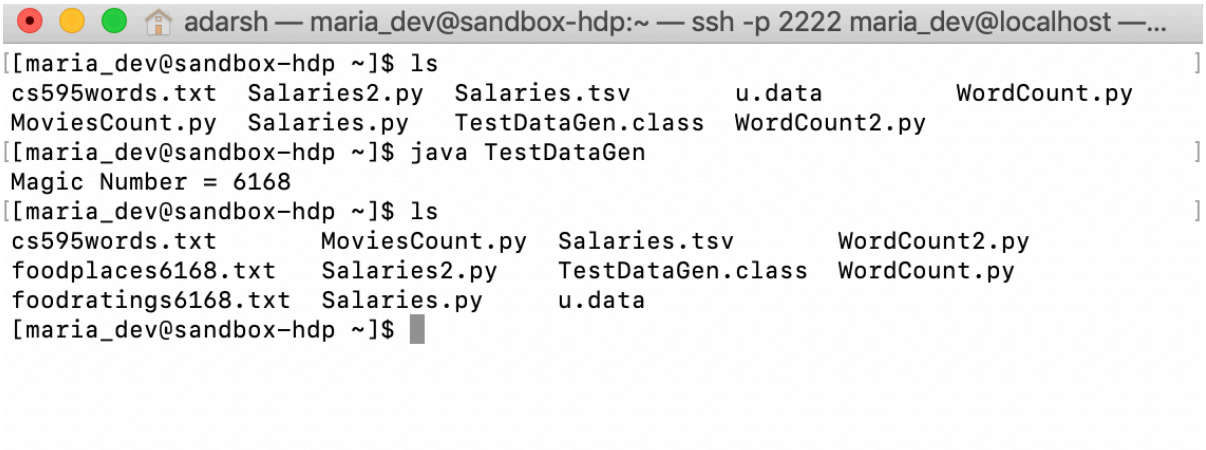
CSP 554 – Assignment #4

Name: Adarsh Mathad Vijayakumar
CWID: A20424847
Email ID: avijayakumar@hawk.iit.edu

Command: java TestDataGen
Output: Magic Number = 6168



Exercise 1:

1. CREATE DATABASE mydb;

2. CREATE DATABASE IF NOT EXISTS mydb;
   use mydb;
   DROP TABLE IF EXISTS foodratings;
   CREATE TABLE IF NOT EXISTS mydb.foodratings (
   name STRING COMMENT 'Food critic name',
   food1 INT COMMENT 'Food1 rating',
   food2 INT COMMENT 'Food2 rating',
   food3 INT COMMENT 'Food3 rating',
   food4 INT COMMENT 'Food4 rating',
   id INT COMMENT 'Restaurant ID')
   COMMENT 'Description of the table foodratings'
   ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
   STORED AS TEXTFILE;

3. DESCRIBE FORMATTED MyDb.foodratings;

   Output:

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

name                    string                  Food critic name
food1                   int                     Food1 rating
food2                   int                     Food2 rating
food3                   int                     Food3 rating
food4                   int                     Food4 rating
id                      int                     Restaurant ID

# Detailed Table Information
Database:               mydb
Owner:                  maria_dev
CreateTime:             Tue Feb 19 04:18:48 UTC 2019
LastAccessTime:         UNKNOWN
Protect Mode:           None
Retention:              0
Location:               hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/mydb.db/foodratings
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        comment                 Description of the table foodratings
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1550549928

# 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: 0.636 seconds, Fetched: 38 row(s)
hive>
```

4. CREATE DATABASE IF NOT EXISTS mydb;
   use mydb;
   DROP TABLE IF EXISTS foodplaces;
   CREATE TABLE IF NOT EXISTS mydb.foodplaces (
   id INT,
   place STRING)
   ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
   STORED AS TEXTFILE;

5. 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:                  maria_dev
CreateTime:             Tue Feb 19 04:35:40 UTC 2019
LastAccessTime:         UNKNOWN
Protect Mode:           None
Retention:              0
Location:               hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/mydb.db/foodplaces
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        numFiles                0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1550550940

# 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: 0.625 seconds, Fetched: 33 row(s)
hive>
```

Exercise 2:

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

LOAD DATA LOCAL INPATH './foodratings6168.txt' OVERWRITE INTO TABLE mydb.foodratings;

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

Magic Number = 6168

Hive command:

SELECT MIN(food3) AS Minimum, MAX(food3) AS Maximum, AVG(food3) AS Average from foodratings;

Output:

Minimum 1, Maximum 50, Average 25.801

Output Screenshot:

```
hive> LOAD DATA LOCAL INPATH './foodratings6168.txt' OVERWRITE INTO TABLE mydb.foodratings;
Loading data to table mydb.foodratings
Table mydb.foodratings stats: [numFiles=1, numRows=0, totalSize=17488, rawDataSize=0]
OK
Time taken: 1.225 seconds
hive> SELECT MIN(food3) AS Minimum, MAX(food3) AS Maximum, AVG(food3) AS Average from foodratings;
Query ID = maria_dev_20190219045025_858c7183-6535-4152-ad0a-5c327fc0ffcc
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_1550548008529_0002)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5.27 s
--------------------------------------------------------------------------------
OK
1       50      25.801
Time taken: 13.673 seconds, Fetched: 1 row(s)
hive>
```

Exercise 3:

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

SELECT name, MIN(food1) as Minimum, MAX(food1) as Maximum, AVG(food1) as Average FROM foodratings GROUP BY name;

[Magic Number: 6168]

Output:

```
hive> SELECT name, MIN(food1) as Minimum, MAX(food1) as Maximum, AVG(food1) as Average FROM foodratings GROUP BY name;
Query ID = maria_dev_20190219045944_0c55eef4-2bf1-4d19-9c71-5e03d2fa28b7
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1550548008529_0002)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 4.98 s
--------------------------------------------------------------------------------
OK
Jill    1       50      24.953051643192488
Joe     1       50      26.018691588785046
Joy     1       50      26.197860962566846
Mel     1       50      25.229508196721312
Sam     1       50      24.788177339901477
Time taken: 6.08 seconds, Fetched: 5 row(s)
hive>
```

Exercise 4:

In MyDb create a partitioned table called 'foodratingspart'

```
CREATE DATABASE IF NOT EXISTS mydb;
use mydb;
DROP TABLE IF EXISTS foodratingspart;
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;
```

Execute a one shot Hive command of 'DESCRIBE FORMATTED MyDb.foodratingspart' and capture its output as the result of this exercise.

```
DESCRIBE FORMATTED mydb.foodratingspart;
```

```
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:                  maria_dev
CreateTime:             Tue Feb 19 05:24:11 UTC 2019
LastAccessTime:         UNKNOWN
Protect Mode:           None
Retention:              0
Location:               hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/mydb.db/foodratingspart
Table Type:             MANAGED_TABLE
Table Parameters:
        transient_lastDdlTime   1550553851

# 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: 0.564 seconds, Fetched: 36 row(s)
hive>
```

Exercise 5:

Use a hive command to copy from MyDB.foodratings into MyDB.foodratingspart to create a partitioned table from a non-partitioned one.

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

Provide a copy of the command you use to load the 'foodratingspart' table as a result of this exercise.

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

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.

```
SELECT MIN(food2) as Minimum, MAX(food2) as Maximum, AVG(food2) as Average FROM foodratingspart WHERE name="Mel" OR name="Jill";
```

Output:

```
hive> SELECT MIN(food2) as Minimum, MAX(food2) as Maximum, AVG(food2) as Average FROM foodratingspart WHERE name="Mel" OR name="Jill";
Query ID = maria_dev_20190219054035_93e01cd4-cbd3-459a-919d-84d899e4ea84
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1550548008529_0003)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5.09 s
--------------------------------------------------------------------------------
OK
1       50      25.474747474747474
Time taken: 6.473 seconds, Fetched: 1 row(s)
hive>
```

Exercise 6:

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

```
LOAD DATA LOCAL INPATH './foodplaces6168.txt' OVERWRITE INTO TABLE mydb.foodplaces;
```

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

```
SELECT fp.place, AVG(fr.food4)
FROM foodratings fr JOIN foodplaces fp ON fr.id = fp.id
WHERE fp.place = 'Soup Bowl'
GROUP BY fp.place;
```

Output:

```
hive> SELECT fp.place, AVG(fr.food4)
    > FROM foodratings fr JOIN foodplaces fp ON fr.id = fp.id
    > WHERE fp.place = 'Soup Bowl'
    > GROUP BY fp.place;
Query ID = maria_dev_20190219060418_772bed3b-c19d-4591-9961-4ee4a8aea922
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_1550548008529_0004)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 3 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 8.41 s
--------------------------------------------------------------------------------
OK
Soup Bowl       25.727272727272727
Time taken: 15.753 seconds, Fetched: 1 row(s)
hive>
```

Exercise 7: (Extra Credit)
Write a half page summary of the following article on the blackboard in section "Articles:"

Pig Latin: A Not-So-Foreign Language for Data Processing

The authors described a new data processing environment being deployed at Yahoo! called Pig, and its associated language, Pig Latin. Pig's target demographic is experienced procedural programmers who prefer map-reduce style programming over the more declarative, SQL-style programming, for stylistic reasons as well as the ability to control the execution plan. Pig aims for a sweet spot between these two extremes, offering high-level data manipulation primitives such as projection and join, but in a much less declarative style than SQL, such as projection.

The author also described a novel debugging environment we are developing for Pig, called Pig Pen. In conjunction with the step-by-step nature of our Pig Latin language, Pig Pen makes it easy and fast for users to construct and debug their programs in an incremental fashion. The author says user can write a prefix of their overall program, examine the output on the sandbox data set, and iterate until the output matches what they intended.

While Pig Pen is still in early stages of development, the core Pig system is fully implemented and available as an open-source Apache incubator project. The Pig system compiles Pig Latin expressions into a sequence of map-reduce jobs, and orchestrates the execution of these jobs on Hadoop, an open-source scalable map-reduce implementation. Pig has an active and growing user base inside Yahoo!, and with their recent open-source release they are beginning to attract users in the broader community.