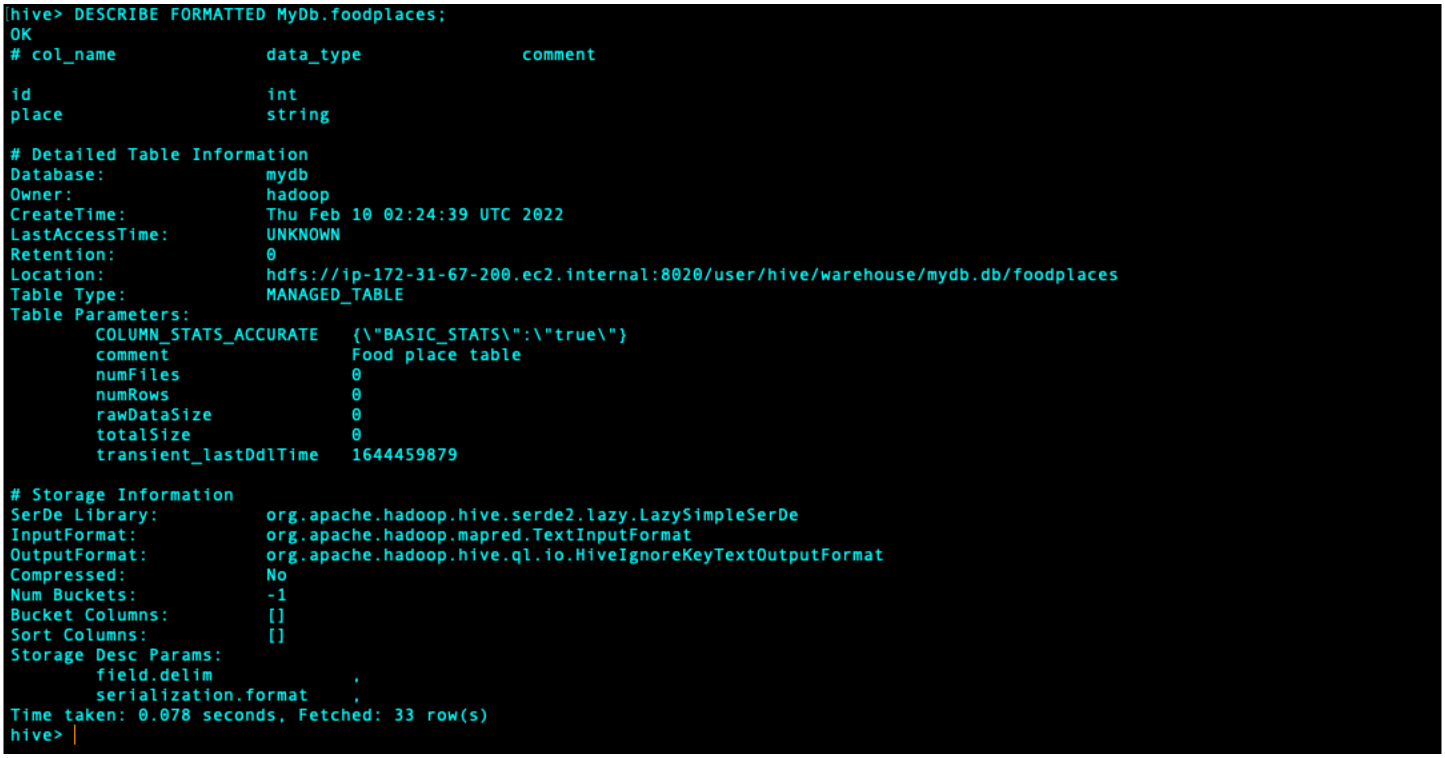


```
hive> CREATE TABLE IF NOT EXISTS MyDb.foodplaces (
    > id INT,
    > place String
    > )
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.163 seconds
hive>
```

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

6) Executing command “DESCRIBE FORMATTED MyDb.foodplaces;”

Output: -



```
hive> DESCRIBE FORMATTED MyDb.foodplaces;
OK
# col_name              data_type               comment

id                      int
place                   string

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Thu Feb 10 02:24:39 UTC 2022
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-67-200.ec2.internal:8020/user/hive/warehouse/mydb.db/foodplaces
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        comment                 Food place table
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1644459879

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        serialization.format    ,
Time taken: 0.078 seconds, Fetched: 33 row(s)
hive>
```

Exercise 2) 2 points

7) Load the foodratings<magic number>.txt file created using TestDataGen from your local file system into the foodratings table.

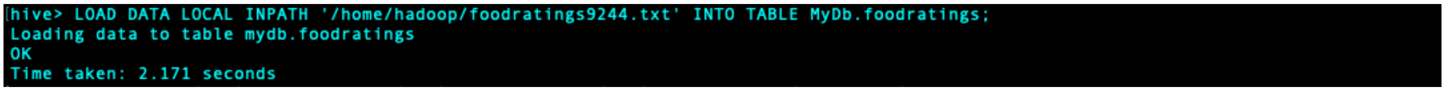
Using Command: -

LOAD DATA LOCAL INPATH '/home/hadoop/foodratings9244.txt' INTO TABLE MyDb.foodratings;

Or we can also use below mentioned command

LOAD DATA LOCAL INPATH '/home/hadoop/foodratings9244.txt' OVERWRITE INTO TABLE MyDb.foodratings;

Output: -



```
hive> LOAD DATA LOCAL INPATH '/home/hadoop/foodratings9244.txt' INTO TABLE MyDb.foodratings;
Loading data to table mydb.foodratings
OK
Time taken: 2.171 seconds
```

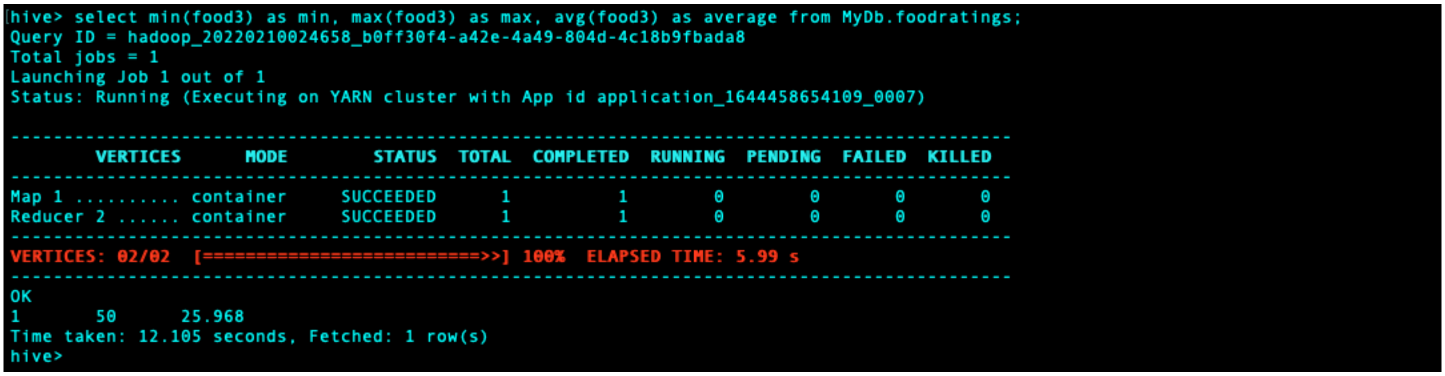
8) Execute a hive command to output the min, max and average of the values of the food3 column of the foodratings table.

Using Command: -

select min(food3) as min, max(food3) as max, avg(food3) as average from MyDb.foodratings;

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

Output: -



```
[hive> select min(food3) as min, max(food3) as max, avg(food3) as average from MyDb.foodratings;
 Query ID = hadoop_20220210024658_b0ff30f4-a42e-4a49-804d-4c18b9fbada8
 Total jobs = 1
 Launching Job 1 out of 1
 Status: Running (Executing on YARN cluster with App id application_1644458654109_0007)

 ----------------------------------------------------------------------------------------------
         VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
 ----------------------------------------------------------------------------------------------
 Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
 Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
 ----------------------------------------------------------------------------------------------
 VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5.99 s
 ----------------------------------------------------------------------------------------------
 OK
 1       50      25.968
 Time taken: 12.105 seconds, Fetched: 1 row(s)
 hive>
```

Magic Number = 9244

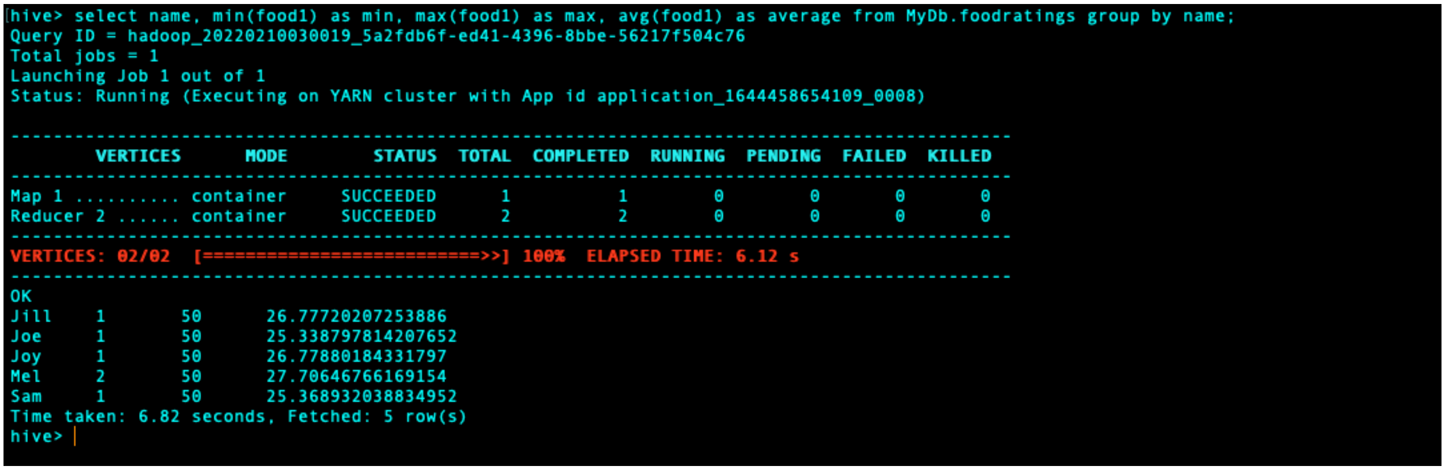
Exercise 3) 2 points

9) Execute a hive command to output the min, max and average of the values of the food1 column grouped by the first column 'name'.

Using Command: -

select name, min(food1) as min, max(food1) as max, avg(food1) as average from MyDb.foodratings group by name;

Output: -



```
[hive> select name, min(food1) as min, max(food1) as max, avg(food1) as average from MyDb.foodratings group by name;
 Query ID = hadoop_20220210030019_5a2fdb6f-ed41-4396-8bbe-56217f504c76
 Total jobs = 1
 Launching Job 1 out of 1
 Status: Running (Executing on YARN cluster with App id application_1644458654109_0008)

 ----------------------------------------------------------------------------------------------
         VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
 ----------------------------------------------------------------------------------------------
 Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
 Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
 ----------------------------------------------------------------------------------------------
 VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 6.12 s
 ----------------------------------------------------------------------------------------------
 OK
 Jill    1       50      26.77720207253886
 Joe     1       50      25.338797814207652
 Joy     1       50      26.77880184331797
 Mel     2       50      27.70646766169154
 Sam     1       50      25.368932038834952
 Time taken: 6.82 seconds, Fetched: 5 row(s)
 hive>
```

Magic Number = 9244

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

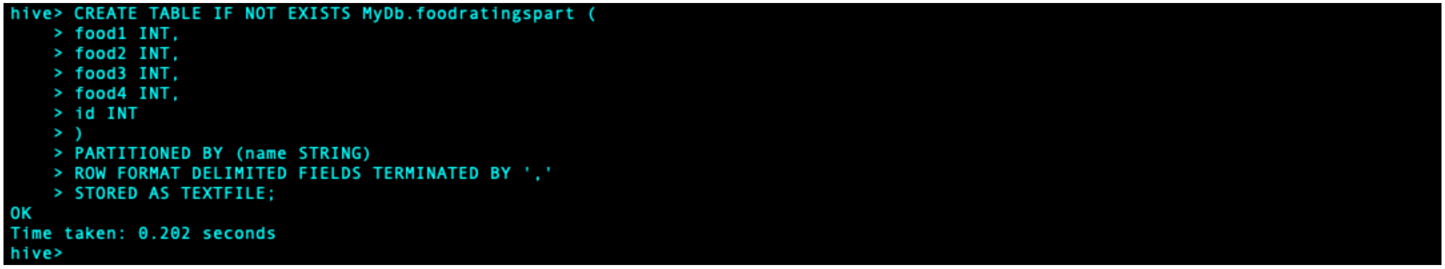
Exercise 4) 2 points

10) In MyDb create a partitioned table called ‘foodratingspart’

Output: -

Command use: -

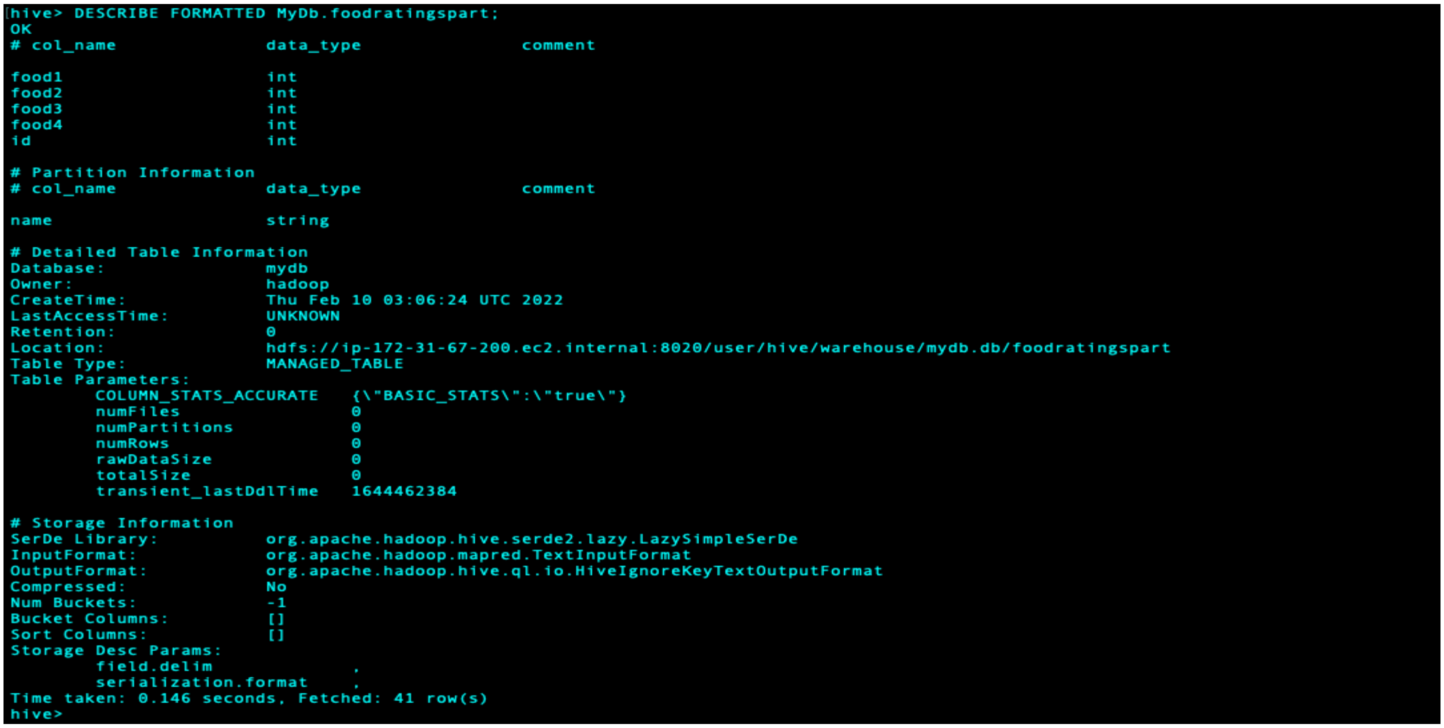
```
CREATE TABLE IF NOT EXISTS MyDb.foodratingspart (
        food1 INT,
        food2 INT,
        food3 INT,
        food4 INT,
        id INT
        )
        PARTITIONED BY (name STRING)
        ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
         STORED AS TEXTFILE;
```



```
hive> CREATE TABLE IF NOT EXISTS MyDb.foodratingspart (
    > food1 INT,
    > food2 INT,
    > food3 INT,
    > food4 INT,
    > id INT
    > )
    > PARTITIONED BY (name STRING)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.202 seconds
hive>
```

11) Execute a Hive command of ‘DESCRIBE FORMATTED MyDb.foodratingspart;’

Output: -



```
hive> DESCRIBE FORMATTED MyDb.foodratingspart;
OK
# col_name              data_type               comment

food1                   int
food2                   int
food3                   int
food4                   int
id                      int

# Partition Information
# col_name              data_type               comment

name                    string

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Thu Feb 10 03:06:24 UTC 2022
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-67-200.ec2.internal:8020/user/hive/warehouse/mydb.db/foodratingspart
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        numFiles                0
        numPartitions           0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1644462384

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        serialization.format    ,
Time taken: 0.146 seconds, Fetched: 41 row(s)
hive>
```

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

Exercise 5) 2 points

12) 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 a good choice for a partition field while using the place id is not.

Sol) We will remove column name so that it can increase the efficiency and to reduce the part size. Also, for faster processing having result in less than 10 columns is easier to create in comparison to 10,000 columns.

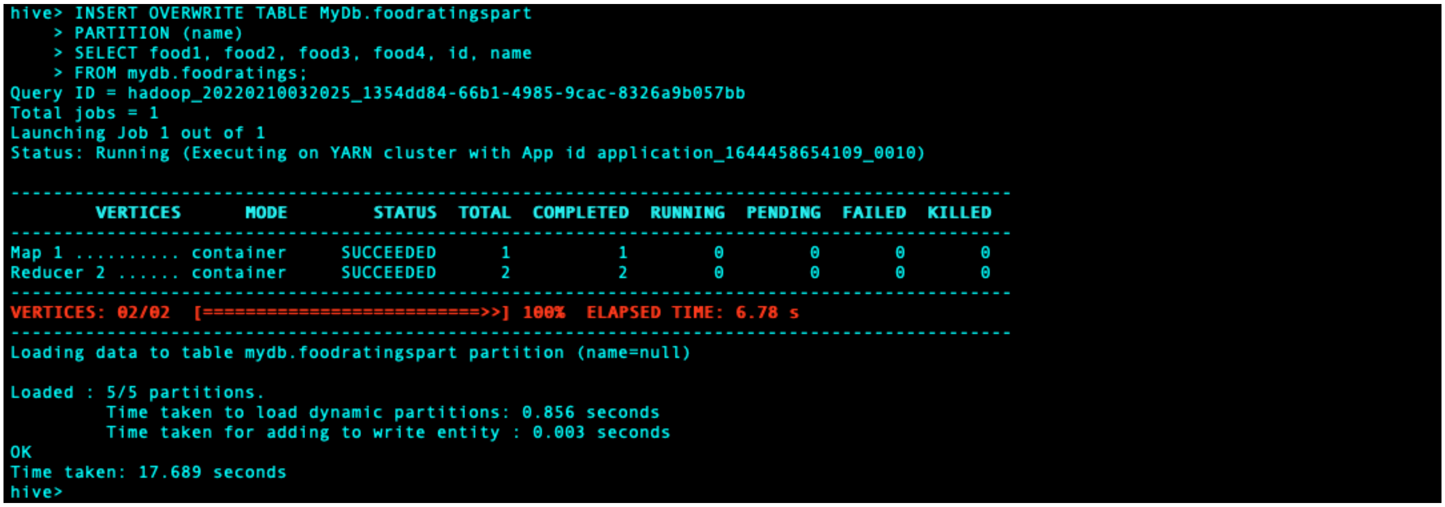
Exercise 6) 2 points

13) Configure Hive to allow dynamic partition creation as described in the lecture. Now, use a hive command to copy from MyDB.foodratings into MyDB.foodratingspart to create a partitioned table from a non-partitioned one.

Output: -

Using Command: -

INSERT OVERWRITE TABLE mydb.foodratingspart
PARTITION (name)
SELECT food1, food2, food3, food4, id, name
FROM mydb.foodratings;

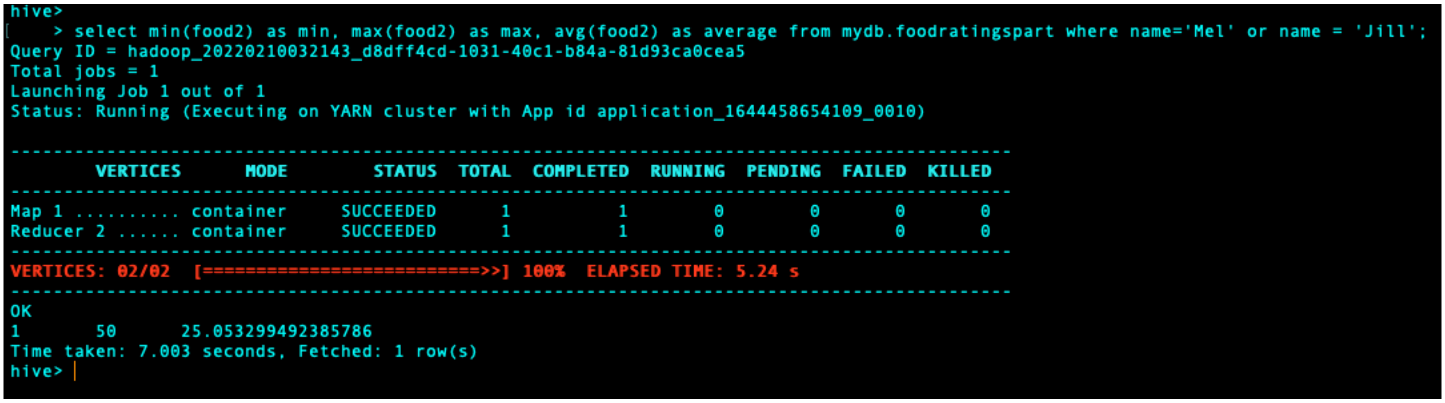


```
hive> INSERT OVERWRITE TABLE MyDb.foodratingspart
    > PARTITION (name)
    > SELECT food1, food2, food3, food4, id, name
    > FROM mydb.foodratings;
Query ID = hadoop_20220210032025_1354dd84-66b1-4985-9cac-8326a9b057bb
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1644458654109_0010)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 6.78 s
----------------------------------------------------------------------------------------------
Loading data to table mydb.foodratingspart partition (name=null)

Loaded : 5/5 partitions.
         Time taken to load dynamic partitions: 0.856 seconds
         Time taken for adding to write entity : 0.003 seconds
OK
Time taken: 17.689 seconds
hive>
```

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554

14) Execute a hive command to output the min, max and average of the values of the food2 column of MyDB.foodratingspart where the food critic 'name' is either Mel or Jill.

Output: -



```
hive>
[    > select min(food2) as min, max(food2) as max, avg(food2) as average from mydb.foodratingspart where name='Mel' or name = 'Jill';
Query ID = hadoop_20220210032143_d8dff4cd-1031-40c1-b84a-81d93ca0cea5
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1644458654109_0010)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5.24 s
----------------------------------------------------------------------------------------------
OK
1       50      25.053299492385786
Time taken: 7.003 seconds, Fetched: 1 row(s)
hive> |
```

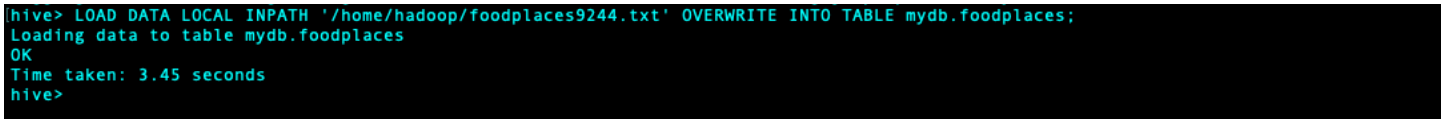
Exercise 7) 2 points

15) Load the foodplaces<.magic number>.txt file created using TestDataGen from your local file system into the foodplaces table.
Use a join operation between the two tables (foodratings and foodplaces) to provide the average rating for field food4 for the restaurant 'Soup Bowl'

Output: -

Using Command: -

LOAD DATA LOCAL INPATH '/home/hadoop/foodplaces9244.txt' OVERWRITE INTO TABLE MyDb.foodplaces;



```
[hive> LOAD DATA LOCAL INPATH '/home/hadoop/foodplaces9244.txt' OVERWRITE INTO TABLE mydb.foodplaces;
Loading data to table mydb.foodplaces
OK
Time taken: 3.45 seconds
hive>
```

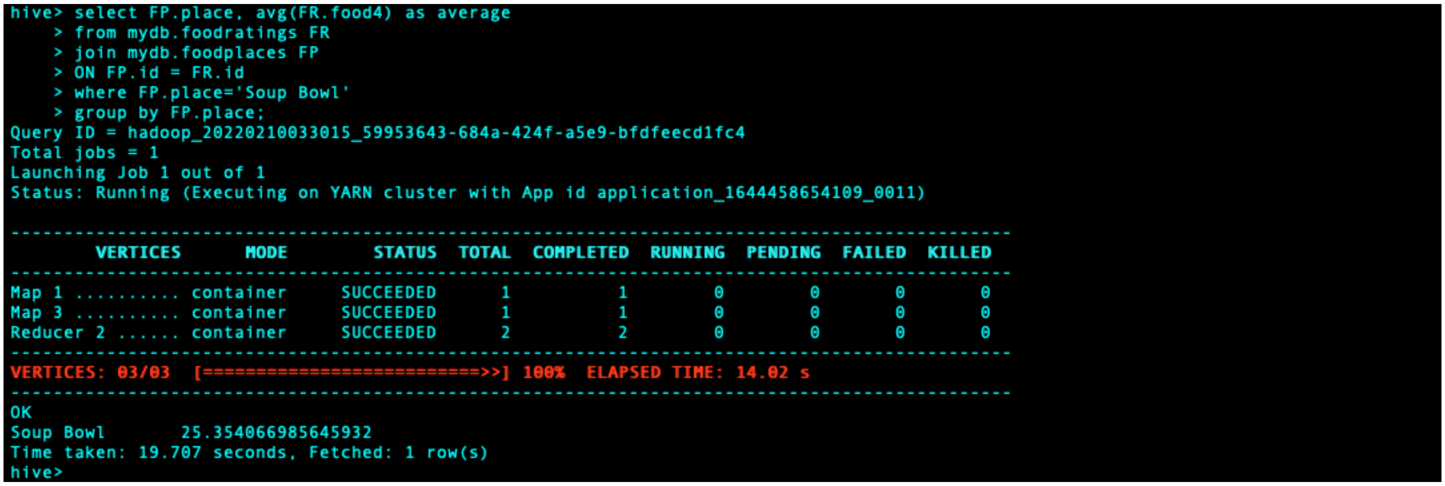
16) Use a join operation between the two tables (foodratings and foodplaces) to provide the average rating for field food4 for the restaurant 'Soup Bowl'

Output: -

Using Command: -

```
select FP.place, avg(FR.food4) as average
from mydb.foodratings FR
join mydb.foodplaces FP
ON FP.id = FR.id
where FP.place='Soup Bowl'
group by FP.place;
```

Name – Rishabh Jain
Mail – rjain35@hawk.iit.edu
Course – CSP 554



```
hive> select FP.place, avg(FR.food4) as average
    > from mydb.foodratings FR
    > join mydb.foodplaces FP
    > ON FP.id = FR.id
    > where FP.place='Soup Bowl'
    > group by FP.place;
Query ID = hadoop_20220210033015_59953643-684a-424f-a5e9-bfdfeecd1fc4
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1644458654109_0011)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 3 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 14.02 s
----------------------------------------------------------------------------------------------
OK
Soup Bowl       25.354066985645932
Time taken: 19.707 seconds, Fetched: 1 row(s)
hive>
```

Exercise 8) 4 points

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

Sol) The most important consideration when choosing a Row or Column Format: -

Column-based storage are useful when you are required to execute analytics queries which require a subset of columns examined over a large data set.

Row-based storage are better when our queries are required to access to all or most of the columns of each row of data.

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

Sol) Splittability can be defined as “Splitting of larger records into smaller records which can be handled independently”. A column-based format will be more amenable to splitting into separate jobs if the query calculation is concerned with a single column at a time. Spittability is important because it help in parallelization process.

c) What can files stored in column format achieve better compression than those stored in row format?

Sol) Columnar data can achieve better compression rate than row row-based data. Storing values by column, with the same data type next to each other, allows you to do more efficient compression on them, instead storing the data on row. For example, storing all dates together in memory will allow you more efficient compression than storing data of multiple types next to each other. Therefore, parquet and ORC file format achieve better compression.

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

Sol) We use Parquet column format, when we are having heavy workload with important factors like splittability, compression and schema evolution support. We also know that parquet file contains binary data organized by row group.