

CSP554 Big Data Technologies

Assignment 4

ID : A20522183

1. java TestDataGen

Magic Number = 190464

```
hadoop@ip-172-31-2-33:~$ https://aws.amazon.com/amazon-linux-2/
37 package(s) needed for security, out of 69 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMM MMMMMMM RRRRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::R
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M RR::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M M:::M R:::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::R R:::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM MMMMMMM RRRRRRR RRRRRR

[hadoop@ip-172-31-2-33 ~]$ hadoop fs -ls
[hadoop@ip-172-31-2-33 ~]$ ls
TestDataGen.class
[hadoop@ip-172-31-2-33 ~]$ java TestDataGen
Magic Number = 190464
[hadoop@ip-172-31-2-33 ~]$
```

2. placed fooddata and foodratings

```
[hadoop@ip-172-31-2-33 ~]$ java TestDataGen
Magic Number = 190464
[hadoop@ip-172-31-2-33 ~]$ ls
foodplaces190464.txt foodratings190464.txt TestDataGen.class
[hadoop@ip-172-31-2-33 ~]$
```

3. View data in food places

```
hadoop@ip-172-31-2-33:~$ E:::E M:::M M:::M R:::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M M:::M R:::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::R R:::R
E:::EEEEEEEEEEEE M:::M M:::M M:::M R:::R R:::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM MMMMMMM RRRRRRR RRRRRR

[hadoop@ip-172-31-2-33 ~]$ hadoop fs -ls
[hadoop@ip-172-31-2-33 ~]$ ls
TestDataGen.class
[hadoop@ip-172-31-2-33 ~]$ java TestDataGen
Magic Number = 190464
[hadoop@ip-172-31-2-33 ~]$ ls
foodplaces190464.txt foodratings190464.txt TestDataGen.class
[hadoop@ip-172-31-2-33 ~]$ cat foodplaces190464.txt
1,China Bistro
2,Atlantic
3,Food Town
4,Jake's
5,Soup Bowl
[hadoop@ip-172-31-2-33 ~]$
```

Create a database on HIVE

Create Database mydb.

```
[hadoop@ip-172-31-2-33 ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.
properties Async: false
hive> CREATE DATABASE MyDb;
OK
Time taken: 1.967 seconds
hive>
```

4. mydb

Exercise 1.

```
[hadoop@ip-172-31-2-33 ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.
properties Async: false
hive> CREATE DATABASE MyDb;
OK
Time taken: 1.967 seconds
hive>
```

5.

Created a database on HIVE

```
java:62)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccess
sorImpl.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 11:10 mismatched input '/' expecting StringLiteral n
ear 'LOCATION' in table location specification
hive> Create External table if not exists MyDb.foodratings(
> name String Comment 'Food Critic Name',
> food1 INT Comment 'Review Rating 1',
> food2 INT Comment 'Review Rating 2',
> food3 INT Comment 'Review Rating 3',
> food4 INT Comment 'Review Rating 4',
> id INT Comment 'Restaurant Id(FK)')
> Comment 'Ratings Data'
> ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
> STORED AS TEXTFILE
> LOCATION '/home/hadoop/foodratings190464.txt';
OK
Time taken: 0.267 seconds
hive>
```

Describe formatted mydb.foodratings;

```
hadoop@ip-172-31-2-33:~
Time taken: 0.267 seconds
hive> DESCRIBE FORMATTED MyDb.foodratings;
OK
# col_name          data_type          comment
name                string            Food Critic Name
food1               int              Review Rating 1
food2               int              Review Rating 2
food3               int              Review Rating 3
food4               int              Review Rating 4
id                  int              Restaurant Id(FK)

# Detailed Table Information
Database:            mydb
Owner:               hadoop
CreateTime:          Thu Sep 28 16:46:07 UTC 2023
LastAccessTime:      UNKNOWN
Retention:           0
Location:             hdfs://ip-172-31-2-33.ec2.internal:8020/home/hadoop/food
ratings190464.txt
Table Type:          EXTERNAL_TABLE
Table Parameters:
    EXTERNAL          TRUE
    comment           Ratings Data
```

```

hadoop@ip-172-31-2-33:~
LastAccessTime: UNKNOWN
Retention: 0
Location: hdfs://ip-172-31-2-33.ec2.internal:8020/home/hadoop/food
ratings190464.txt
Table Type: EXTERNAL_TABLE
Table Parameters:
    EXTERNAL TRUE
    comment Ratings Data
    transient_lastDdlTime 1695919567

# Storage Information
SerDe Library: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat: org.apache.hadoop.mapred.TextInputFormat
OutputFormat: org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputForm
at
Compressed: No
Num Buckets: -1
Bucket Columns: []
Sort Columns: []
Storage Desc Params:
    field.delim ,
    serialization.format ,
Time taken: 0.127 seconds, Fetched: 33 row(s)
hive>

```

```

hadoop@ip-172-31-2-33:~
Database: mydb
Owner: hadoop
CreateTime: Thu Sep 28 16:51:39 UTC 2023
LastAccessTime: UNKNOWN
Retention: 0
Location: hdfs://ip-172-31-2-33.ec2.internal:8020/home/hadoop/food
places190464.txt
Table Type: EXTERNAL_TABLE
Table Parameters:
    EXTERNAL TRUE
    comment Restaurant Details
    transient_lastDdlTime 1695919899

# Storage Information
SerDe Library: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat: org.apache.hadoop.mapred.TextInputFormat
OutputFormat: org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputForm
at
Compressed: No
Num Buckets: -1
Bucket Columns: []
Sort Columns: []
Storage Desc Params:
    field.delim ,
    serialization.format ,
Time taken: 0.058 seconds, Fetched: 29 row(s)
hive> |

```

```
hadoop@ip-172-31-2-33:~  
Num Buckets: -1  
Bucket Columns: []  
Sort Columns: []  
Storage Desc Params:  
    field.delim  
    serialization.format  
Time taken: 0.058 seconds, Fetched: 29 row(s)  
hive> select name, min(food3) as MIN, max(food3) as MAX, avg(food3) as AVG from  
MyDb.foodratings;  
FAILED: SemanticException [Error 10025]: Line 1:7 Expression not in GROUP BY key  
(  
  'name'  
)  
hive> select "food3" as Column_name, min(food3) as MIN, max(food3) as MAX, avg(f  
ood3) as AVG from MyDb.foodratings;  
Query ID = hadoop_20230928165721_d74a90f9-538d-4b4d-a67b-e4d1bd9a0269  
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_1695917019180  
_0002)  
  
Map 1: -/-      Reducer 2: 0/1  
Map 1: -/-      Reducer 2: 0(+1)/1  
Map 1: -/-      Reducer 2: 1/1  
OK  
food3  NULL    NULL    NULL  
Time taken: 9.586 seconds, Fetched: 1 row(s)  
hive>
```

Select name, min(food1) as min, max(food1) as max, avg(food1) as avg from mydb.foodratings group by name.

```
hive> SELECT name, min(food1) AS MIN, max(food1) as MAX, AVG(food1) as AVG from MyDb.foodratings Group by name;  
FAILED: ParseException line 1:97 missing BY at 'ny' near '<EOF>'  
line 1:100 extraneous input 'name' expecting EOF near '<EOF>'  
hive> SELECT name, min(food1) AS MIN, max(food1) as MAX, AVG(food1) as AVG from MyDb.foodratings Group by name;  
Query ID = hadoop_20221001083424_93670229-6565-4bd4-b9e9-400b38a1ed61  
Total jobs = 1  
Launching Job 1 out of 1  
Status: Running (Executing on YARN cluster with App id application_1664609429932_0002)  
  
-----  
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: 4.61 s  
-----  
OK  
Jill    1      50      24.737373737373737  
Joe     1      50      24.555555555555557  
Joy     1      50      25.646464646464647  
Mel     1      50      24.989583333333332  
Sam     1      50      24.653658536585365  
Time taken: 5.285 seconds, Fetched: 5 row(s)
```

```

hive> DESCRIBE FORMATTED MyDB.foodratingspart;
OK
# col_name          data_type          comment
food1               int                Review rating 1
food2               int                Review rating 2
food3               int                Review rating 3
food4               int                Review rating 4
id                  int                Restaurant ID (FK)

# Partition Information
# col_name          data_type          comment
name                string             Name of food Critic

# Detailed Table Information
Database:            mydb
Owner:               hadoop
CreateTime:          Thu Sep 28 17:10:17 UTC 2023
LastAccessTime:      UNKNOWN
Retention:           0
Location:             hdfs://ip-172-31-2-33.ec2.internal:8020/user/hive/warehouse/mydb.db/foodratings
part
Table Type:          EXTERNAL_TABLE
Table Parameters:
    COLUMN_STATS_ACCURATE {\"BASIC_STATS\": \"true\"}
    EXTERNAL              TRUE
    comment                Rating data
    numFiles               0
    numPartitions           0
    numRows                0
    rawDataSize            0
    totalSize              0
    transient_lastDdlTime  1695921017

# 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: 0.049 seconds, Fetched: 43 row(s)
hive>

```

5

Answer: Partition on critic names would help querying faster and easier than partitioning on data of number of places which are large.

```

Time taken: 0.049 seconds, Fetched: 43 row(s)
hive> SET hive.exec.dynamic.partition.mode = non-strict;
hive> SET hive.exec.dynamic.partition;
hive.exec.dynamic.partition=true
hive> SET hive.exec.dynamic.partition.mode;
hive.exec.dynamic.partition.mode=non-strict
hive> |

```

6.

Set hive.exec.dynamic.partition.mode = non-strict;

```
hive> INSERT OVERWRITE TABLE MyDb.foodratingspart
> PARTITION (name)
> SELECT food1,food2,food3,food4,id,name FROM MyDb.foodratings;
Query ID = hadoop_20230928171547_fcf547e0-b3df-43e1-bf85-da64c061a8c4
Total jobs = 1
Launching Job 1 out of 1
tez session was closed. Reopening...
Session re-established.
```

```
-----
VERTICES    MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1        container  SUCCEEDED  0      0           0         0         0         0
-----
VERTICES: 00/01  [>>-----] 0%    ELAPSED TIME: 0.47 s
-----
loading data to table mydb.foodratingspart partition (name=null)

Time taken to load dynamic partitions: 0.009 seconds
Time taken for adding to write entity : 0.0 seconds
OK
Time taken: 5.447 seconds
hive>
```

INSERT OVERWRITE TABLE Mydb.foodratingspart.

SELECT MIN(FOOD1) AS MIN, MAX(FOOD1) AS MAX, AVG(FOOD1) AS AVG FROM
MYDB.FOODRATINGSPART WHERE NAME = "JILL" OR NAME="MEL"

```
hive> SELECT min(food1) as MIN, MAX(Food1) AS MAX, AVG(food1) as AVG from MyDb.foodratingspart where name ="Jill" or nam
e ="Mel";
Query ID = hadoop_20221001085131_153301c9-5ac1-4ada-aaeb-c01c20261dc1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1664609429932_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: 5.09 s
-----
OK
1      50      24.861538461538462
Time taken: 6.039 seconds, Fetched: 1 row(s)
```

SELECT FOODP.PLACE, AVG(FOODR,FOOD4) AS AVG FROM MYDB.FOODPLACES FOODP,
MYDB.FOODRATINGS WHERE FOODP.PLACE = 'SOUP BOWL' AND FOODP.ID =FOODR.ID GROUP BY
FOODP.PLACE

```

time taken: 0.725 seconds
hive> select foodp.place, avg(foodr.food4) as AVG from mydb.foodplaces foodp, Mydb.foodratings foodr where foodp.place =
'Soup Bowl' and foodp.id =foodr.id group by foodp.place;
Query ID = hadoop_20221001085612_6756f362-678f-493b-8035-99bf080a8256
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1664609429932_0003)

```

VERTICES	MODE	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	container	SUCCEEDED	1	1	0	0	0	0
Map 2	container	SUCCEEDED	1	1	0	0	0	0
Reducer 3	container	SUCCEEDED	2	2	0	0	0	0

```

VERTICES: 03/03 [=====] 100% ELAPSED TIME: 7.29 s
OK
Soup Bowl      25.21578947368421
Time taken: 8.056 seconds, Fetched: 1 row(s)
hive>

```

8.

a) Row format is useful when user has to access data w.r.t row values and need to access many rows at a time. This is for optimal data reading and writing.

Column format is useful when computation is focused on specific column without the need to search row values. This is for optimal data computation.

b) The capability to divide a file into independent smaller components is termed "splitability." When it comes to columnar file formats, splitability is achievable when query processing is concentrated on a specific column. This allows the data to be partitioned based on columns, enhancing computational efficiency.

c) Storing similar data types adjacent to each other enables more efficient compression compared to organizing them in a row-wise manner.

d) Parquet is used in Hadoop analytical database like (Impala). It is specially used in analysing huge dataset with multiple columns for computations.