Scripts Execution

Screenshots of the execution of the scripts written

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

1. Start hive from command prompt. Create new database namely capstone_project and switch to use capstone_project.

```
-bash-4.2$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>
    >
    >
    >
    >
    > create database capstone_project;
OK
```

2. Set below parameters for the hive session

```
hive>
    >
    > set hive.auto.convert.join=false;
hive> set hive.stats.autogather=true;
hive> set orc.compress=SNAPPY;
hive> set hive.exec.compress.output=true;
hive> set mapred.output.compression.codec=org.apache.hadoop.io.compress.SnappyCodec;
hive> set mapred.output.compression.type=BLOCK;
hive> set mapreduce.map.java.opts=-Xmx5G;
hive> set mapreduce.reduce.java.opts=-Xmx5G;
hive> set mapred.child.java.opts=-Xmx5G -XX:+UseConcMarkSweepGC -XX:-UseGCOverheadLimit;
```

3. Create external table card_transactions_ext table which will point to HDFS location created earlier.

```
CREATE EXTERNAL TABLE IF NOT EXISTS CARD_TRANSACTIONS_EXT(
`CARD_ID` STRING,
`MEMBER_ID` STRING,
`AMOUNT` DOUBLE,
`POSTCODE` STRING,
`POS_ID` STRING,
`TRANSACTION_DT` STRING,
`STATUS` STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/capstone_project/card_transactions'
TBLPROPERTIES ("skip.header.line.count"="1");
```

```
hive>
    > CREATE EXTERNAL TABLE IF NOT EXISTS CARD_TRANSACTIONS_EXT(
    > `CARD_ID` STRING,
    > `MEMBER_ID` STRING,
    > `AMOUNT` DOUBLE,
    > `POSTCODE` STRING,
    > `POS_ID` STRING,
    > `TRANSACTION_DT` STRING,
    > `STATUS` STRING)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LOCATION '/capstone_project/card_transactions'
    > TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.441 seconds
```

4. Create table card_transactions_orc. ORC format will help in better performance.

```
CREATE TABLE IF NOT EXISTS CARD_TRANSACTIONS_ORC(
`CARD_ID` STRING,
`MEMBER_ID` STRING,
`AMOUNT` DOUBLE,
`POSTCODE` STRING,
`POS_ID` STRING,
`TRANSACTION_DT` TIMESTAMP,
`STATUS` STRING)
STORED AS ORC
TBLPROPERTIES ("orc.compress"="SNAPPY");
```

```
    >
    > CREATE TABLE IF NOT EXISTS CARD_TRANSACTIONS_ORC(
    > `CARD_ID` STRING,
    > `MEMBER_ID` STRING,
    > `AMOUNT` DOUBLE,
    > `POSTCODE` STRING,
    > `POS_ID` STRING,
    > `TRANSACTION_DT` TIMESTAMP,
    > `STATUS` STRING)
    > STORED AS ORC
    > TBLPROPERTIES ("orc.compress"="SNAPPY");
OK
Time taken: 0.408 seconds
```

5. Load data in card_transactions_orc while casting timestamp format for transaction_dt column.

```
INSERT OVERWRITE TABLE CARD_TRANSACTIONS_ORC
```

SELECT CARD_ID, MEMBER_ID, AMOUNT, POSTCODE, POS_ID,
CAST(FROM_UNIXTIME(UNIX_TIMESTAMP(TRANSACTION_DT,'dd-MM-yyyy HH:mm:ss')) AS TIMESTAMP), STATUS
FROM CARD_TRANSACTIONS_EXT;

```
hive>
    > INSERT OVERWRITE TABLE CARD_TRANSACTIONS_ORC
    > SELECT CARD_ID, MEMBER_ID, AMOUNT, POSTCODE, POS_ID, CAST(FROM_UNIXTIME(UNIX_TIMESTAMP(TRANSACTION_DT,'dd-MM-yyyy
 HH:mm:ss')) AS TIMESTAMP), STATUS
    > FROM CARD_TRANSACTIONS_EXT;
Query ID = hdfs_20230518065643_3fdbdfb5-2d1a-4164-9a3d-bb08d432cfa9
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1684390665990_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 6.12 s
--------------------------------------------------------------------------------
Loading data to table capstone_project.card_transactions_orc
OK
Time taken: 16.33 seconds
```

6. Verify transaction_dt and year in card_transactions_orc table.

select year(transaction_dt), transaction_dt from card_transactions_orc limit 10;

```
hive>
    > select year(transaction_dt), transaction_dt from card_transactions_orc limit 10;
OK
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
2018    2018-02-11 00:00:00
Time taken: 0.236 seconds, Fetched: 10 row(s)
hive>
```

7. Create card_transactions_hbase hive-hbase integrated table which will be visible in HBase as well. - **This table will have all transactions – historical as well as new incoming from streaming layer.**

CREATE TABLE CARD_TRANSACTIONS_HBASE(
`TRANSACTION_ID` STRING,
`CARD_ID` STRING,
`MEMBER_ID` STRING,
`AMOUNT` DOUBLE,
`POSTCODE` STRING,
`POS_ID` STRING,
`TRANSACTION_DT` TIMESTAMP,
`STATUS` STRING)

ROW FORMAT DELIMITED
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES
("hbase.columns.mapping"=":key, card_transactions_family:card_id,
card_transactions_family:member_id, card_transactions_family:amount,
card_transactions_family:postcode, card_transactions_family:pos_id,
card_transactions_family:transaction_dt, card_transactions_family:status")
TBLPROPERTIES ("hbase.table.name"="card_transactions_hive");

```
hive> CREATE TABLE CARD_TRANSACTIONS_HBASE(
    > `TRANSACTION_ID` STRING,
    > `CARD_ID` STRING,
    > `MEMBER_ID` STRING,
    > `AMOUNT` DOUBLE,
    > `POSTCODE` STRING,
    > `POS_ID` STRING,
    > `TRANSACTION_DT` TIMESTAMP,
    > `STATUS` STRING)
    > ROW FORMAT DELIMITED
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES
    > ("hbase.columns.mapping"=":key, card_transactions_family:card_id, card_transactions_family:member_id, card_transa
ctions_family:amount, card_transactions_family:postcode, card_transactions_family:pos_id, card_transactions_family:tran
saction_dt, card_transactions_family:status")
    > TBLPROPERTIES ("hbase.table.name"="card_transactions_hive");
OK
Time taken: 2.645 seconds
```

8. Load data in card_transactions_hbase which will be visible in HBase as well with name as card_transactions_hive. Use randomUUID to populate TRANSACTION_ID field which will become row key in HBase effectively.

INSERT OVERWRITE TABLE CARD_TRANSACTIONS_HBASE
SELECT
reflect('java.util.UUID', 'randomUUID') as TRANSACTION_ID, CARD_ID, MEMBER_ID, AMOUNT,
POSTCODE, POS_ID, TRANSACTION_DT, STATUS
FROM CARD_TRANSACTIONS_ORC;

```
hive> INSERT OVERWRITE TABLE CARD_TRANSACTIONS_HBASE
    > SELECT
    > reflect('java.util.UUID', 'randomUUID') as TRANSACTION_ID, CARD_ID, MEMBER
_ID, AMOUNT, POSTCODE, POS_ID, TRANSACTION_DT, STATUS
    > FROM CARD_TRANSACTIONS_ORC;
Query ID = hdfs_20230518094547_e328dfed-48a8-4c16-8dbf-67b94f22f298
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1684390665990
_0005)

Map 1: 0/1
Map 1: 0/1
Map 1: 0(+1)/1
Map 1: 0(+1)/1
Map 1: 0(+1)/1
Map 1: 1/1
OK
Time taken: 24.903 seconds
```

9. Check some data in card_transactions_hbase.

select * from card_transactions_hbase limit 10;

```
hive> select * from card_transactions_hbase limit 10;
OK
0000e47c-07b6-46fd-892e-54f2eb4a65cc    4863127030291206        7883348231400969
180153.0        12495   572821677272050 2017-10-18 19:09:36     GENUINE
000360f1-a497-48a8-b471-feb27897ab41    4115056128951163        8405266842701411
590603.0        76566   167237778761295 2017-09-15 22:53:48     GENUINE
0003c2a4-7bb0-4643-b474-02cf06c437e2    341920010925925 185524819205415 4237138.
0       65734   935721714228987 2017-03-31 11:45:13     GENUINE
0003e302-418c-4883-8776-4d74123dc865    345969858843082 035217612427824 4497071.
0       14745   457146580055026 2018-01-06 03:59:03     GENUINE
00056a09-f965-48fa-93e7-ad0aa72a2c72    5137733204831953        0499338382710826
993746.0        58835   282691592931003 2017-02-19 09:41:09     GENUINE
00071039-a80f-4bbf-ac2a-bf41d95dd49e    5397412643360495        9448781983330798
528396.0        80520   770813269233619 2017-06-20 14:42:43     GENUINE
000790d4-365e-4985-a131-f30ecd7f2c90    347816334672492 314841462077900 2098057.
0       10993   693115489753967 2016-02-06 20:27:52     GENUINE
00082e95-5595-4495-a5cd-c98a83a70a14    5342400571435088        0087322675886723
5304.0  14887   202751295246195 2017-03-01 18:58:50     GENUINE
000995ed-fa3c-42f4-aab0-221b2eccadc9    5140973081437202        5060194806436666
577155.0        49918   560182011727900 2017-01-05 18:46:15     GENUINE
00099ab5-76f4-45c6-868e-dd2fcd70dcff    4750699680073601        6772957250877343
398741.0        93204   981461330778216 2017-11-22 06:02:49     GENUINE
Time taken: 0.269 seconds, Fetched: 10 row(s)
hive>
```

HBase Commands

10. Start HBase shell from command prompt. In HBase, check details of card_transactions_hive hive-hbase integrated table.

```
[root@ip-172-31-92-226 ~]# hbase shell
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
Version 1.4.13, rUnknown, Fri Apr 17 15:18:24 UTC 2020

hbase(main):001:0> describe 'card_transactions_hive'
Table card_transactions_hive is ENABLED
card_transactions_hive
COLUMN FAMILIES DESCRIPTION
{NAME => 'card_transactions_family', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_M
EMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'tru
e', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.3950 seconds
```

11. In HBase, check count in card_transactions_hive in HBase

count 'card_transactions_hive'

```
hbase(main):002:0> count 'card_transactions_hive'
Current count: 1000, row: 04d1f3b4-6a0f-4ce6-86d3-4e260481d1de
Current count: 2000, row: 099765ca-3222-4739-b554-8db61ef13a51
Current count: 3000, row: 0e932cba-409b-47f9-bb2c-a61e8891daad
Current count: 4000, row: 1331ade7-9799-4263-87cf-0b58f975e4ad
Current count: 5000, row: 18071f80-c43c-4d3f-a883-2dbda062f203
Current count: 6000, row: 1d09746a-937b-4258-969f-e7990a4c4122
Current count: 7000, row: 21c863aa-fcea-4cf4-adc0-3c8a8497e0ea
Current count: 8000, row: 265d3730-7161-4cbc-8b60-d7589be900f1
Current count: 9000, row: 2b2e8a6b-df9a-43ab-8f2d-5394a26f0583
Current count: 10000, row: 2ffd6027-1797-47d0-9ac2-090146e47604
Current count: 11000, row: 349e205f-e75c-426a-915a-801f1fe0fd1d
Current count: 12000, row: 3966a7e3-0be0-42d9-b6e0-02da867ab4d1
Current count: 13000, row: 3e387af9-7159-4f85-9c17-190d2a42e3b5
Current count: 14000, row: 42d8adeb-5dab-43f5-ae02-46aabe7d3616
Current count: 15000, row: 47d35c83-f072-4b5b-a118-e07e8c6e345b
Current count: 16000, row: 4c6c5981-8948-4125-97a0-366a1091ba28
Current count: 17000, row: 513d62fb-cd92-4483-889c-ea3db022834d
Current count: 18000, row: 5635a79e-72dc-4de2-8a09-b956a2b5e23c
Current count: 19000, row: 5adb2e30-8fc4-4f96-8c01-6eaaf347c21f
Current count: 20000, row: 5f76caa5-72bd-43bd-b3ba-f813580560e3
Current count: 21000, row: 64501917-cc13-45bd-9c60-f30eb6b2e526
Current count: 22000, row: 691f4aa0-1c2a-4ba8-af34-9120e1c724be
Current count: 23000, row: 6e0a21c4-f23c-41d0-b98c-7a86db37eeef
Current count: 24000, row: 7310ff96-ddd2-4ca8-9946-9db56bcd0c14
Current count: 25000, row: 77ddd7b9-c792-44e9-9155-8856aa419d51
Current count: 26000, row: 7cda3499-0382-4804-a1b5-029418153206
Current count: 27000, row: 818964b5-cc56-4990-ab00-edc58f3406e8
Current count: 28000, row: 86568c96-d150-4b78-a213-fbfcd3357bd2
Current count: 29000, row: 8b455c01-aff5-4c2d-acb7-64987f1b7cd5
Current count: 30000, row: 8ffe4acf-6712-4be5-8b42-eb164ea6eeb4
Current count: 31000, row: 94ee05c1-4fda-478e-bcab-9f823e4c46cf
Current count: 32000, row: 99c95dc6-5cfe-4005-a8e7-024628ab5b3a
Current count: 33000, row: 9e70dfe4-8722-45a0-855d-10800de96226
Current count: 34000, row: a2f4900d-a12a-441b-88ff-b84c5513df3a
Current count: 35000, row: a77ff886-964f-4bfc-87ae-82933a6ba0e6
Current count: 36000, row: ac88c76b-dc08-4b4f-84d9-e41e33370dce
Current count: 37000, row: b12a25e3-9608-4919-8318-94a43870709f
Current count: 38000, row: b5fdcd7b-87c0-492d-9b21-6527b72b3f8f
Current count: 39000, row: bab99808-7393-4d14-b32d-7da182f09c99
Current count: 40000, row: bf808783-abdb-4d51-8ddb-f2244489d0de
Current count: 41000, row: c45d17af-c641-4e6e-bc61-94b3ffda6816
Current count: 42000, row: c92ed8ab-35eb-47a4-8832-40c112b53177
Current count: 43000, row: ce27b524-6a9d-401b-ac09-9672d70702e6
Current count: 44000, row: d32f4a39-f997-4839-a6f1-81548fff5716
Current count: 45000, row: d81f5858-2a50-4a2f-b9ac-f792667242d6
Current count: 46000, row: dcfd9f3b-76e5-4792-aef2-c1c2e8a138c4
Current count: 47000, row: e1ff170e-6ed8-4aad-9015-a699c514652d
Current count: 48000, row: e6b6b5fd-612e-49cb-82b3-51e7089915ba
Current count: 49000, row: eb677868-b820-4cf4-b556-962f65ad3b38
Current count: 50000, row: f050703e-3bf8-4b4a-ab34-d4b5817e52da
Current count: 51000, row: f50cd6bc-9e18-466f-b405-ab8445661124
Current count: 52000, row: f9b5cfde-03f7-4bcd-afca-b4040e48cf3d
Current count: 53000, row: fe987fec-642c-4460-bac4-ce8d6c47bb0f
53292 row(s) in 3.4500 seconds

=> 53292
```

Task 2: Write a script to ingest the relevant data from AWS RDS to Hadoop

Sqoop Commands.

1. Run below Sqoop command to import member_score table from RDS into HDFS, from command prompt.

sqoop import --connect jdbc:mysql://upgradawsrds1.cyaielc9bmnf.us-east-1.rds.amazonaws.com/cred_financials_data --username upgraduser --password upgraduser --table member_score --null-string 'NA' --null-non-string '\\N' --delete-target-dir --target-dir '/capstone_project/member_score' -m 1

```
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
23/05/21 07:21:43 INFO sqoop.Sqoop: Running Sqoop version: 1.4.7
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/share/aws/redshift/jdbc/redshift-jdbc42-1.2.37.1061.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
23/05/21 07:21:44 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
23/05/21 07:21:44 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
23/05/21 07:21:44 INFO tool.CodeGenTool: Beginning code generation
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatical
ly registered via the SPI and manual loading of the driver class is generally unnecessary.
23/05/21 07:21:45 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `member_score` AS t LIMIT 1
23/05/21 07:21:45 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `member_score` AS t LIMIT 1
23/05/21 07:21:45 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /usr/lib/hadoop-mapreduce
Note: /tmp/sqoop-hadoop/compile/d8ce8213316f36df6c5fa5f56a4c7384/member_score.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
23/05/21 07:21:49 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-hadoop/compile/d8ce8213316f36df6c5fa5f56a4c7384/member_score
.jar
23/05/21 07:21:51 INFO tool.ImportTool: Destination directory /capstone_project/member_score is not present, hence not deleting.
23/05/21 07:21:51 WARN manager.MySQLManager: It looks like you are importing from mysql.
23/05/21 07:21:51 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
23/05/21 07:21:51 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
23/05/21 07:21:51 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
23/05/21 07:21:51 INFO mapreduce.ImportJobBase: Beginning import of member_score
23/05/21 07:21:51 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
23/05/21 07:21:51 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
23/05/21 07:21:52 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-92-66.ec2.internal/172.31.92.66:8032
23/05/21 07:21:55 INFO db.DBInputFormat: Using read commited transaction isolation
23/05/21 07:21:55 INFO mapreduce.JobSubmitter: number of splits:1
23/05/21 07:21:55 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684652299850_0002
23/05/21 07:21:56 INFO impl.YarnClientImpl: Submitted application application_1684652299850_0002
23/05/21 07:21:56 INFO mapreduce.Job: The url to track the job: http://ip-172-31-92-66.ec2.internal:20888/proxy/application_168465229985
0_0002/
23/05/21 07:21:56 INFO mapreduce.Job: Running job: job_1684652299850_0002
23/05/21 07:22:04 INFO mapreduce.Job: Job job_1684652299850_0002 running in uber mode : false
23/05/21 07:22:04 INFO mapreduce.Job:  map 0% reduce 0%
23/05/21 07:22:11 INFO mapreduce.Job:  map 100% reduce 0%
23/05/21 07:22:11 INFO mapreduce.Job: Job job_1684652299850_0002 completed successfully
23/05/21 07:22:11 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=189631
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=87
                HDFS: Number of bytes written=19980
                HDFS: Number of read operations=4
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=213456
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=4447
                Total vcore-milliseconds taken by all map tasks=4447
                Total megabyte-milliseconds taken by all map tasks=6830592
        Map-Reduce Framework
                Map input records=999
                Map output records=999
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=75
                CPU time spent (ms)=2170
                Physical memory (bytes) snapshot=266850304
                Virtual memory (bytes) snapshot=3281862656
                Total committed heap usage (bytes)=242745344
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=19980
23/05/21 07:22:11 INFO mapreduce.ImportJobBase: Transferred 19.5117 KB in 19.6785 seconds (1,015.3227 bytes/sec)
23/05/21 07:22:11 INFO mapreduce.ImportJobBase: Retrieved 999 records.
```

2. Run below Sqoop command to import card_member table from RDS into HDFS, from command prompt.

sqoop import --connect jdbc:mysql://upgradawsrds1.cyaielc9bmnf.us-east-1.rds.amazonaws.com/cred_financials_data --username upgraduser --password upgraduser --table card_member --null-string 'NA' --null-non-string '\\N' --delete-target-dir --target-dir '/capstone_project/card_member' -m 1

```
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
23/05/21 07:28:19 INFO sqoop.Sqoop: Running Sqoop version: 1.4.7
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/share/aws/redshift/jdbc/redshift-jdbc42-1.2.37.1061.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
23/05/21 07:28:19 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
23/05/21 07:28:19 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
23/05/21 07:28:19 INFO tool.CodeGenTool: Beginning code generation
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatical
ly registered via the SPI and manual loading of the driver class is generally unnecessary.
23/05/21 07:28:19 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `member_score` AS t LIMIT 1
23/05/21 07:28:20 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `member_score` AS t LIMIT 1
23/05/21 07:28:20 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /usr/lib/hadoop-mapreduce
Note: /tmp/sqoop-hadoop/compile/93e78845b87762a455efc0b8dd316de7/member_score.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
23/05/21 07:28:22 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-hadoop/compile/93e78845b87762a455efc0b8dd316de7/member_score
.jar
23/05/21 07:28:24 INFO tool.ImportTool: Destination directory /capstone_project/card_member is not present, hence not deleting.
23/05/21 07:28:24 WARN manager.MySQLManager: It looks like you are importing from mysql.
23/05/21 07:28:24 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
23/05/21 07:28:24 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
23/05/21 07:28:24 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
23/05/21 07:28:24 INFO mapreduce.ImportJobBase: Beginning import of member_score
23/05/21 07:28:24 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
23/05/21 07:28:24 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
23/05/21 07:28:24 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-92-66.ec2.internal/172.31.92.66:8032
23/05/21 07:28:26 INFO db.DBInputFormat: Using read commited transaction isolation
23/05/21 07:28:26 INFO mapreduce.JobSubmitter: number of splits:1
23/05/21 07:28:26 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684652299850_0003
23/05/21 07:28:27 INFO impl.YarnClientImpl: Submitted application application_1684652299850_0003
23/05/21 07:28:27 INFO mapreduce.Job: The url to track the job: http://ip-172-31-92-66.ec2.internal:20888/proxy/application_168465229985
0_0003/
23/05/21 07:28:27 INFO mapreduce.Job: Running job: job_1684652299850_0003
23/05/21 07:28:35 INFO mapreduce.Job: Job job_1684652299850_0003 running in uber mode : false
23/05/21 07:28:35 INFO mapreduce.Job:  map 0% reduce 0%
23/05/21 07:28:41 INFO mapreduce.Job:  map 100% reduce 0%
23/05/21 07:28:41 INFO mapreduce.Job: Job job_1684652299850_0003 completed successfully
23/05/21 07:28:41 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=189630
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=87
                HDFS: Number of bytes written=19980
                HDFS: Number of read operations=4
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=174816
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=3642
                Total vcore-milliseconds taken by all map tasks=3642
                Total megabyte-milliseconds taken by all map tasks=5594112
        Map-Reduce Framework
                Map input records=999
                Map output records=999
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=74
                CPU time spent (ms)=2160
                Physical memory (bytes) snapshot=269864960
                Virtual memory (bytes) snapshot=3286827008
                Total committed heap usage (bytes)=244842496
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=19980
23/05/21 07:28:41 INFO mapreduce.ImportJobBase: Transferred 19.5117 KB in 17.3427 seconds (1.1251 KB/sec)
23/05/21 07:28:41 INFO mapreduce.ImportJobBase: Retrieved 999 records.
```

Hive Commands

1. Start hive from command prompt. Create external table card_member_ext which will point to HDFS location to hold data from card_member table in RDS. Sqoop command will write in this location.

```
CREATE EXTERNAL TABLE IF NOT EXISTS CARD_MEMBER_EXT(
`CARD_ID` STRING,
`MEMBER_ID` STRING,
`MEMBER_JOINING_DT` TIMESTAMP,
`CARD_PURCHASE_DT` STRING,
`COUNTRY` STRING,
`CITY` STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/capstone_project/card_member';
```

```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>
    >
    > CREATE EXTERNAL TABLE IF NOT EXISTS CARD_MEMBER_EXT(
    > `CARD_ID` STRING,
    > `MEMBER_ID` STRING,
    > `MEMBER_JOINING_DT` TIMESTAMP,
    > `CARD_PURCHASE_DT` STRING,
    > `COUNTRY` STRING,
    > `CITY` STRING)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LOCATION '/capstone_project/card_member';
OK
Time taken: 0.723 seconds
```

2. Create external table member_score_ext which will point to HDFS location to hold data from member_score table in RDS. Sqoop command will write in this location.

```
CREATE EXTERNAL TABLE IF NOT EXISTS MEMBER_SCORE_EXT(
`MEMBER_ID` STRING,
`SCORE` INT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/capstone_project/member_score';
```

```
hive>
    > CREATE EXTERNAL TABLE IF NOT EXISTS MEMBER_SCORE_EXT(
    > `MEMBER_ID` STRING,
    > `SCORE` INT)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LOCATION '/capstone_project/member_score';
OK
Time taken: 0.058 seconds
```

3. Create card_member_orc table. ORC format will help in better performance.

```
CREATE TABLE IF NOT EXISTS CARD_MEMBER_ORC(
`CARD_ID` STRING,
`MEMBER_ID` STRING,
`MEMBER_JOINING_DT` TIMESTAMP,
`CARD_PURCHASE_DT` STRING,
`COUNTRY` STRING,
`CITY` STRING)
```

STORED AS ORC
TBLPROPERTIES ("orc.compress"="SNAPPY");

4. Create member_score_orc table. ORC format will help in better performance.

CREATE TABLE IF NOT EXISTS MEMBER_SCORE_ORC(
`MEMBER_ID` STRING,
`SCORE` INT)
STORED AS ORC
TBLPROPERTIES ("orc.compress"="SNAPPY");

```
hive>
    >
    > CREATE TABLE IF NOT EXISTS CARD_MEMBER_ORC(
    > `CARD_ID` STRING,
    > `MEMBER_ID` STRING,
    > `MEMBER_JOINING_DT` TIMESTAMP,
    > `CARD_PURCHASE_DT` STRING,
    > `COUNTRY` STRING,
    > `CITY` STRING)
    > STORED AS ORC
    > TBLPROPERTIES ("orc.compress"="SNAPPY");
OK
Time taken: 0.426 seconds
hive>
    > CREATE TABLE IF NOT EXISTS MEMBER_SCORE_ORC(
    > `MEMBER_ID` STRING,
    > `SCORE` INT)
    > STORED AS ORC
    > TBLPROPERTIES ("orc.compress"="SNAPPY");
OK
Time taken: 0.252 seconds
```

5. Load data into card_member_orc from card_member_ext.

INSERT OVERWRITE TABLE CARD_MEMBER_ORC
SELECT CARD_ID, MEMBER_ID, MEMBER_JOINING_DT, CARD_PURCHASE_DT, COUNTRY, CITY
FROM CARD_MEMBER_EXT;

```
hive>
    > INSERT OVERWRITE TABLE CARD_MEMBER_ORC
    > SELECT CARD_ID, MEMBER_ID, MEMBER_JOINING_DT, CARD_PURCHASE_DT, COUNTRY, CITY FROM CARD_MEMBER_EXT;
Query ID = hadoop_20230521074040_ee2e72f5-76e9-4f39-9de8-572bf6f5fcf8
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1684652299850_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 5.12 s
--------------------------------------------------------------------------------
Loading data to table default.card_member_orc
OK
Time taken: 16.71 seconds
hive>
    >
```

6. Load data into member_score_orc from member_score_ext.

INSERT OVERWRITE TABLE MEMBER_SCORE_ORC
SELECT MEMBER_ID, SCORE FROM MEMBER_SCORE_EXT;

```
hive>
    > INSERT OVERWRITE TABLE MEMBER_SCORE_ORC
    > SELECT MEMBER_ID, SCORE FROM MEMBER_SCORE_EXT;
Query ID = hadoop_20230521074149_742d60b3-7099-43d8-8a4f-afbfdca2c817
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1684652299850_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 5.63 s
--------------------------------------------------------------------------------
Loading data to table default.member_score_orc
OK
Time taken: 6.863 seconds
```

7. Verify some data in card_member_orc table.

SELECT * FROM CARD_MEMBER_ORC LIMIT 10;

```
hive> SELECT * FROM CARD_MEMBER_ORC LIMIT 10;
OK
340028465709212 009250698176266 2012-02-08 06:04:13     05/13   United States   Barberton
340054675199675 835873341185231 2017-03-10 09:24:44     03/17   United States   Fort Dodge
340082915339645 512969555857346 2014-02-15 06:30:30     07/14   United States   Graham
340134186926007 887711945571282 2012-02-05 01:21:58     02/13   United States   Dix Hills
340265728490548 680324265406190 2014-03-29 07:49:14     11/14   United States   Rancho Cucamonga
340268219434811 929799084911715 2012-07-08 02:46:08     08/12   United States   San Francisco
340379737226464 089615510858348 2010-03-10 00:06:42     09/10   United States   Clinton
340383645652108 181180599313885 2012-02-24 05:32:44     10/16   United States   West New York
340803866934451 417664728506297 2015-05-21 04:30:45     08/17   United States   Beaverton
340889618969736 459292914761635 2013-04-23 08:40:11     11/15   United States   West Palm Beach
Time taken: 0.097 seconds, Fetched: 10 row(s)
```

8. Verify some data in member_score_orc table.
SELECT * FROM MEMBER_SCORE_ORC LIMIT 10;

```
hive>
    > SELECT * FROM MEMBER_SCORE_ORC LIMIT 10;
OK
000037495066290 339
000117826301530 289
001147922084344 393
001314074991813 225
001739553947511 642
003761426295463 413
004494068832701 217
006836124210484 504
006991872634058 697
007955566230397 372
Time taken: 0.125 seconds, Fetched: 10 row(s)
```

Task 3: Create Look-up table with columns specified earlier in the problem statement.
Write a script to calculate the moving average and standard deviation of the last 10 transactions for each card_id for the data present in Hadoop and NoSQL database. If the total number of transactions for a particular card_id is less than 10, then calculate the parameters based on the total number of records available for that card_id. The script should be able to extract and feed the other relevant data ('postcode', 'transaction_dt', 'score', etc.) for the look-up table along with card_id and UCL.

Hive Commands

1. Create lookup_data_hbase hive-hbase integrated table which will be visible in HBase as well with name as lookup_data_hive.

```
CREATE TABLE LOOKUP_DATA_HBASE(`CARD_ID` STRING,`UCL` DOUBLE, `SCORE` INT,
`POSTCODE` STRING, `TRANSACTION_DT` TIMESTAMP)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping"=":key, lookup_card_family:ucl,
lookup_card_family:score, lookup_transaction_family:postcode,
lookup_transaction_family:transaction_dt")
TBLPROPERTIES ("hbase.table.name" = "lookup_data_hive");
```

```
hive>
    >
    >
    >
    > CREATE TABLE LOOKUP_DATA_HBASE(`CARD_ID` STRING,`UCL` DOUBLE, `SCORE` INT, `POSTCODE` STRING, `TRANSACTION_DT` TI
MESTAMP)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping"=":key, lookup_card_family:ucl, lookup_card_family:score, lookup_tra
nsaction_family:postcode, lookup_transaction_family:transaction_dt")
    > TBLPROPERTIES ("hbase.table.name" = "lookup_data_hive");
OK
Time taken: 2.382 seconds
hive>
```

2. In HBase, check details of lookup_data_hive hive-hbase integrated table

```
describe 'lookup_data_hive'
```

```
hbase(main):030:0* describe 'lookup_data_hive'
Table lookup_data_hive is ENABLED
lookup_data_hive
COLUMN FAMILIES DESCRIPTION
{NAME => 'lookup_card_family', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BL
OCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'lookup_transaction_family', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_
MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
 TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'tr
ue', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
2 row(s) in 0.0190 seconds

hbase(main):031:0>
```

3. In HBase, alter lookup_data_hive table and set VERSIONS to 10 for lookup_transaction_family. We are supposed to store last 10 transactions in lookup table so altering VERSIONS to 10. I have created 2 column families in lookup table namely lookup_card_family and lookup_transaction_family. Column family lookup_card_family has score and ucl as columns and will store only 1 VERSION. Column family lookup_transaction_family has postcode and transaction_dt and will store 10 VERSIONS

alter 'lookup_data_hive', {NAME => 'lookup_transaction_family', VERSIONS => 10}

```
hbase(main):031:0> alter 'lookup_data_hive', {NAME => 'lookup_transaction_family
', VERSIONS => 10}
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 2.4200 seconds
```

4. In HBase, check details of lookup_data_hive and confirm that VERSIONS is set to 10 for lookup_transaction_family.
   describe 'lookup_data_hive'

```
hbase(main):032:0> describe 'lookup_data_hive'
Table lookup_data_hive is ENABLED
lookup_data_hive
COLUMN FAMILIES DESCRIPTION
{NAME => 'lookup_card_family', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BL
OCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'lookup_transaction_family', BLOOMFILTER => 'ROW', VERSIONS => '10', IN
_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE'
, TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 't
rue', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
2 row(s) in 0.0170 seconds
```

5. Create table ranked_card_transactions_orc to store last 10 transactions for each card_id. ORC format will help in better performance.

```
CREATE TABLE IF NOT EXISTS RANKED_CARD_TRANSACTIONS_ORC(
`CARD_ID` STRING,
`AMOUNT` DOUBLE,
`POSTCODE` STRING,
`TRANSACTION_DT` TIMESTAMP,
`RANK` INT)
STORED AS ORC
TBLPROPERTIES ("orc.compress"="SNAPPY");
```

```
hive> CREATE TABLE IF NOT EXISTS RANKED_CARD_TRANSACTIONS_ORC(
    > `CARD_ID` STRING,
    > `AMOUNT` DOUBLE,
    > `POSTCODE` STRING,
    > `TRANSACTION_DT` TIMESTAMP,
    > `RANK` INT)
    > STORED AS ORC
    > TBLPROPERTIES ("orc.compress"="SNAPPY");
OK
Time taken: 0.052 seconds
```

6. Create table card_ucl_orc to store UCL values for each card_id. ORC format will help in better performance.

```
CREATE TABLE IF NOT EXISTS CARD_UCL_ORC(
`CARD_ID` STRING,
`UCL` DOUBLE)
STORED AS ORC
TBLPROPERTIES ("orc.compress"="SNAPPY");
```

```
hive> CREATE TABLE IF NOT EXISTS CARD_UCL_ORC(
    > `CARD_ID` STRING,
    > `UCL` DOUBLE)
    > STORED AS ORC
    > TBLPROPERTIES ("orc.compress"="SNAPPY");
OK
Time taken: 0.051 seconds
```

7. Load data in ranked_card_transactions_orc table. Here for each card id get top 10 transactions based on the amount column. This is done with SQL using Rank() function partition by card_id sorted by amount in descrending order with max # of transactions <= 10.

```
INSERT OVERWRITE TABLE RANKED_CARD_TRANSACTIONS_ORC
SELECT B.CARD_ID, B.AMOUNT, B.POSTCODE, B.TRANSACTION_DT, B.RANK FROM
(SELECT A.CARD_ID, A.AMOUNT, A.POSTCODE, A.TRANSACTION_DT, RANK()
OVER(PARTITION BY A.CARD_ID ORDER BY A.TRANSACTION_DT DESC, AMOUNT DESC) AS
RANK FROM
(SELECT CARD_ID, AMOUNT, POSTCODE, TRANSACTION_DT FROM
CARD_TRANSACTIONS_HBASE WHERE
STATUS = 'GENUINE') A ) B WHERE B.RANK <= 10;
```

```
hive>
    > INSERT OVERWRITE TABLE RANKED_CARD_TRANSACTIONS_ORC
    > SELECT B.CARD_ID, B.AMOUNT, B.POSTCODE, B.TRANSACTION_DT, B.RANK FROM
    > (SELECT A.CARD_ID, A.AMOUNT, A.POSTCODE, A.TRANSACTION_DT, RANK() OVER(PARTITION BY A.CARD_ID ORDER BY A.TRANSACTION_DT DESC, AMOU
NT DESC) AS RANK FROM
    > (SELECT CARD_ID, AMOUNT, POSTCODE, TRANSACTION_DT FROM CARD_TRANSACTIONS_HBASE WHERE
    > STATUS = 'GENUINE') A ) B WHERE B.RANK <= 10;
Query ID = hadoop_20230521080530_07c8a249-1e60-4471-91fe-5a84b56e3d9e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1684652299850_0006)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 11.55 s
----------------------------------------------------------------------------------------------
Loading data to table capstone_project.ranked_card_transactions_orc
OK
Time taken: 15.746 seconds
hive>
```

8. Load data in card_ucl_orc table. In innermost query, select card_id, average of amount and standard deviation of amount from card_transactions_orc. In outermost query, select card_id and compute UCL using average and standard deviation with formula (avg + (3 * stddev)). Insert all this data in card_ucl_orc.

```
INSERT OVERWRITE TABLE CARD_UCL_ORC
SELECT A.CARD_ID, (A.AVERAGE + (3 * A.STANDARD_DEVIATION)) AS UCL FROM (
SELECT CARD_ID, AVG(AMOUNT) AS AVERAGE, STDDEV(AMOUNT) AS STANDARD_DEVIATION
FROM
RANKED_CARD_TRANSACTIONS_ORC
GROUP BY CARD_ID) A;
```

```
hive>
    >
    > INSERT OVERWRITE TABLE CARD_UCL_ORC
    > SELECT A.CARD_ID, (A.AVERAGE + (3 * A.STANDARD_DEVIATION)) AS UCL FROM (
    > SELECT CARD_ID, AVG(AMOUNT) AS AVERAGE, STDDEV(AMOUNT) AS STANDARD_DEVIATION FROM
    > RANKED_CARD_TRANSACTIONS_ORC
    > GROUP BY CARD_ID) A;
Query ID = hadoop_20230521081015_dcf978e9-667c-4cc8-80bc-df2c0ffc81a7
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1684652299850_0006)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 6.43 s
----------------------------------------------------------------------------------------------
Loading data to table capstone_project.card_ucl_orc
OK
Time taken: 7.568 seconds
```

9. Load data in lookup_data_hbase table. Create intermediate table or sort of inline view which can be used in JOIN condition by selecting card_id, score from card_member_orc joining member_score_orc on member_id and name it as CMS. In main query, select card_id, UCL, score, postcode, transaction_dt from ranked_card_transactions_orc joining card_ucl_orc on card_id column and joining cms on card_id where rank is 1. This will ensure that we have obtained data of latest transaction for each card_id.

```
INSERT OVERWRITE TABLE LOOKUP_DATA_HBASE
SELECT RCTO.CARD_ID, CUO.UCL, CMS.SCORE, RCTO.POSTCODE, RCTO.TRANSACTION_DT
FROM RANKED_CARD_TRANSACTIONS_ORC RCTO
JOIN CARD_UCL_ORC CUO
ON CUO.CARD_ID = RCTO.CARD_ID
JOIN (
SELECT DISTINCT CARD.CARD_ID, SCORE.SCORE
FROM CARD_MEMBER_ORC CARD
JOIN MEMBER_SCORE_ORC SCORE
ON CARD.MEMBER_ID = SCORE.MEMBER_ID) AS CMS
ON RCTO.CARD_ID = CMS.CARD_ID
WHERE RCTO.RANK = 1;
```

```
hive> INSERT OVERWRITE TABLE LOOKUP_DATA_HBASE
    > SELECT RCTO.CARD_ID, CUO.UCL, CMS.SCORE, RCTO.POSTCODE, RCTO.TRANSACTION_DT
    > FROM RANKED_CARD_TRANSACTIONS_ORC RCTO
    > JOIN CARD_UCL_ORC CUO
    > ON CUO.CARD_ID = RCTO.CARD_ID
    > JOIN (
    > SELECT DISTINCT CARD.CARD_ID, SCORE.SCORE
    > FROM CARD_MEMBER_ORC CARD
    > JOIN MEMBER_SCORE_ORC SCORE
    > ON CARD.MEMBER_ID = SCORE.MEMBER_ID) AS CMS
    > ON RCTO.CARD_ID = CMS.CARD_ID
    > WHERE RCTO.RANK = 1;
No Stats for capstone_project@ranked_card_transactions_orc, Columns: postcode, rank, transaction_dt, card_id
No Stats for capstone_project@card_ucl_orc, Columns: card_id, ucl
No Stats for capstone_project@card_member_orc, Columns: member_id, card_id
No Stats for capstone_project@member_score_orc, Columns: member_id, score
Query ID = hadoop_20230521081303_7c55ebf6-e939-46cd-b1e7-c1e5927a5d6e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1684652299850_0006)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 2 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 3 .......... container     SUCCEEDED      1          1        0        0       0       0
Map 5 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      2          2        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 05/05  [==========================>>] 100%  ELAPSED TIME: 11.78 s
--------------------------------------------------------------------------------
OK
Time taken: 17.342 seconds
```

10. Verify count in lookup_data_hbase table.

select count(*) from lookup_data_hbase;

```
hive> select count(*) from lookup_data_hbase;
Query ID = hadoop_20230521090038_7f2fb1fb-8a5b-4310-bd20-66b208b350e0
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1684652299850_0008)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 6.28 s
--------------------------------------------------------------------------------
OK
999
Time taken: 10.841 seconds, Fetched: 1 row(s)
```

11. Verify some data in lookup_data_hbase table.

select * from lookup_data_hbase limit 10;

select * from lookup_data_hbase limit 10;

```
hive> select * from lookup_data_hbase limit 10;
OK
340028465709212 1.6331555548882348E7    233     24658   2018-01-02 03:25:35
340054675199675 1.4156079786189131E7    631     50140   2018-01-15 19:43:23
340082915339645 1.5285685330791473E7    407     17844   2018-01-26 19:03:47
340134186926007 1.5239767522438556E7    614     67576   2018-01-18 23:12:50
340265728490548 1.608491671255562E7     202     72435   2018-01-21 02:07:35
340268219434811 1.2507323937605347E7    415     62513   2018-01-16 04:30:05
340379737226464 1.4198310998368107E7    229     26656   2018-01-27 00:19:47
340383645652108 1.4091750460468251E7    645     34734   2018-01-29 01:29:12
340803866934451 1.0843341196185412E7    502     87525   2018-01-31 04:23:57
340889618969736 1.3217942365515321E7    330     61341   2018-01-31 21:57:18
Time taken: 0.176 seconds, Fetched: 10 row(s)
```

Hbase Commands

1. Start HBase shell from command prompt. In HBase, check count in lookup_data_hive table.

count 'lookup_data_hive';

```
hbase(main):004:0> count 'lookup_data_hive'
999 row(s) in 0.2340 seconds

=> 999
```

2. In HBase, check data in lookup_data_hive table.

scan 'lookup_data_hive'

```
 6591175617713393                column=lookup_card_family:score, timestamp=1684659024744, value=568
 6591175617713393                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.6242373363420745E7
 6591175617713393                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=21048
 6591175617713393                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-31 13:10:37
 6592184145413632                column=lookup_card_family:score, timestamp=1684659024744, value=456
 6592184145413632                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.3323882099122094E7
 6592184145413632                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=53186
 6592184145413632                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-28 00:54:30
 6594248319343442                column=lookup_card_family:score, timestamp=1684659024744, value=350
 6594248319343442                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.4567957140418548E7
 6594248319343442                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=24927
 6594248319343442                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-31 23:42:38
 6595638658736751                column=lookup_card_family:score, timestamp=1684659024744, value=310
 6595638658736751                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.356629177577566E7
 6595638658736751                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=68328
 6595638658736751                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-30 10:50:34
 6595814135833988                column=lookup_card_family:score, timestamp=1684659024744, value=210
 6595814135833988                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.3926273240525039E7
 6595814135833988                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=22508
 6595814135833988                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-30 02:03:54
 6595928469079750                column=lookup_card_family:score, timestamp=1684659024744, value=412
 6595928469079750                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.142797041440079E7
 6595928469079750                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=98349
 6595928469079750                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-24 12:38:22
 6597703848279563                column=lookup_card_family:score, timestamp=1684659024744, value=218
 6597703848279563                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.4718634149498457E7
 6597703848279563                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=95699
 6597703848279563                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-27 10:51:49
 6598830758632447                column=lookup_card_family:score, timestamp=1684659024744, value=293
 6598830758632447                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.2227949982601807E7
 6598830758632447                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=19421
 6598830758632447                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-30 00:18:34
 6599900931314251                column=lookup_card_family:score, timestamp=1684659024744, value=297
 6599900931314251                column=lookup_card_family:ucl, timestamp=1684659024744, value=1.2121408572464656E7
 6599900931314251                column=lookup_transaction_family:postcode, timestamp=1684659024744, value=97423
 6599900931314251                column=lookup_transaction_family:transaction_dt, timestamp=1684659024744, value=2018-01-31 11:25:16
999 row(s) in 2.4440 seconds

hbase(main):006:0>
```