PROJECT DESCRIPTION:- The Project aims to display the Agent loging report and Agent performance data of "I-Neuron technical consultant team". Both datasets are real-time generated that further contains the details of agents working for the team. The two datasets are stored in the form of CSV files and the project aims to load it to HDFS for further detailed analysis using HIVE.

DESCRIPTION OF DATASET:- The first dataset is Agent_Loging_report dataset which contains almost 1000 records and 7 attributes of the Consultant team agents. The attributes are :- Slno, Agent,Date,Login_time,Logout_time and duration of activity.

Link- "https://drive.google.com/file/d/1WrG-9qv6atP-W3P_-gYln1hHyFKRKMHP/view"

The second dataset is Agent_performance dataset which contains approximately 2200 records and 8 attributes. This dataset mostly depicts the performance analysis of agents working in the technical consultant team. The attributes are:- Slno,Date,Agent_name,Total_chats,Average_response_time,Average_resolution_time,Average_rating and Total Feedback.

Link - "https://drive.google.com/file/d/1-JIPCZ34dyN6k9CqJa-Y8yxIGq6vTVXU/view"

PROBLEM STATEMENT:-

There is a list of problem statements which we need to solve for further analysis on the dataset.

1.Create a schema based on the given dataset

2. Dump the data inside the hdfs in the given schema location.

3. List of all agents' names.

4. Find out agent average rating.

5. Total working days for each agents

6. Total query that each agent have taken

7. Total Feedback that each agent have received

8. Agent name who have average rating between 3.5 to 4

9. Agent name who have rating less than 3.5

10. Agent name who have rating more than 4.5

11. How many feedback agents have received more than 4.5 average

12. average weekly response time for each agent

13. average weekly resolution time for each agents

14. Find the number of chat on which they have received a feedback

15. Total contribution hour for each and every agents weekly basis

16. Perform inner join, left join and right join based on the agent column and after joining the table export that data into your local system.

17. Perform partitioning on top of the agent column and then on top of that perform bucketing for each partitioning.

Hence , the solutions for the above problem statements are shown on the next page. I have performed the analysis on HIVE Engine on Cloudera Platform . The screenshots of the HIve-Query-language is also pasted below for better understanding.

1) **Create a schema based on the given dataset.**
2) Dump the data inside the hdfs in the given schema location.

First of All, open the Hive Shell in Cloudera terminal and either use the existing database or create a new database. Here, I have used the existing database “hive_assignment” and created an empty table based on the schema of the two datasets.

```
Create table agent_loging_report (
sl_no int,Agent string,Date string,
login_time string,logout_time string,duration string)
row format delimited
fields terminated by ','
tblproperties ("skip.header.line.count" = "1");

Create table agent_performance(
sl_no int,Date string,Agent_name string,
total_chats int,average_response_time string,
average_resolution string,average_rating float,
total_feedback int)
row format delimited
fields terminated by ','
tblproperties ("skip.header.line.count" = "1");
```

After creating the table, its time to load the csv files in the empty table, At first I have downloaded both the CSV files in my local cloudera manager.

```
Load data local inpath ‘/home/cloudera/Downloads/
AgentLogingReport.csv’ into table agent_loging_report;

Load data local inpath ‘/home/cloudera/Downloads/
AgentPerformance.csv’ into table agent_performance;
```

The screenshot of the above code part is pasted below.

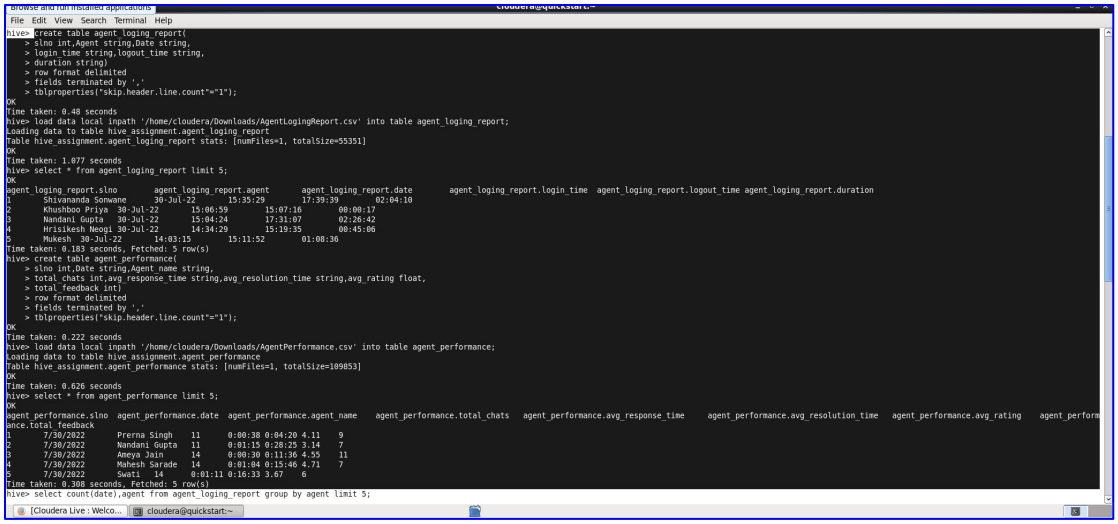
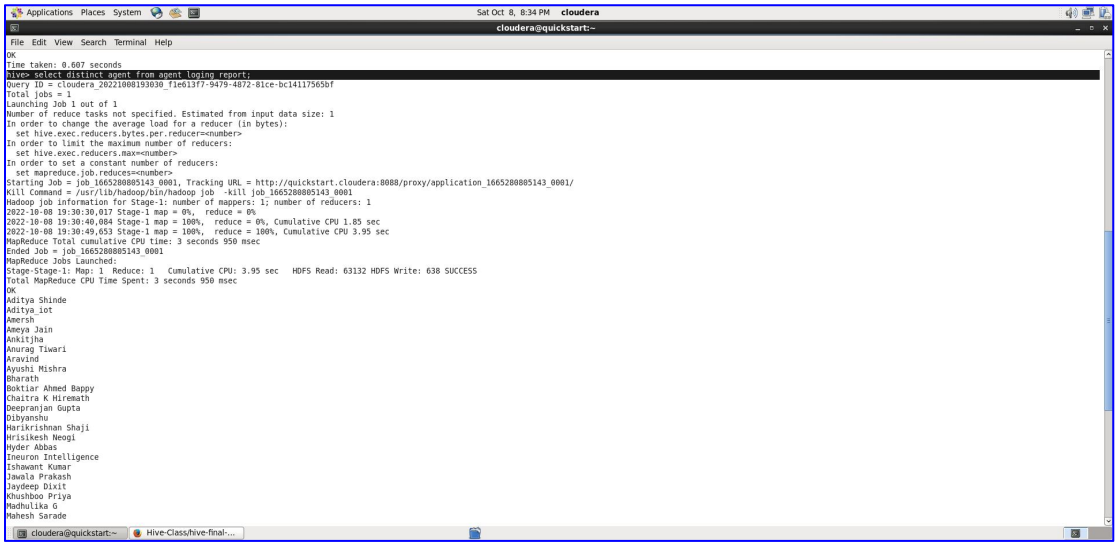


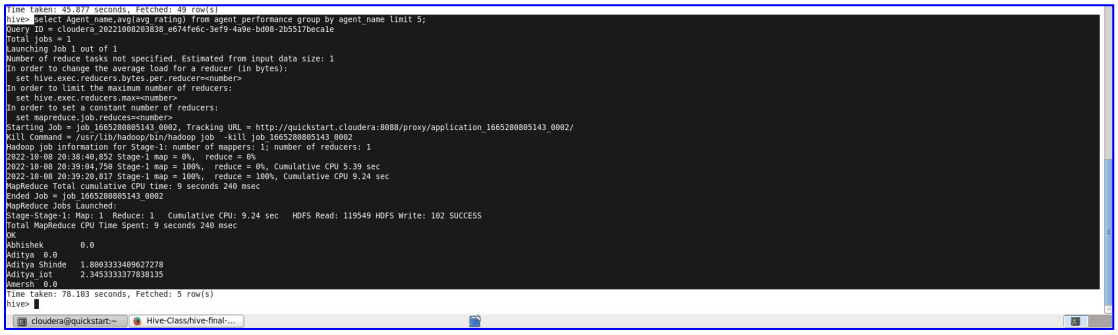
Fig 1

3. List of all agents' names.



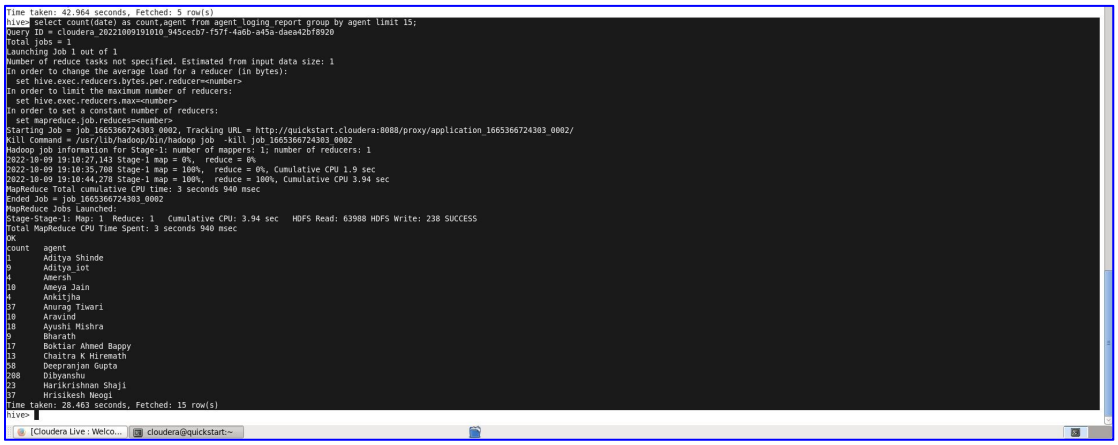
Select distinct agent from agent_logging_report ;

4. Find out agent average rating.



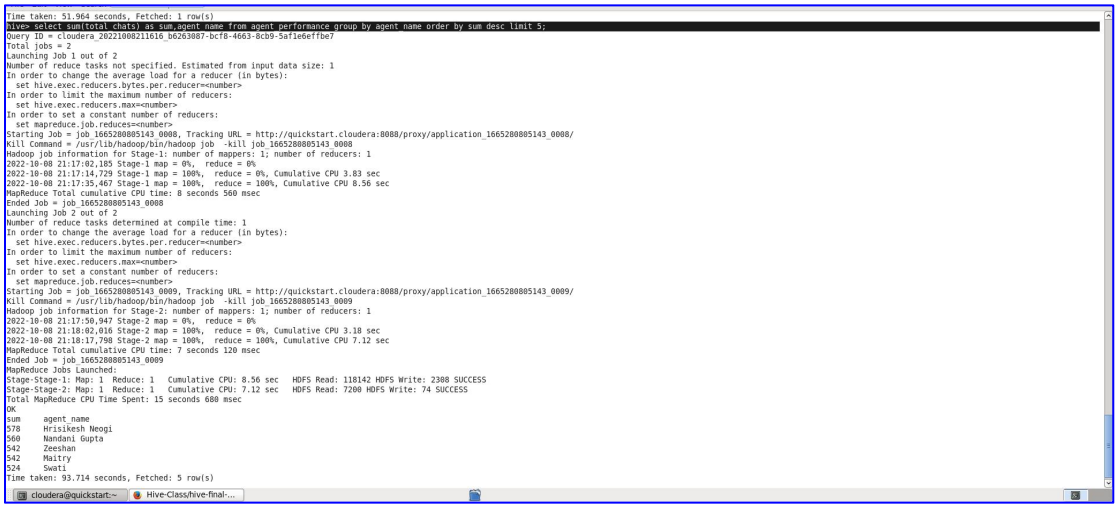
select agent_name,avg(average_rating), from agent_performance group by agent_name limit 5;

5. Total working days for each agents



Select count(date),agent from agent_logging_report group by agent limit 5;

6. Total query that each agent have taken

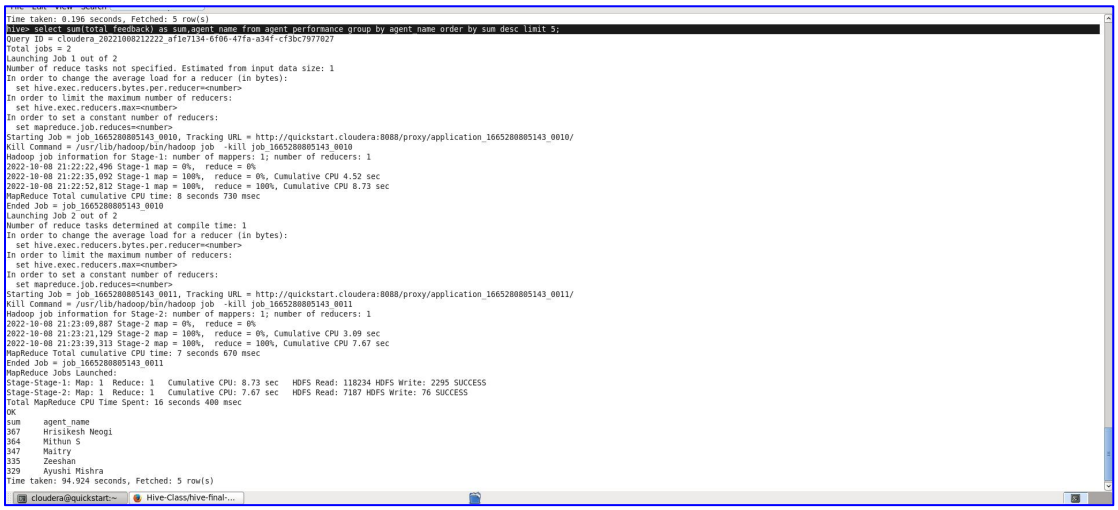


Select Sum(total_chats) as sum,agent_name
from agent_performance group by agent_name
Order by sum Desc limit 5;

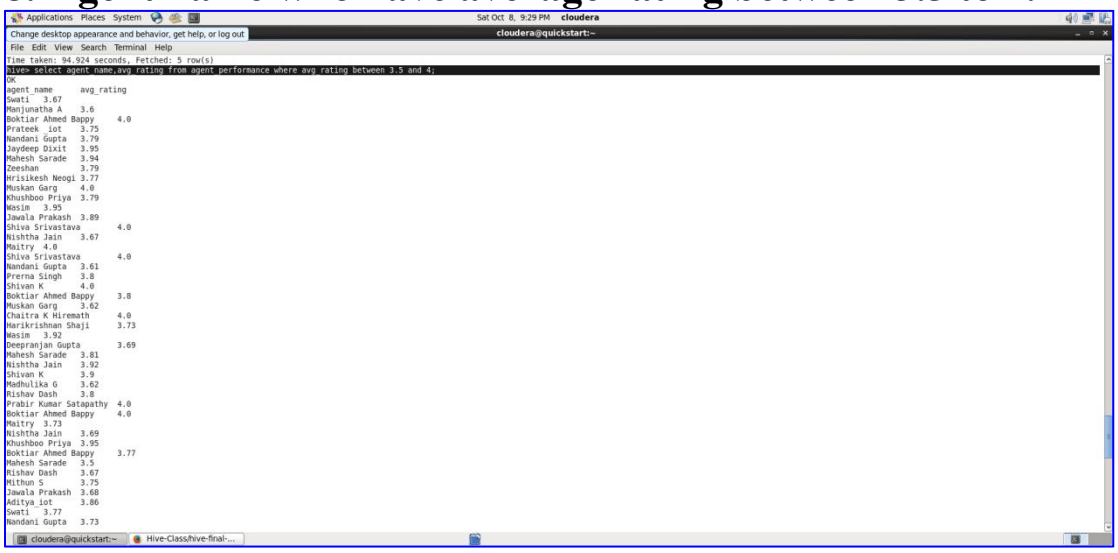
7. Total Feedback that each agent have received

select sum(total_feedback) as sum,agent_name from agent_performance group by agent_name

Order by sum desc limit 5;



8. Agent name who have average rating between 3.5 to 4.



Select agent_name from agent_performance where average_rating BETWEEN 3.5 AND 4;

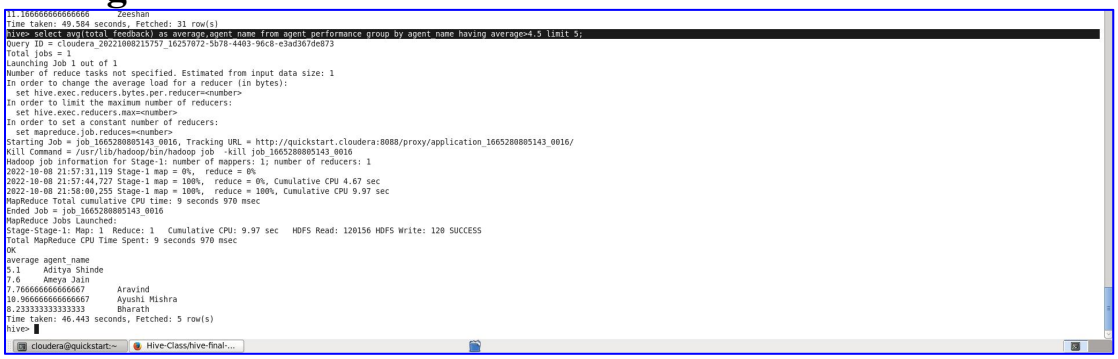
9. Agent name who have rating less than 3.5 .

Select agent_name from agent_performance where average_rating < 3.5 limit 10;

10. Agent name who have rating more than 4.5.

Select agent_name from agent_performance where average_rating > 4.5 limit 5;

11. How many feedback agents have received more than 4.5 average.

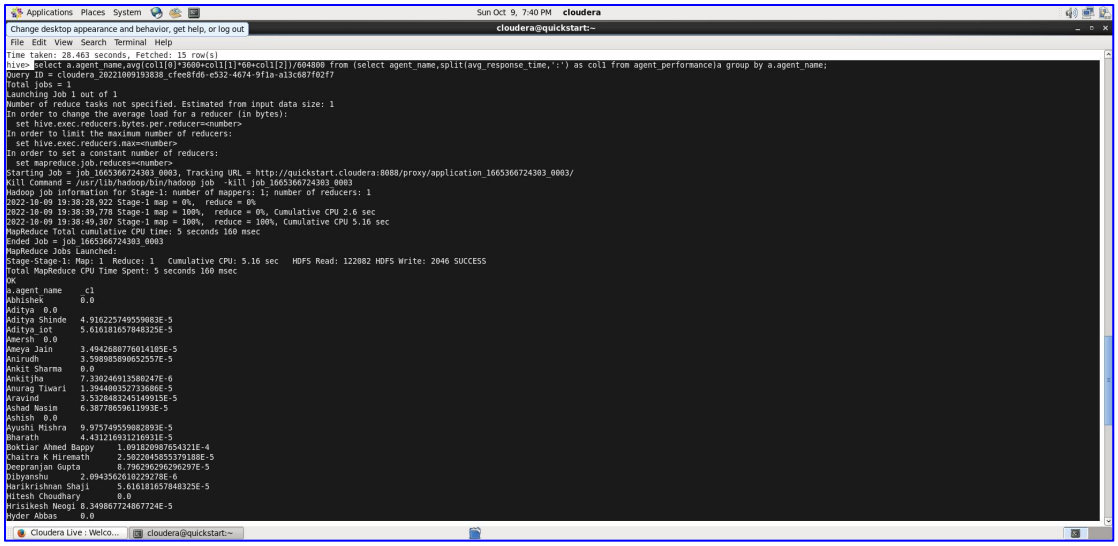


Select avg(total_feedback) as average, agent_name from agent_performance group by agent_name having average>4.5;

12.Average weekly response time for each agent.

Here,to find average weekly response time, we have to first convert the “avg_response_time” column into seconds and then convert it into week. Since the column is in the format hh:mm:ss, I have first converted it into seconds and then divided it by (60*60*24*7) to convert it into week format.

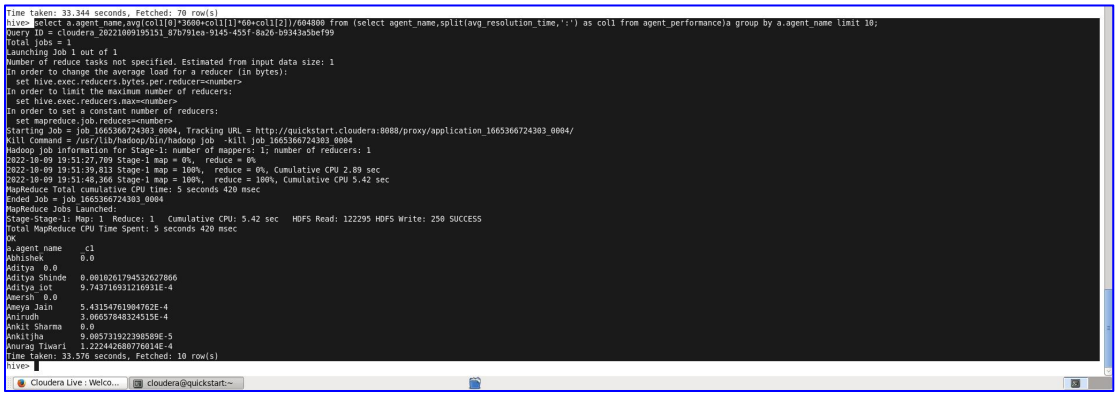
Select a.agent_name, avg(col1[0]*3600+col1[1]*60+col1[2])/604800 from (select Agent_name,split(avg_response_time,’:’) as col1 from agent_performance) a
Group by a.agent_name;



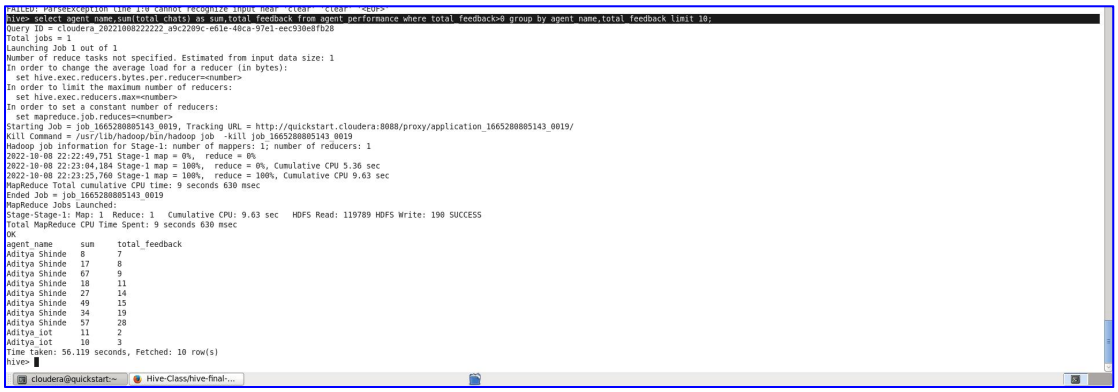
13. Average weekly resolution time for each agents .

Here also,we have to do the same thing as in previous question, the only difference is here we have to use “avg_resolution-time”.

Select a.agent_name, avg(col1[0]*3600+col1[1]*60+col1[2])/604800 from (select Agent_name,split(avg_resolution_time,’:’) as col1 from agent_performance) a
Group by a.agent_name limit 10;



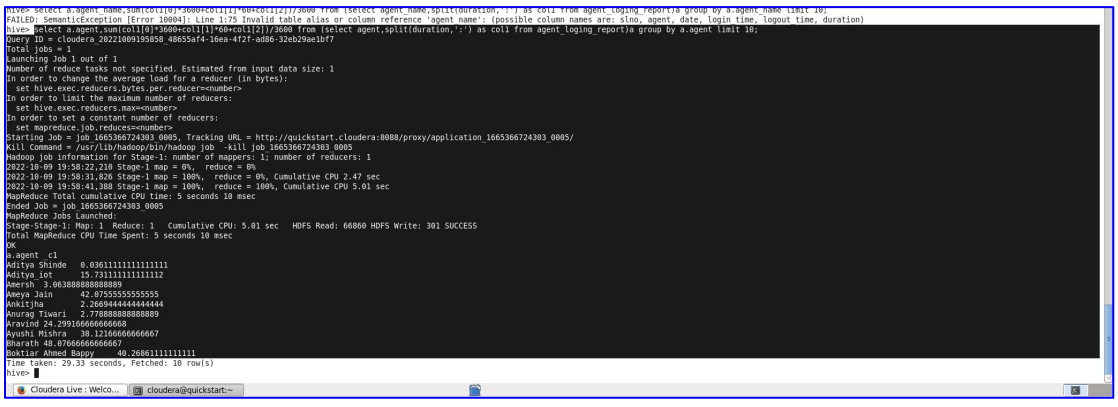
14. Find the number of chat on which they have received a feedback .



Select agent_name,sum(total_chats),total_feedback from agent_performance where total_feedback>0 group by agent_name,total_feedback limit 10;

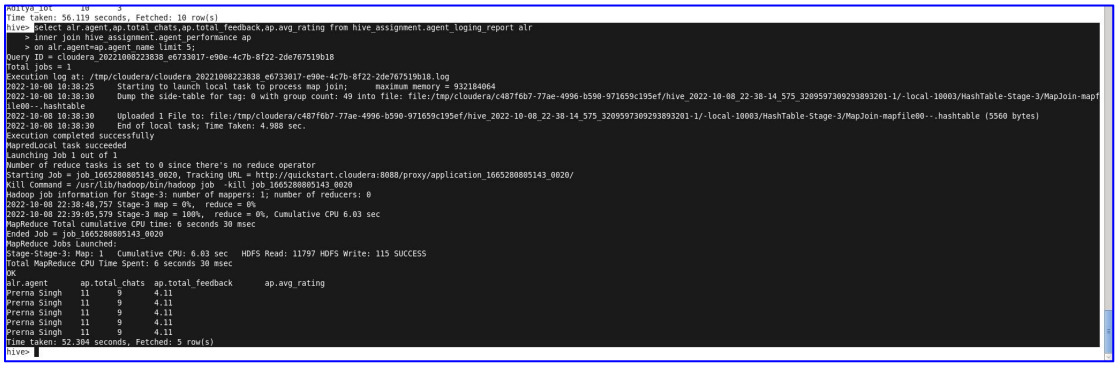
15. Total contribution hour for each and every agents weekly basis.

Select a.agent, avg(col1[0]*3600+col1[1]*60+col1[2])/3600 from (select agent,split(duration,':') as col1 from agent_performance) a Group by a.agent_name limit 10;



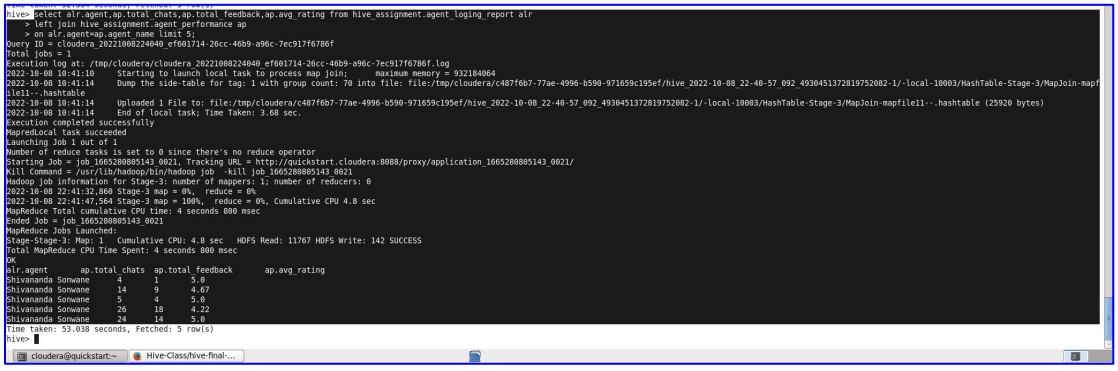
16. Perform inner join, left join and right join based on the agent column and after joining the table export that data into your local system.

- **INNER JOIN-**



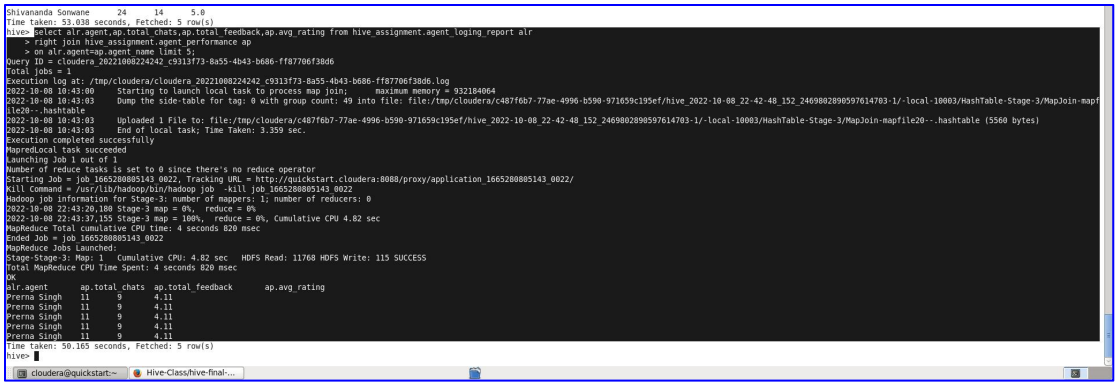
Select alr.agent,ap.total_chats,ap.avg_rating,ap.total_feedback
From hive_assignment.agent_loging_report alr
Inner join hive_assignment.agent_performance ap
On alr.agent=ap.agent_name limit 5;

- **LEFT JOIN-**



Select alr.agent,ap.total_chats,ap.avg_rating,ap.total_feedback
From hive_assignment.agent_loging_report alr
left join hive_assignment.agent_performance ap
On alr.agent=ap.agent_name limit 5;

- **RIGHT JOIN-**




```
Select alr.agent,ap.total_chats,ap.avg_rating,ap.total_feedback
From hive_assignment.agent_loging_report alr
right join hive_assignment.agent_performance ap
On alr.agent=ap.agent_name limit 5;
```

Now, we have to export the joined output in the local HDFS system. For this, we have to exit the hive terminal or start a new terminal and then perform the following query.

```
hive -e 'Select alr.agent,alr.date,ap.total_chats,ap.total_feedback
from hive_assignment2.agent_logging_report alr join
hive_assignment2.agent_performance ap on alr.agent =
ap.agent_name limit 5' >
/home/cloudera/hive_Assignment2/inner.join.csv;

hive -e 'Select alr.agent,alr.date,ap.total_chats,ap.total_feedback
from hive_assignment2.agent_logging_report alr left join
hive_assignment2.agent_performance ap on alr.agent =
ap.agent_name limit 5' >
/home/cloudera/hive_Assignment2/left.join.csv;
```


```
hive -e 'Select alr.agent,alr.date,ap.total_chats,ap.total_feedback
from hive_assignment2.agent_logging_report alr right join
hive_assignment2.agent_performance ap on alr.agent =
ap.agent_name limit 5' >
/home/cloudera/hive_Assignment2/right.join.csv;
```

17. Perform partitioning on top of the agent column and then on top of that perform bucketing for each partitioning.

First of all, I have performed partitioning and Bucketing for Agent_loging_report table.

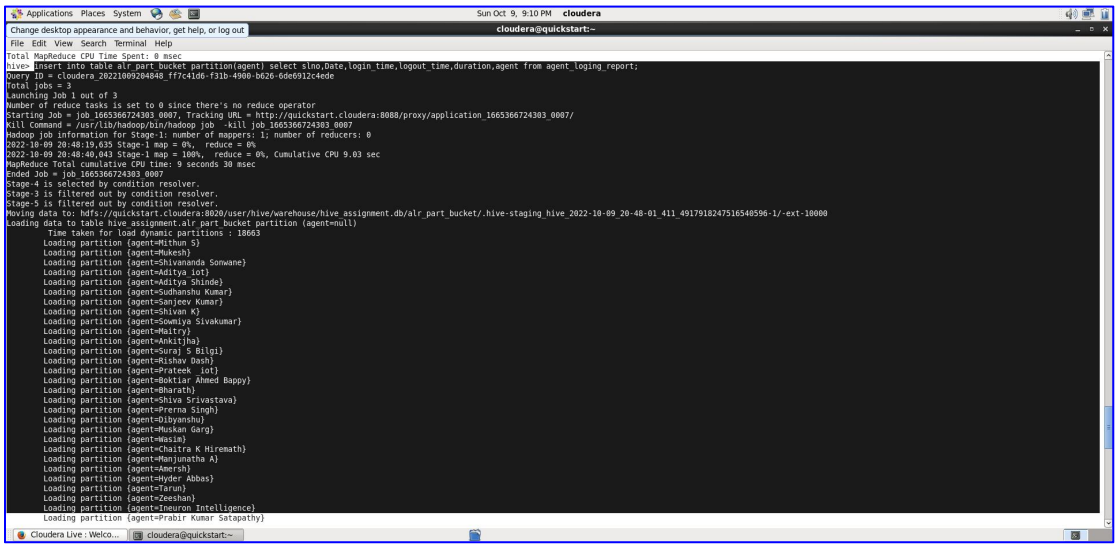
```
Create table alr_part_bucket(
Slno int, Date string,login_time string,logout_time string,
Duration string) partitioned by (agent string)
Clustered by (Date) sorted by (Date) into 4 buckets
Row format delimited
Fields terminated by ‘,’;
```

Then set the Hive Dynamic mode partition.

```
Set hive.exec.dynamic.partition=true;
Set hive.exec.dynamic.partition.mode=nonstrict;
```

And then loading the data into the table using insert overwrite command.

Insert into table alr_part_bucket partition (agent)
Select slno,Date,login_time,logout_time,duration,agent from agent_loging_report;



From the above image, it can be seen that the partitioning and bucketing part is created successfully.
Now , the steps have to be repeated for the second table “Agent_performance”.

Create table AP_partition_Bucket(
slno int,Date date,
Total_chat string,
Average_Response_Time string,
Average_Resolution_Time string,
Average_Rating float,Total_Feedback int)
partitioned by (agent_name string)
CLUSTERED BY (Date) sorted by (Date) INTO 8 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

Insert into table ap_part_bucket partition (agent_name)
Select
slno,Date,total_chats,avg_response_time,avg_resolution_time,
Avg_rating,total_feedback,Agent_name from
agent_performance;