CSP554—Big Data Technologies

Prachi Kotadia (A20549927)

Assignment #4

Exercise 1) 2 points

Create a Hive database called “MyDb”.

Execute a Hive command of ‘DESCRIBE FORMATTED MyDb.foodratings;’ and capture its output as one of the results of this exercise.

```
[hadoop@ip-172-31-57-126 ~]$ java TestDataGen
Magic Number = 168082
```

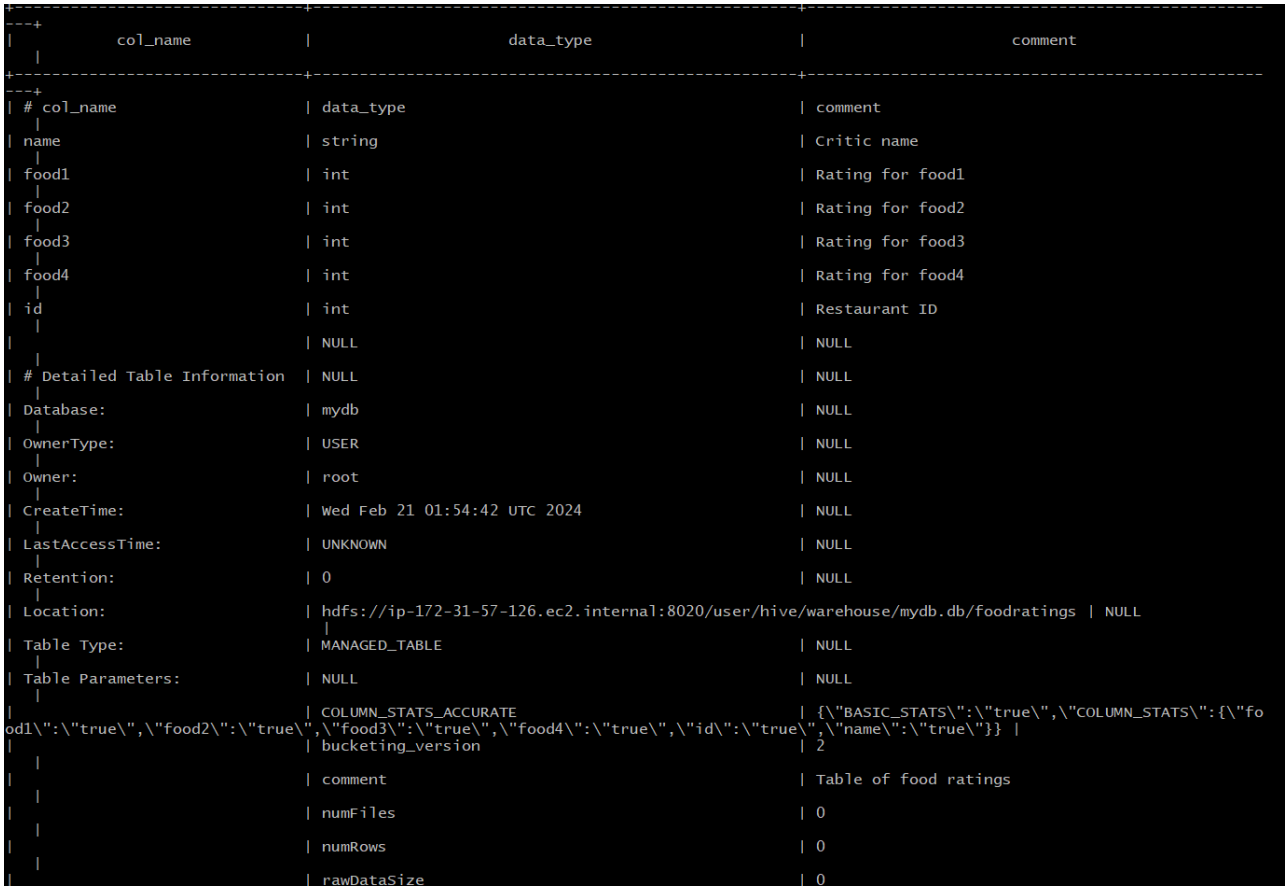
```
0: jdbc:hive2://localhost:10000> CREATE DATABASE MyDb;
```

```
0: jdbc:hive2://localhost:10000> USE MyDb;
```

```
0: jdbc:hive2://localhost:10000> CREATE TABLE MyDb.foodratings (
. . . . . .>   name STRING COMMENT 'Critic name',
. . . . . .>   food1 INT COMMENT 'Rating for food1',
. . . . . .>   food2 INT COMMENT 'Rating for food2',
. . . . . .>   food3 INT COMMENT 'Rating for food3',
. . . . . .>   food4 INT COMMENT 'Rating for food4',
. . . . . .>   id INT COMMENT 'Restaurant ID'
. . . . . .> )
. . . . . .> COMMENT 'Table of food ratings'
. . . . . .> ROW FORMAT DELIMITED
. . . . . .> FIELDS TERMINATED BY ','
. . . . . .> STORED AS TEXTFILE;
```

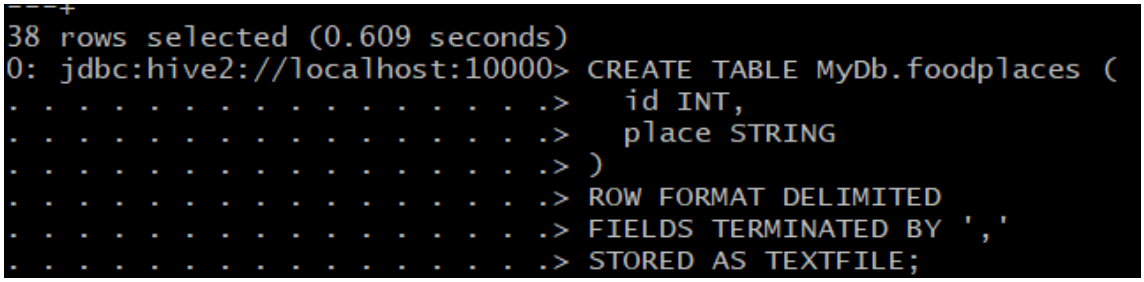
```
0: jdbc:hive2://localhost:10000> DESCRIBE FORMATTED MyDb.foodratings;
```

col_name	data_type	comment
# col_name	data_type	comment
name	string	Critic name
food1	int	Rating for food1
food2	int	Rating for food2
food3	int	Rating for food3
food4	int	Rating for food4
id	int	Restaurant ID



col_name	data_type	comment
# col_name	data_type	comment
name	string	Critic name
food1	int	Rating for food1
food2	int	Rating for food2
food3	int	Rating for food3
food4	int	Rating for food4
id	int	Restaurant ID
	NULL	NULL
# Detailed Table Information	NULL	NULL
Database:	mydb	NULL
OwnerType:	USER	NULL
Owner:	root	NULL
CreateTime:	Wed Feb 21 01:54:42 UTC 2024	NULL
LastAccessTime:	UNKNOWN	NULL
Retention:	0	NULL
Location:	hdfs://ip-172-31-57-126.ec2.internal:8020/user/hive/warehouse/mydb.db/foodratings	NULL
Table Type:	MANAGED_TABLE	NULL
Table Parameters:	NULL	NULL
	COLUMN_STATS_ACCURATE	{\"BASIC_STATS\":\"true\",\"COLUMN_STATS\":{\"food1\":\"true\",\"food2\":\"true\",\"food3\":\"true\",\"food4\":\"true\",\"id\":\"true\",\"name\":\"true\"}}
	bucketing_version	2
	comment	Table of food ratings
	numFiles	0
	numRows	0
	rawDataSize	0

Execute a Hive command of 'DESCRIBE FORMATTED MyDb.foodplaces' and capture its output as another of the results of this exercise



```
38 rows selected (0.609 seconds)
0: jdbc:hive2://localhost:10000> CREATE TABLE MyDb.foodplaces (
. . . . . .>   id INT,
. . . . . .>   place STRING
. . . . . .> )
. . . . . .> ROW FORMAT DELIMITED
. . . . . .> FIELDS TERMINATED BY ','
. . . . . .> STORED AS TEXTFILE;
```

```
0: jdbc:hive2://localhost:10000> DESCRIBE FORMATTED MyDb.foodplaces;
```

```
+-------------------------------+----------------------------------------------------+----------------------------------------------------+
|           col_name            |                     data_type                      |                      comment                       |
+-------------------------------+----------------------------------------------------+----------------------------------------------------+
| # col_name                    | data_type                                          | comment                                            |
| id                            | int                                                |                                                    |
| place                         | string                                             |                                                    |
|                               | NULL                                               | NULL                                               |
| # Detailed Table Information  | NULL                                               | NULL                                               |
| Database:                     | mydb                                               | NULL                                               |
| OwnerType:                    | USER                                               | NULL                                               |
| Owner:                        | root                                               | NULL                                               |
| CreateTime:                   | Wed Feb 21 01:57:14 UTC 2024                       | NULL                                               |
| LastAccessTime:               | UNKNOWN                                            | NULL                                               |
| Retention:                    | 0                                                  | NULL                                               |
| Location:                     | hdfs://ip-172-31-57-126.ec2.internal:8020/user/hive/warehouse/mydb.db/foodplaces | NULL                                               |
| Table Type:                   | MANAGED_TABLE                                      | NULL                                               |
| Table Parameters:             | NULL                                               | NULL                                               |
|                               | COLUMN_STATS_ACCURATE                              | {\"BASIC_STATS\":\"true\",\"COLUMN_STATS\":{\"id\":\"true\
|                               | bucketing_version                                  | 2                                                  |
|                               | numFiles                                           | 0                                                  |
|                               | numRows                                            | 0                                                  |
|                               | rawDataSize                                        | 0                                                  |
|                               | totalSize                                          | 0                                                  |
|                               | transient_lastDdlTime                              | 1708480634                                         |
|                               | NULL                                               | NULL                                               |
| # Storage Information         | NULL                                               | NULL                                               |
| SerDe Library:                | org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe | NULL                                               |
| InputFormat:                  | org.apache.hadoop.mapred.TextInputFormat           | NULL                                               |
| OutputFormat:                 | org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat | NULL                                               |
| Compressed:                   | No                                                 | NULL                                               |
| Num Buckets:                  | -1                                                 | NULL                                               |
| Bucket Columns:               | []                                                 | NULL                                               |
| Sort Columns:                 | []                                                 | NULL                                               |
| Storage Desc Params:          | NULL                                               | NULL                                               |
|                               | field.delim                                        | ,                                                  |
|                               | serialization.format                               | ,                                                  |
+-------------------------------+----------------------------------------------------+----------------------------------------------------+
33 rows selected (0.099 seconds)
```

Exercise 2) 2 points

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

Execute a hive command to output the min, max and average of the values of the food3 column of the foodratings table. This should be one hive command, not three separate ones.

```
0: jdbc:hive2://localhost:10000> LOAD DATA LOCAL INPATH '/home/hadoop/foodratings168082.txt' INTO TABLE MyDb.foodratings;
```

```
0: jdbc:hive2://localhost:10000> SELECT MIN(food3) AS Min_Food3, MAX(food3) AS Max_Food3, AVG(food3) AS Avg_Food3 FROM MyDb.foodratings;
```

```
+-------------+-------------+-------------+
| min_food3   | max_food3   | avg_food3   |
+-------------+-------------+-------------+
| 1           | 50          | 25.006      |
+-------------+-------------+-------------+
1 row selected (27.717 seconds)
```

Exercise 3) 2 points

Execute a hive command to output the min, max and average of the values of the food1 column grouped by the first column ‘name’. This should be one hive command, not three separate ones.

```
SELECT name, MIN(food1) AS Min_Food1, MAX(food1) AS Max_Food1, AVG(food1) AS Avg_Food1 FROM MyDb.foodratings GROUP BY name;
```

```
+-------+------------+------------+---------------------+
| name  | min_food1  | max_food1  |      avg_food1      |
+-------+------------+------------+---------------------+
| Joy   | 1          | 50         | 28.292035398230087  |
| Jill  | 1          | 50         | 25.120218579234972  |
| Joe   | 1          | 50         | 25.734299516908212  |
| Mel   | 1          | 50         | 24.926108374384235  |
| Sam   | 1          | 50         | 25.58011049723757   |
+-------+------------+------------+---------------------+
5 rows selected (6.22 seconds)
```

Exercise 4) 2 Points
In MyDb create a partitioned table called ‘foodratingspart’

The partition field should be called ‘name’ and its type should be a string. The names of the non-partition columns should be food1, food2, food3, food4 and id and their types each an integer. The table should have storage format TEXTFILE and column separator a “,”. That is the underlying format should be a CSV file. No comments are needed for this table.

Execute a Hive command of ‘DESCRIBE FORMATTED MyDb.foodratingspart;’ and capture its output as the result of this exercise.

```
0: jdbc:hive2://localhost:10000> 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;
```

```
0: jdbc:hive2://localhost:10000> DESCRIBE FORMATTED MyDb.foodratingspart;
```

col_name	data_type	comment
# col_name	data_type	comment
food1	int	
food2	int	
food3	int	
food4	int	
id	int	
	NULL	NULL
# Partition Information	NULL	NULL
# col_name	data_type	comment
name	string	
	NULL	NULL
# Detailed Table Information	NULL	NULL
Database:	mydb	NULL
OwnerType:	USER	NULL
Owner:	root	NULL
CreateTime:	Wed Feb 21 02:18:09 UTC 2024	NULL
LastAccessTime:	UNKNOWN	NULL
Retention:	0	NULL
Location:	hdfs://ip-172-31-57-126.ec2.internal:8020/user/hive/warehouse/mydb.db/foodratingspart	
Table Type:	MANAGED_TABLE	NULL
Table Parameters:	NULL	NULL
	COLUMN_STATS_ACCURATE	{\"BASIC_STATS\":\"true\"}
	bucketing_version	2
	numFiles	0
	numPartitions	0
	numRows	0
	rawDataSize	0
	totalSize	0
	transient_lastDdlTime	1708481889
	NULL	NULL
# Storage Information	NULL	NULL
SerDe Library:	org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe	NULL
InputFormat:	org.apache.hadoop.mapred.TextInputFormat	NULL
OutputFormat:	org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat	NULL
Compressed:	No	NULL
Num Buckets:	-1	NULL
Bucket Columns:	[]	NULL
Sort Columns:	[]	NULL
Storage Desc Params:	NULL	NULL
	field.delim	,
	serialization.format	,

41 rows selected (0.112 seconds)

Exercise 5) 2 points

Assume that the number of food critics is relatively small, say less than 10 and the number of 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.

A. It is better to utilize the critic's name as a partition field when there are few reviewers because it reduces the number of partitions, which is useful for managing and retrieving data. However, because there are so many locations, employing the place ID would result in a lot of partitions, which could cause performance problems because it can be difficult to manage and query such a large number of partitions, which would slow down query processing.

Exercise 6) 2 points

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.

```
0: jdbc:hive2://localhost:10000> SET hive.exec.dynamic.partition = true;
No rows affected (0.014 seconds)
0: jdbc:hive2://localhost:10000> SET hive.exec.dynamic.partition.mode = nonstrict;
No rows affected (0.007 seconds)
0: jdbc:hive2://localhost:10000> INSERT OVERWRITE TABLE MyDb.foodratingspart PARTITION (name)
. . . . . .> SELECT food1, food2, food3, food4, id, name FROM MyDb.foodratings;
```

```
0: jdbc:hive2://localhost:10000> SELECT MIN(food2) AS Min_Food2, MAX(food2) AS Max_Food2, AVG(food2) AS Avg_Food2
. . . . . .> FROM MyDb.foodratingspart
. . . . . .> WHERE name IN ('Mel', 'Jill');
```

```
+------------+------------+--------------------+
| min_food2  | max_food2  |      avg_food2     |
+------------+------------+--------------------+
| 1          | 50         | 25.901554404145077 |
+------------+------------+--------------------+
1 row selected (5.655 seconds)
```

Exercise 7) 2 points

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'

```
LOAD DATA LOCAL INPATH '/home/hadoop/foodplaces168082.txt' INTO TABLE MyDb.foodratings;
```

```
0: jdbc:hive2://localhost:10000> SELECT fp.place, AVG(fr.food4) AS Avg_Rating
. . . . . .> FROM MyDb.foodratings fr
. . . . . .> JOIN MyDb.foodplaces fp ON (fr.id = fp.id)
. . . . . .> WHERE fp.place = 'Soup Bowl'
. . . . . .> GROUP BY fp.place;
```

```
+-------------+---------------------+
|  fp.place   |         avg         |
+-------------+---------------------+
| Soup Bowl   | 25.862433862433864  |
+-------------+---------------------+
```

Exercise 8) <u>4 points</u>

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) The choice between a row format and a column format for big data files depends on the nature of the queries to be performed. Column-based storage is most beneficial for analytics queries that need only a subset of columns over large datasets, while row-based storage is better suited for queries requiring access to most or all columns of each row.

b) "Splittability" refers to the ability to break down a file into smaller, independent chunks that can be processed in parallel, which is crucial for efficient processing of large volumes of data. It enhances parallel processing capabilities, significantly impacting performance.

c) Files stored in a column format can achieve better compression than those in a row format because storing values of the same type together allows for more efficient compression algorithms. This results in significant savings in storage space and improves query performance due to reduced I/O.

d) The Parquet column file format is best used in scenarios where the dataset is wide (i.e., has many columns) and read-heavy workloads are common. It is particularly suited for use cases involving analytics and querying large datasets, where its efficient compression and splittability significantly improve performance and reduce storage costs.