1. Create a schema based on the given dataset

create table agentperf

(

slno int,

dates string,

agent\_name string,

totalchat int,

avgresptime string,

avgresoltime string,

avgrating float,

totalfeedback int

)

row format delimited

fields terminated by ','

LINES TERMINATED BY '\n'

STORED AS TEXTFILE

tblproperties("skip.header.line.count"="1");

create table agentlog

(

slno int

agent\_name string

dates string

logintime string

logouttime string

duration string

)

row format delimited

fields terminated by ','

LINES TERMINATED BY '\n'

STORED AS TEXTFILE

tblproperties("skip.header.line.count"="1");

#Tables transformed by formatting dates and changing durations to seconds

create table agentlogging

(

agent\_name string,

dated date,

durationsec int

)

row format delimited

fields terminated by ','

tblproperties("skip.header.line.count"="1");

insert overwrite table agentlogging select t.agent\_name, to\_date(from\_unixtime(unix\_timestamp(t.dates,'dd-MMM-yy'))), t.durationsec from (select agent\_name, dates, (cast(split(duration,':')[0] as int)\*3600 + cast(split(duration,':')[1] as int)\* 60 + cast(split(duration,':')[2] as int)) as durationsec from agentlog) t;

create table agentperformance (

agent\_name string,

dated date,

totalchat int,

avgresp\_timesec int,

avgresol\_timesec int,

avgrating float,

totalfeedback int

)

row format delimited

fields terminated by ','

tblproperties("skip.header.line.count"="1");

insert overwrite table agentperformance select t.agentname, to\_date(from\_unixtime(unix\_timestamp(t.adate,'MM/dd/yyyy'))),t.totalchats, t.averageresponsetime,t.averageresolutiontime,t.averagerating,t.total\_feedback from (select agentname, adate, totalchats, (cast(split(averageresponsetime,':')[0] as int)\*3600 + cast(split(averageresponsetime,':')[1] as int)\* 60 + cast(split(averageresponsetime,':')[2] as int)) as averageresponsetime, (cast(split(averageresolutiontime ,':')[0] as int)\*3600 + cast(split(averageresolutiontime ,':')[1] as int)\* 60 + cast(split(averageresolutiontime ,':')[2] as int)) as averageresolutiontime, averagerating,total\_feedback from agentperf) t;

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

PS C:\HiveDockerSetup> docker cp AgentPerformance.csv namenode:AgentPerformance.csv

PS C:\HiveDockerSetup> docker cp AgentLogingReport.csv namenode:AgentLogingReport.csv

root@44a76142d7c6:/# hadoop fs -put AgentLogingReport.csv /hivemini1/AgentLogingReport.csv

root@44a76142d7c6:/# hadoop fs -put AgentPerformance.csv /hivemini1/AgentPerformance.csv

3. List of all agents' names.

select distinct agentname from agentperf;

4. Find out agent average rating.

For the entire month the average rating would be

select agentname, avg(averagerating) from agentperf group by agentname;

5. Total working days for each agents

select distinct p.agentname, tmp.no\_of\_working\_days from agentperf p

left join

(select agent\_name, count(\*) as no\_of\_working\_days from agentlog group by agent\_name) tmp

on p.agentname=tmp.agent\_name ;

6. Total query that each agent have taken

select agentname,sum(totalchats) from agentperf group by agentname;

7. Total Feedback that each agent have received

select agentname,sum(totalfeedback) from agentperf group by agentname;

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

select distinct tb.agentname from agentperf tb

left join

(select agentname, avg(averagerating) avgs from agentperf group by agentname) tmp

on tb.agentname=tmp.agentname

where tmp.avgs between 3.5 and 4;

9. Agent name who have rating less than 3.5

select distinct tb.agentname from agentperf tb

left join

(select agentname, avg(averagerating) avgs from agentperf group by agentname) tmp

on tb.agentname=tmp.agentname

where tmp.avgs < 3.5;

10. Agent name who have rating more than 4.5

select distinct tb.agentname from agentperf tb

left join

(select agentname, avg(averagerating) avgs from agentperf group by agentname) tmp

on tb.agentname=tmp.agentname

where tmp.avgs >4.5;

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

#Considering monthly average above 4.5

select count(distinct tb.agentname) from agentperf tb

left join

(select agentname, avg(averagerating) avgs from agentperf group by agentname) tmp

on tb.agentname=tmp.agentname

where tmp.avgs >4.5;

#Considering daily average > 4.5

select count(distinct agentname) from agentperf where averagerating >4.5;

12. average weekly response time for each agent

Ans: In minutes

SELECT agent\_name, WEEKOFYEAR(dated) as week,

Round( AVG(avgresp\_timesec)/60 ,2) as weekly\_avg\_response\_time

FROM agentperformance

GROUP BY agent\_name, WEEKOFYEAR(dated);

13. average weekly resolution time for each agents

Ans: In minutes

SELECT agent\_name, WEEKOFYEAR(dated) as week,

Round(AVG(avgresol\_timesec)/60,2) as weekly\_avg\_resolution\_time

FROM agentperformance

GROUP BY agent\_name, WEEKOFYEAR(dated);

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

select agentname, sum(total\_feedback) from agentperf group by agentname ;

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

select agent\_name, weekofyear(dated) as week, round(SUM(durationsec)/60,2) from

agentlogging

group by agent\_name, weekofyear(dated);

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.

INSERT OVERWRITE LOCAL DIRECTORY /user/hive/warehouse/minipr1.db

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

SELECT \*

FROM agentlog l

INNER JOIN agentperf p

ON l.agent\_name = p.agentname;

INSERT OVERWRITE LOCAL DIRECTORY /user/hive/warehouse/minipr1.db

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

SELECT \*

FROM agentlog l

LEFT JOIN agentperf p

ON l.agent\_name = p.agentname;

INSERT OVERWRITE LOCAL DIRECTORY /user/hive/warehouse/minipr1.db

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

SELECT \*

FROM agentlog l

RIGHT JOIN agentperf p

ON l.agent\_name = p.agentname;

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

create table agentperformancepartitionbucket (

dated date,

totalchat int,

avgresp\_timesec int,

avgresol\_timesec int,

avgrating float,

totalfeedback int

)

partitioned by(agent\_name string)

clustered by(dated) into 4 buckets;

insert overwrite table agentperformancepartitionbucket partition(agent\_name) select dated,totalchat, avgresp\_timesec,

avgresol\_timesec,

avgrating,

totalfeedback,agent\_name from agentperformance;

# Same is for agent logging table