Bhasheyam Krishnan

A20380078

Magic number: 55179

Que1:

Comand :

foot_ratings = LOAD '/user/maria_dev/foodratings55179.txt' USING PigStorage(',')

AS (name:chararray,f1:int, f2:int, f3:int, f4:int, placeid :int

);

DESCRIBE foot_ratings;

output:

```
grunt> DESCRIBE foot_ratings
foot_ratings: {name: chararray,f1: int,f2: int,f3: int,f4: int,placeid:
```

Que 2:

Command:

food_ratings_subset = FOREACH foot_ratings GENERATE name, f4;

STORE food_ratings_subset INTO ‘my_food_rating_output’ USING PigStorage(‘*’);

que2_output = LIMIT food_ratings_subset 6;

dump que2_output;

```
(Jill,38)
(Joe,22)
(Jill,25)
(Mel,12)
(Joe,22)
(Jill,22)
```

Que 3:

Command:

Group_rating = GROUP foot_ratings All;

food_ratings_profile = FOREACH grop_ratings GENERATE MIN(foot_ratings.f2),MAX(foot_ratings.f2),AVG(foot_ratings.f2),MIN(foot_ratings.f3),MAX(foot_ratings.f3),AVG(foot_ratings.f3);

dump food_ratings_profile;

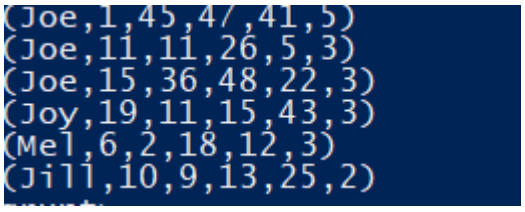
Que 4:

Command:

SPLIT foot_ratings INTO food_ratings_filtered IF (f1 < 20 and f3 > 5) , x IF (f1 > 20 and f3 < 5);

que4_output =LIMIT food_ratings_filtered 6;

dump que4_output



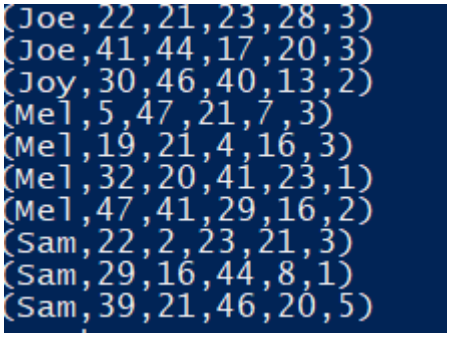
Que 5:

Command:

food_ratings_2percent = SAMPLE foot_ratings 0.02;

ques5_output = LIMIT foot_ratings_2percent 10;

dump qeus5_output;



Que6

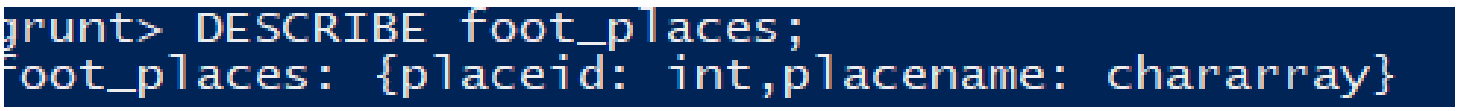
Command:

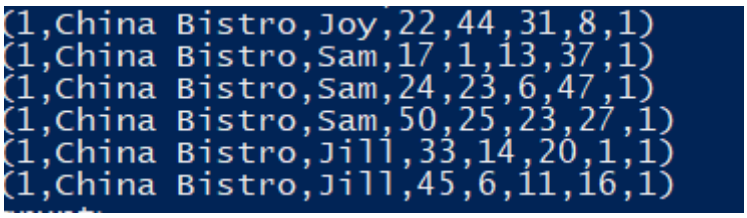
foot_places = LOAD '/user/maria_dev/foodplaces55179.txt' USING PigStorage(',') AS (placeid:int,placename:chararray);

DESCRIBE foot_places;

food_ratings_w_place_names = JOIN food_places BY (placeid), foot_ratings BY (placeid);

ques6_output = LIMIT food_ratings_w_place_names 6;





Que 7:

- Spark System(RDD's) is motivated by complex multi stage application and interactive ad-hoc queries as map-reduce simplified big data analysis but as soon as it got popular , used wanted it perform more.
- To perform such task a good data sharing technique is required which map – reduce lack as it use instance storage which is slow. Instead RDDs use the in -memory which faster to share data.
- To process all data in in-memory is not feasible all the time, so to overcome this, RDDs use the portioned data like filter, map etc.
- Spark is implementation helps to over come the issue faced in map – reduce for these advanced task.
- Spark produce better results in ML, data mining Processes as it can perform better in the parallel and iterative environment.
- a. RDDs would be less suitable for applications that make asynchronous fine-grained updates to shared state, such as a storage system for a web application or an incremental web crawler
- RDDs performance is better as restrictions on RDDs have little impact in many parallel applications.
- RDDs offer an API based on coarse-grained transformations that lets them recover data efficiently using lineage.
- Spark system is tent to work better than Hadoop for the iterative application.