Capstone Project
Script Execution
(Credit Card Fraud Detection System)

Devashish Kapadia

Content

- SQOOP-JOB-CREATION-CARD_MEMBER- DATA IMPORT FROM AWS RDS.
- SQOOP-JOB-CREATION-MEMBER_SCORE- DATA IMPORT FROM AWS RDS.
- HBASE TABLE CREATION(card_transaction)
- HBASE TABLE CREATION(lookup table)
- HBASE TABLE CREATION(card member)
- Create mapping table in hive for HBASE card_transaction table
- Card transaction staging table in hive
- Load data from local path into staging table.
- Insert card transactions which has valid card id
- Dropping staging table
- Create mapping table in hive for Hbase card_member table
- Card member staging table in hive
- Insert data into HBase mapping table
- Create member score table
- Create lookup UCL staging table
- Calculate UCL and insert into lookup UCL staging table
- Create lookup full staging table for UCL & memscore
- Insert UCL and Score into lookup full staging table
- Create mapping table for HBase lookup table for loading limit attributes
- Insert UCL and Score into mapping lookup from staging table
- Create mapping table for HBase lookup table for loading status attributes
- Insert postal code and transaction date into mapping lookup from card_transations_hbase
- Create mapping table for Hbase lookup table

SQOOP-JOB-CREATION-CARD_MEMBER- DATA IMPORT FROM AWS RDS.

--Create sqoop job card_member ingest from AWS RDS to HADOOP

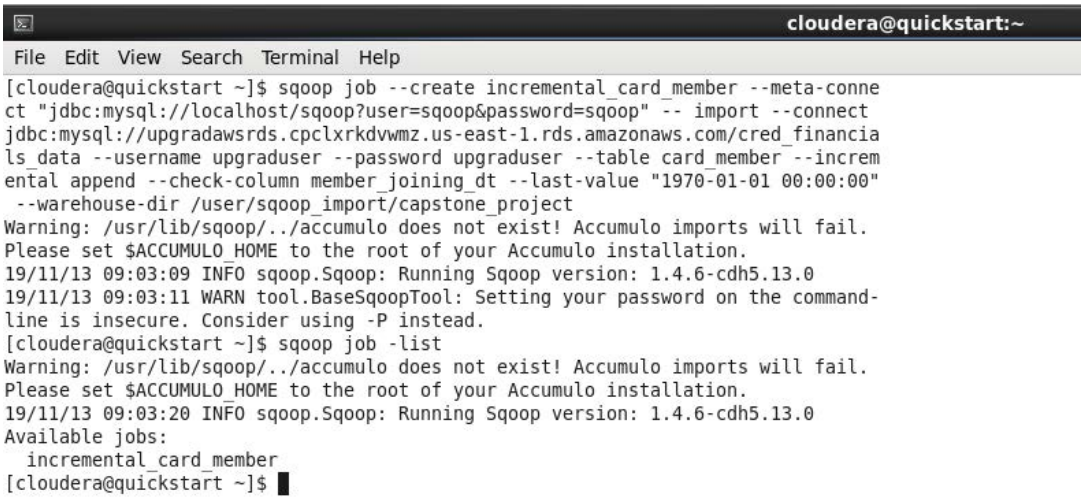
sqoop job --create incremental_card_member --meta-connect "jdbc:mysql://localhost/sqoop?user=sqoop&password=sqoop" -- import --connect jdbc:mysql://upgradawsrds.cpclxrkdvwmz.us-east-1.rds.amazonaws.com/cred_financials_data --username upgraduser --password upgraduser --table card_member --incremental append --check-column member_joining_dt --last-value "1970-01-01 00:00:00" --warehouse-dir /user/sqoop_import/capstone_project

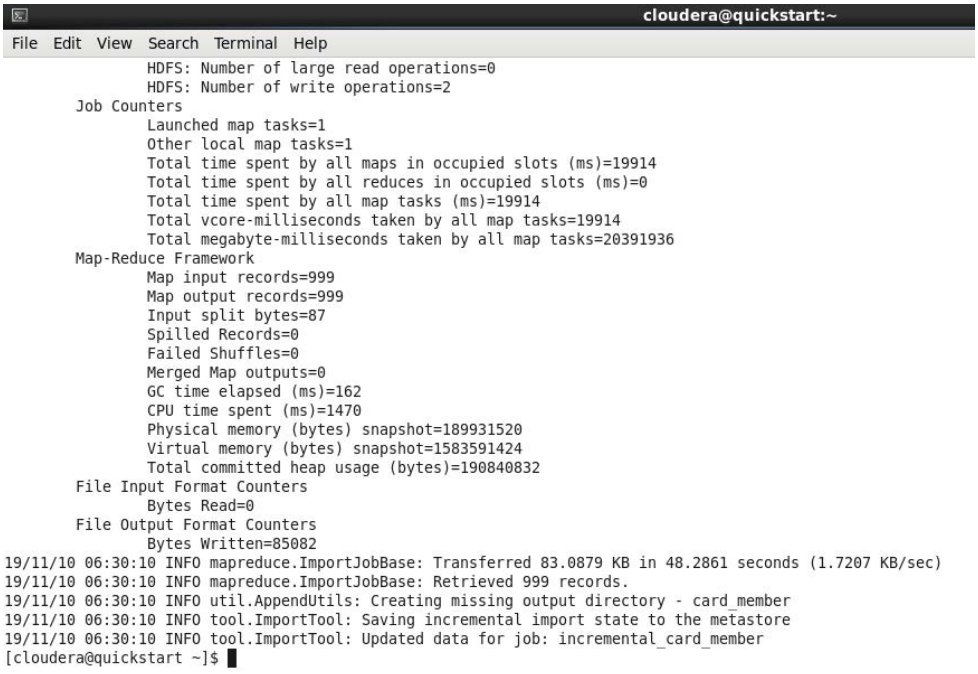
--JOB-verification

sqoop job --list

-- Execute the job once to get the initial data load

sqoop job --exec incremental_card_member





SQOOP-JOB-CREATION-MEMBER_SCORE- DATA IMPORT FROM AWS RDS.

--import for member_score table which is full load every time

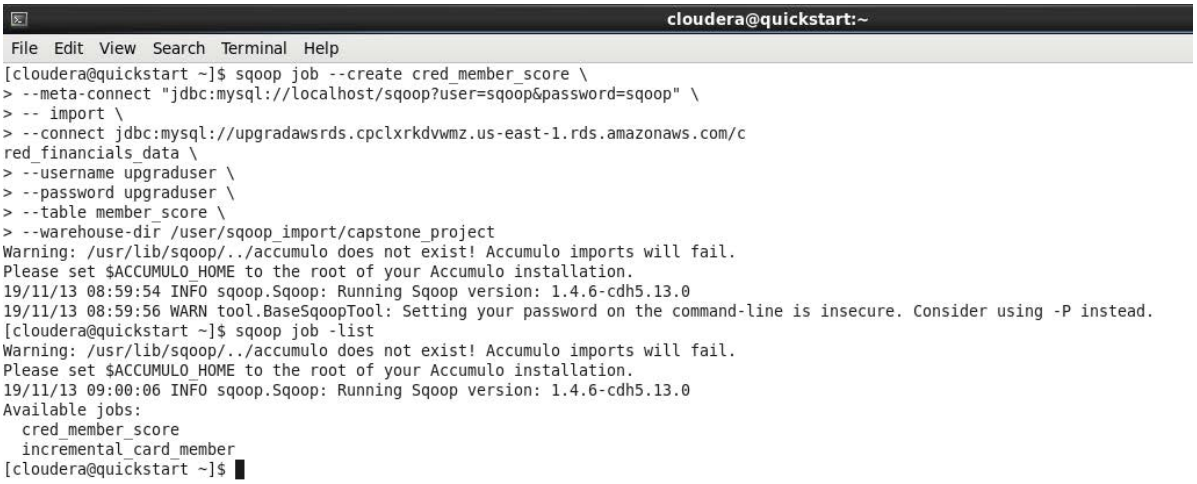
```
sqoop job --create cred_member_score \
--meta-connect "jdbc:mysql://localhost/sqoop?user=sqoop&password=sqoop" \
-- import \
--connect jdbc:mysql://upgradawsrds.cpclxrkdvwmz.us-east-
1.rds.amazonaws.com/cred_financials_data \
--username upgraduser \
--password upgraduser \
--table member_score \
--warehouse-dir /user/sqoop_import/capstone_project
```

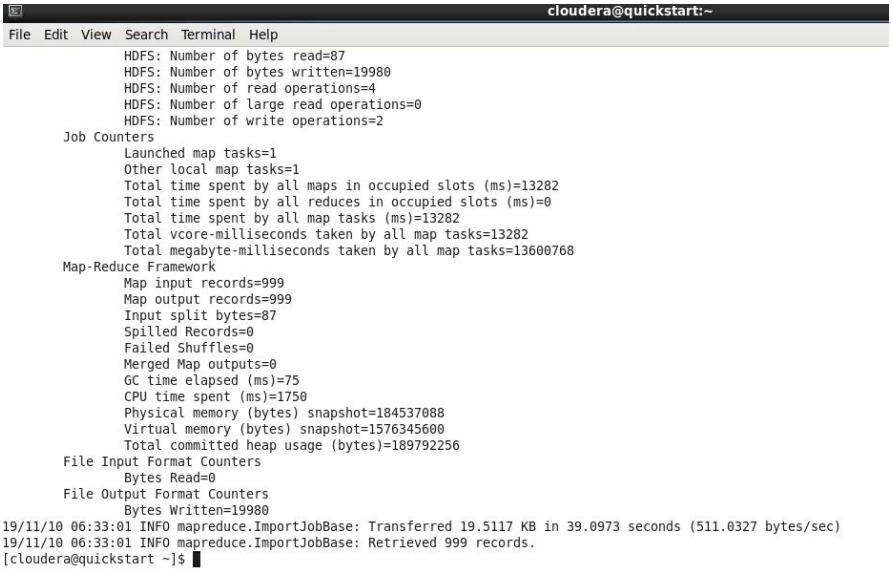
--JOB-verification

```
sqoop job --list
```

-- Execute the job once to get the initial data load

```
sqoop job --exec cred_member_score
```

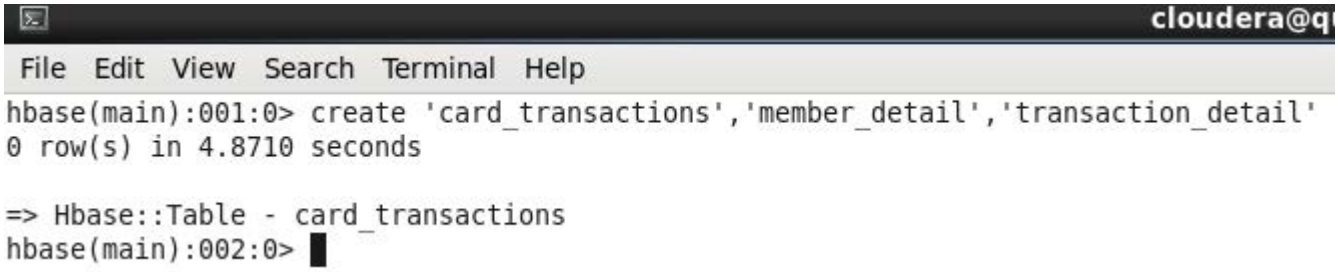




HBASE TABLE CREATION(card_transaction)

--Hbase shell command for table creation

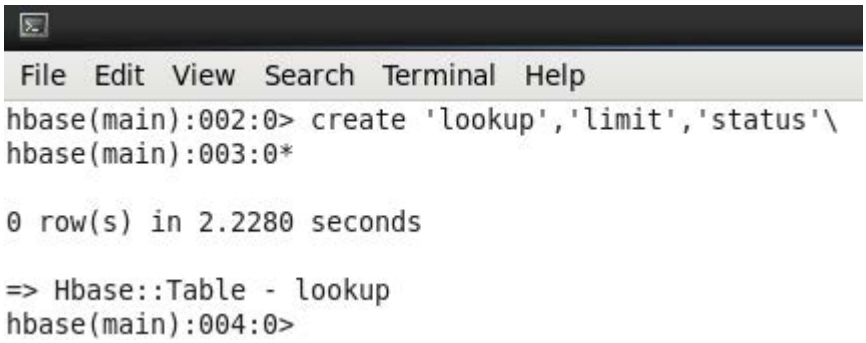
create 'card_transactions','member_detail','transaction_detail'



HBASE TABLE CREATION (lookup table)

--Hbase shell command for table creation

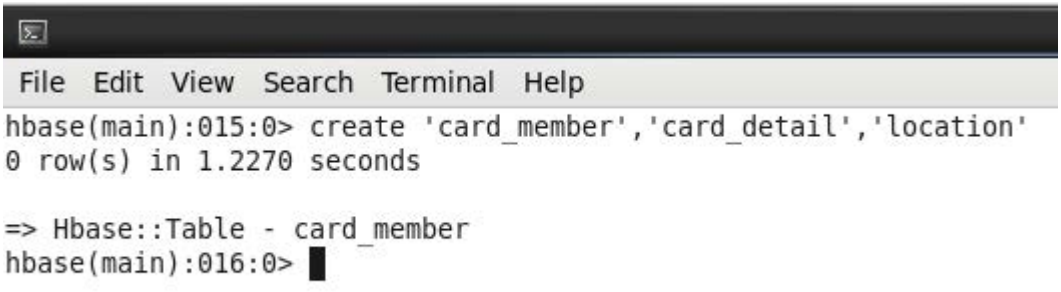
create 'lookup','limit','status'



HBASE TABLE CREATION (card member)

-- Hbase shell command for table creation

create 'card_member','card_detail','location'



Create mapping table in hive for HBASE card_transaction table

--create mapping table in hive for HBase card transaction table

create external table card_transations_hbase(key struct<member_id:bigint, transaction_dt:string, amount:bigint>,

card_id bigint, post_code bigint,

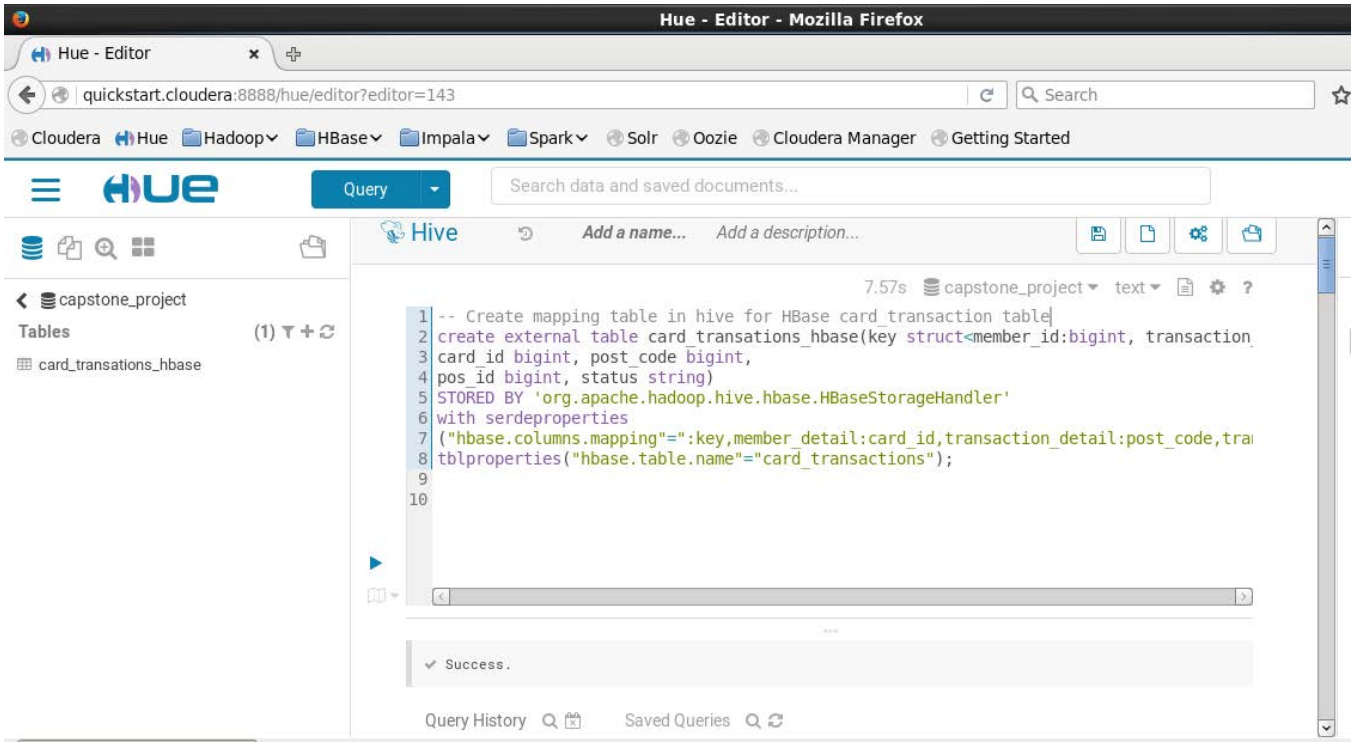
pos_id bigint, status string)

STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

with serdeproperties

("hbase.columns.mapping"=":key,member_detail:card_id,transaction_detail:post_code,transaction_det ail:pos_id,transaction_detail:status")

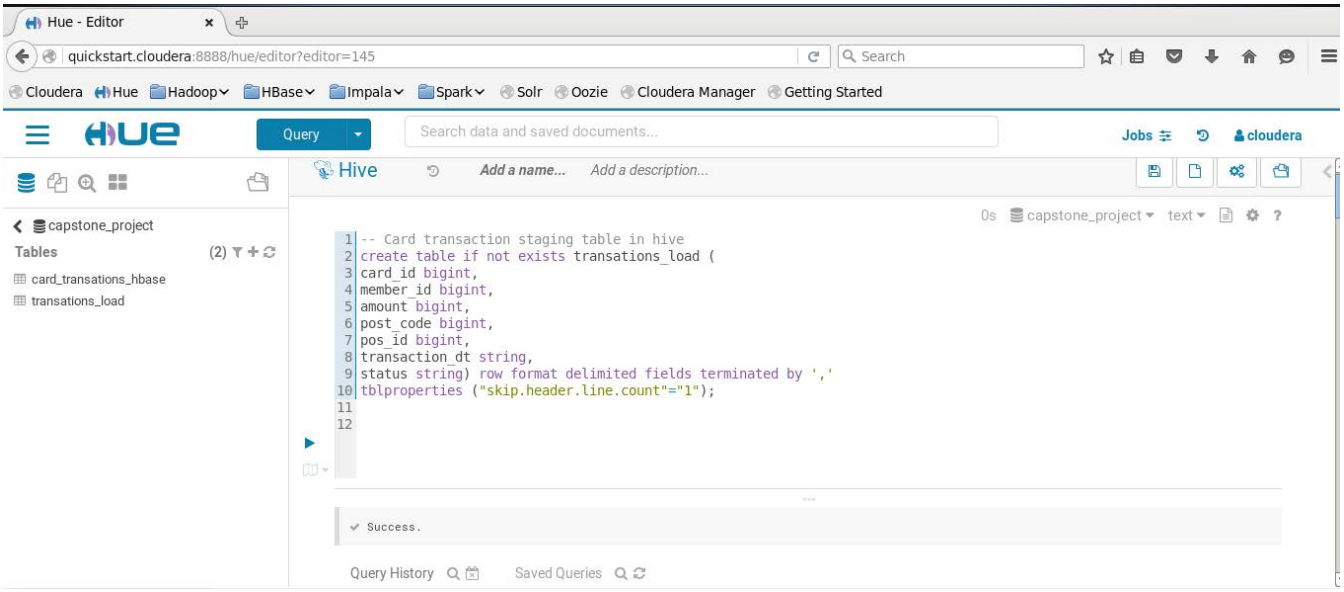
tblproperties("hbase.table.name"="card_transactions");



Card transaction staging table in hive

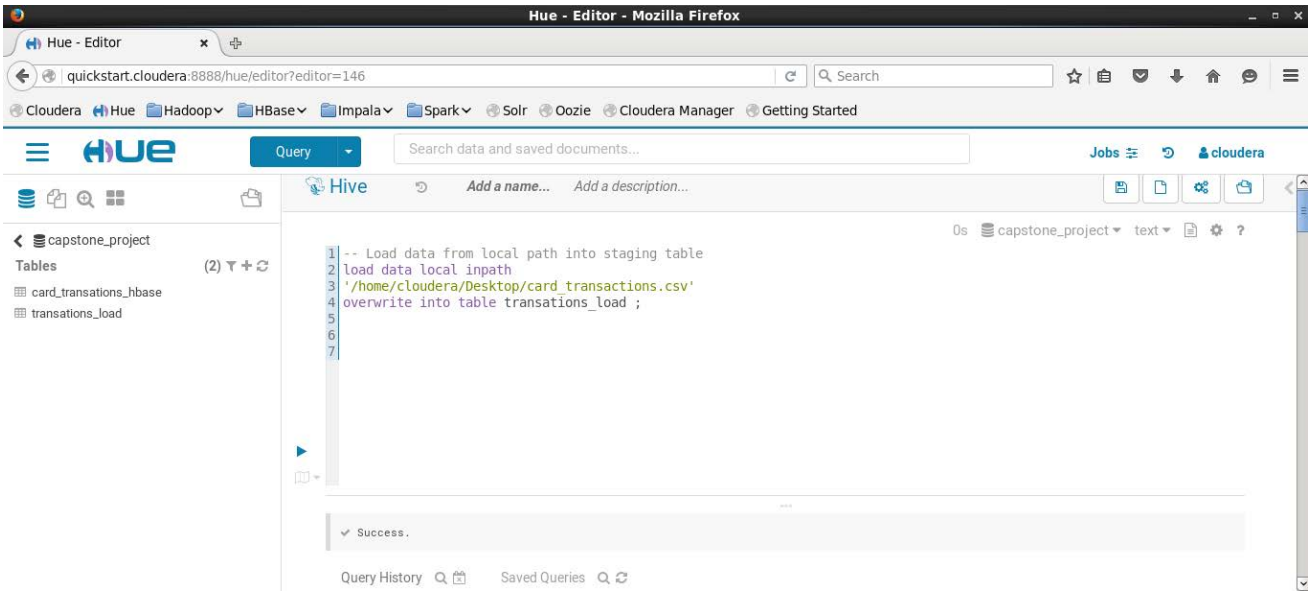
--Card transaction staging table in hive

```
create table if not exists transations_load (
card_id bigint,
member_id bigint,
amount bigint,
post_code bigint,
pos_id bigint,
transaction_dt string,
status string) row format delimited fields terminated by ',' tblproperties
("skip.header.line.count"="1");
```



Load data from local path into staging table.

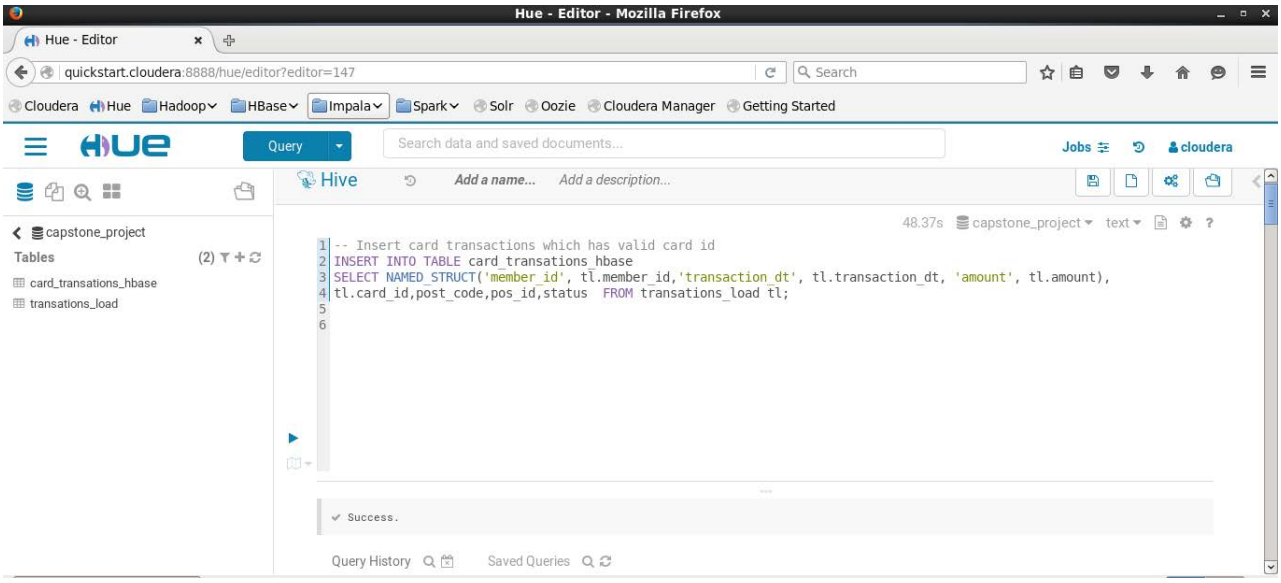
load data local inpath
'/home/cloudera/Desktop/card_transactions.csv'
overwrite into table transations_load ;



Inserting data into card transaction HBase

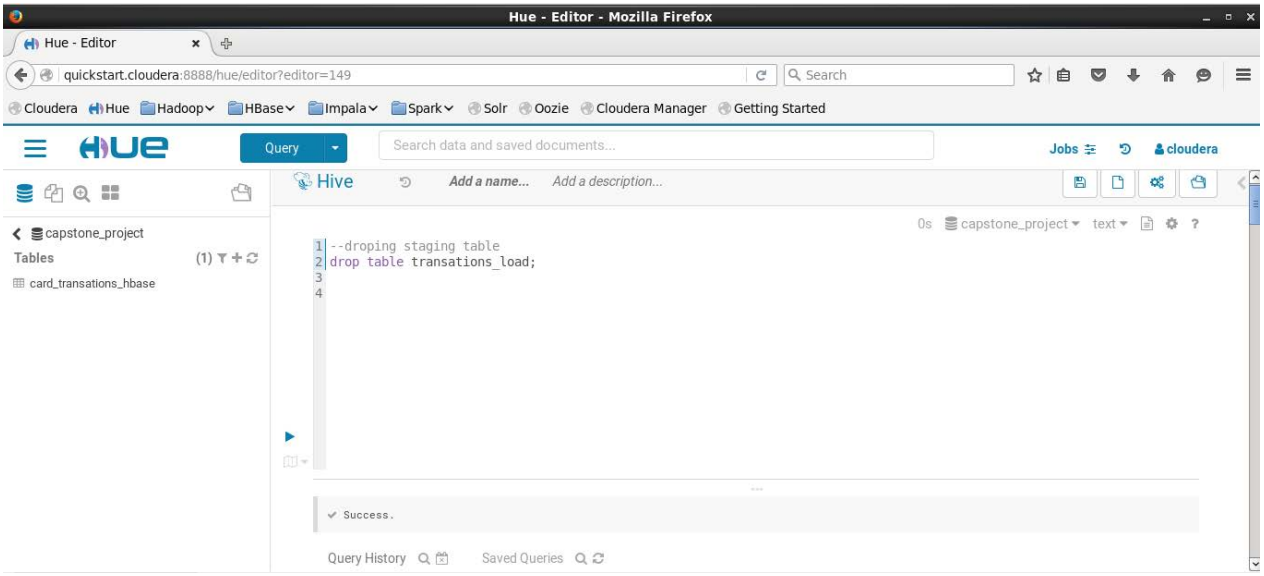
--Insert card transactions which has valid card id

INSERT INTO TABLE card_transations_hbase
SELECT NAMED_STRUCT('member_id', tl.member_id,'transaction_dt',
tl.transaction_dt, 'amount', tl.amount),
tl.card_id,post_code,pos_id,status FROM transations_load tl;



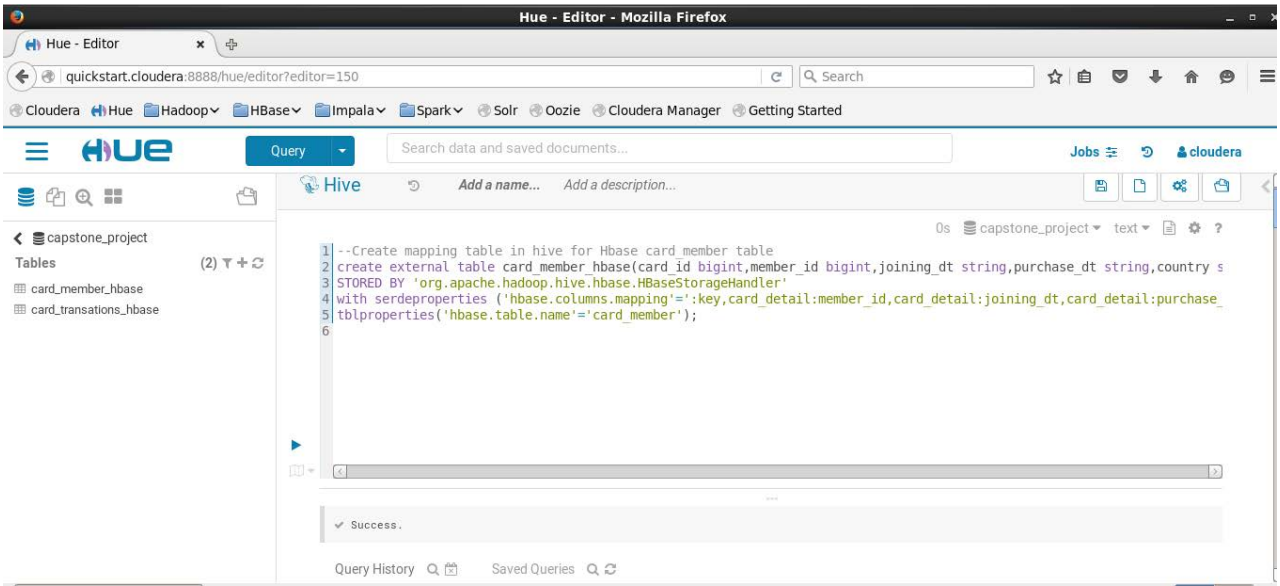
Dropping staging table

drop table transations_load;



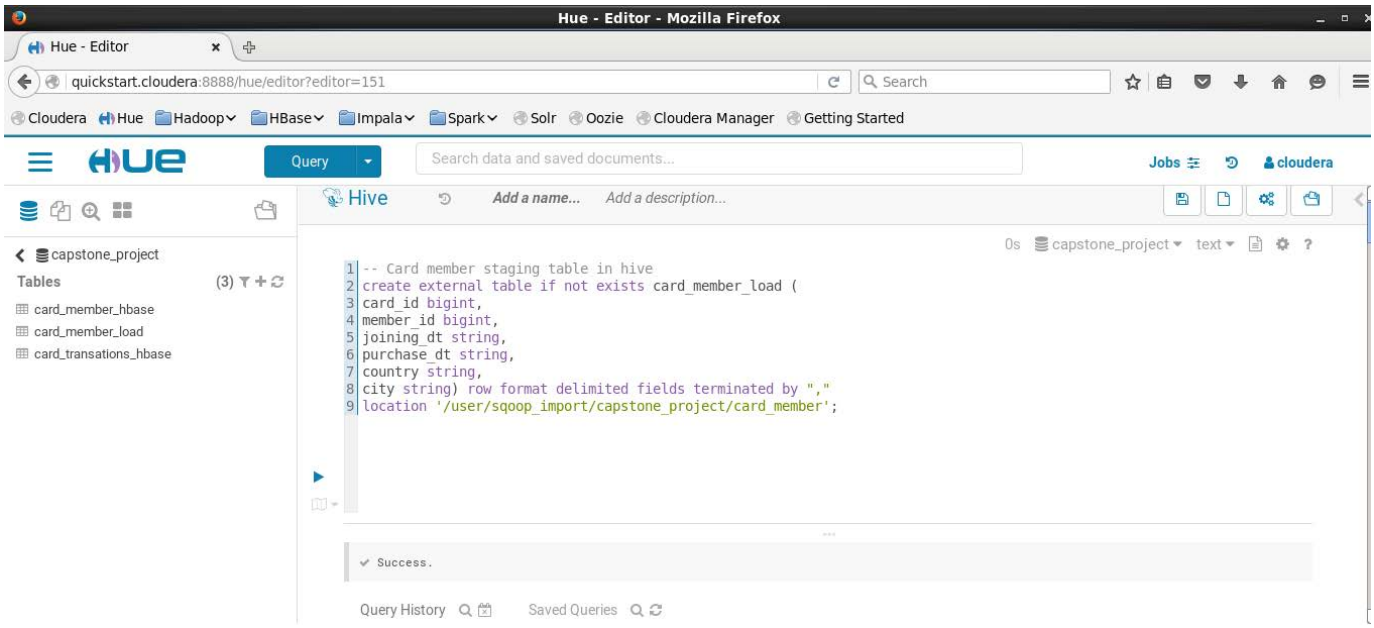
Create mapping table in hive for Hbase card_member table

create external table card_member_hbase(card_id bigint,member_id bigint,joining_dt string,purchase_dt string,country string,city string)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
('hbase.columns.mapping'=':key,card_detail:member_id,card_detail:joining_dt,card_detail:purchase_ dt ,location:country,location:city')
tblproperties('hbase.table.name'='card_member');



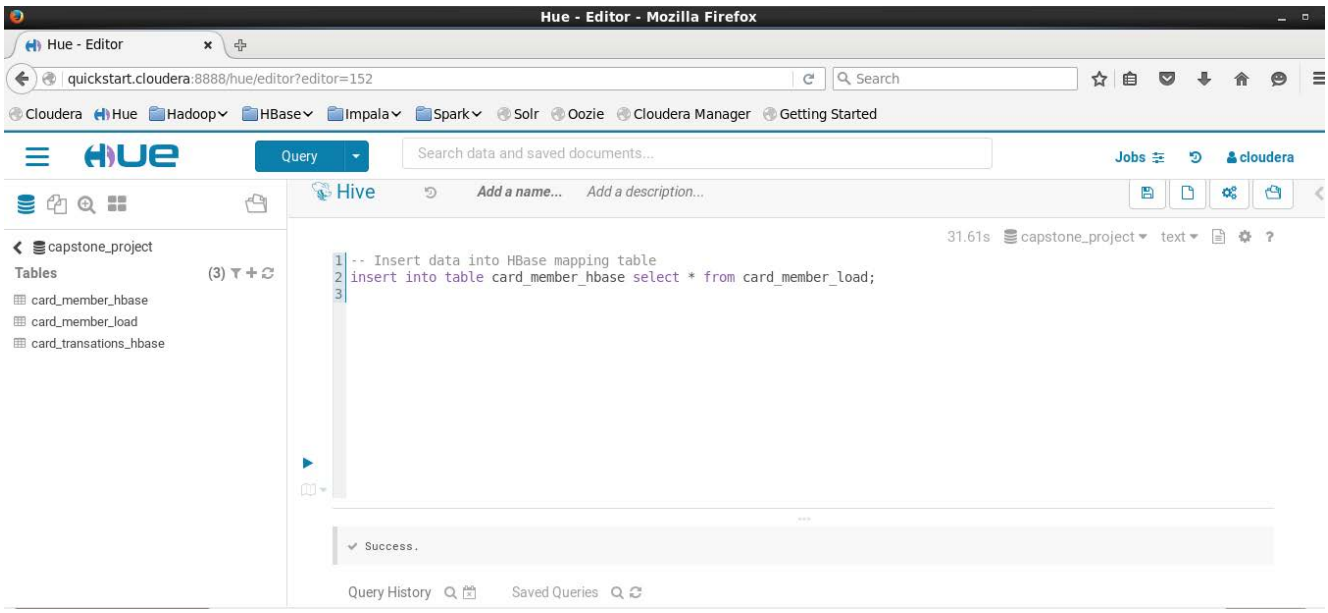
Card member staging table in hive

```
create external table if not exists card_member_load (
card_id bigint,
member_id bigint,
joining_dt string,
purchase_dt string,
country string,
city string) row format delimited fields terminated by "," location '/user/sqoop_import/
capstone_project/card_member';
```



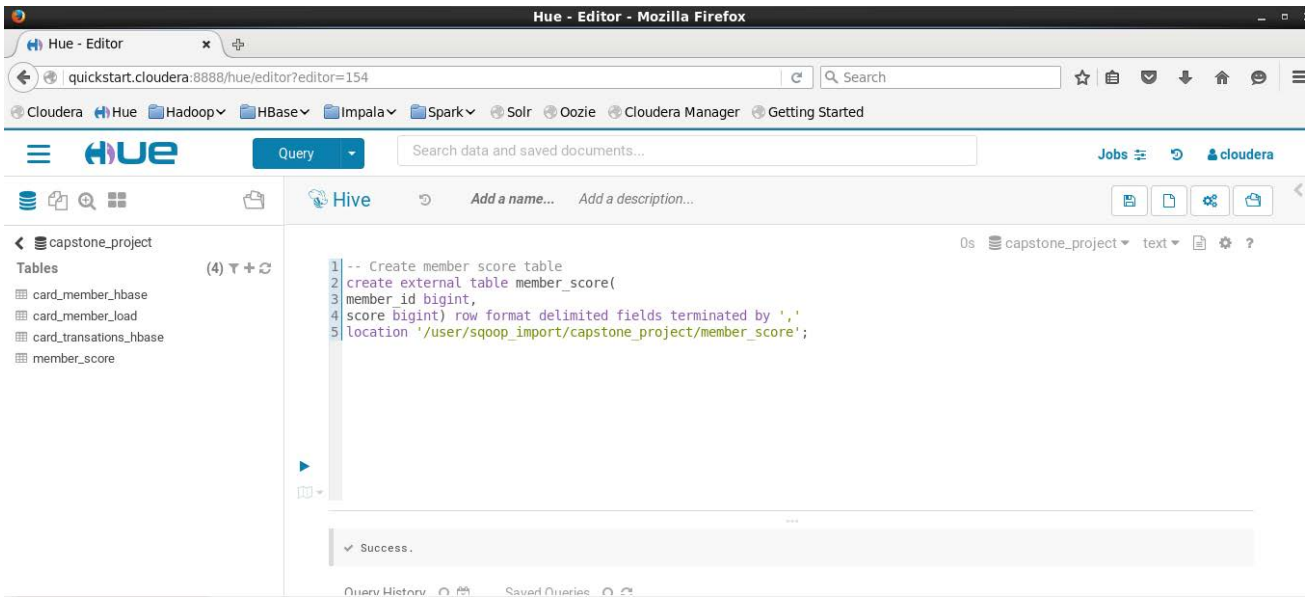
Insert data into HBase mapping table

```
insert into table card_member_hbase select * from card_member_load;
```



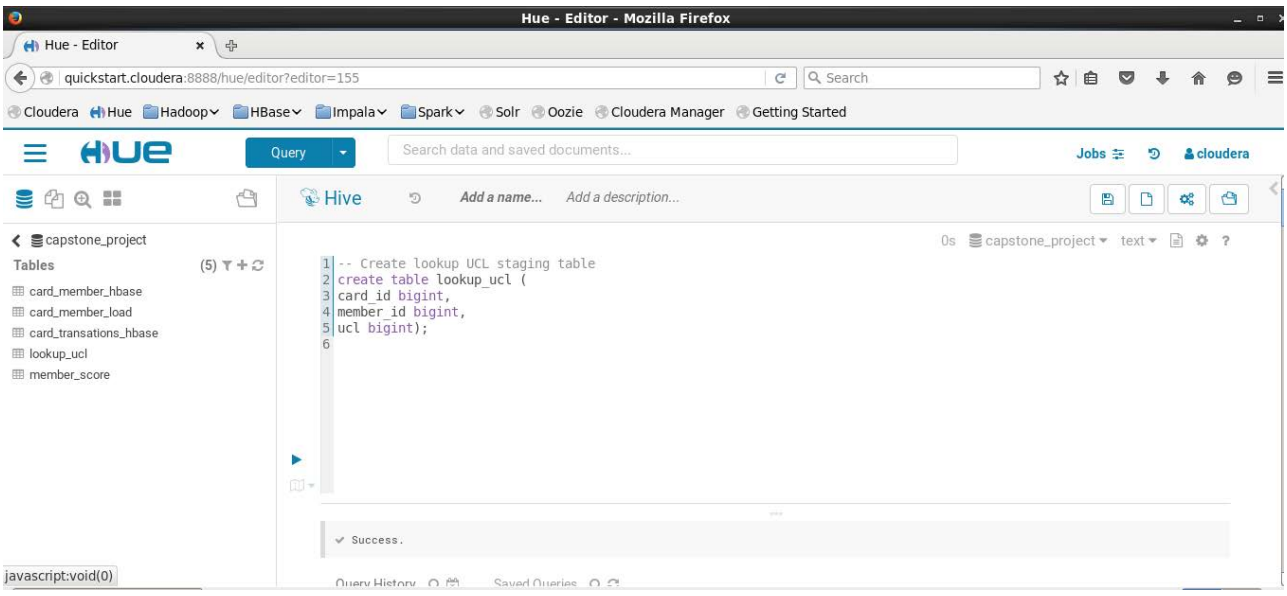
Create member score table

create external table member_score(
member_id bigint,
score bigint) row format delimited fields terminated by ',' location '/user/sqoop_import/capstone_project/member_score';



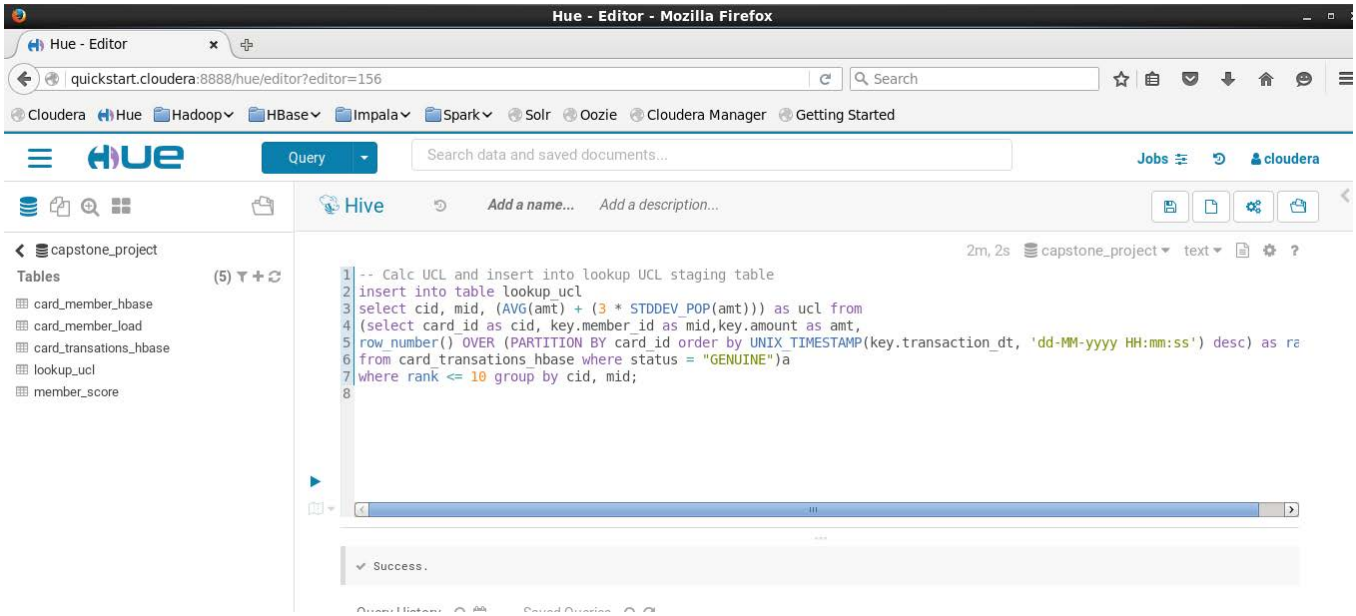
Create lookup UCL staging table

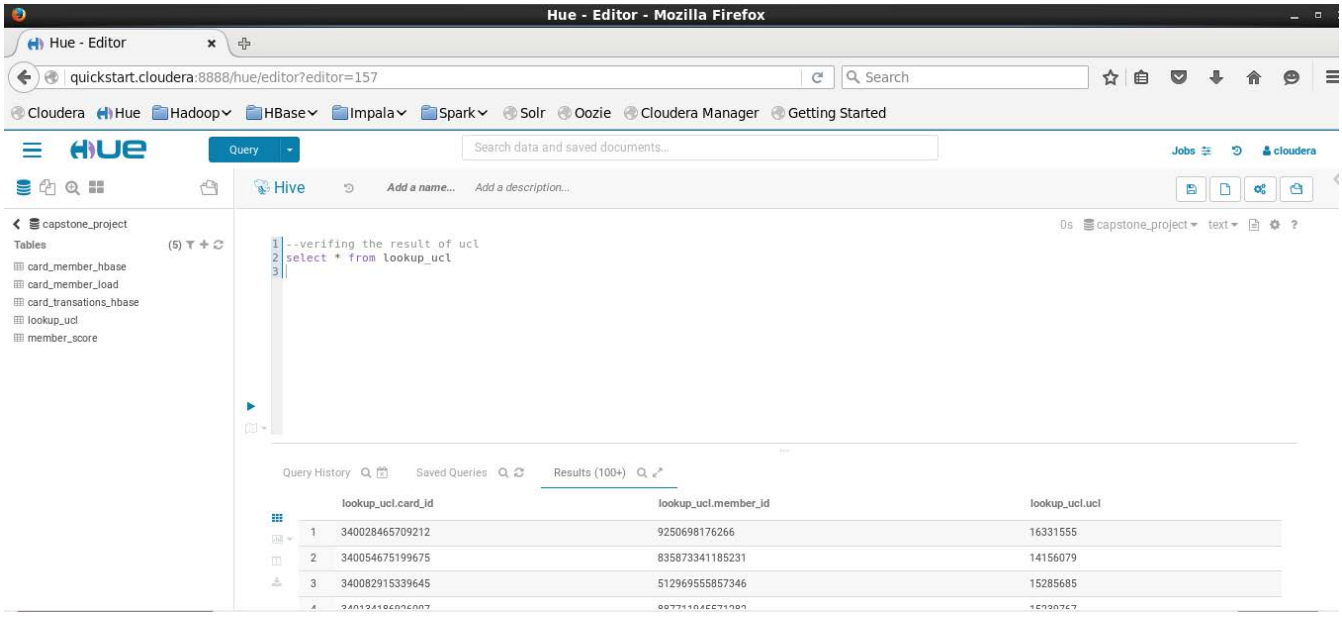
create table lookup_ucl (card_id bigint, member_id bigint, ucl bigint);



Calculate UCL and insert into lookup UCL staging table

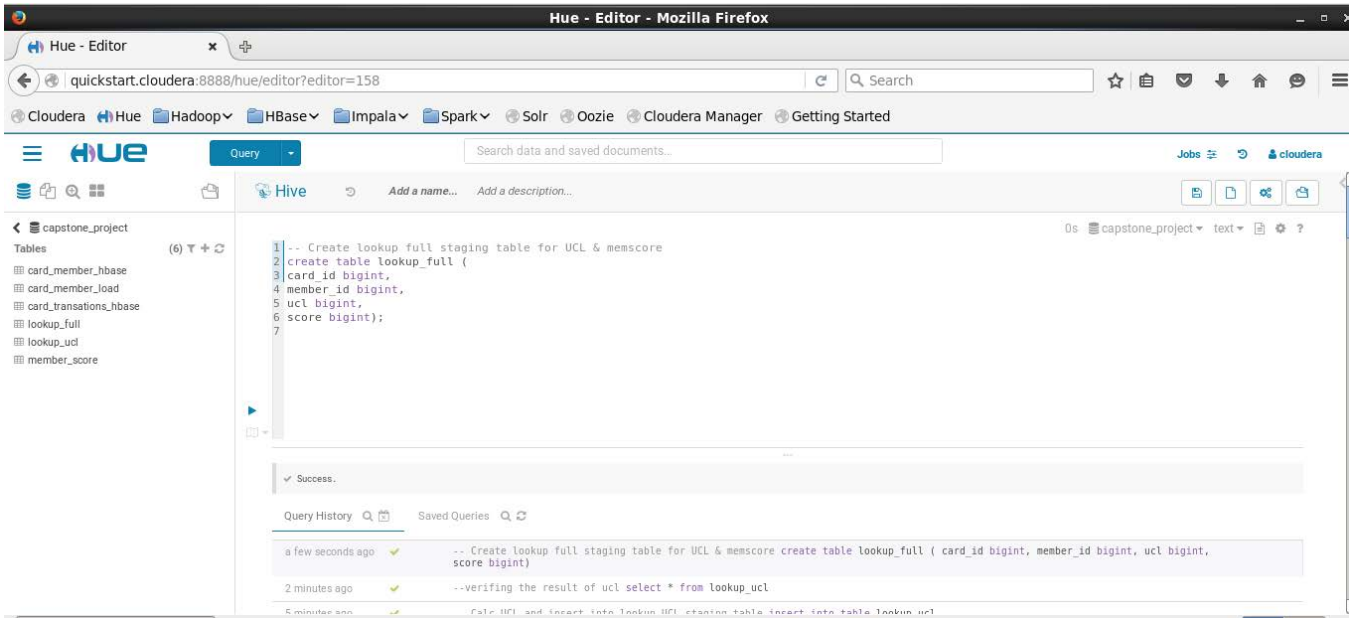
```
insert into table lookup_ucl
select cid, mid, (AVG(amt) + (3 * STDDEV_POP(amt))) as ucl from
(select card_id as cid, key.member_id as mid,key.amount as amt,
row_number() OVER (PARTITION BY card_id order by UNIX_TIMESTAMP(key.transaction_dt, 'dd-MM-yyyy HH:mm:ss') desc) as rank
from card_transations_hbase where status = "GENUINE")a
where rank <= 10 group by cid, mid;

--Verifying result
select * from lookup_ucl
```





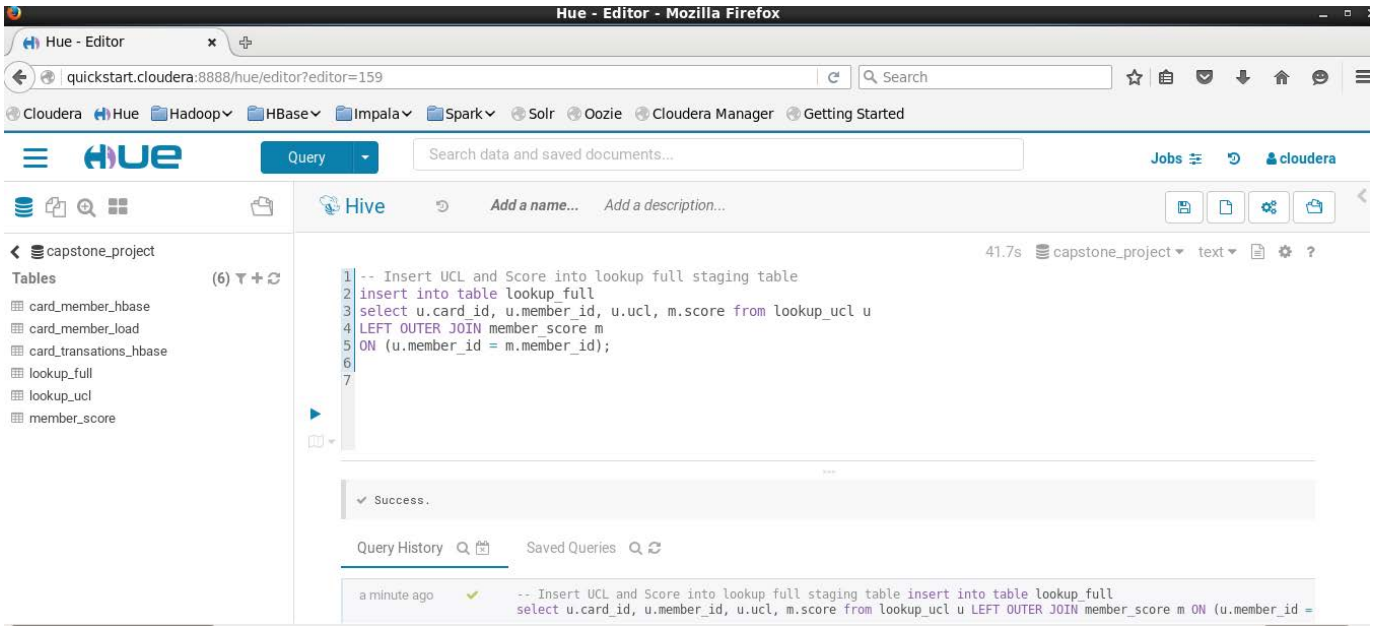
Create lookup full staging table for UCL & memscore

create table lookup_full (card_id bigint, member_id bigint, ucl bigint, score bigint);



Insert UCL and Score into lookup full staging table

insert into table lookup_full
select u.card_id, u.member_id, u.ucl, m.score from lookup_ucl u LEFT OUTER JOIN member_score m
ON (u.member_id = m.member_id);



Create mapping table for HBase lookup table for loading limit attributes

create external table lookup_limit_hbase(card_id bigint,

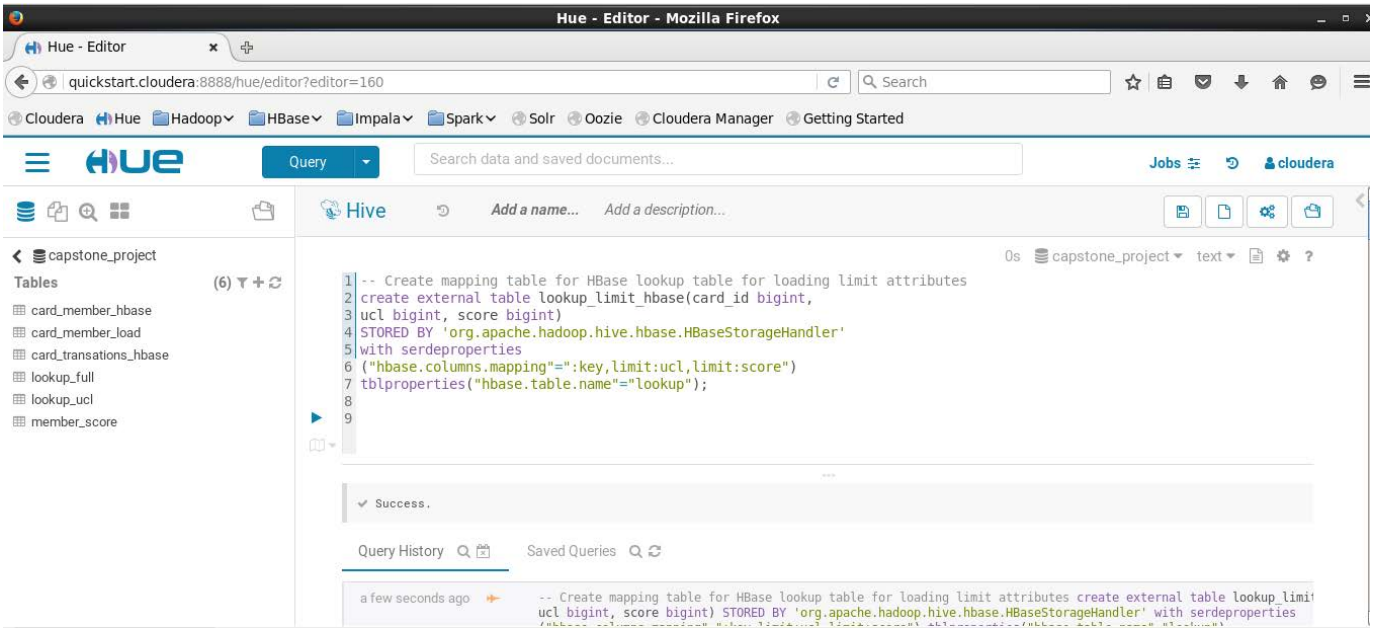
ucl bigint, score bigint)

STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

with serdeproperties

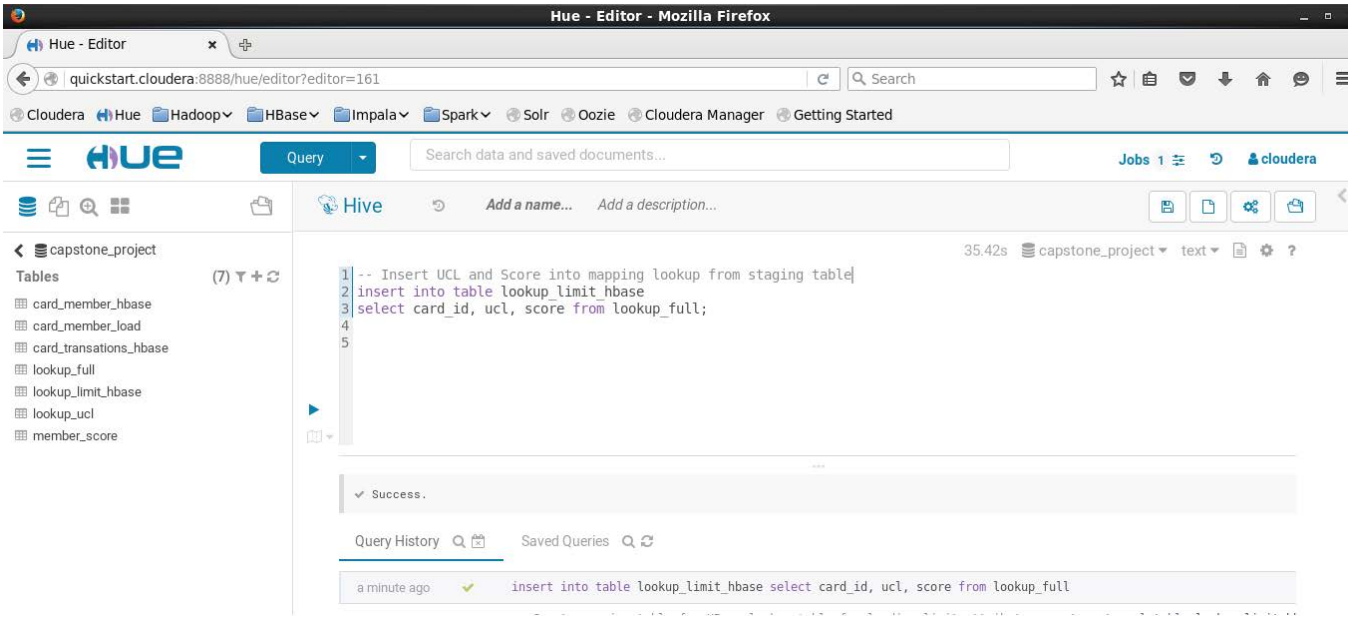
("hbase.columns.mapping"=":key,limit:ucl,limit:score")

tblproperties("hbase.table.name"="lookup");



Insert UCL and Score into mapping lookup from staging table

insert into table lookup_limit_hbase select card_id, ucl, score from lookup_full;

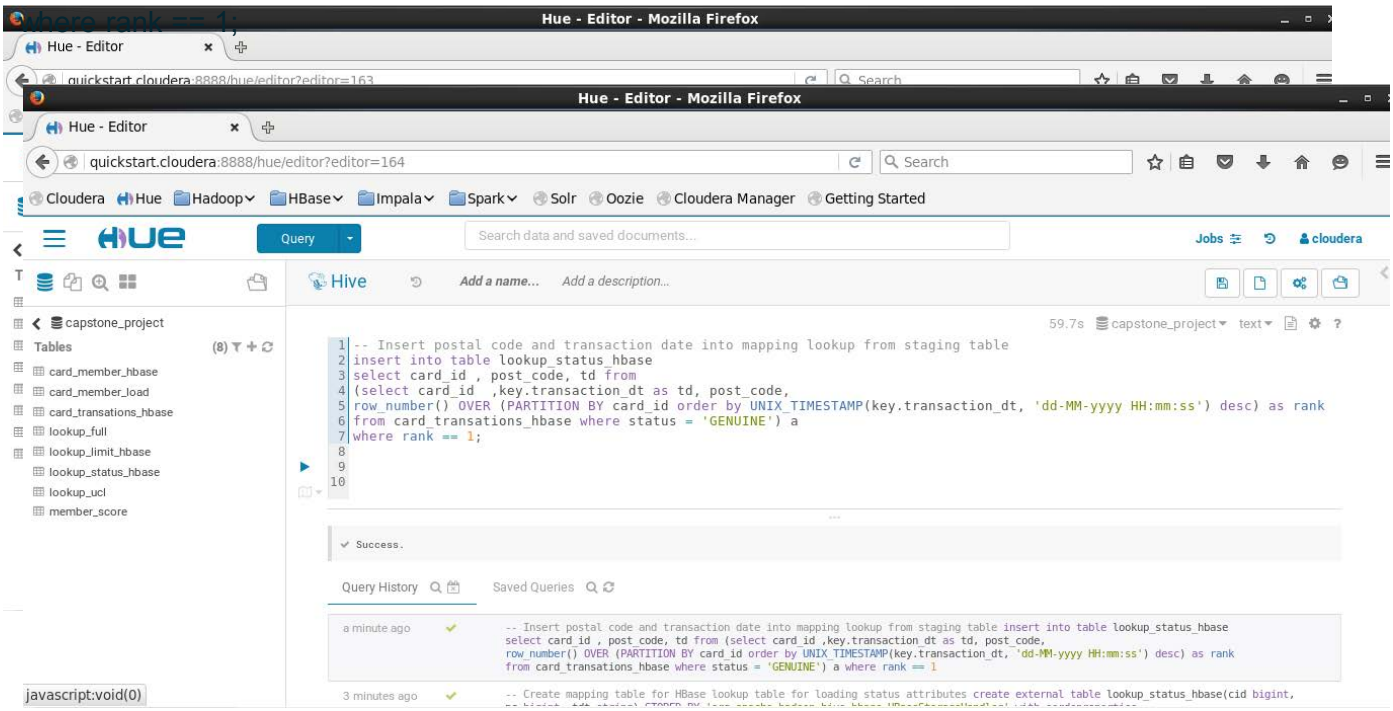


Create mapping table for HBase lookup table for loading status attributes

```
create external table lookup_status_hbase(cid bigint,
pc bigint, tdt string)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"=":key,status:pc,status:tdt")
tblproperties("hbase.table.name"="lookup");
```

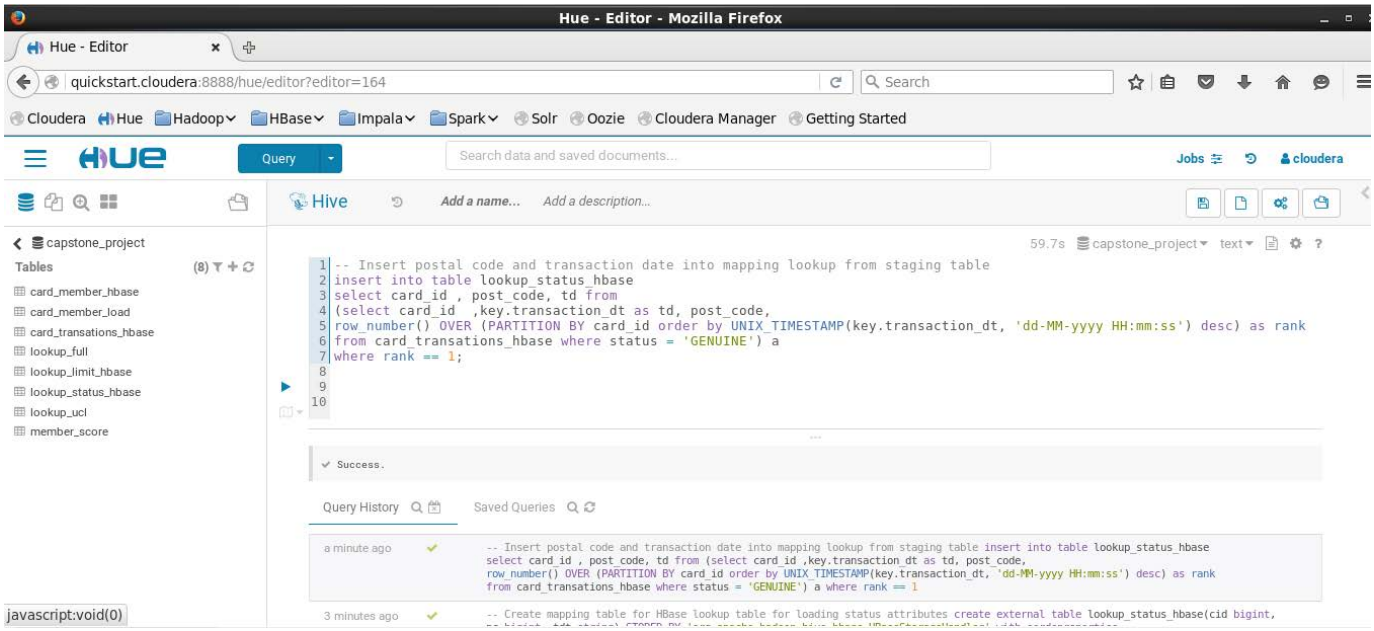
Insert postal code and transaction date into mapping lookup from card_transations_hbase

```
insert into table lookup_status_hbase
select card_id , post_code, td from
(select card_id ,key.transaction_dt as td, post_code,
row_number() OVER (PARTITION BY card_id order by UNIX_TIMESTAMP(key.transaction_dt, 'dd-MM-yyyy HH:mm:ss') desc) as rank
from card_transations_hbase where status = 'GENUINE') a
where rank == 1;
```



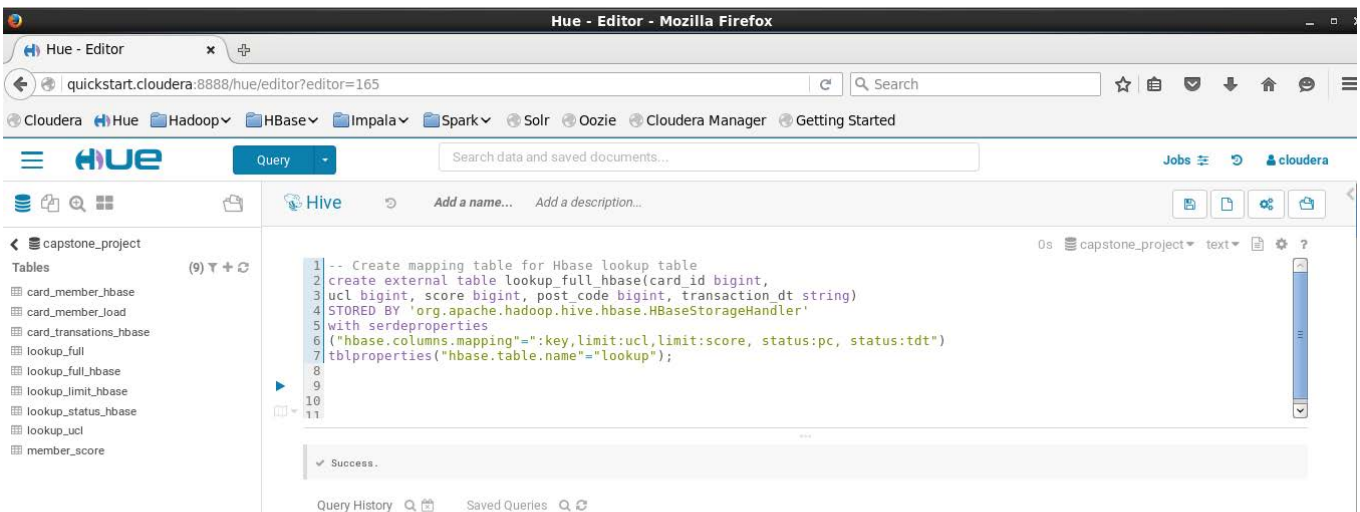
Insert postal code and transaction date into mapping lookup from card_transations_hbase

insert into table lookup_status_hbase
select card_id , post_code, td from
(select card_id ,key.transaction_dt as td, post_code,
row_number() OVER (PARTITION BY card_id order by UNIX_TIMESTAMP(key.transaction_dt, 'dd-MM-yyyy HH:mm:ss') desc) as rank
from card_transations_hbase where status = 'GENUINE') a
where rank == 1;



Create mapping table for Hbase lookup table

create external table lookup_full_hbase(card_id bigint,
ucl bigint, score bigint, post_code bigint, transaction_dt string)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler' with serdeproperties
("hbase.columns.mapping"=":key,limit:ucl,limit:score, status:pc, status:tdt")
tblproperties("hbase.table.name"="lookup");



Drop tables

--Dropping all staging table

drop table lookup_ucl;
drop table lookup_full;
drop lookup_limit_hbase
drop lookup_limit_hbase
drop table lookup_status_hbase;