## Scripts Execution

**Explanation of the solution to the batch layer problem**

Task 1: Load the transactions history data (card\_transactions.csv) in a NoSQL database.

Load the transaction history data to hbase and also to hive using the relavent commands mentioned in documents.

Task 2: Ingest the relevant data from AWS RDS to Hadoop.

Ingest data to hdfs from rds path/details given and load this as well to hive. Specifically member\_score data

Task 3: Create a look-up table with columns specified earlier in the problem statement.

First, create lookup table from hive which maps to hbase and next step is to run a query which basically joins the card\_transactions and member\_score table to get the latest entry for each member.

Later, join the above query with UCL computed select statement which computed moving average and standard deviation and applies formula (avg + 3 x Std.Deveation) to have all fields to fill data in loop\_up\_table.

Finally add these columns to newely created look up table which basically reflects in hbase too

SCRIPT:

insert into transactions.look\_up\_table\_hbase ( card\_id, ucl, postcode, transaction\_dt, score)

select t1.card\_id, us.ucl, t1.postcode, t1.transaction\_dt, t1.score

from

(SELECT t.card\_id

,t.postcode

,t.transaction\_dt

,t.score

FROM (

SELECT s.member\_id, c.card\_id

,c.postcode

,c.transaction\_dt

,s.score

,ROW\_NUMBER() OVER (

PARTITION BY card\_id ORDER BY unix\_timestamp(transaction\_dt,'dd-MM-yyyy hh:mm:ss') DESC

) AS ROW\_NUMBER

FROM transactions.card\_transactions c

join transactions.member\_score s on c.member\_id=s.member\_id

order by c.card\_id

) t

WHERE t.ROW\_NUMBER <= 1) t1

join (SELECT t.card\_id, cast (avg(t.amount)+3\*STDDEV(t.amount) as int) as ucl

FROM (

SELECT s.member\_id, c.card\_id

,c.postcode

,c.amount

,c.transaction\_dt

,s.score

,ROW\_NUMBER() OVER (

PARTITION BY card\_id ORDER BY unix\_timestamp(transaction\_dt,'dd-MM-yyyy hh:mm:ss') DESC

) AS ROW\_NUMBER

FROM transactions.card\_transactions c

join transactions.member\_score s on c.member\_id=s.member\_id

order by c.card\_id

) t

WHERE t.ROW\_NUMBER <= 10

group by t.card\_id) us on us.card\_id=t1.card\_id;

Task 4: After creating the table, you need to load the relevant data in the lookup table.

hive> select \* from transactions.look\_up\_table limit 10;

OK

340028465709212 319613580 24658 02-01-2018 03:25:35 233

340054675199675 14156079 50140 15-01-2018 19:43:23 631

340082915339645 15285685 17844 26-01-2018 19:03:47 407

340134186926007 909246042 67576 18-01-2018 23:12:50 614

340265728490548 16084916 72435 21-01-2018 02:07:35 202

340268219434811 12507323 62513 16-01-2018 04:30:05 415

340379737226464 14198310 26656 27-01-2018 00:19:47 229

340383645652108 14091750 34734 29-01-2018 01:29:12 645

340803866934451 10843341 87525 31-01-2018 04:23:57 502

340889618969736 13217942 61341 31-01-2018 21:57:18 330

Time taken: 0.088 seconds, Fetched: 10 row(s)





