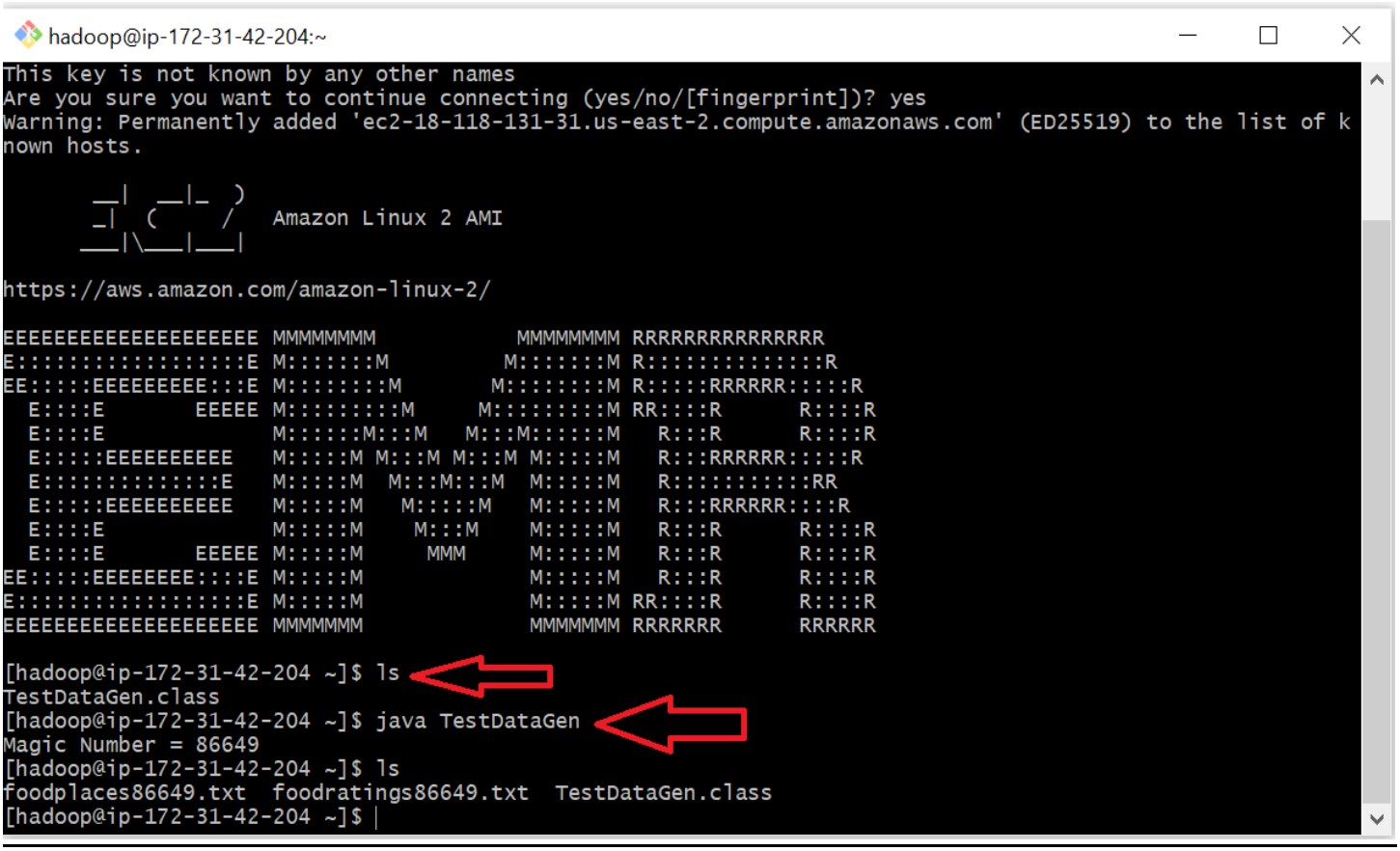
CSP – 554 Big Data Technologies

Assignment #4

Rutul Mehta
A20476293

- Copy “TestDataGen.class” file to the /home/Hadoop directory
- Run Java TestDataGen command to generate magic number.
- Magic number is 86649.
- This command also generates two separate files name foodratings86649.txt and foodplaces86649.txt files in /home/Hadoop directory.



Exercise 1

hadoop@ip-172-31-42-204:~

```
  E::::E              M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-42-204 ~]$ ls
TestDataGen.class
[hadoop@ip-172-31-42-204 ~]$ java TestDataGen
Magic Number = 86649
[hadoop@ip-172-31-42-204 ~]$ ls
foodplaces86649.txt  foodratings86649.txt  TestDataGen.class
[hadoop@ip-172-31-42-204 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> CREATE DATABASE MyDB
    >
    > show databases
    > CREATE DATABASE MyDB;
FAILED: ParseException line 3:0 missing EOF at 'show' near 'MyDB'
hive> CREATE DATABASE MyDB;
OK
Time taken: 0.368 seconds
hive> SHOW DATABASES;\
    > SHOW DATABASES;
OK
default
mydb
Time taken: 0.229 seconds, Fetched: 2 row(s)
FAILED: ParseException line 1:0 character '\' not supported here
hive> SHOW DATABASES;
OK
default
mydb
Time taken: 0.031 seconds, Fetched: 2 row(s)
hive>
```

hadoop@ip-172-31-42-204:~

```
        at org.apache.hadoop.hive.cli.CliDriver.executeDriver(CliDriver.java:821)
        at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:759)
        at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:686)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:244)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:158)
FAILED: ParseException line 3:9 mismatched input 'Food' expecting StringLiteral near 'COMMENT' in table's comment
hive> [hadoop@ip-172-31-42-204 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> CREATE TABLE IF NOT EXISTS MyDb.foodratings (name string COMMENT 'comment for names', food1 int COMMENT 'com
ment for food1',
    > food2 int COMMENT 'comment for food2', food3 int COMMENT 'comment for food3', food4 int COMMENT 'comment for
 food4', id int COMMENT 'comment for Id')
    > COMMENT "FoodRatings"
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE;
OK
Time taken: 1.297 seconds
hive> DESCRIBE FORMATTED MyDb.foodratings;
OK
# col_name              data_type               comment

name                    string                  comment for names
food1                   int                     comment for food1
food2                   int                     comment for food2
food3                   int                     comment for food3
food4                   int                     comment for food4
id                      int                     comment for Id

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Tue Sep 28 04:40:29 UTC 2021
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-42-204.us-east-2.compute.internal:8020/user/hive/warehouse/mydb.db/foodra
tings
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        comment                 FoodRatings
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1632804029

# 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             ,
        line.delim              \n
        serialization.format    ,
Time taken: 0.236 seconds, Fetched: 38 row(s)
hive>
```

```
hadoop@ip-172-31-42-204:~
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        comment                 FoodRatings
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1632804029

# 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             ,
        line.delim              \n
        serialization.format    ,
Time taken: 0.236 seconds, Fetched: 38 row(s)
hive> CREATE TABLE IF NOT EXISTS MyDb.foodplaces (id int COMMENT 'comment for id', place string COMMENT 'comment for place')
    > COMMENT "FoodPlaces"
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE;
OK
Time taken: 0.096 seconds
hive> DESCRIBE FORMATTED MyDb.foodplaces;
OK
# col_name              data_type               comment

id                      int                     comment for id
place                   string                  comment for place

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Tue Sep 28 04:48:24 UTC 2021
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-42-204.us-east-2.compute.internal:8020/user/hive/warehouse/mydb.db/foodplaces
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        comment                 FoodPlaces
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1632804504

# 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             ,
        line.delim              \n
        serialization.format    ,
Time taken: 0.082 seconds, Fetched: 34 row(s)
hive>
```

Exercise 2

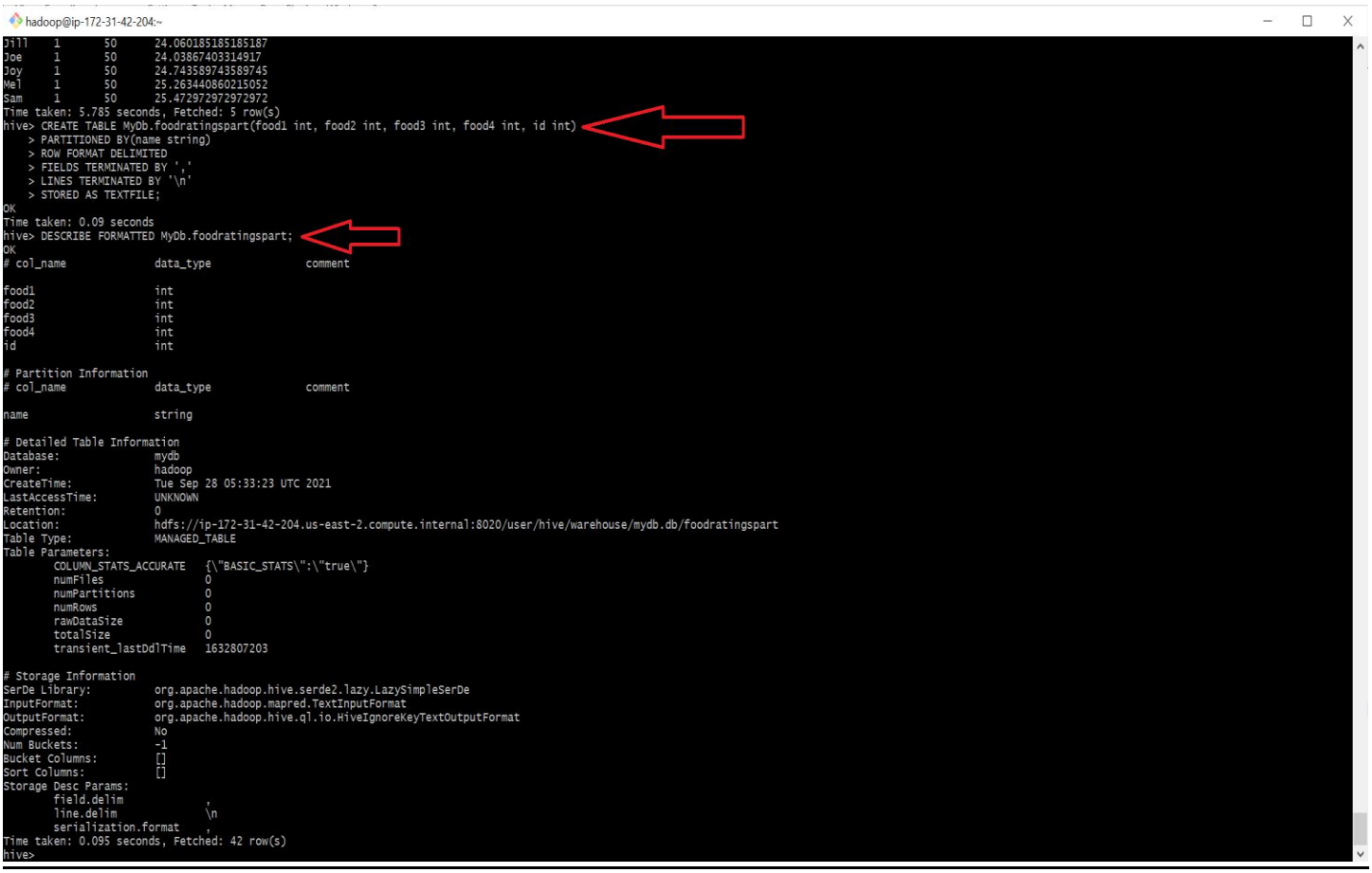
```
hadoop@ip-172-31-42-204:~
# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        line.delim              \n
        serialization.format    ,
Time taken: 0.082 seconds, Fetched: 34 row(s)
hive>  INTO TABLE
    > ;
NoViableAltException(153@[])
        at org.apache.hadoop.hive.ql.parse.HiveParser.statement(HiveParser.java:1300)
        at org.apache.hadoop.hive.ql.parse.ParseDriver.parse(ParseDriver.java:208)
        at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:77)
        at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:70)
        at org.apache.hadoop.hive.ql.Driver.compile(Driver.java:468)
        at org.apache.hadoop.hive.ql.Driver.compileInternal(Driver.java:1317)
        at org.apache.hadoop.hive.ql.Driver.runInternal(Driver.java:1457)
        at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1237)
        at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1227)
        at org.apache.hadoop.hive.cli.CliDriver.processLocalCmd(CliDriver.java:233)
        at org.apache.hadoop.hive.cli.CliDriver.processCmd(CliDriver.java:184)
        at org.apache.hadoop.hive.cli.CliDriver.processLine(CliDriver.java:403)
        at org.apache.hadoop.hive.cli.CliDriver.executeDriver(CliDriver.java:821)
        at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:759)
        at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:686)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:244)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:158)
FAILED: ParseException line 1:1 cannot recognize input near 'INTO' 'TABLE' '<EOF>'
hive> LOAD DATA LOCAL INPATH 'foodratings86649.txt' OVERWRITE INTO TABLE MyDb.foodratings;
Loading data to table mydb.foodratings
OK
Time taken: 0.727 seconds
hive> select * from MyDb.foodratings;
OK
Sam     7       11      15      22      1
Joy     30      16      21      40      3
Joe     16      14      43      2       4
Joy     48      30      16      34      5
Joe     23      3       15      3       2
Sam     10      19      33      19      1
Mel     41      49      12      35      1
Mel     32      35      5       38      5
Joe     2       11      8       32      5
Mel     7       14      7       31      1
Jill    7       39      6       49      3
Joy     30      12      8       4       5
Joy     34      30      24      49      4
Sam     29      12      50      10      2
Jill    39      8       34      28      1
Joe     41      28      11      36      3
Joy     36      18      4       3       5
Jill    8       28      33      13      5
Joy     48      44      19      36      2
Sam     36      1       10      12      4
Joy     22      15      41      31      4
Joe     26      37      2       3       3
Sam     23      22      39      50      1
```

```
hadoop@ip-172-31-42-204:~
Joe     44      37      22      8       5
Jill    22      37      24      46      3
Sam     37      30      47      49      5
Jill    28      24      7       35      5
Sam     10      7       25      45      4
Time taken: 3.392 seconds, Fetched: 1000 row(s)
hive> select
    >  max(ApproxValue) as MaxValue, avg(ApproxValue) as AvgValue, min(ApproxValue) as MinValue
    > from
    > ;
NoViableAltException(-1@[])
        at org.apache.hadoop.hive.ql.parse.HiveParser_FromClauseParser.fromSource(HiveParser_FromClauseParser.java:1612)
        at org.apache.hadoop.hive.ql.parse.HiveParser_FromClauseParser.fromClause(HiveParser_FromClauseParser.java:1312)
        at org.apache.hadoop.hive.ql.parse.HiveParser.fromClause(HiveParser.java:42048)
        at org.apache.hadoop.hive.ql.parse.HiveParser.atomSelectStatement(HiveParser.java:36735)
        at org.apache.hadoop.hive.ql.parse.HiveParser.selectStatement(HiveParser.java:36987)
        at org.apache.hadoop.hive.ql.parse.HiveParser.regularBody(HiveParser.java:36633)
        at org.apache.hadoop.hive.ql.parse.HiveParser.queryStatementExpressionBody(HiveParser.java:35822)
        at org.apache.hadoop.hive.ql.parse.HiveParser.queryStatementExpression(HiveParser.java:35710)
        at org.apache.hadoop.hive.ql.parse.HiveParser.execStatement(HiveParser.java:2284)
        at org.apache.hadoop.hive.ql.parse.HiveParser.statement(HiveParser.java:1333)
        at org.apache.hadoop.hive.ql.parse.ParseDriver.parse(ParseDriver.java:208)
        at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:77)
        at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:70)
        at org.apache.hadoop.hive.ql.Driver.compile(Driver.java:468)
        at org.apache.hadoop.hive.ql.Driver.compileInternal(Driver.java:1317)
        at org.apache.hadoop.hive.ql.Driver.runInternal(Driver.java:1457)
        at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1237)
        at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1227)
        at org.apache.hadoop.hive.cli.CliDriver.processLocalCmd(CliDriver.java:233)
        at org.apache.hadoop.hive.cli.CliDriver.processCmd(CliDriver.java:184)
        at org.apache.hadoop.hive.cli.CliDriver.processLine(CliDriver.java:403)
        at org.apache.hadoop.hive.cli.CliDriver.executeDriver(CliDriver.java:821)
        at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:759)
        at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:686)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:244)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:158)
FAILED: ParseException line 4:0 cannot recognize input near '<EOF>' '<EOF>' '<EOF>' in join source
hive> select min(food3) as MinValue, max(food3) as MaxValue, avg(food3) as AvgValue
    > from MyDb.foodrating
    > ;
FAILED: SemanticException [Error 10001]: Line 2:5 Table not found 'foodrating'
hive> select min(food3) as MinValue, max(food3) as MaxValue, avg(food3) as AvgValue
    > from MyDb.foodratings;
Query ID = hadoop_20210928051032_7c342004-d9d3-479e-aaaf-20e1244ca44c
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1632800415710_0003)

--------------------------------------------------------------------------------
        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: 6.16 s
--------------------------------------------------------------------------------
OK
1       50      26.168
Time taken: 14.845 seconds, Fetched: 1 row(s)
hive>
```

Exercise 3

```
hadoop@ip-172-31-42-204:~
Status: Running (Executing on YARN cluster with App id application_1632800415710_0003)

--------------------------------------------------------------------------------
        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: 6.16 s
--------------------------------------------------------------------------------
OK
1       50      26.168
Time taken: 14.845 seconds, Fetched: 1 row(s)
hive> select name, min(food3) as MinValue, max(food3) as MaxValue, avg(food3) as AvgValue
    > from MyDb.foodratings
    > group by name;
Query ID = hadoop_20210928051304_dd31b20c-e536-49f2-a63b-9a7d10bba02a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1632800415710_0003)

--------------------------------------------------------------------------------
        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: 5.93 s
--------------------------------------------------------------------------------
OK
Jill    1       50      25.90740740740741
Joe     1       50      25.425414364640883
Joy     1       50      26.358974358974358
Mel     1       50      25.586021505376344
Sam     1       50      27.346846846846848
Time taken: 6.721 seconds, Fetched: 5 row(s)
hive> select name, min(food1) as MinValue, max(food1) as MaxValue, avg(food1) as AvgValue
    > from MyDb.foodratings
    > group by name;
Query ID = hadoop_20210928051721_f50c4447-0ced-4a1f-b88d-6a3f70b4fbeb
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1632800415710_0003)

--------------------------------------------------------------------------------
        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: 5.10 s
--------------------------------------------------------------------------------
OK
Jill    1       50      24.060185185185187
Joe     1       50      24.03867403314917
Joy     1       50      24.743589743589745
Mel     1       50      25.263440860215052
Sam     1       50      25.472972972972972
Time taken: 5.785 seconds, Fetched: 5 row(s)
hive>
```

Exercise 4



Exercise 5

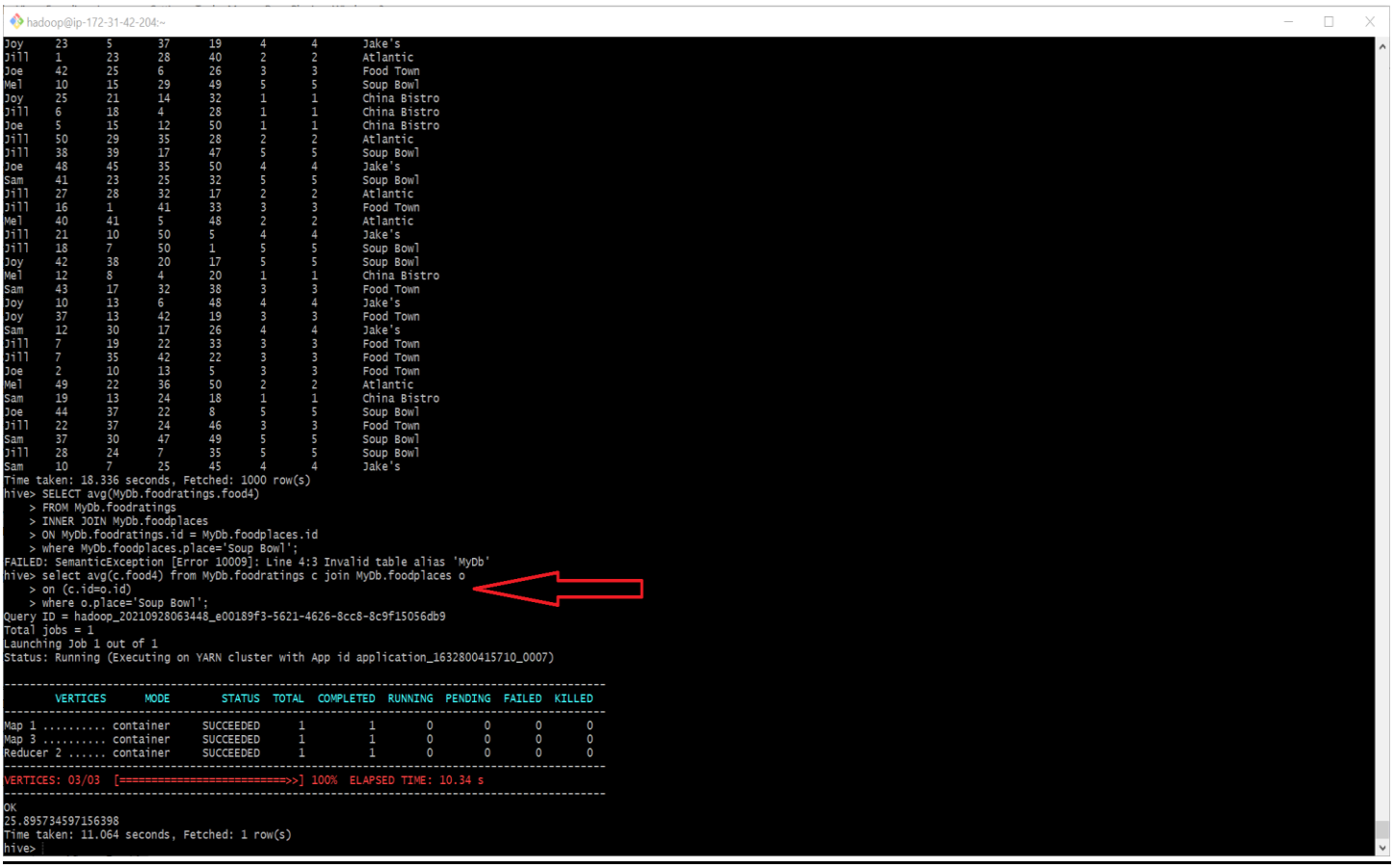
- Hive partitions are used to split the larger table into smaller parts just like partition tables in SQL server or any other RDBMS databases.
- Each partition table in the hive is identified by the partition key. It is very easy to do queries on the slices of the data (partition tables).
- But when we divide the table into so many partitions, it immensely affects the performance of the query executions. Now imagine having thousands of queries every day, scanning thousands of partitions per table.

Exercise 6

```
hadoop@ip-172-31-42-204:~
hive> INSERT INTO TABLE MyDb.foodratingspart PARTITION(name)
    > SELECT food1,food2,food3,food4,id,name FROM MyDb.foodratings;
Query ID = hadoop_20210928060734_85b157e2-b232-470d-a059-4f073057b9e3
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1632800415710_0005)

--------------------------------------------------------------------------------
        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: 6.24 s
--------------------------------------------------------------------------------
Loading data to table mydb.foodratingspart partition (name=null)

Loaded : 5/5 partitions.
        Time taken to load dynamic partitions: 0.684 seconds
        Time taken for adding to write entity : 0.001 seconds
OK
Time taken: 15.099 seconds
hive> select * from MyDb.foodratingspart;
OK
7       39      12      31      4       Jill
7       8       44      28      5       Jill
45      17      46      4       5       Jill
17      25      22      16      5       Jill
38      9       40      17      3       Jill
9       50      16      24      5       Jill
29      2       8       11      5       Jill
12      1       6       37      5       Jill
11      10      39      41      5       Jill
39      37      10      15      2       Jill
26      1       48      50      2       Jill
29      3       19      44      4       Jill
12      43      3       49      4       Jill
12      39      49      8       4       Jill
30      41      8       24      3       Jill
48      13      17      31      4       Jill
13      27      11      5       3       Jill
16      32      1       5       4       Jill
30      4       48      7       1       Jill
11      16      35      42      1       Jill
31      5       17      49      3       Jill
39      3       48      13      5       Jill
11      19      35      34      1       Jill
10      44      37      41      2       Jill
39      8       34      28      1       Jill
46      25      1       40      1       Jill
19      4       36      14      5       Jill
49      49      45      8       3       Jill
20      37      33      29      2       Jill
8       43      44      29      5       Jill
15      14      34      12      4       Jill
44      18      3       46      1       Jill
39      24      13      42      2       Jill
5       49      46      45      2       Jill
```

```
hadoop@ip-172-31-42-204:~
11      10      8       28      5       Sam
32      48      50      8       4       Sam
29      19      24      12      2       Sam
10      35      28      36      1       Sam
39      27      28      4       5       Sam
35      33      36      14      2       Sam
47      28      50      4       1       Sam
41      14      48      5       1       Sam
27      43      30      10      2       Sam
11      36      13      4       5       Sam
41      23      25      32      5       Sam
43      17      32      38      3       Sam
12      30      17      26      4       Sam
19      13      24      18      1       Sam
37      30      47      49      5       Sam
10      7       25      45      4       Sam
Time taken: 0.155 seconds, Fetched: 1000 row(s)
hive> select name, min(food2) as MinValue, max(food2) as MaxValue, avg(food2) as AvgValue
    > from MyDb.foodratingspart
    > WHERE name='Mel' or name='Jill'
    > group by name;
Query ID = hadoop_20210928061614_6b017c51-7e74-4908-9da3-b5c8b748d173
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1632800415710_0006)

--------------------------------------------------------------------------------
        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: 6.18 s
--------------------------------------------------------------------------------
OK
Jill    1       50      24.805555555555557
Mel     1       50      26.155913978494624
Time taken: 14.987 seconds, Fetched: 2 row(s)
hive> select min(food2) as MinValue, max(food2) as MaxValue, avg(food2) as AvgValue
    > from MyDb.foodratingspart
    > WHERE name='Mel' or name='Jill';
Query ID = hadoop_20210928062056_a18ea409-81f1-4661-bf39-d6ed768da690
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1632800415710_0006)

--------------------------------------------------------------------------------
        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: 5.76 s
--------------------------------------------------------------------------------
OK
1       50      25.430348258706466
Time taken: 6.737 seconds, Fetched: 1 row(s)
hive>
```

Exercise 7



Exercise 8

a)

Column-based storage is most useful when performing analytics queries that require only a subset of columns examined over very large datasets. While if your query require access to all or most of the columns of each row, row-based storage is well-suited.

For Example,

EXAMPLE: SAMPLE TRANSACTION DATA

Customer Name	Product ID	Sale Amount	Transaction Date
Emma	Prod 1	100.00	2018-04-02
Liam	Prod 2	79.99	2018-04-02
Noah	Prod 3	19.99	2018-04-01
Olivia	Prod 2	79.99	2018-04-03

Row-based format

In a database, this data would be stored by row, as follows

Emma, Prod1, 100.00, 2018-04-02;
Liam, Prod2, 79.99, 2018-04-02;
Noah, Prod3, 19.99, 2018-04-01;
Olivia, Prod2, 79.99, 2018-04-03;

- To process this data, a computer would read this data from left to right, starting at the first row and then reading each subsequent row.
- Storing data in this format is ideal when you need to access one or more entries and all or many columns for each entry.

Column-based format

Emma, Liam, Noah, Olivia;
Prod1, Prod2, Prod3, Prod2;
100.00, 79.99, 19.99, 79.99;
2018-04-02, 2018-04-02, 2018-04-01, 2018-04-03;

- Data is stored sequentially by column, from top to bottom not by row, left to right.
- Having data grouped by column makes it more efficient to easily focus computation on specific columns of data. Reading only relevant columns of data saves compute costs as irrelevant columns are ignored.

b)

- Splitability means breaking large chunk of data into smaller ones.
- Large-scale parallelization of processing is key to performance. Your choice of the format can critically affect the implementation of the parallelization.
- If the query calculation is concerned with a single column at a time, a column-based format is more suitable.

C)

- Data compression reduces the information and resources needed to store and transmit the data, eventually saving time and money.
- Column-based data can achieve better compression rates than row-based data. Storing values by column, with the same type to each other, allows you to do more efficient compression than row-based data.
- For example, storing same datatypes in memory provide more efficient compression than storing data of various types next to each other like row-based data.

d)

- Parquet is used especially when data is column-based and also when we want to analyze wide dataset having many columns.
- Each Parquet file contains binary data organized by “row group.” For each row group, the data values are organized by column. This is how we can achieve better compression and we can read heavy workloads.