HIVE Mini-Project 1:

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.

1.

```
create table AgentLogingReport
(
Agent varchar(25),
Date DATE FORMAT ‘dd-mm-yy’,
Login_Time TIMESTAMP,
Logout_Time TIMESTAMP,
Duration TIMESTAMP,
PRIMARY KEY (Agent) DISABLE NOVALIDATE
)
row format delimited
fields terminated by ‘,’
;

create table AgentPerformance
(
Date DATE ,
Agent_Name varchar(25),
Total_Chats int,
Average_Response_Time TIMESTAMP,
Average_Resolution_Time TIMESTAMP,
Average_Rating float,
Total_Feedback int,
PRIMARY KEY (Agent_Name) DISABLE NOVALIDATE
)
row format delimited
fields terminated by ‘,’
;
```

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

```
LOAD DATA INPATH ‘file:///path/to/AgentLogingReport.csv’ INTO TABLE
AgentLogingReport;
```

To copy the file from your local file system to the HDFS, you can use the **hdfs dfs -put** command

3. List of all agents' names.

```
select distinct Agent from AgentLogingReport;
```

4. Find out agent average rating.

```
select Agent_Name, avg(Average_Rating) over(partition by Agent_Name) from
AgentPerformance
sort by Agent_Name asc;
```

Total working days for each agents:

```
select Agent, count(distinct(Date)) as Total_Working_Days from
AgentLogingReport
group by Agent
sort by Total_Working_Days;
```

Total query that each agent have taken:

```
select Agent_Name, sum(Total_Chats) from AgentPerformance
group by Agent_Name;
```

Total Feedback that each agent have received:

```
select Agent_Name, sum(Total_Feedback) as TotalFeedback from
AgentPerformance
group by Agent_Name
sort by TotalFeedback desc;
```

Agent name who have average rating between 3.5 to 4:

```
select Agent_Name, avg(Average_Rating) as Avg_Rating from AgentPerformance
where Avg_Rating BETWEEN 3.5 AND 4
group by Agent_Name
sort by Avg_Rating desc;
```

Agent name who have rating less than 3.5

```
select Agent_Name, avg(Average_Rating) as Avg_Rating from AgentPerformance
where Avg_Rating< 3.5
group by Agent_Name
sort by Avg_Rating desc;
```

Agent name who have rating more than 4.5:

```
select Agent_Name, avg(Average_Rating) as Avg_Rating from AgentPerformance
where Avg_Rating> 4.5
group by Agent_Name
sort by Avg_Rating desc;
```

How many feedback agents have received more than 4.5 average:

```
select Agent_Name, sum(Total_Feedback) from AgentPerformance
where Average_Rating > 4.5
group by Agent_Name;
```

Average weekly response time for each agent

```
select Agent_Name, DATEPART(week, Date) as week,
avg(Average_Response_Time) as avg_time from AgentPerformance
group by Agent_Name, week
sort by avg_time desc;
```

Average weekly resolution time for each agents:

```
select Agent_Name, DATEPART(week, Date) as week,
avg(Average_Resolution_Time) as as avg_time from AgentPerformance
group by Agent_Name, week
sort by avg_time desc;
```

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

```
select Agent_Name, count(Total_Feedback) from AgentPerformance
where Total_Feedback != 0
group by Agent_Name;
```

Total contribution hour for each and every agents weekly basis:

```
select Agent_Name, DATEPART(week, Date), sum(HOUR(TIMEDIFF(Logout_Time,
Login_Time))) from AgentLogingReport
group by 1, 2
sort by 1;
```

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.

```
select a.*,b.* from AgentLogingReport a INNER JOIN AgentPerformance b ON
```

a.Agent = b.Agent_Name;
select a.*,b.* from AgentLogingReport a LEFT JOIN AgentPerformance b ON a.Agent = b.Agent_Name;
select a.*,b.* from AgentLogingReport a RIGHT JOIN AgentPerformance b ON a.Agent = b.Agent_Name;

INSERT OVERWRITE LOCAL DIRECTORY '/path/to/local/directory'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
SELECT *
FROM my_table;

# The **INSERT OVERWRITE LOCAL DIRECTORY** statement will create a new file in the specified local directory for each task that produces output

INSERT OVERWRITE LOCAL DIRECTORY '/path/to/local/directory'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
SELECT t1.*, t2.*
FROM table1 t1
JOIN table2 t2
ON t1.key = t2.key;

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

ALTER TABLE AgentPerformance **PARTITIONED BY** (Agent_Name);
CLUSTER AgentPerformance **BY** (Agent_Name) INTO 32 BUCKETS; # 2166 rows —> 2166/128 = 16.92 —> 2^5=32