Akash Tanwani A20448831

CSP554—Big Data Technologies

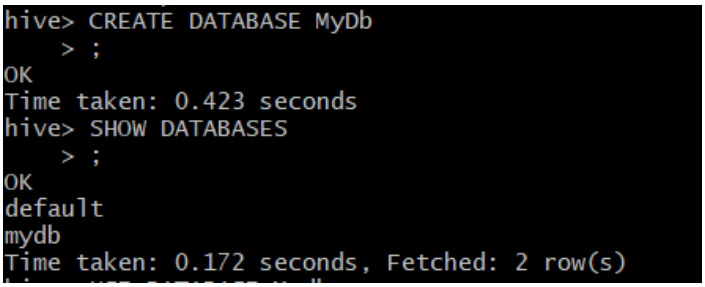
Assignment #4

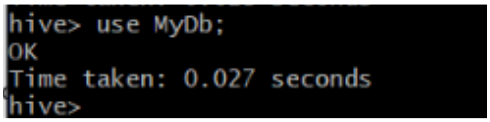
Exercise 1) 2 points

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';
```

```
hive> CREATE TABLE IF NOT EXISTS MyDb.foodratings(
    > name STRING COMMENT 'Critic Name',
    > food1 INT COMMENT 'Food Item1',
    > food2 INT COMMENT 'Food Item2',
    > food3 INT COMMENT 'Food Item3',
    > food4 INT COMMENT 'Food Item4',
    > ID INT COMMENT 'Restaurant ID')
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE
    > LOCATION '/home/hadoop';
OK
Time taken: 0.433 seconds
hive> DESCRIBE FORMATTED MYDB.foodratings;
OK
# col_name              data_type               comment

name                    string                  Critic Name
food1                   int                     Food Item1
food2                   int                     Food Item2
food3                   int                     Food Item3
food4                   int                     Food Item4
id                      int                     Restaurant ID

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Tue Sep 22 04:02:15 UTC 2020
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-87-61.ec2.internal:8020/home/hadoop
Table Type:             MANAGED_TABLE
Table Parameters:
        transient_lastDdlTime   1600747335

# 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: 0.174 seconds, Fetched: 31 row(s)
```

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’;
```

```
hive> CREATE TABLE IF NOT EXISTS MyDb.foodplaces(
    > ID INT,
    > Place STRING)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE
    > LOCATION '/home/hadoop';
OK
Time taken: 0.074 seconds
hive> DESCRIBE FORMATTED MyDb.foodplaces;
OK
# col_name              data_type               comment

id                      int
place                   string

# Detailed Table Information
Database:               mydb
Owner:                  hadoop
CreateTime:             Tue Sep 22 05:35:28 UTC 2020
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-87-61.ec2.internal:8020/home/hadoop
Table Type:             MANAGED_TABLE
Table Parameters:
        transient_lastDdlTime   1600752928

# 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: 0.046 seconds, Fetched: 27 row(s)
```

Exercise 2) 2 points

Loading Data:

LOAD DATA LOCAL INPATH '/home/hadoop/foodratings60618.txt'
OVERWRITE INTO TABLE foodratings;

```
hive> LOAD DATA LOCAL INPATH '/home/hadoop/foodratings60618.txt'
    > OVERWRITE INTO TABLE foodratings;
Loading data to table mydb.foodratings
OK
Time taken: 1.096 seconds
```

SELECT MIN(food3) AS MINIMUM, MAX(food3) AS MAXIMUM, AVG(food3) AS
AVERAGE FROM foodratings;

```
hive> select MIN(food3) AS Minimum, MAX(food3) as Maximum, AVG(food3) as Average from foodratings;
Query ID = hadoop_20200922044104_28c4e876-c34e-4dad-8f0e-3f7e17e0765a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1600743238725_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: 5.23 s
--------------------------------------------------------------------------------
OK
1       50      25.988
Time taken: 5.922 seconds, Fetched: 1 row(s)
```

Exercise 3) 2 points

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

```
hive> select name, MIN(food1), MAX(food1), AVG(food1) from foodratings group by name;
Query ID = hadoop_20200922044807_c15bb72c-67a6-4958-9454-7ef7bfb874b1
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_1600743238725_0004)

--------------------------------------------------------------------------------
        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: 6.93 s
--------------------------------------------------------------------------------
OK
Jill    1       50      25.072727272727274
Joe     1       50      25.61627906976744
Joy     1       50      24.28021978021978
Mel     1       50      26.666666666666668
Sam     1       50      24.896103896103895
Time taken: 15.494 seconds, Fetched: 5 row(s)
```

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’;
```

```
hive> CREATE TABLE 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
    > LOCATION '/home/hadoop';
OK
Time taken: 0.06 seconds
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:             Tue Sep 22 04:56:12 UTC 2020
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://ip-172-31-87-61.ec2.internal:8020/home/hadoop
Table Type:             MANAGED_TABLE
Table Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        numFiles                0
        numPartitions           0
        numRows                 0
        rawDataSize             0
        totalSize               0
        transient_lastDdlTime   1600750572

# 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: 0.079 seconds, Fetched: 41 row(s)
```

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.

Akash Tanwani **A20448831**

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;

```
hive> INSERT OVERWRITE TABLE MyDb.foodratingspart
    > PARTITION (name) SELECT food1, food2, food3, food4, ID, name from foodratings;
Query ID = hadoop_20200922050654_337ab940-3719-4e5d-b1ff-f45e7a1dba3d
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_1600743238725_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: 7.24 s
--------------------------------------------------------------------------------
Loading data to table mydb.foodratingspart partition (name=null)

Loaded : 5/5 partitions.
         Time taken to load dynamic partitions: 0.446 seconds
         Time taken for adding to write entity : 0.002 seconds
OK
Time taken: 16.595 seconds
```

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

```
hive> SELECT MIN(food2), MAX(food2), AVG(food2) from foodratingspart where name='Mel' or name='Jill';
Query ID = hadoop_20200922051219_76ce352c-17ec-480a-a5dc-2b42ad63e6f8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1600743238725_0005)

--------------------------------------------------------------------------------
        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: 5.58 s
--------------------------------------------------------------------------------
OK
1       50      25.219277108433737
Time taken: 6.734 seconds, Fetched: 1 row(s)
```

Akash Tanwani **A20448831**

Exercise 7) 2 points

LOAD DATA LOCAL INPATH '/home/hadoop/foodplaces60618.txt'
OVERWRITE INTO TABLE MyDb.foodplaces;

```
hive> LOAD DATA LOCAL INPATH '/home/hadoop/foodplaces60618.txt'
    > OVERWRITE INTO TABLE MyDb.foodplaces;
Loading data to table mydb.foodplaces
OK
Time taken: 0.514 seconds
```

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;

```
hive> 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;
Query ID = hadoop_20200922054430_672f0929-20e9-418f-820e-88dd41255237
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1600743238725_0006)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1            container     SUCCEEDED      0          0        0        0       0       0
Map 3            container     SUCCEEDED      0          0        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/03  [==========================>>] 100%  ELAPSED TIME: 7.99 s
--------------------------------------------------------------------------------
OK
Time taken: 8.706 seconds
```

Verification:

SELECT * FROM MyDb.foodplaces where place = 'Super Bowl';
SELECT AVG(food4) from foodratings where id = 5;
SELECT food4 FROM foodratinngs WHERE ID = 5;

```
hive> select * from mydb.foodplaces where place='Soup Bowl';
OK
5       Soup Bowl
Time taken: 0.335 seconds, Fetched: 1 row(s)
hive> select AVG(food4) from foodratings where id=5
    > ;
Query ID = hadoop_20200922060843_71dfeb8c-6b73-4224-9d25-6412704e02b5
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_1600743238725_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: 6.63 s
--------------------------------------------------------------------------------
OK
NULL
Time taken: 14.12 seconds, Fetched: 1 row(s)
hive> select food4 from foodratings where id=5;
OK
Time taken: 0.209 seconds
```

Akash Tanwani **A20448831**

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:

a) 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 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?

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.

c) 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.

d) 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.