Banking data analysis

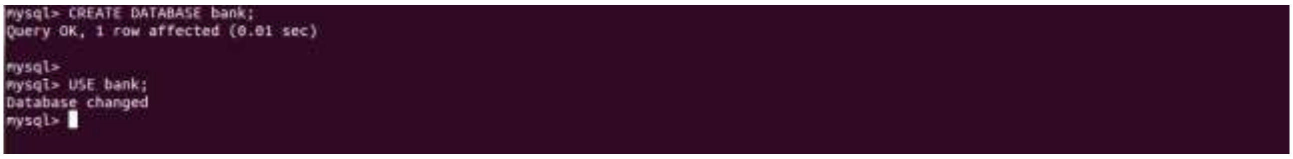
1.Loading data into mysql

Entering into mysql shell

mysql -u root -p

Creating database bank in mysql

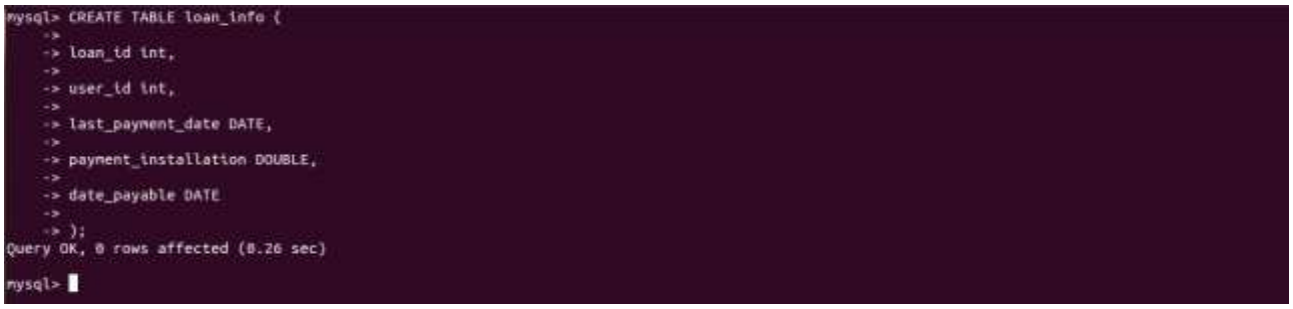
CREATE DATABASE bank;
USE bank;



```
mysql> CREATE DATABASE bank;
Query OK, 1 row affected (0.01 sec)

mysql>
mysql> USE bank;
Database changed
mysql> 
```

Creating tables in mysql and inserting the data into mysql tables

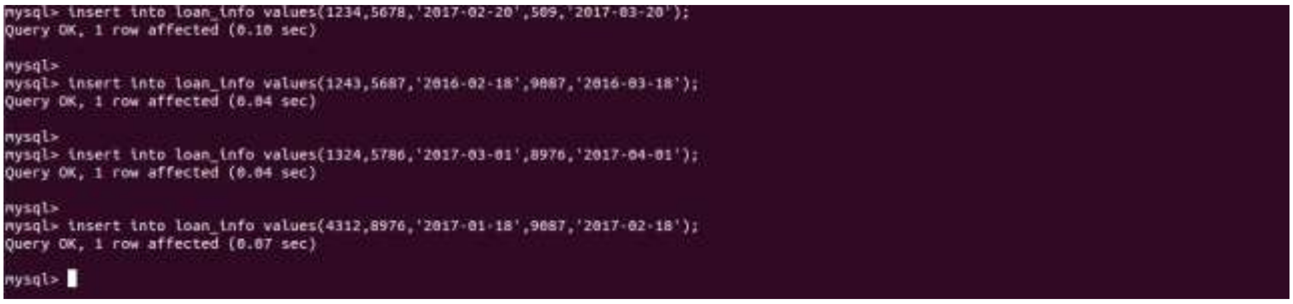
Creating table loan_info
CREATE TABLE loan_info (
loan_id int,
user_id int,
last_payment_date DATE,
payment_installation DOUBLE,
date_payable DATE
);



```
mysql> CREATE TABLE loan_info (
    ->
    -> loan_id int,
    ->
    -> user_id int,
    ->
    -> last_payment_date DATE,
    ->
    -> payment_installation DOUBLE,
    ->
    -> date_payable DATE
    ->
    -> );
Query OK, 0 rows affected (0.26 sec)

mysql> 
```

Inserting data into loan_info table

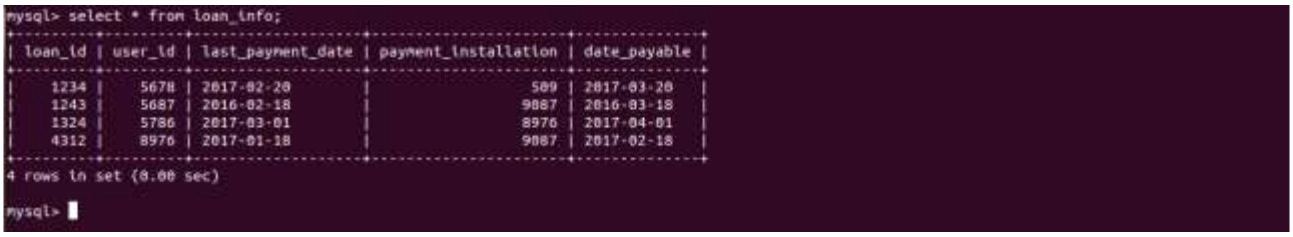
insert into loan_info values(1234,5678,'2017-02-20',509,'2017-03-20');
insert into loan_info values(1243,5687,'2016-02-18',9087,'2016-03-18');
insert into loan_info values(1324,5786,'2017-03-01',8976,'2017-04-01');
insert into loan_info values(4312,8976,'2017-01-18',9087,'2017-02-18');



```
mysql> insert into loan_info values(1234,5678,'2017-02-20',509,'2017-03-20');
Query OK, 1 row affected (0.10 sec)

mysql>
mysql> insert into loan_info values(1243,5687,'2016-02-18',9087,'2016-03-18');
Query OK, 1 row affected (0.04 sec)

mysql>
mysql> insert into loan_info values(1324,5786,'2017-03-01',8976,'2017-04-01');
Query OK, 1 row affected (0.04 sec)

mysql>
mysql> insert into loan_info values(4312,8976,'2017-01-18',9087,'2017-02-18');
Query OK, 1 row affected (0.07 sec)

mysql> █
```

Checking the data in loan_info table

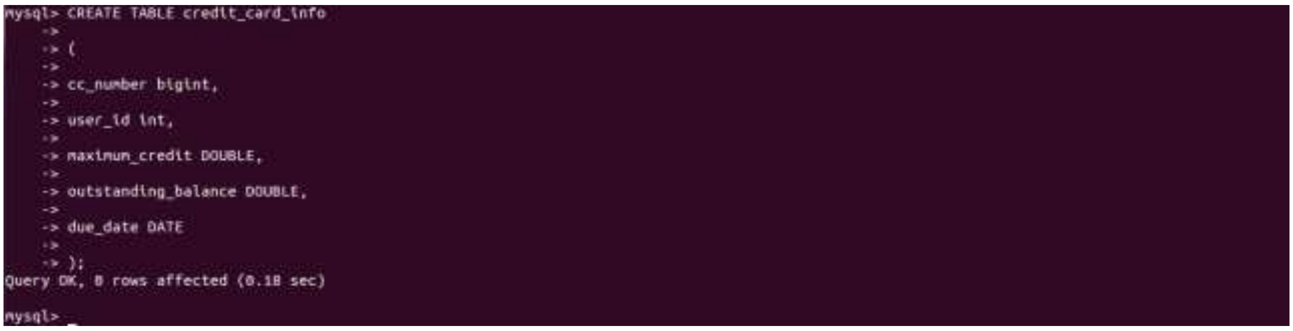
select * from loan_info



```
mysql> select * from loan_info;
+---------+---------+-------------------+----------------------+--------------+
| loan_id | user_id | last_payment_date | payment_installation | date_payable |
+---------+---------+-------------------+----------------------+--------------+
|    1234 |    5678 | 2017-02-20        |                  509 | 2017-03-20   |
|    1243 |    5687 | 2016-02-18        |                 9087 | 2016-03-18   |
|    1324 |    5786 | 2017-03-01        |                 8976 | 2017-04-01   |
|    4312 |    8976 | 2017-01-18        |                 9087 | 2017-02-18   |
+---------+---------+-------------------+----------------------+--------------+
4 rows in set (0.00 sec)

mysql> █
```

Creating table credit_card_info

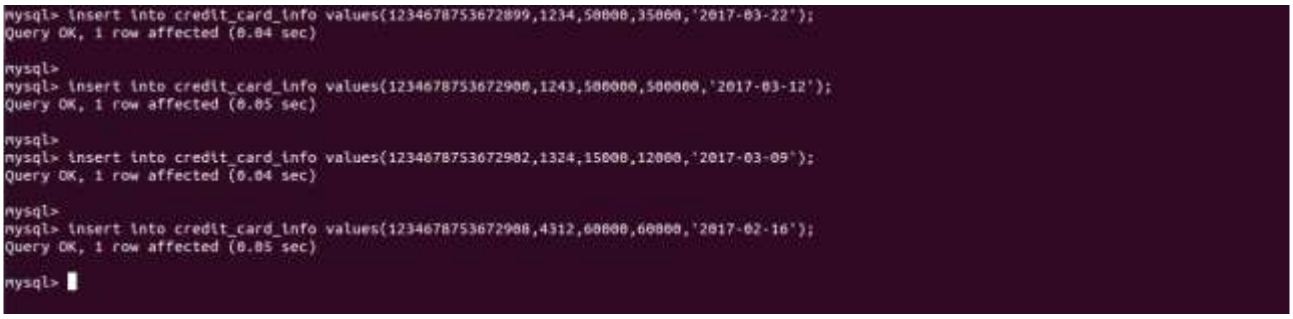
```
CREATE TABLE credit_card_info
(
cc_number bigint,
user_id int,
maximum_credit DOUBLE,
outstanding_balance DOUBLE,
due_date DATE
);
```



```
mysql> CREATE TABLE credit_card_info
    ->
    -> (
    ->
    -> cc_number bigint,
    ->
    -> user_id int,
    ->
    -> maximum_credit DOUBLE,
    ->
    -> outstanding_balance DOUBLE,
    ->
    -> due_date DATE
    ->
    -> );
Query OK, 0 rows affected (0.18 sec)

mysql> _
```

Inserting data into the credit_card_info table

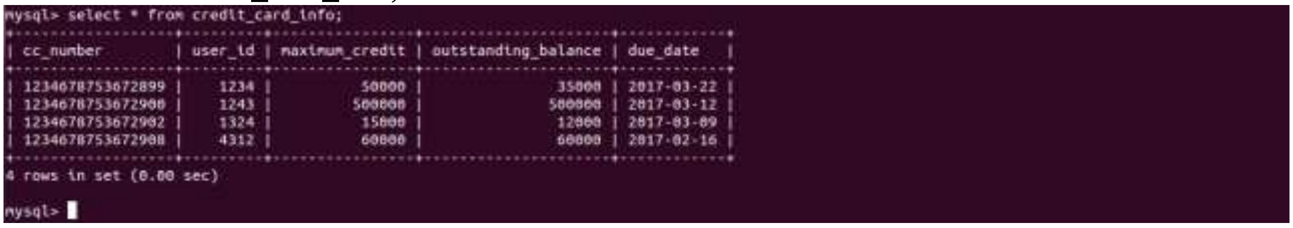
```
insert into credit_card_info values(1234678753672899,1234,50000,35000,'2017-03-22');
insert into credit_card_info values(1234678753672900,1243,500000,500000,'2017-03-12');
insert into credit_card_info values(1234678753672902,1324,15000,12000,'2017-03-09');
insert into credit_card_info values(1234678753672908,4312,60000,60000,'2017-02-16');
```



```
mysql> insert into credit_card_info values(1234678753672899,1234,50000,35000,'2017-03-22');
Query OK, 1 row affected (0.04 sec)

mysql>
mysql> insert into credit_card_info values(1234678753672900,1243,500000,500000,'2017-03-12');
Query OK, 1 row affected (0.05 sec)

mysql>
mysql> insert into credit_card_info values(1234678753672902,1324,15000,12000,'2017-03-09');
Query OK, 1 row affected (0.04 sec)

mysql>
mysql> insert into credit_card_info values(1234678753672908,4312,60000,60000,'2017-02-16');
Query OK, 1 row affected (0.05 sec)

mysql> █
```

Checking the data in credit_card_info table

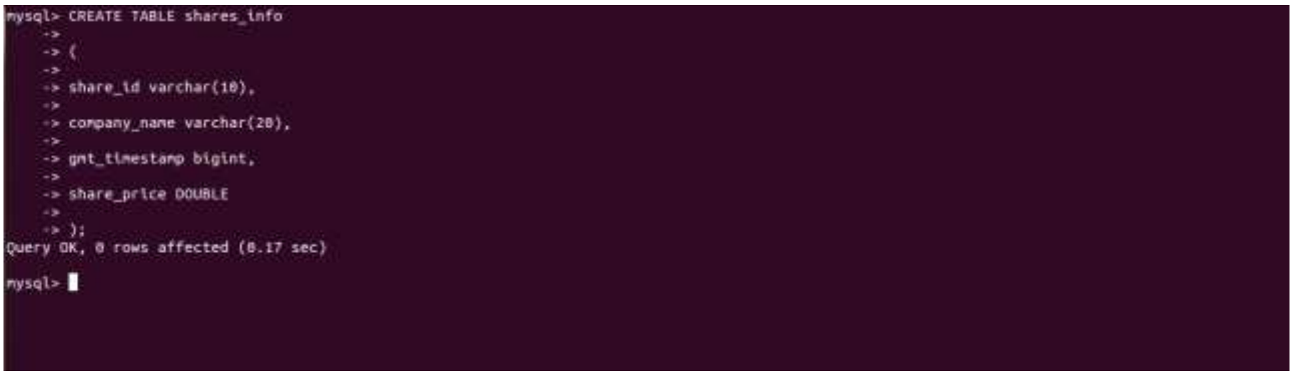
select * from credit_card_info;



```
mysql> select * from credit_card_info;
+------------------+---------+----------------+---------------------+------------+
| cc_number        | user_id | maximum_credit | outstanding_balance | due_date   |
+------------------+---------+----------------+---------------------+------------+
| 1234678753672899 |    1234 |          50000 |               35000 | 2017-03-22 |
| 1234678753672900 |    1243 |         500000 |              500000 | 2017-03-12 |
| 1234678753672902 |    1324 |          15000 |               12000 | 2017-03-09 |
| 1234678753672908 |    4312 |          60000 |               60000 | 2017-02-16 |
+------------------+---------+----------------+---------------------+------------+
4 rows in set (0.00 sec)

mysql> █
```

Creating table shares_info

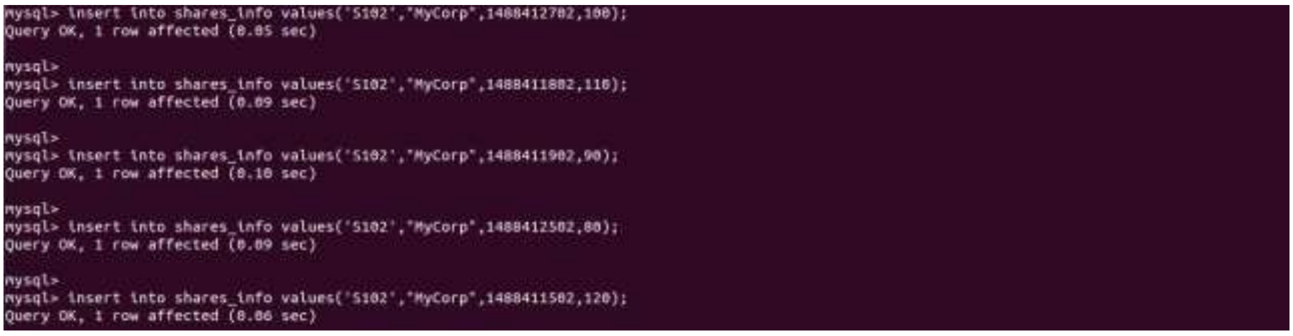
```
CREATE TABLE shares_info
(
share_id varchar(10),
company_name varchar(20),
gmt_timestamp bigint,
share_price DOUBLE
);
```



```
mysql> CREATE TABLE shares_info
    ->
    -> (
    ->
    -> share_id varchar(10),
    ->
    -> company_name varchar(20),
    ->
    -> gmt_timestamp bigint,
    ->
    -> share_price DOUBLE
    ->
    -> );
Query OK, 0 rows affected (0.17 sec)

mysql> █
```

Inserting data into shares_info table

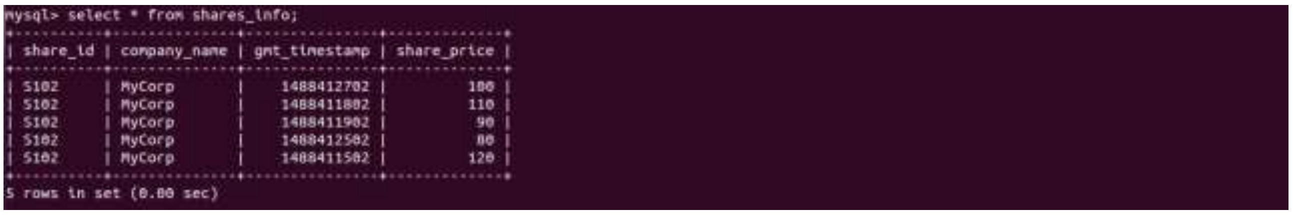
```
insert into shares_info values('S102',"MyCorp",1488412702,100);
insert into shares_info values('S102',"MyCorp",1488411802,110);
insert into shares_info values('S102',"MyCorp",1488411902,90);
insert into shares_info values('S102',"MyCorp",1488412502,80);
insert into shares_info values('S102',"MyCorp",1488411502,120);
```



```
mysql> insert into shares_info values('S102','MyCorp',1488412702,100);
Query OK, 1 row affected (0.05 sec)

mysql>
mysql> insert into shares_info values('S102','MyCorp',1488411802,110);
Query OK, 1 row affected (0.09 sec)

mysql>
mysql> insert into shares_info values('S102','MyCorp',1488411902,90);
Query OK, 1 row affected (0.10 sec)

mysql>
mysql> insert into shares_info values('S102','MyCorp',1488412502,80);
Query OK, 1 row affected (0.09 sec)

mysql>
mysql> insert into shares_info values('S102','MyCorp',1488411502,120);
Query OK, 1 row affected (0.06 sec)
```

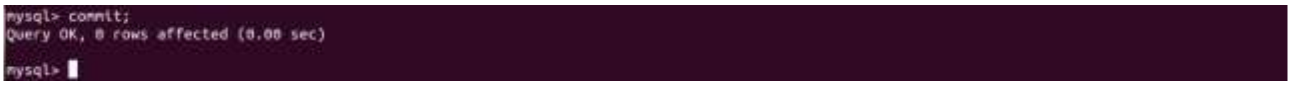
Checking the data in shares_info table

select * from shares_info;



```
mysql> select * from shares_info;
+----------+--------------+---------------+-------------+
| share_id | company_name | gmt_timestamp | share_price |
+----------+--------------+---------------+-------------+
| S102     | MyCorp       |    1488412702 |         100 |
| S102     | MyCorp       |    1488411802 |         110 |
| S102     | MyCorp       |    1488411902 |          90 |
| S102     | MyCorp       |    1488412502 |          80 |
| S102     | MyCorp       |    1488411502 |         120 |
+----------+--------------+---------------+-------------+
5 rows in set (0.00 sec)
```

Commit;



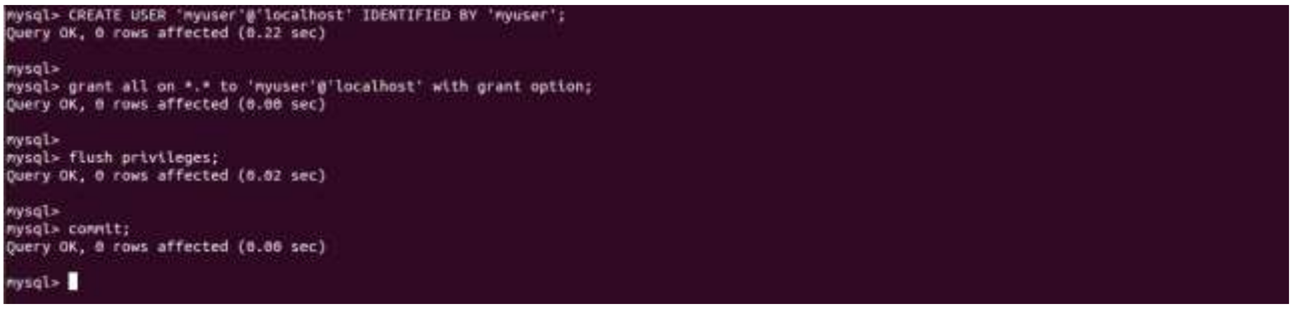
```
mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> █
```

2.Exporting data from Mysql to HDFS using sqoop

Now we have data in mysql. We need to export this data into HDFS. We will do it using sqoop.

For exporting data into HDFS we will first create an user in mysql.

CREATE USER 'myuser'@'localhost' IDENTIFIED BY 'myuser';
grant all on *.* to 'myuser'@'localhost' with grant option;
flush privileges;
commit;



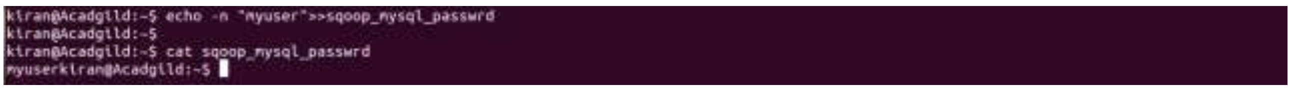
```
mysql> CREATE USER 'myuser'@'localhost' IDENTIFIED BY 'myuser';
Query OK, 0 rows affected (0.22 sec)

mysql>
mysql> grant all on *.* to 'myuser'@'localhost' with grant option;
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> flush privileges;
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> █
```

Now let's transfer these tables into HDFS by writing sqoop jobs.

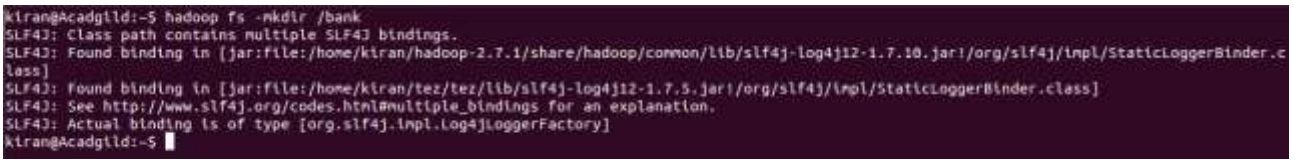
We will protect our mysql password by saving the password in a file.

echo -n "myuser">>sqoop_mysql_passwrd

You need to use the option **-n.** Otherwise, a new line will be created unknowingly and while reading the password, Sqoop throws an error **Access Denied for User**.



```
kiran@Acadgild:~$ echo -n "myuser">>sqoop_mysql_passwrd
kiran@Acadgild:~$
kiran@Acadgild:~$ cat sqoop_mysql_passwrd
myuserkiran@Acadgild:~$ █
```



```
kiran@Acadgild:~$ hadoop fs -mkdir /bank
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.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]
kiran@Acadgild:~$ █
```

/*

Sqoop job to transfer data in loan_info table

Create a directory in HDFS loan_info_stg to store the table data

hadoop fs -mkdir /bank

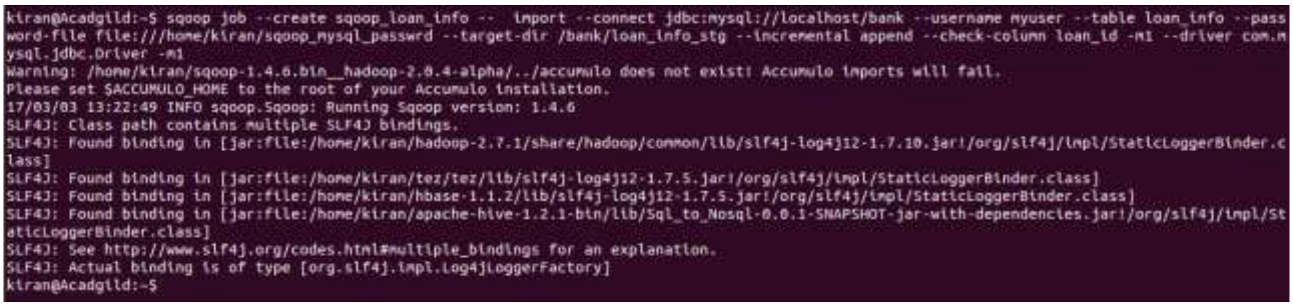
hadoop fs -mkdir /bank/loan_info_stg

*/
Above step is not required while performing normal import.

Creating sqoop job for sqoop_loan_info

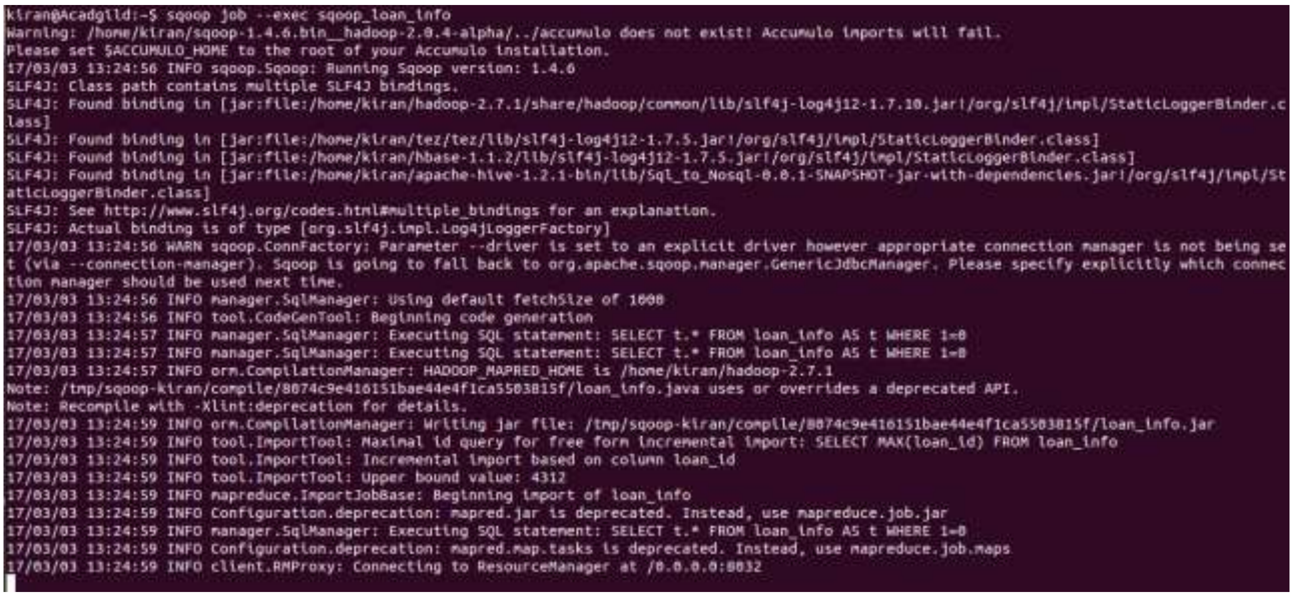
sqoop job --create sqoop_loan_info -- import --connect jdbc:mysql://localhost/bank --username myuser --table loan_info --password-file file:///home/acadgild/sqoop_mysql_passwrd --target-dir /bank/loan_info_stg -m1

sqoop job --list



```
kiran@Acadgild:~$ sqoop job --create sqoop_loan_info --  import --connect jdbc:mysql://localhost/bank --username myuser --table loan_info --pass
word-file file:///home/kiran/sqoop_mysql_passwrd --target-dir /bank/loan_info_stg --incremental append --check-column loan_id -m1 --driver com.m
ysql.jdbc.Driver -m1
Warning: /home/kiran/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
17/03/03 13:22:49 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-1.1.2/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/Sql_to_Nosql-0.0.1-SNAPSHOT-jar-with-dependencies.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
kiran@Acadgild:~$
```

Executing the sqoop job sqoop_loan_info
sqoop job --exec sqoop_loan_info



```
kiran@Acadgild:~$ sqoop job --exec sqoop_loan_info
Warning: /home/kiran/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
17/03/03 13:24:56 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-1.1.2/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/Sql_to_Nosql-0.0.1-SNAPSHOT-jar-with-dependencies.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
17/03/03 13:24:56 WARN sqoop.ConnFactory: Parameter --driver is set to an explicit driver however appropriate connection manager is not being se
t (via --connection-manager). Sqoop is going to fall back to org.apache.sqoop.manager.GenericJdbcManager. Please specify explicitly which connec
tion manager should be used next time.
17/03/03 13:24:56 INFO manager.SqlManager: Using default fetchSize of 1000
17/03/03 13:24:56 INFO tool.CodeGenTool: Beginning code generation
17/03/03 13:24:57 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM loan_info AS t WHERE 1=0
17/03/03 13:24:57 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM loan_info AS t WHERE 1=0
17/03/03 13:24:57 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/kiran/hadoop-2.7.1
Note: /tmp/sqoop-kiran/compile/8074c9e416151bae44e4f1ca5503815f/loan_info.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
17/03/03 13:24:59 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-kiran/compile/8074c9e416151bae44e4f1ca5503815f/loan_info.jar
17/03/03 13:24:59 INFO tool.ImportTool: Maximal id query for free form incremental import: SELECT MAX(loan_id) FROM loan_info
17/03/03 13:24:59 INFO tool.ImportTool: Incremental import based on column loan_id
17/03/03 13:24:59 INFO tool.ImportTool: Upper bound value: 4312
17/03/03 13:24:59 INFO mapreduce.ImportJobBase: Beginning import of loan_info
17/03/03 13:24:59 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
17/03/03 13:24:59 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM loan_info AS t WHERE 1=0
17/03/03 13:24:59 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
17/03/03 13:24:59 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
```

Sqoop job to transfer data in credit_card_info table

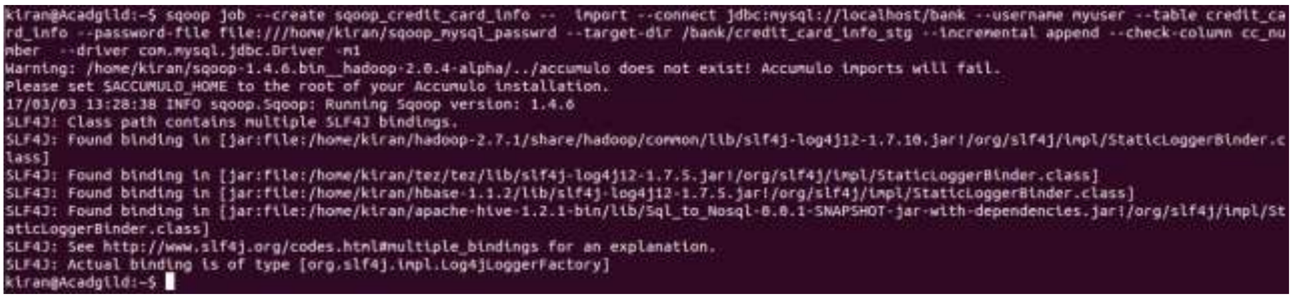
Create a directory in HDFS for storing credit_card_info table data

hadoop fs -mkdir /bank/credit_card_info_stg

Creating sqoop job credit_card_info

sqoop job --create sqoop_credit_card_info -- import --connect jdbc:mysql://localhost/bank --username myuser --table credit_card_info --password-file file:///home/acadgild/sqoop_mysql_passwrd --target-dir /bank/credit_card_info_stg -m 1

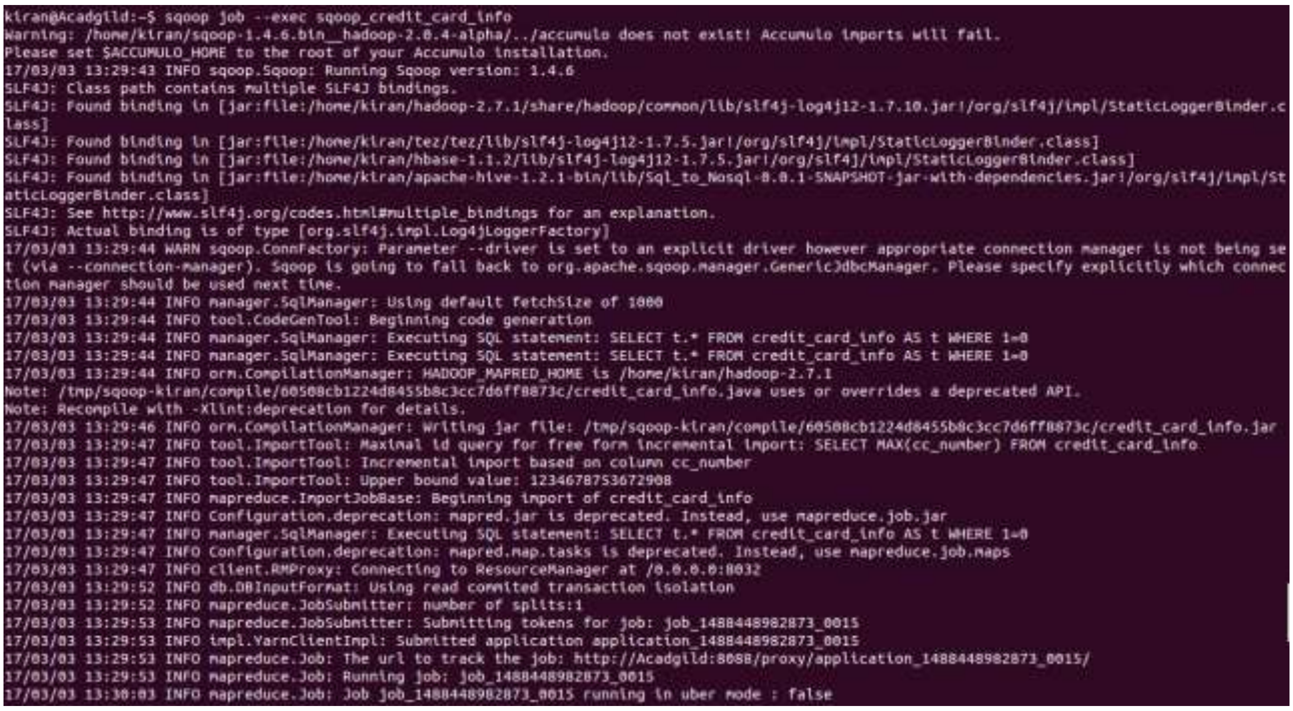
sqoop job --list



```
kiran@Acadgild:~$ sqoop job --create sqoop_credit_card_info --  import --connect jdbc:mysql://localhost/bank --username myuser --table credit_ca
rd_info --password-file file:///home/kiran/sqoop_mysql_passwrd --target-dir /bank/credit_card_info_stg --incremental append --check-column cc_nu
mber  --driver com.mysql.jdbc.Driver -m1
Warning: /home/kiran/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
17/03/03 13:28:38 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-1.1.2/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/Sql_to_Nosql-0.0.1-SNAPSHOT-jar-with-dependencies.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
kiran@Acadgild:~$
```

Executing sqoop job credit_card_info

sqoop job --exec sqoop_credit_card_info



```
kiran@Acadgild:~$ sqoop job --exec sqoop_credit_card_info
Warning: /home/kiran/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
17/03/03 13:29:43 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-1.1.2/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/Sql_to_Nosql-0.0.1-SNAPSHOT-jar-with-dependencies.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
17/03/03 13:29:44 WARN sqoop.ConnFactory: Parameter --driver is set to an explicit driver however appropriate connection manager is not being se
t (via --connection-manager). Sqoop is going to fall back to org.apache.sqoop.manager.GenericJdbcManager. Please specify explicitly which connec
tion manager should be used next time.
17/03/03 13:29:44 INFO manager.SqlManager: Using default fetchSize of 1000
17/03/03 13:29:44 INFO tool.CodeGenTool: Beginning code generation
17/03/03 13:29:44 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM credit_card_info AS t WHERE 1=0
17/03/03 13:29:44 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM credit_card_info AS t WHERE 1=0
17/03/03 13:29:44 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/kiran/hadoop-2.7.1
Note: /tmp/sqoop-kiran/compile/60508cb1224d8455b8c3cc7d6ff8873c/credit_card_info.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
17/03/03 13:29:46 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-kiran/compile/60508cb1224d8455b8c3cc7d6ff8873c/credit_card_info.jar
17/03/03 13:29:47 INFO tool.ImportTool: Maximal id query for free form incremental import: SELECT MAX(cc_number) FROM credit_card_info
17/03/03 13:29:47 INFO tool.ImportTool: Incremental import based on column cc_number
17/03/03 13:29:47 INFO tool.ImportTool: Upper bound value: 1234678753672908
17/03/03 13:29:47 INFO mapreduce.ImportJobBase: Beginning import of credit_card_info
17/03/03 13:29:47 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
17/03/03 13:29:47 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM credit_card_info AS t WHERE 1=0
17/03/03 13:29:47 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
17/03/03 13:29:47 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/03/03 13:29:52 INFO db.DBInputFormat: Using read commited transaction isolation
17/03/03 13:29:52 INFO mapreduce.JobSubmitter: number of splits:1
17/03/03 13:29:53 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1488448982873_0015
17/03/03 13:29:53 INFO impl.YarnClientImpl: Submitted application application_1488448982873_0015
17/03/03 13:29:53 INFO mapreduce.Job: The url to track the job: http://Acadgild:8088/proxy/application_1488448982873_0015/
17/03/03 13:29:53 INFO mapreduce.Job: Running job: job_1488448982873_0015
17/03/03 13:30:03 INFO mapreduce.Job: Job job_1488448982873_0015 running in uber mode : false
```

Sqoop job to transfer data in shares_info_table

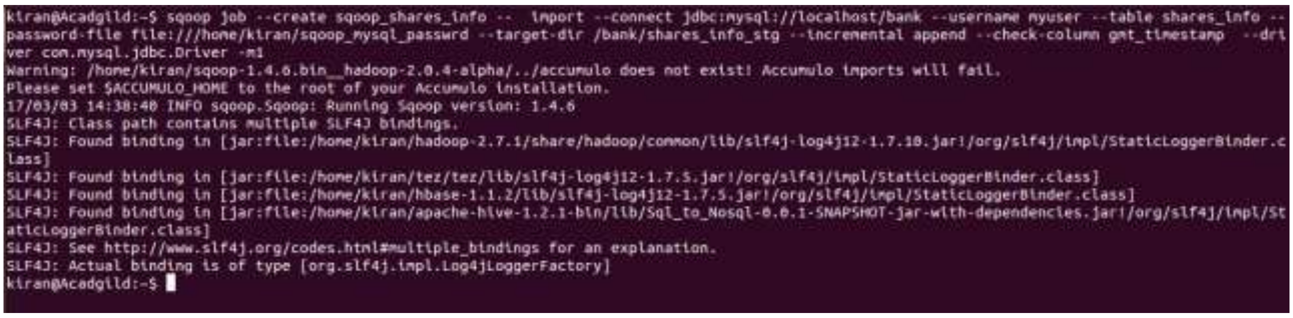
Creating HDFS directory to store shares_info table data

hadoop fs -mkdir /bank/shares_info_stg

Creating sqoop job shares_info

sqoop job --create sqoop_shares_info -- import --connect jdbc:mysql://localhost/bank --username myuser --table shares_info --password-file file:///home/acadgild/sqoop_mysql_passwrd --target-dir /bank/shares_info_stg -m 1

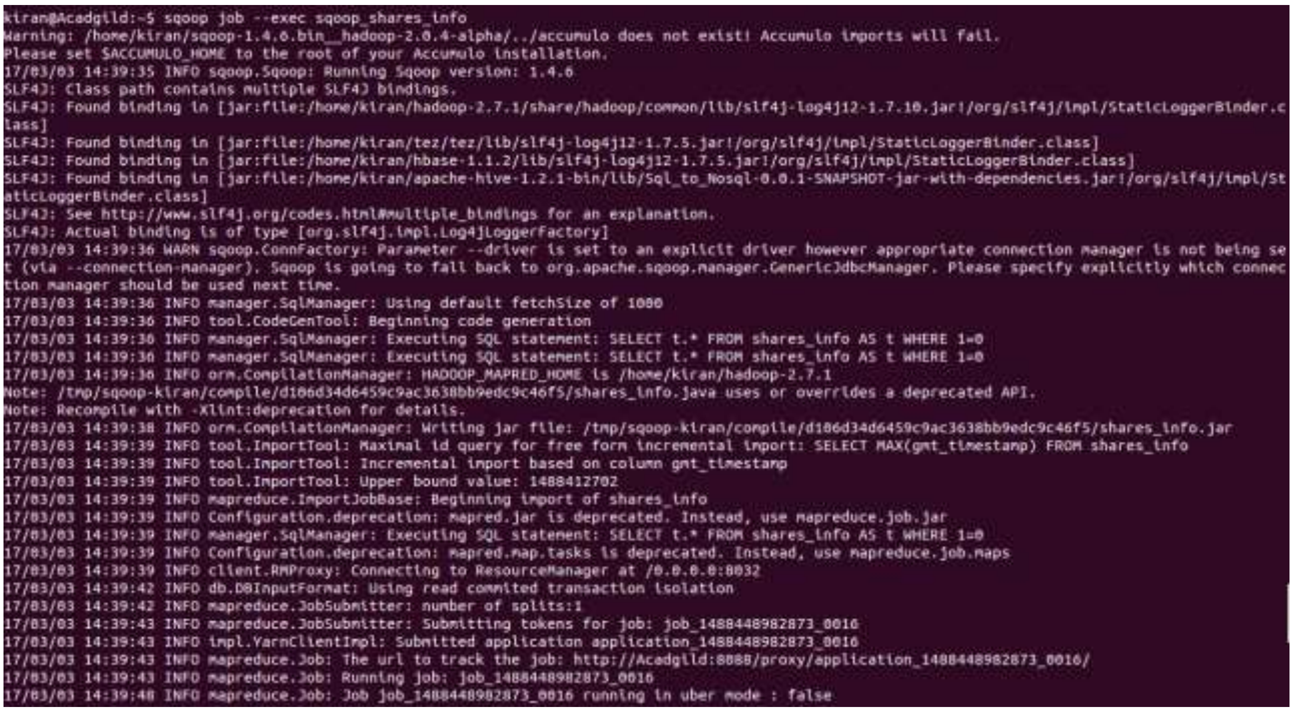
sqoop job --list



```
kiran@Acadgild:~$ sqoop job --create sqoop_shares_info --  import --connect jdbc:mysql://localhost/bank --username myuser --table shares_info --
password-file file:///home/kiran/sqoop_mysql_passwrd --target-dir /bank/shares_info_stg --incremental append --check-column gmt_timestamp  --dri
ver com.mysql.jdbc.Driver -m1
Warning: /home/kiran/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
17/03/03 14:38:40 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-1.1.2/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/Sql_to_Nosql-0.0.1-SNAPSHOT-jar-with-dependencies.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
kiran@Acadgild:~$
```

Executing sqoop_job shares_info

sqoop job --exec sqoop_shares_info



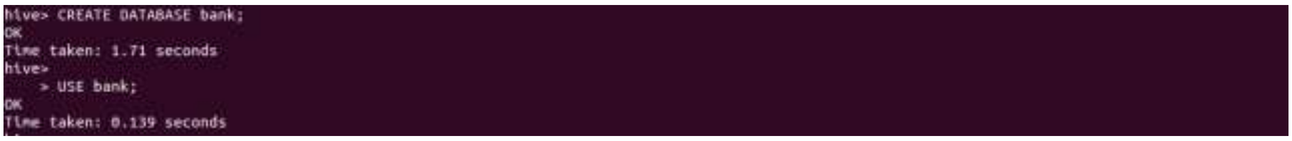
```
kiran@Acadgild:~$ sqoop job --exec sqoop_shares_info
Warning: /home/kiran/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
17/03/03 14:39:35 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-1.1.2/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/Sql_to_Nosql-0.0.1-SNAPSHOT-jar-with-dependencies.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
17/03/03 14:39:36 WARN sqoop.ConnFactory: Parameter --driver is set to an explicit driver however appropriate connection manager is not being se
t (via --connection-manager). Sqoop is going to fall back to org.apache.sqoop.manager.GenericJdbcManager. Please specify explicitly which connec
tion manager should be used next time.
17/03/03 14:39:36 INFO manager.SqlManager: Using default fetchSize of 1000
17/03/03 14:39:36 INFO tool.CodeGenTool: Beginning code generation
17/03/03 14:39:36 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM shares_info AS t WHERE 1=0
17/03/03 14:39:36 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM shares_info AS t WHERE 1=0
17/03/03 14:39:36 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/kiran/hadoop-2.7.1
Note: /tmp/sqoop-kiran/compile/d106d34d6459c9ac3638bb9edc9c46f5/shares_info.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
17/03/03 14:39:38 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-kiran/compile/d106d34d6459c9ac3638bb9edc9c46f5/shares_info.jar
17/03/03 14:39:39 INFO tool.ImportTool: Maximal id query for free form incremental import: SELECT MAX(gmt_timestamp) FROM shares_info
17/03/03 14:39:39 INFO tool.ImportTool: Incremental import based on column gmt_timestamp
17/03/03 14:39:39 INFO tool.ImportTool: Upper bound value: 1488412702
17/03/03 14:39:39 INFO mapreduce.ImportJobBase: Beginning import of shares_info
17/03/03 14:39:39 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
17/03/03 14:39:39 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM shares_info AS t WHERE 1=0
17/03/03 14:39:39 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
17/03/03 14:39:39 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/03/03 14:39:42 INFO db.DBInputFormat: Using read commited transaction isolation
17/03/03 14:39:42 INFO mapreduce.JobSubmitter: number of splits:1
17/03/03 14:39:43 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1488448982873_0016
17/03/03 14:39:43 INFO impl.YarnClientImpl: Submitted application application_1488448982873_0016
17/03/03 14:39:43 INFO mapreduce.Job: The url to track the job: http://Acadgild:8088/proxy/application_1488448982873_0016/
17/03/03 14:39:43 INFO mapreduce.Job: Running job: job_1488448982873_0016
17/03/03 14:39:48 INFO mapreduce.Job: Job job_1488448982873_0016 running in uber mode : false
```

3.Creating external tables in hive

Create database

CREATE DATABASE bank;

USE bank;

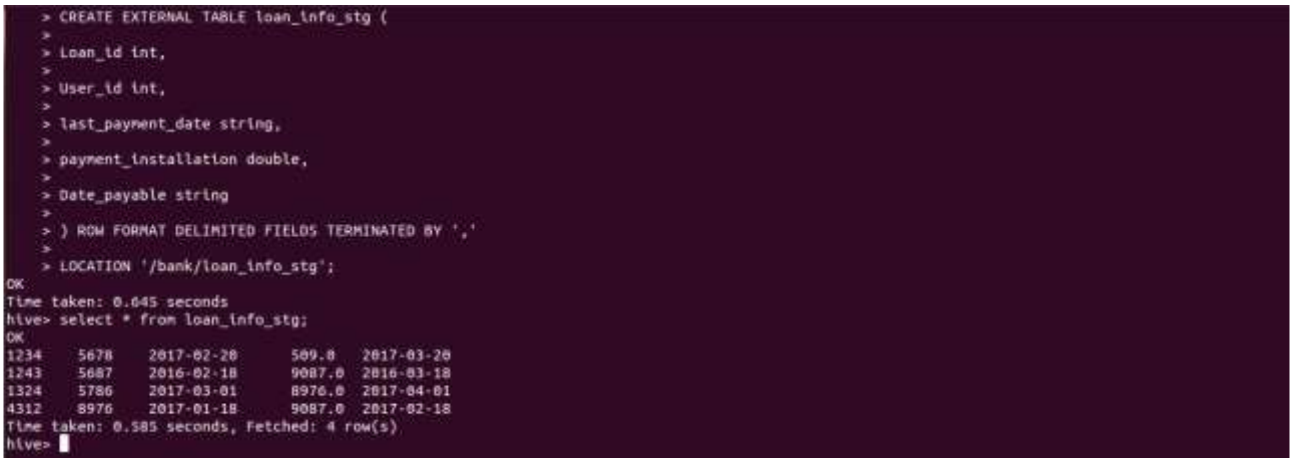


```
hive> CREATE DATABASE bank;
OK
Time taken: 1.71 seconds
hive>
    > USE bank;
OK
Time taken: 0.139 seconds
```

Creating table loan_info_stg

As this table is an external table, we just need to give the location of the data.

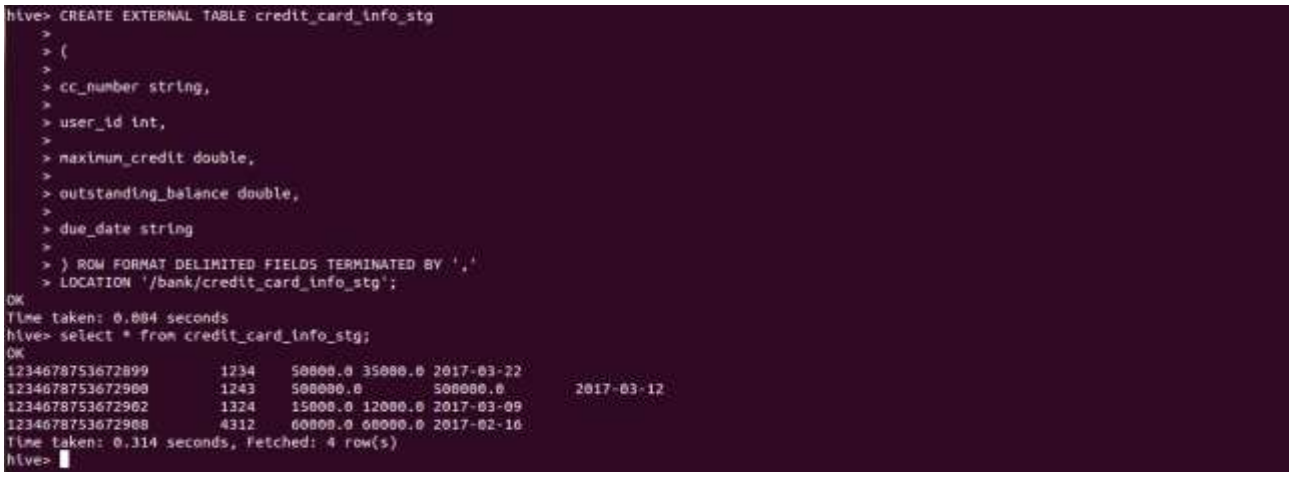
```
CREATE EXTERNAL TABLE loan_info_stg (
Loan_id int,
User_id int,
last_payment_date string,
payment_installation double,
Date_payable string
) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/bank/loan_info_stg';
```



```
    > CREATE EXTERNAL TABLE loan_info_stg (
    >
    > Loan_id int,
    >
    > User_id int,
    >
    > last_payment_date string,
    >
    > payment_installation double,
    >
    > Date_payable string
    >
    > ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    >
    > LOCATION '/bank/loan_info_stg';
OK
Time taken: 0.045 seconds
hive> select * from loan_info_stg;
OK
1234    5678    2017-02-20      509.0   2017-03-20
1243    5687    2016-02-18      9087.0  2016-03-18
1324    5786    2017-03-01      8976.0  2017-04-01
4312    8976    2017-01-18      9087.0  2017-02-18
Time taken: 0.585 seconds, Fetched: 4 row(s)
hive>
```

Creating table credit_card_info_stg

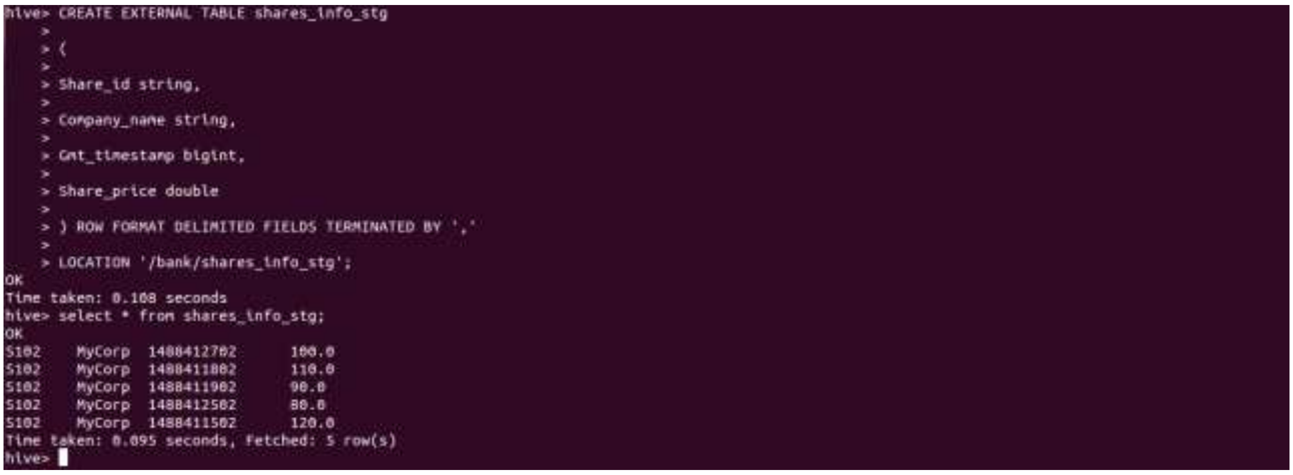
```
CREATE EXTERNAL TABLE credit_card_info_stg
(
cc_number string,
user_id int,
maximum_credit double,
outstanding_balance double,
due_date string
) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/bank/credit_card_info_stg';
```



```
hive> CREATE EXTERNAL TABLE credit_card_info_stg
    >
    > (
    >
    > cc_number string,
    >
    > user_id int,
    >
    > maximum_credit double,
    >
    > outstanding_balance double,
    >
    > due_date string
    >
    > ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LOCATION '/bank/credit_card_info_stg';
OK
Time taken: 0.084 seconds
hive> select * from credit_card_info_stg;
OK
1234678753672899        1234    50000.0 35000.0 2017-03-22
1234678753672900        1243    500000.0        500000.0        2017-03-12
1234678753672902        1324    15000.0 12000.0 2017-03-09
1234678753672908        4312    60000.0 60000.0 2017-02-16
Time taken: 0.314 seconds, Fetched: 4 row(s)
hive>
```

Creating table shares_info_stg

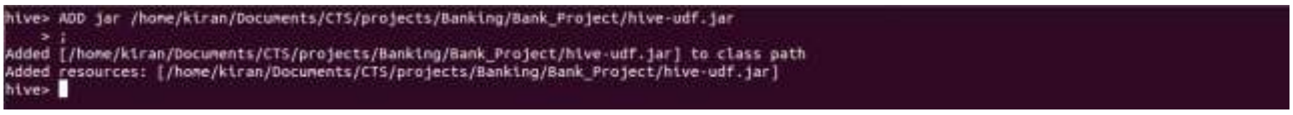
```
CREATE EXTERNAL TABLE shares_info_stg
(
Share_id string,
Company_name string,
Gmt_timestamp bigint,
Share_price double
) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/bank/shares_info_stg';
```



```
hive> CREATE EXTERNAL TABLE shares_info_stg
    >
    > (
    >
    > Share_id string,
    >
    > Company_name string,
    >
    > Gmt_timestamp bigint,
    >
    > Share_price double
    >
    > ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    >
    > LOCATION '/bank/shares_info_stg';
OK
Time taken: 0.108 seconds
hive> select * from shares_info_stg;
OK
S102    MyCorp  1488412702      100.0
S102    MyCorp  1488411802      110.0
S102    MyCorp  1488411902      90.0
S102    MyCorp  1488412502      80.0
S102    MyCorp  1488411502      120.0
Time taken: 0.095 seconds, Fetched: 5 row(s)
hive>
```

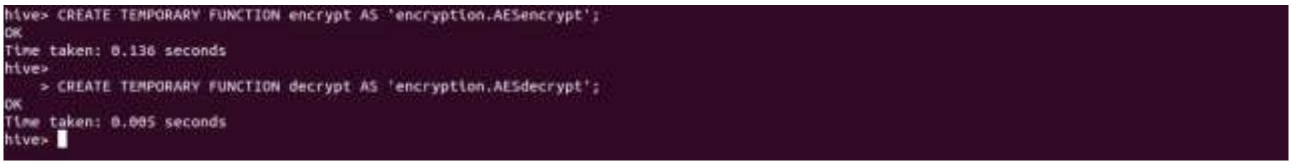
Creating core tables and loading the data into the core tables from stg tables

Adding the udf into hive shell.



```
hive> ADD jar /home/kiran/Documents/CTS/projects/Banking/Bank_Project/hive-udf.jar
    > ;
Added [/home/kiran/Documents/CTS/projects/Banking/Bank_Project/hive-udf.jar] to class path
Added resources: [/home/kiran/Documents/CTS/projects/Banking/Bank_Project/hive-udf.jar]
hive>
```

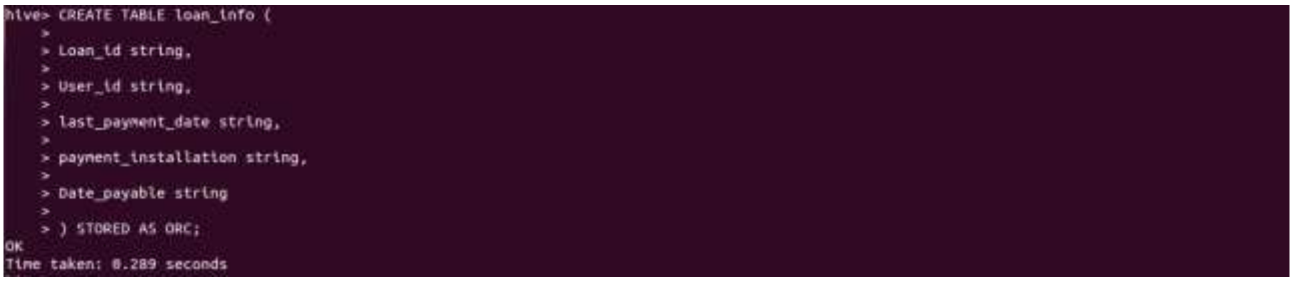
CREATE TEMPORARY FUNCTION encrypt AS 'encryption.AESencrypt';
CREATE TEMPORARY FUNCTION decrypt AS 'encryption.AESdecrypt';



```
hive> CREATE TEMPORARY FUNCTION encrypt AS 'encryption.AESencrypt';
OK
Time taken: 0.136 seconds
hive>
    > CREATE TEMPORARY FUNCTION decrypt AS 'encryption.AESdecrypt';
OK
Time taken: 0.005 seconds
hive>
```

Creating loan_info table

CREATE TABLE loan_info (
Loan_id string,
User_id string,
last_payment_date string,
payment_installation string,
Date_payable string
) STORED AS ORC;

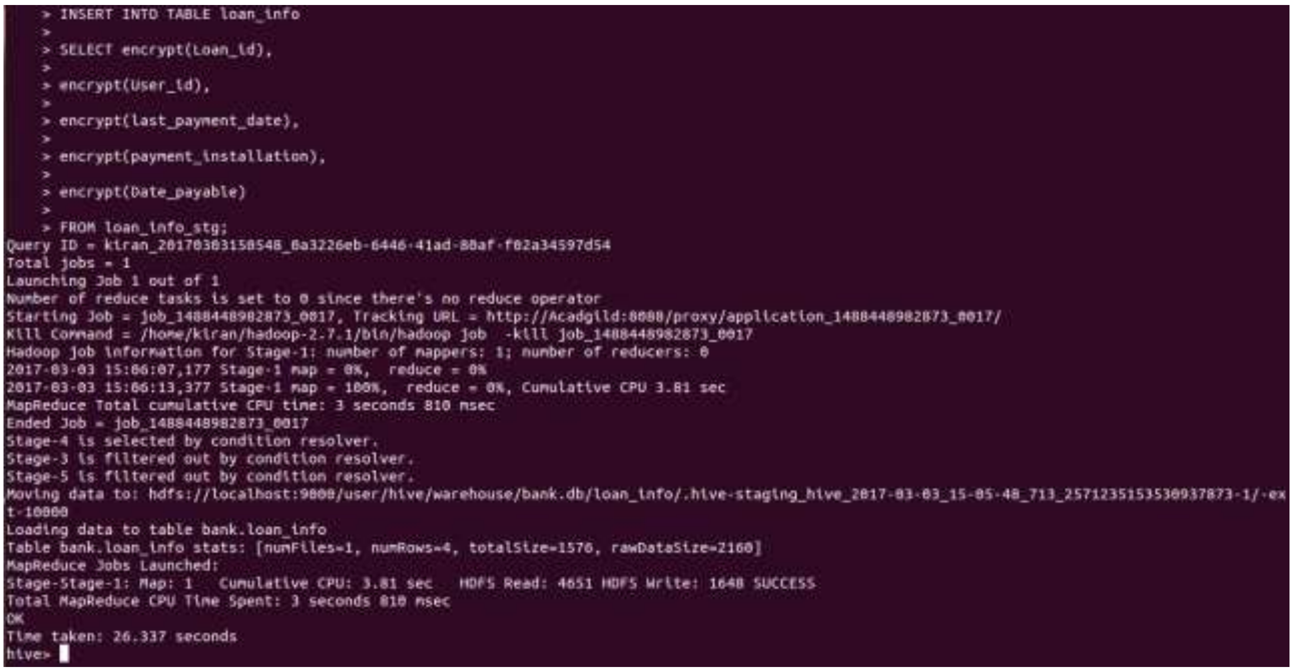


```
hive> CREATE TABLE loan_info (
    >
    > Loan_id string,
    >
    > User_id string,
    >
    > last_payment_date string,
    >
    > payment_installation string,
    >
    > Date_payable string
    >
    > ) STORED AS ORC;
OK
Time taken: 0.289 seconds
```

Inserting data into loan_info table

INSERT INTO TABLE loan_info

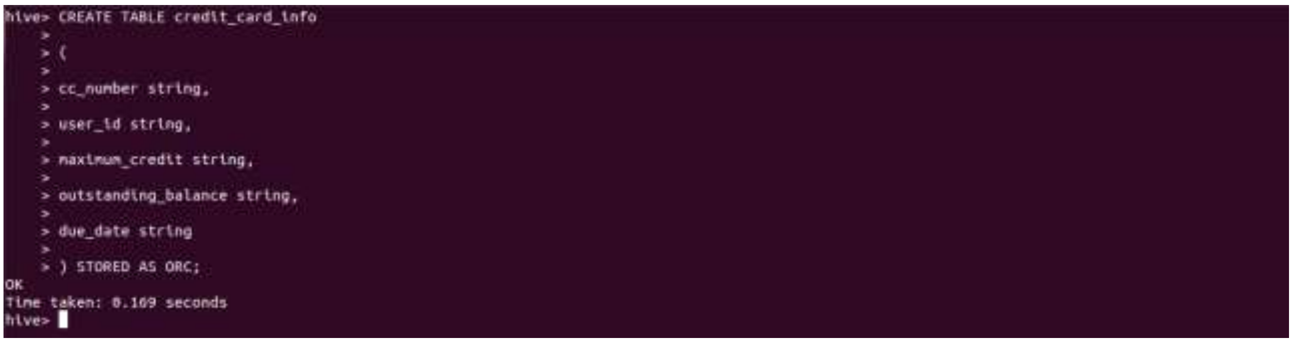
SELECT encrypt(Loan_id),
encrypt(User_id),
encrypt(last_payment_date),
encrypt(payment_installation),
encrypt(Date_payable)
FROM loan_info_stg;



```
    > INSERT INTO TABLE loan_info
    >
    > SELECT encrypt(Loan_id),
    >
    > encrypt(User_id),
    >
    > encrypt(last_payment_date),
    >
    > encrypt(payment_installation),
    >
    > encrypt(Date_payable)
    >
    > FROM loan_info_stg;
Query ID = kiran_20170303150548_0a3226eb-6446-41ad-80af-f02a34597d54
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1488448982873_0017, Tracking URL = http://Acadgild:8088/proxy/application_1488448982873_0017/
Kill Command = /home/kiran/hadoop-2.7.1/bin/hadoop job  -kill job_1488448982873_0017
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-03-03 15:06:07,177 Stage-1 map = 0%,  reduce = 0%
2017-03-03 15:06:13,377 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.81 sec
MapReduce Total cumulative CPU time: 3 seconds 810 msec
Ended Job = job_1488448982873_0017
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:9000/user/hive/warehouse/bank.db/loan_info/.hive-staging_hive_2017-03-03_15-05-48_713_2571235153530937873-1/-ex
t-10000
Loading data to table bank.loan_info
Table bank.loan_info stats: [numFiles=1, numRows=4, totalSize=1576, rawDataSize=2160]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 3.81 sec   HDFS Read: 4651 HDFS Write: 1648 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 810 msec
OK
Time taken: 26.337 seconds
hive>
```

Creating credit_card_info table

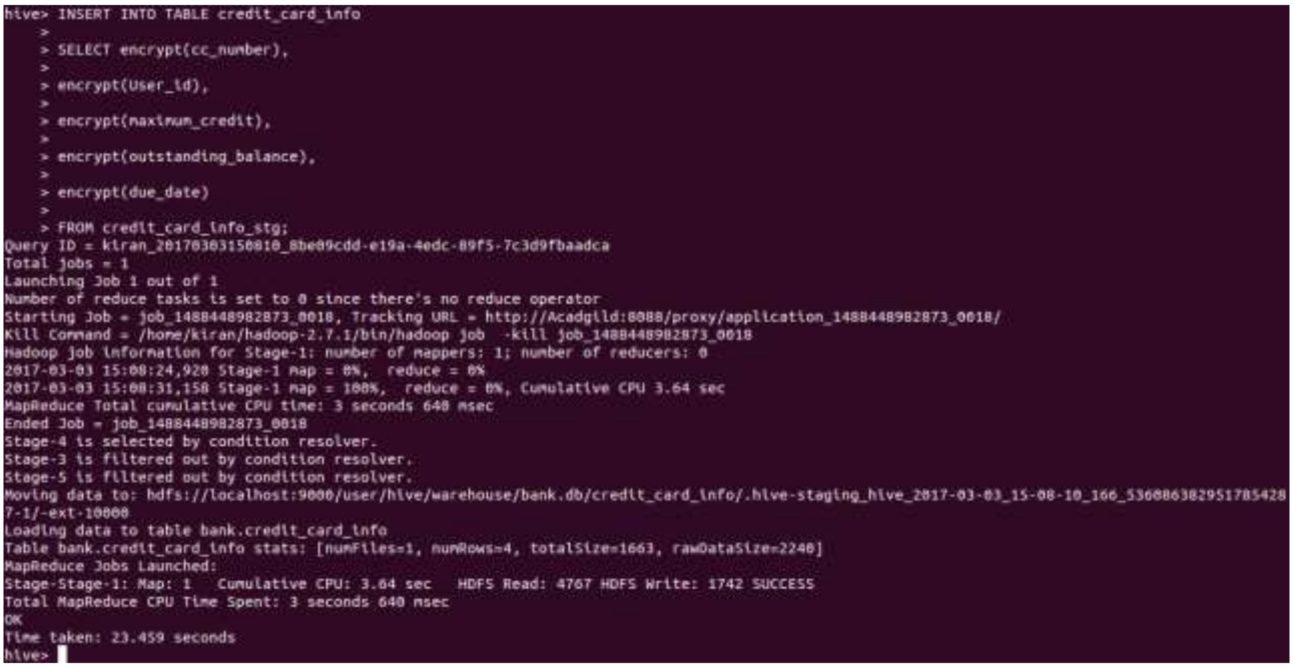
CREATE TABLE credit_card_info
(
cc_number string,
user_id string,
maximum_credit string,
outstanding_balance string,
due_date string
) STORED AS ORC;



```
hive> CREATE TABLE credit_card_info
    >
    > (
    >
    > cc_number string,
    >
    > user_id string,
    >
    > maximum_credit string,
    >
    > outstanding_balance string,
    >
    > due_date string
    >
    > ) STORED AS ORC;
OK
Time taken: 0.169 seconds
hive>
```

Inserting data into credit_card_info table

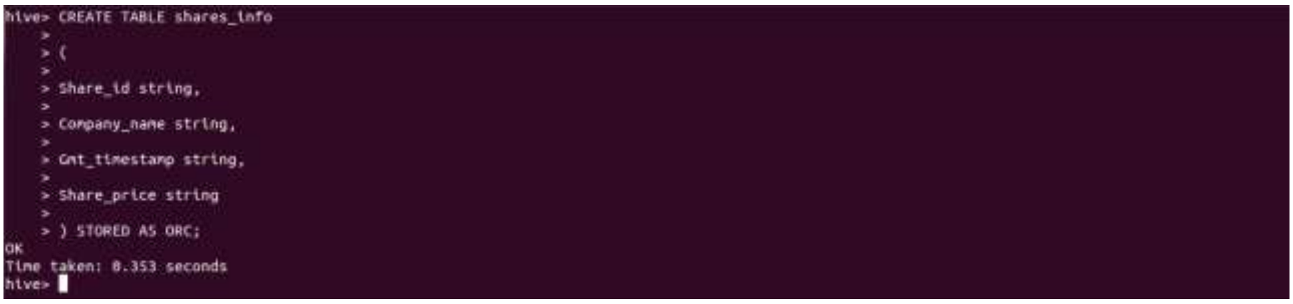
INSERT INTO TABLE credit_card_info
SELECT encrypt(cc_number),
encrypt(User_id),
encrypt(maximum_credit),
encrypt(outstanding_balance),
encrypt(due_date)
FROM credit_card_info_stg;



```
hive> INSERT INTO TABLE credit_card_info
    >
    > SELECT encrypt(cc_number),
    >
    > encrypt(User_id),
    >
    > encrypt(maximum_credit),
    >
    > encrypt(outstanding_balance),
    >
    > encrypt(due_date)
    >
    > FROM credit_card_info_stg;
Query ID = kiran_20170303150810_8be09cdd-e19a-4edc-89f5-7c3d9fbaadca
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1488448982873_0018, Tracking URL = http://Acadgild:8088/proxy/application_1488448982873_0018/
Kill Command = /home/kiran/hadoop-2.7.1/bin/hadoop job  -kill job_1488448982873_0018
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-03-03 15:08:24,928 Stage-1 map = 0%,  reduce = 0%
2017-03-03 15:08:31,158 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.64 sec
MapReduce Total cumulative CPU time: 3 seconds 640 msec
Ended Job = job_1488448982873_0018
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:9000/user/hive/warehouse/bank.db/credit_card_info/.hive-staging_hive_2017-03-03_15-08-10_166_5360863829517854287-1/-ext-10000
Loading data to table bank.credit_card_info
Table bank.credit_card_info stats: [numFiles=1, numRows=4, totalSize=1663, rawDataSize=2240]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 3.64 sec   HDFS Read: 4767 HDFS Write: 1742 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 640 msec
OK
Time taken: 23.459 seconds
hive>
```

Creating shares_info table

CREATE TABLE shares_info
(
Share_id string,
Company_name string,
Gmt_timestamp string,
Share_price string
) STORED AS ORC;

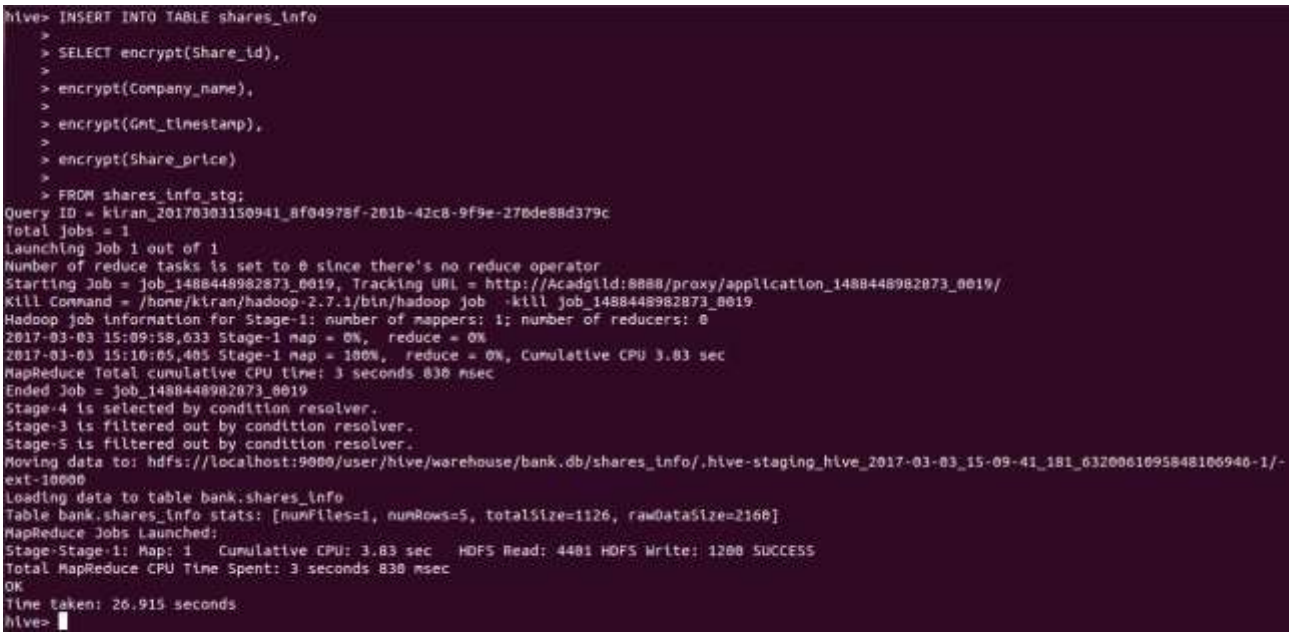


```
hive> CREATE TABLE shares_info
    >
    > (
    >
    > Share_id string,
    >
    > Company_name string,
    >
    > Gmt_timestamp string,
    >
    > Share_price string
    >
    > ) STORED AS ORC;
OK
Time taken: 0.353 seconds
hive>
```

Inserting data into shares_info table

INSERT INTO TABLE shares_info
SELECT encrypt(Share_id),
encrypt(Company_name),
encrypt(Gmt_timestamp),

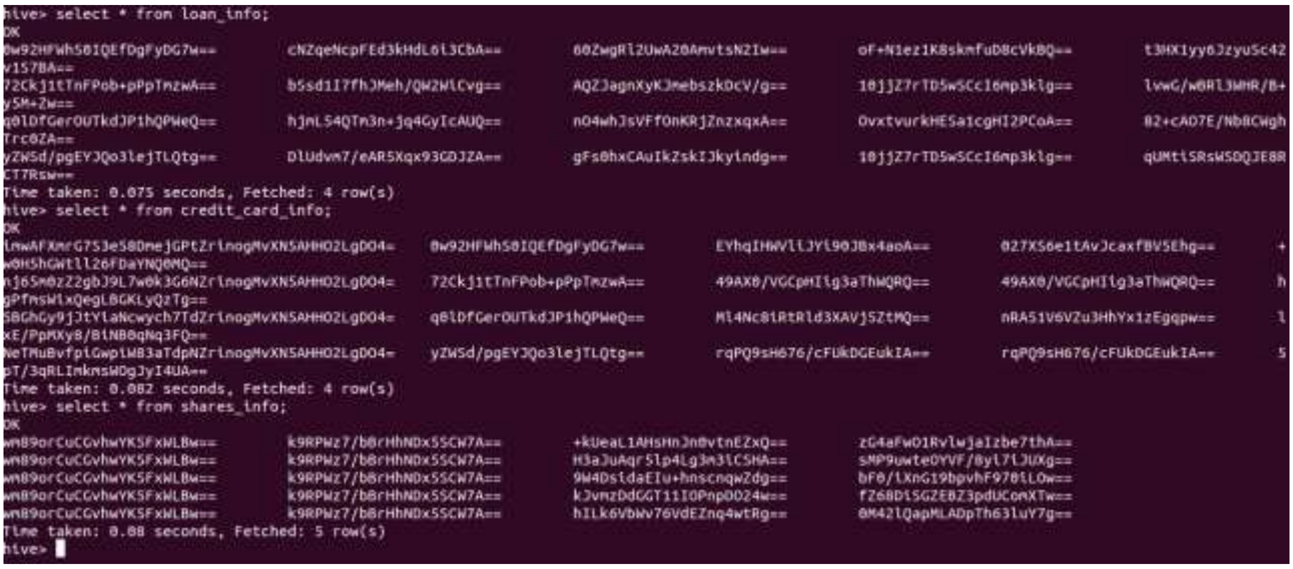
encrypt(Share_price)
FROM shares_info_stg;



```
hive> INSERT INTO TABLE shares_info
    >
    > SELECT encrypt(Share_id),
    >
    > encrypt(Company_name),
    >
    > encrypt(Gmt_timestamp),
    >
    > encrypt(Share_price)
    >
    > FROM shares_info_stg;
Query ID = kiran_20170303150941_8f04978f-201b-42c8-9f9e-270de88d379c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1488448982873_0019, Tracking URL = http://Acadgild:8088/proxy/application_1488448982873_0019/
Kill Command = /home/kiran/hadoop-2.7.1/bin/hadoop job  -kill job_1488448982873_0019
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-03-03 15:09:58,633 Stage-1 map = 0%,  reduce = 0%
2017-03-03 15:10:05,405 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.83 sec
MapReduce Total cumulative CPU time: 3 seconds 830 msec
Ended Job = job_1488448982873_0019
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:9000/user/hive/warehouse/bank.db/shares_info/.hive-staging_hive_2017-03-03_15-09-41_181_6320061095848106946-1/-
ext-10000
Loading data to table bank.shares_info
Table bank.shares_info stats: [numFiles=1, numRows=5, totalSize=1126, rawDataSize=2160]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 3.83 sec   HDFS Read: 4401 HDFS Write: 1200 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 830 msec
OK
Time taken: 26.915 seconds
hive>
```

Checking the data in the three tables

As the data is bank data, we have encrypted the data.



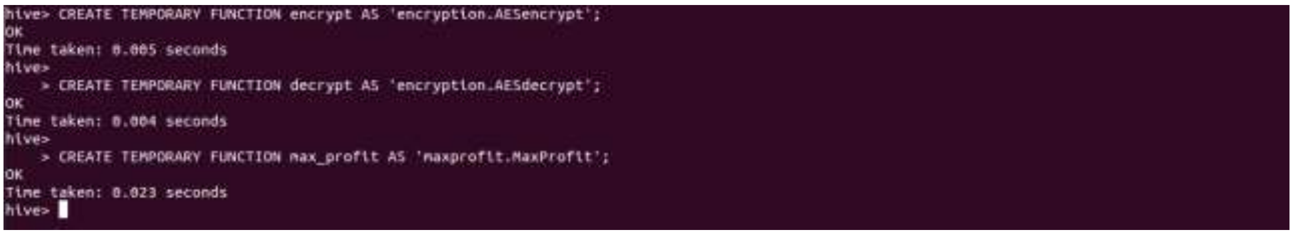
```
hive> select * from loan_info;
OK
0w92HFWhS0IQEfDgFyDG7w==        cNZqeNcpFEd3kHdL6i3CbA==        60ZwgRl2UwA20AmvtsNZIw==        oF+N1ez1K8skmfuD8cVkBQ==        t3HX1yy6JzyuSc42
v157BA==
72Ckj1tTnFPob+pPpTmzwA==        b5sd1I7fhJMeh/QW2WlCvg==        AQZJagnXyKJmebszkOcV/g==        10jjZ7rTD5wSCcI6mp3klg==        lvwG/w0Rl3WHR/B+
y5M+Zw==
q0lDfGerOUTkdJP1hQPWeQ==        hjmL54QTm3n+jq4GyIcAUQ==        nO4whJsVFfOnKRjZnzxqxA==        OvxtvurkHESa1cgHI2PCoA==        82+cAD7E/NbRCWgh
Trc0ZA==
yZWSd/pgEYJQo3lejTLQtg==        DlUdvm7/eAR5Xqx93GDJZA==        gFs0hxCAuIkZskIJkyindg==        10jjZ7rTD5wSCcI6mp3klg==        qUMtiSRsWSDQJE8R
CT7Rsw==
Time taken: 0.075 seconds, Fetched: 4 row(s)
hive> select * from credit_card_info;
OK
lmwAFXmrG7S3e58DmejGPtZrinogMvXNSAHHO2LgDO4=    0w92HFWhS0IQEfDgFyDG7w==        EYhqIHWVlLJYi90JBx4aoA==        027XS6e1tAvJcaxfBV5Ehg==        +
w0HShGWtll26fDaYNQ0MQ==
nj65m0zZ2gbJ9L7w0k3G6NZrinogMvXNSAHHO2LgDO4=    72Ckj1tTnFPob+pPpTmzwA==        49AX0/VGCpHIlg3aThWQRQ==        49AX0/VGCpHIlg3aThWQRQ==        h
gPfmsWlxQegLBGKLyQzTg==
5BGhGy9jJtYiaNcwych7TdZrinogMvXNSAHHO2LgDO4=    q0lDfGerOUTkdJP1hQPWeQ==        Ml4Nc0iRtRld3XAVj5ZtMQ==        nRA51V6VZu3HhYx1zEgqpw==        l
xE/PpMXy8/BiNB0qNq3FQ==
NeTMuBvfpiGwpiW83aTdpNZrinogMvXNSAHHO2LgDO4=    yZWSd/pgEYJQo3lejTLQtg==        rqPQ9sH676/cFUkDGEukIA==        rqPQ9sH676/cFUkDGEukIA==        5
pT/3qRLImkmsWDgJyI4UA==
Time taken: 0.082 seconds, Fetched: 4 row(s)
hive> select * from shares_info;
OK
wm89orCuCGvhwYK5FxWLBw==        k9RPWz7/b0rHhNDx5SCW7A==        +kUeaL1AHsHnJn0vtnEZxQ==        zG4aFwO1RvlwjaIzbe7thA==
wm89orCuCGvhwYK5FxWLBw==        k9RPWz7/b0rHhNDx5SCW7A==        H3aJuAqr5lp4Lg3m3lCSHA==        sMP9uwteOYVF/0yl7lJUXg==
wm89orCuCGvhwYK5FxWLBw==        k9RPWz7/b0rHhNDx5SCW7A==        9W4DsidaEIu+hnscnqwZdg==        bF0/lXnG19bpvhF970lLOw==
wm89orCuCGvhwYK5FxWLBw==        k9RPWz7/b0rHhNDx5SCW7A==        kJvmzDdGGT11IOPnpDO24w==        fZ68DlSGZEBZ3pdUComXTw==
wm89orCuCGvhwYK5FxWLBw==        k9RPWz7/b0rHhNDx5SCW7A==        hILk6VbWv76VdEZnq4wtRg==        0M42lQapMLADpTh63luY7g==
Time taken: 0.08 seconds, Fetched: 5 row(s)
hive>
```

You can truncate the data from stg tables.

5.Analysis

Decrypting the data for analysis

CREATE TEMPORARY FUNCTION encrypt AS 'encryption.AESencrypt';
CREATE TEMPORARY FUNCTION decrypt AS 'encryption.AESdecrypt';
CREATE TEMPORARY FUNCTION max_profit AS 'maxprofit.MaxProfit';

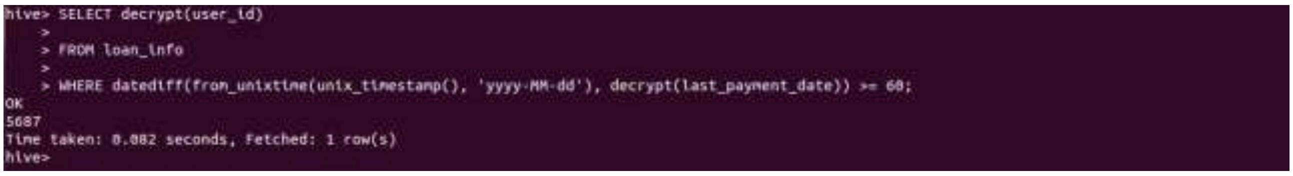


```
hive> CREATE TEMPORARY FUNCTION encrypt AS 'encryption.AESencrypt';
OK
Time taken: 0.005 seconds
hive>
    > CREATE TEMPORARY FUNCTION decrypt AS 'encryption.AESdecrypt';
OK
Time taken: 0.004 seconds
hive>
    > CREATE TEMPORARY FUNCTION max_profit AS 'maxprofit.MaxProfit';
OK
Time taken: 0.023 seconds
hive> 
```

SET hive.auto.convert.join=false;

6.1. Find out the list of users who have at least 2 loan instalments pending.

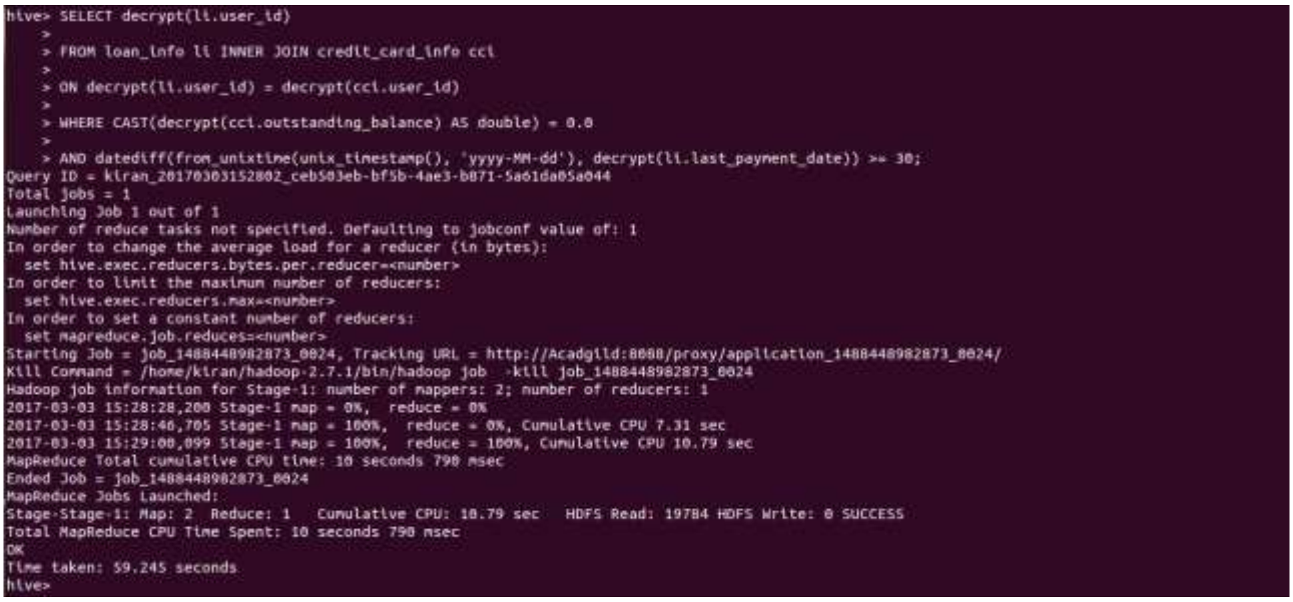
SELECT decrypt(user_id)
FROM loan_info
WHERE datediff(from_unixtime(unix_timestamp(), 'yyyy-MM-dd'), decrypt(last_payment_date)) >= 60;



```
hive> SELECT decrypt(user_id)
    >
    > FROM loan_info
    >
    > WHERE datediff(from_unixtime(unix_timestamp(), 'yyyy-MM-dd'), decrypt(last_payment_date)) >= 60;
OK
5687
Time taken: 0.082 seconds, Fetched: 1 row(s)
hive>
```

6.2. Find the list of users who have a healthy credit card but outstanding loan account. Healthy credit card means no outstanding balance.

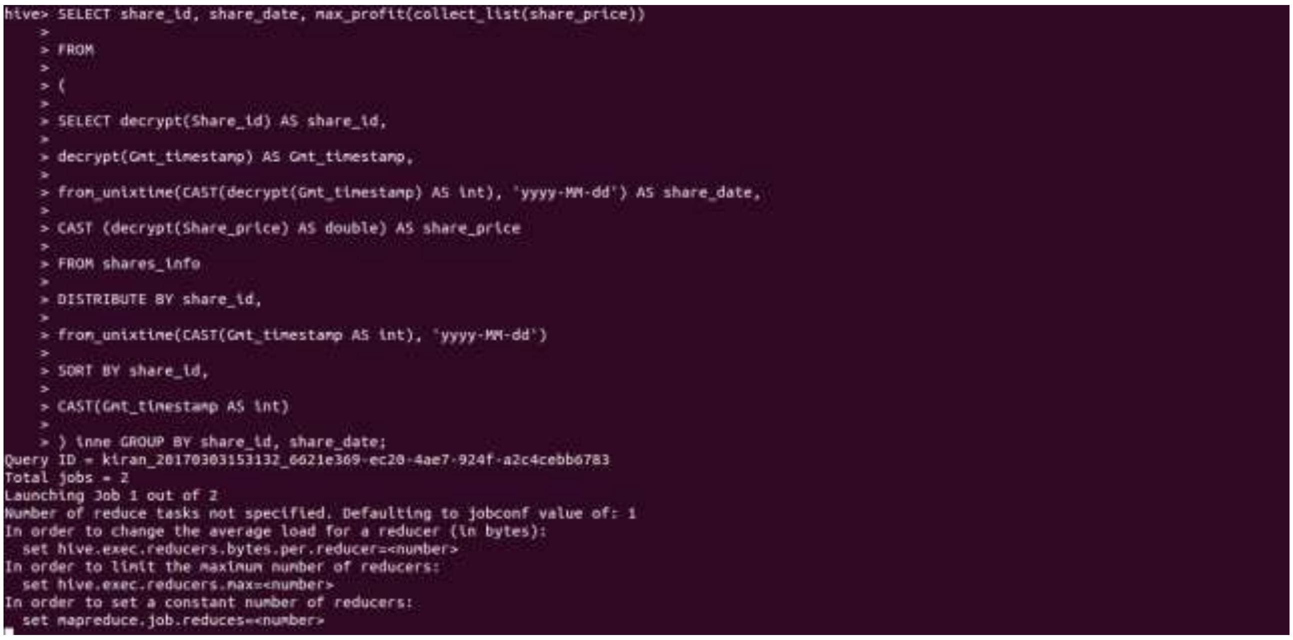
SELECT decrypt(li.user_id)
FROM loan_info li INNER JOIN credit_card_info cci
ON decrypt(li.user_id) = decrypt(cci.user_id)
WHERE CAST(decrypt(cci.outstanding_balance) AS double) = 0.0
AND datediff(from_unixtime(unix_timestamp(), 'yyyy-MM-dd'), decrypt(li.last_payment_date)) >= 30;



```
hive> SELECT decrypt(li.user_id)
    >
    > FROM loan_info li INNER JOIN credit_card_info cci
    >
    > ON decrypt(li.user_id) = decrypt(cci.user_id)
    >
    > WHERE CAST(decrypt(cci.outstanding_balance) AS double) = 0.0
    >
    > AND datediff(from_unixtime(unix_timestamp(), 'yyyy-MM-dd'), decrypt(li.last_payment_date)) >= 30;
Query ID = kiran_20170303152802_ceb503eb-bf5b-4ae3-b871-5a61da05a044
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Defaulting to jobconf value of: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1488448982873_0024, Tracking URL = http://Acadgild:8088/proxy/application_1488448982873_0024/
Kill Command = /home/kiran/hadoop-2.7.1/bin/hadoop job  -kill job_1488448982873_0024
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2017-03-03 15:28:28,200 Stage-1 map = 0%,  reduce = 0%
2017-03-03 15:28:46,705 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 7.31 sec
2017-03-03 15:29:00,099 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 10.79 sec
MapReduce Total cumulative CPU time: 10 seconds 790 msec
Ended Job = job_1488448982873_0024
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 1   Cumulative CPU: 10.79 sec   HDFS Read: 19784 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 10 seconds 790 msec
OK
Time taken: 59.245 seconds
hive>
```

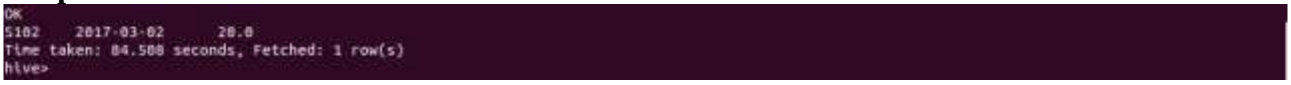
6.3. For every share and for every date, find the maximum profit one could have made on the share. Bear in mind that a share purchase must be before share sell and if share prices fall throughout the day, maximum possible profit may be negative.

```
SELECT share_id, share_date, max_profit(collect_list(share_price))
FROM
(
SELECT decrypt(Share_id) AS share_id,
decrypt(Gmt_timestamp) AS Gmt_timestamp,
from_unixtime(CAST(decrypt(Gmt_timestamp) AS int), 'yyyy-MM-dd') AS share_date,
CAST (decrypt(Share_price) AS double) AS share_price
FROM shares_info
DISTRIBUTE BY share_id,
from_unixtime(CAST(Gmt_timestamp AS int), 'yyyy-MM-dd')
SORT BY share_id,
CAST(Gmt_timestamp AS int)
) inne GROUP BY share_id, share_date;
```



```
hive> SELECT share_id, share_date, max_profit(collect_list(share_price))
    >
    > FROM
    >
    > (
    >
    > SELECT decrypt(Share_id) AS share_id,
    >
    > decrypt(Gmt_timestamp) AS Gmt_timestamp,
    >
    > from_unixtime(CAST(decrypt(Gmt_timestamp) AS int), 'yyyy-MM-dd') AS share_date,
    >
    > CAST (decrypt(Share_price) AS double) AS share_price
    >
    > FROM shares_info
    >
    > DISTRIBUTE BY share_id,
    >
    > from_unixtime(CAST(Gmt_timestamp AS int), 'yyyy-MM-dd')
    >
    > SORT BY share_id,
    >
    > CAST(Gmt_timestamp AS int)
    >
    > ) inne GROUP BY share_id, share_date;
Query ID = kiran_20170303153132_6621e369-ec20-4ae7-924f-a2c4cebb6783
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Defaulting to jobconf value of: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
```

Output



```
OK
S102    2017-03-02      20.0
Time taken: 84.508 seconds, Fetched: 1 row(s)
hive>
```

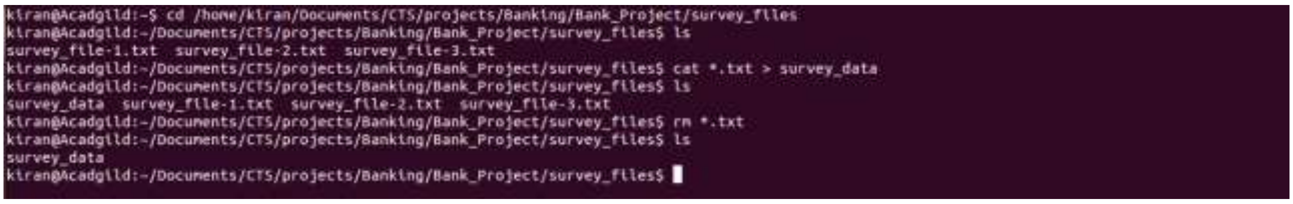
7.Archival

8.Survey data analysis

We have 3 survery part files. So we will copy the contents into a single file using the below linux commands.

```
cd /home/acadgild/survey_files
cat *.txt > survey_data
```

rm *.txt

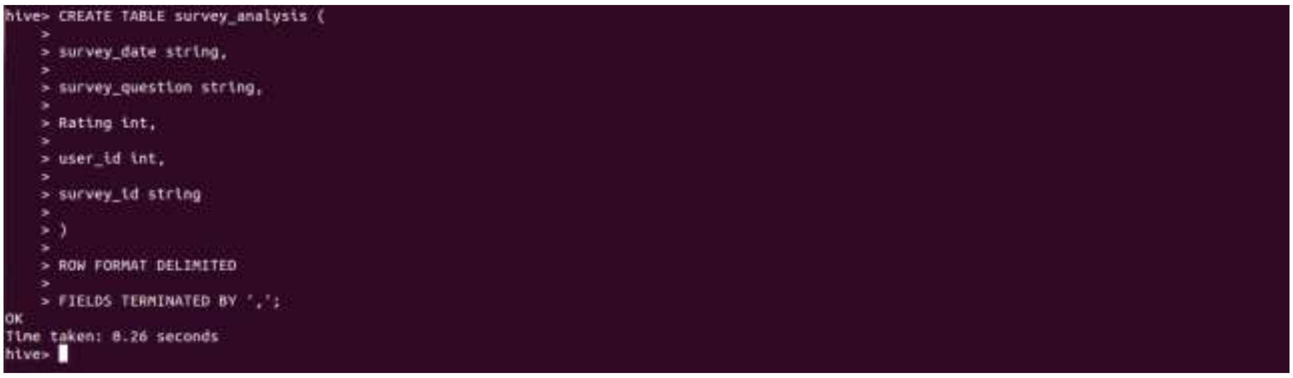


```
kiran@Acadgild:~$ cd /home/kiran/Documents/CTS/projects/Banking/Bank_Project/survey_files
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/survey_files$ ls
survey_file-1.txt  survey_file-2.txt  survey_file-3.txt
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/survey_files$ cat *.txt > survey_data
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/survey_files$ ls
survey_data  survey_file-1.txt  survey_file-2.txt  survey_file-3.txt
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/survey_files$ rm *.txt
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/survey_files$ ls
survey_data
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/survey_files$ 
```

Now we have the concated data in survey_data file.

Creating hive table to load survey_data

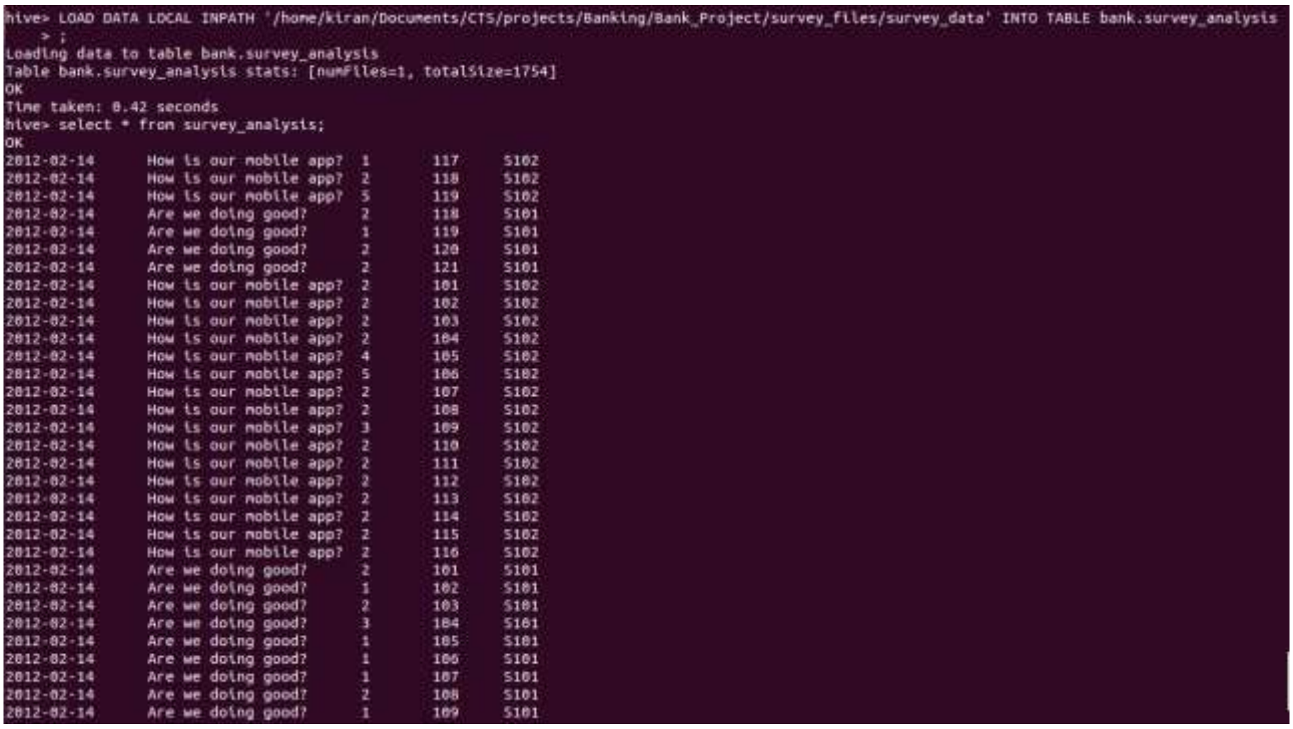
```
CREATE TABLE survey_analysis (
survey_date string,
survey_question string,
Rating int,
user_id int,
survey_id string
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```



```
hive> CREATE TABLE survey_analysis (
    >
    > survey_date string,
    >
    > survey_question string,
    >
    > Rating int,
    >
    > user_id int,
    >
    > survey_id string
    >
    > )
    >
    > ROW FORMAT DELIMITED
    >
    > FIELDS TERMINATED BY ',';
OK
Time taken: 0.26 seconds
hive> 
```

Loading data into survey_analysis table

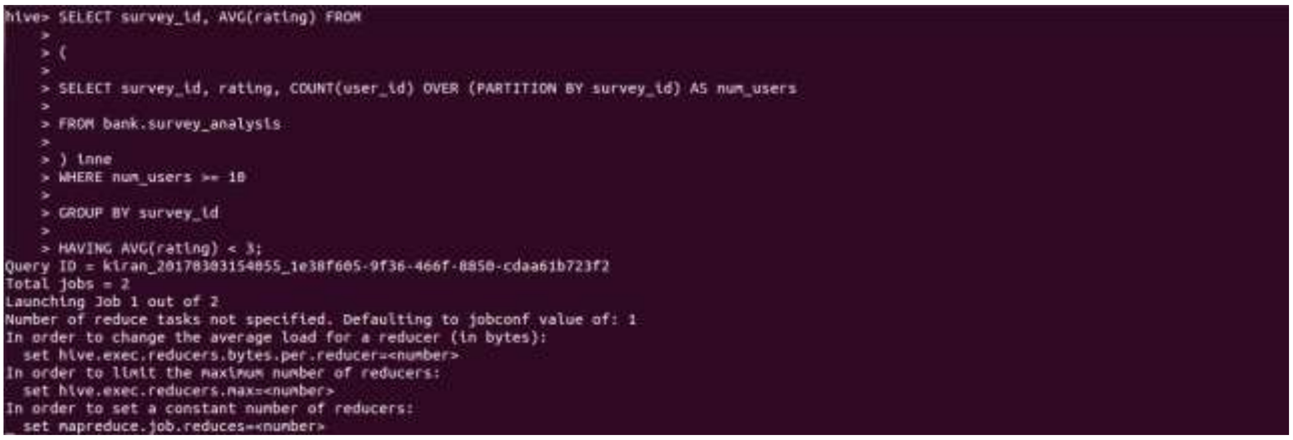
```
LOAD DATA LOCAL INPATH '/home/acadgild/survey_files/survey_data' INTO TABLE bank.survey_analysis;
```



```
hive> LOAD DATA LOCAL INPATH '/home/kiran/Documents/CTS/projects/Banking/Bank_Project/survey_files/survey_data' INTO TABLE bank.survey_analysis
    > ;
Loading data to table bank.survey_analysis
Table bank.survey_analysis stats: [numFiles=1, totalSize=1754]
OK
Time taken: 0.42 seconds
hive> select * from survey_analysis;
OK
2012-02-14      How is our mobile app?  1       117     S102
2012-02-14      How is our mobile app?  2       118     S102
2012-02-14      How is our mobile app?  5       119     S102
2012-02-14      Are we doing good?      2       118     S101
2012-02-14      Are we doing good?      1       119     S101
2012-02-14      Are we doing good?      2       120     S101
2012-02-14      Are we doing good?      2       121     S101
2012-02-14      How is our mobile app?  2       101     S102
2012-02-14      How is our mobile app?  2       102     S102
2012-02-14      How is our mobile app?  2       103     S102
2012-02-14      How is our mobile app?  2       104     S102
2012-02-14      How is our mobile app?  4       105     S102
2012-02-14      How is our mobile app?  5       106     S102
2012-02-14      How is our mobile app?  2       107     S102
2012-02-14      How is our mobile app?  2       108     S102
2012-02-14      How is our mobile app?  3       109     S102
2012-02-14      How is our mobile app?  2       110     S102
2012-02-14      How is our mobile app?  2       111     S102
2012-02-14      How is our mobile app?  2       112     S102
2012-02-14      How is our mobile app?  2       113     S102
2012-02-14      How is our mobile app?  2       114     S102
2012-02-14      How is our mobile app?  2       115     S102
2012-02-14      How is our mobile app?  2       116     S102
2012-02-14      Are we doing good?      2       101     S101
2012-02-14      Are we doing good?      1       102     S101
2012-02-14      Are we doing good?      2       103     S101
2012-02-14      Are we doing good?      3       104     S101
2012-02-14      Are we doing good?      1       105     S101
2012-02-14      Are we doing good?      1       106     S101
2012-02-14      Are we doing good?      1       107     S101
2012-02-14      Are we doing good?      2       108     S101
2012-02-14      Are we doing good?      1       109     S101
```

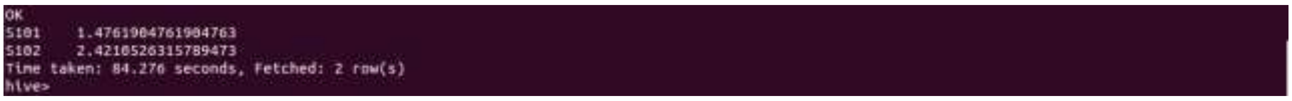
8.1. How many surveys got the average rating less than 3, provided at least 10 distinct users gave the rating?

```
SELECT survey_id, AVG(rating) FROM
(
SELECT survey_id, rating, COUNT(user_id) OVER (PARTITION BY survey_id) AS num_users
FROM bank.survey_analysis
) inne
WHERE num_users >= 10
GROUP BY survey_id
HAVING AVG(rating) < 3;
```



```
hive> SELECT survey_id, AVG(rating) FROM
    >
    > (
    >
    > SELECT survey_id, rating, COUNT(user_id) OVER (PARTITION BY survey_id) AS num_users
    >
    > FROM bank.survey_analysis
    >
    > ) inne
    > WHERE num_users >= 10
    >
    > GROUP BY survey_id
    >
    > HAVING AVG(rating) < 3;
Query ID = kiran_20170303154055_1e38f605-9f36-466f-8850-cdaa61b723f2
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Defaulting to jobconf value of: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
```

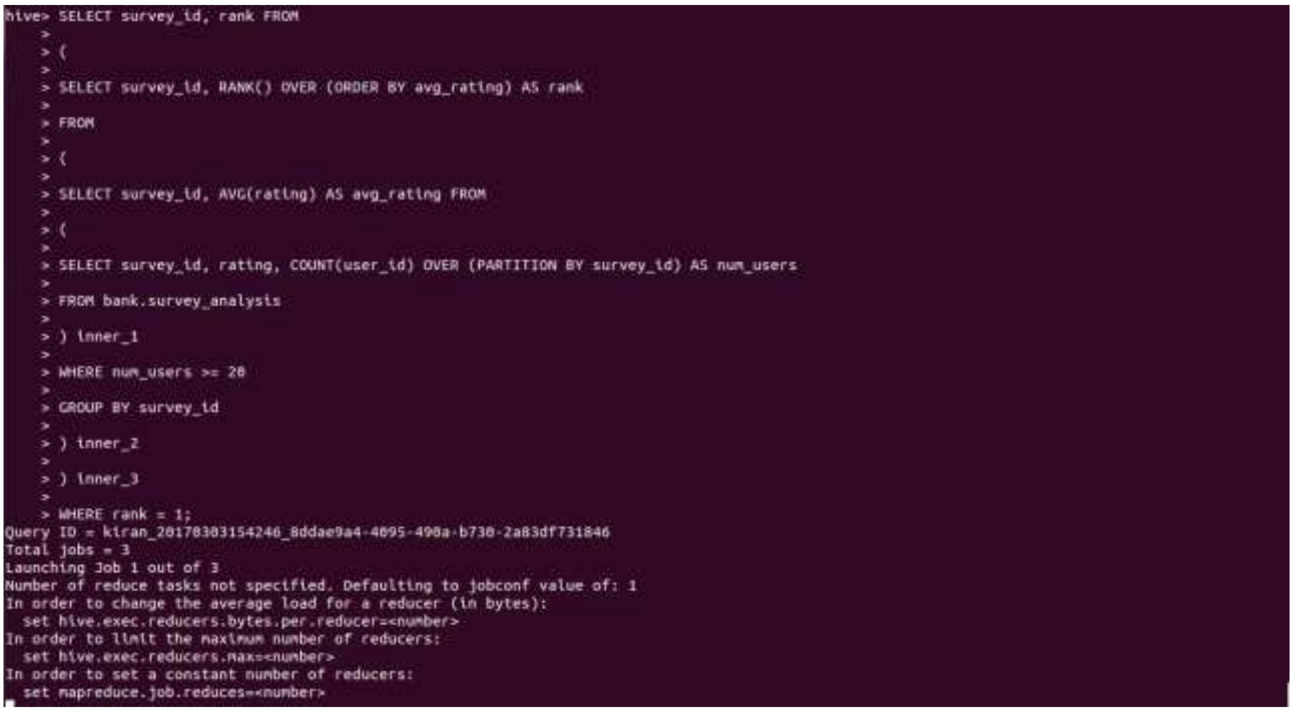
Output



```
OK
S101    1.4761904761904763
S102    2.4210526315789473
Time taken: 84.276 seconds, Fetched: 2 row(s)
hive>
```

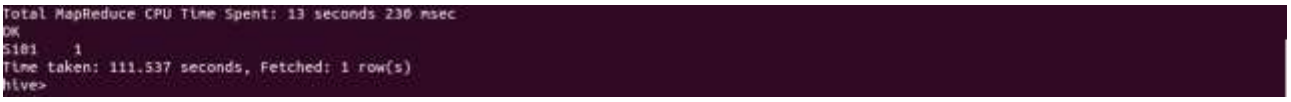
8.2. Find the details of the survey which received the minimum rating. The condition is that the survey must have been rated by at least 20 users.

SELECT survey_id, rank FROM
(
SELECT survey_id, RANK() OVER (ORDER BY avg_rating) AS rank
FROM
(
SELECT survey_id, AVG(rating) AS avg_rating FROM
(
SELECT survey_id, rating, COUNT(user_id) OVER (PARTITION BY survey_id) AS num_users
FROM bank.survey_analysis
) inner_1
WHERE num_users >= 20
GROUP BY survey_id
) inner_2
) inner_3
WHERE rank = 1;



```
hive> SELECT survey_id, rank FROM
    >
    > (
    >
    > SELECT survey_id, RANK() OVER (ORDER BY avg_rating) AS rank
    >
    > FROM
    >
    > (
    >
    > SELECT survey_id, AVG(rating) AS avg_rating FROM
    >
    > (
    >
    > SELECT survey_id, rating, COUNT(user_id) OVER (PARTITION BY survey_id) AS num_users
    >
    > FROM bank.survey_analysis
    >
    > ) inner_1
    >
    > WHERE num_users >= 20
    >
    > GROUP BY survey_id
    >
    > ) inner_2
    >
    > ) inner_3
    >
    > WHERE rank = 1;
Query ID = kiran_20170303154246_8ddae9a4-4095-498a-b730-2a83df731846
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks not specified. Defaulting to jobconf value of: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
```

Output



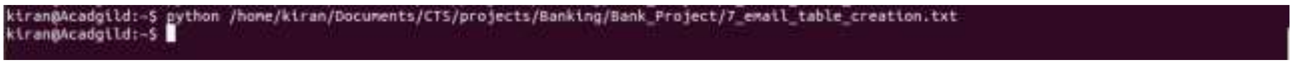
```
Total MapReduce CPU Time Spent: 13 seconds 230 msec
OK
S101    1
Time taken: 111.537 seconds, Fetched: 1 row(s)
hive>
```

Email data analysis

The organisation also has lots of emails stored in small files.
The metadata about the email is present in an XML file email_metadata.xml
Read the XML file for email structure and pack all the email files in HDFS.

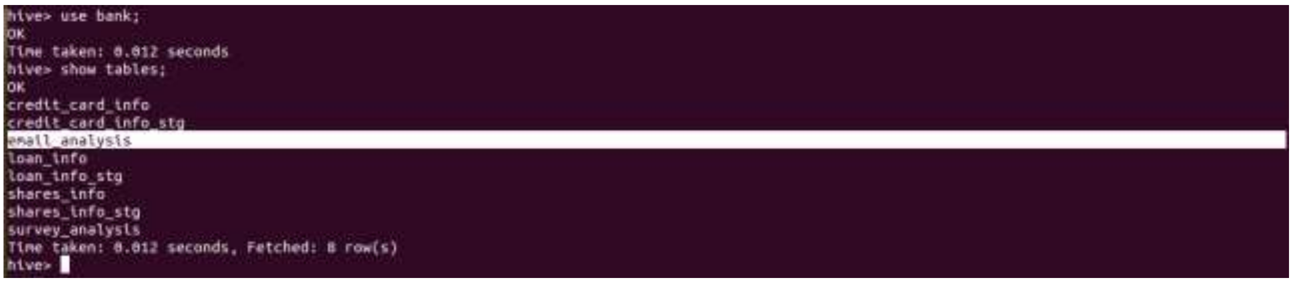
Run in the python shell

```
import xml.etree.ElementTree as ET
import commands
base_str = file("/home/acadgild/email_schema.xml", "r").read().replace("\t","").replace(" ","")
root = ET.fromstring(base_str)

structure_list = []
for each_col in root.findall("column"):
        name = each_col.find("name").text
        type = each_col.find("type").text
        structure_list.append(name + " " + type)

create_table  = "CREATE TABLE email_analysis (" + ",".join(structure_list) + ") ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';"

hive_file = file("/home/acadgild/hive_query.hql", "w")
hive_file.write("CREATE DATABASE IF NOT EXISTS bank;\n")
hive_file.write("USE bank;\n")
hive_file.write(create_table)
hive_file.close()
status, output = commands.getstatusoutput("hive -f " + hive_file.name)
```



```
kiran@Acadgild:~$ python /home/kiran/Documents/CTS/projects/Banking/Bank_Project/7_email_table_creation.txt
kiran@Acadgild:~$
```

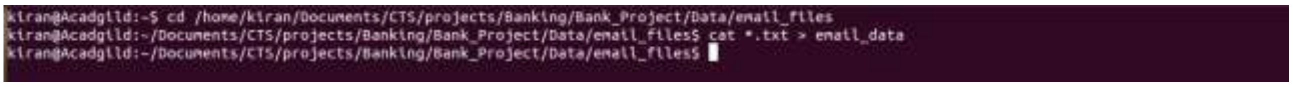
A file with name hive_query.hql and a table will get created in the bank database with name **email_analysis.**



```
hive> use bank;
OK
Time taken: 0.012 seconds
hive> show tables;
OK
credit_card_info
credit_card_info_stg
email_analysis
loan_info
loan_info_stg
shares_info
shares_info_stg
survey_analysis
Time taken: 0.012 seconds, Fetched: 8 row(s)
hive>
```

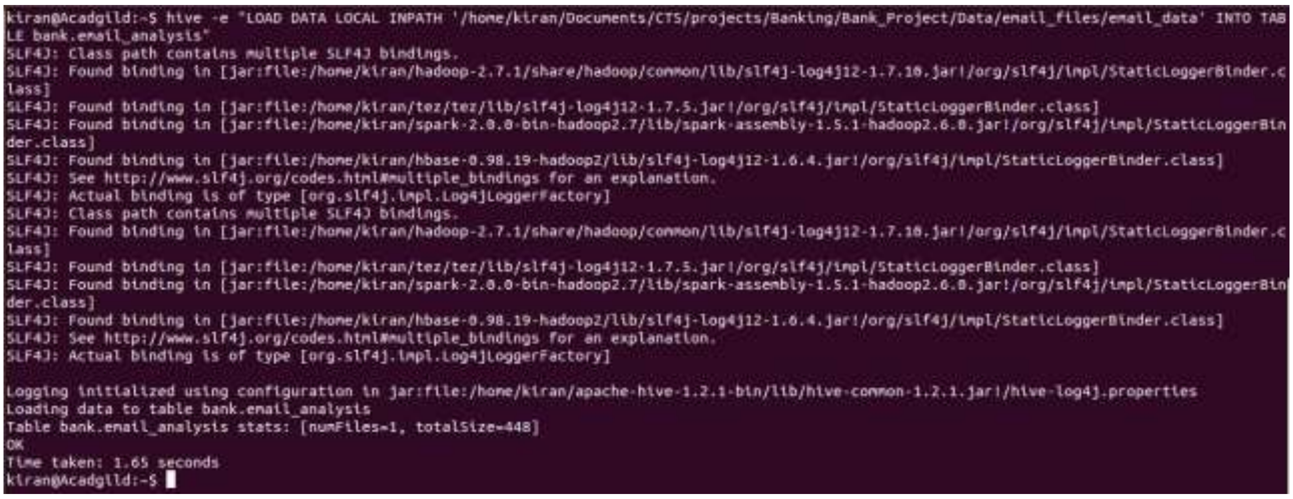
Concatenating the small files

```
cd /home/acadgild/email_files
cat *.txt > email_data
rm *.txt
```



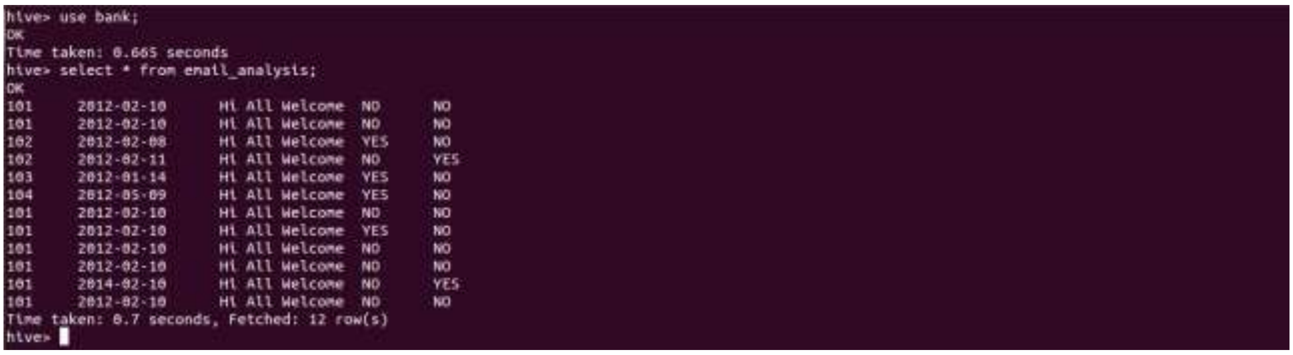
```
kiran@Acadgild:~$ cd /home/kiran/Documents/CTS/projects/Banking/Bank_Project/Data/email_files
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/Data/email_files$ cat *.txt > email_data
kiran@Acadgild:~/Documents/CTS/projects/Banking/Bank_Project/Data/email_files$
```

hive -e "LOAD DATA LOCAL INPATH '/home/acadgild/email_files/email_data' INTO TABLE bank.email_analysis"



```
kiran@Acadgild:~$ hive -e "LOAD DATA LOCAL INPATH '/home/kiran/Documents/CTS/projects/Banking/Bank_Project/Data/email_files/email_data' INTO TAB
LE bank.email_analysis"
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/spark-2.0.0-bin-hadoop2.7/lib/spark-assembly-1.5.1-hadoop2.6.0.jar!/org/slf4j/impl/StaticLoggerBin
der.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-0.98.19-hadoop2/lib/slf4j-log4j12-1.6.4.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]
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/kiran/hadoop-2.7.1/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.c
lass]
SLF4J: Found binding in [jar:file:/home/kiran/tez/tez/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/kiran/spark-2.0.0-bin-hadoop2.7/lib/spark-assembly-1.5.1-hadoop2.6.0.jar!/org/slf4j/impl/StaticLoggerBin
der.class]
SLF4J: Found binding in [jar:file:/home/kiran/hbase-0.98.19-hadoop2/lib/slf4j-log4j12-1.6.4.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]

Logging initialized using configuration in jar:file:/home/kiran/apache-hive-1.2.1-bin/lib/hive-common-1.2.1.jar!/hive-log4j.properties
Loading data to table bank.email_analysis
Table bank.email_analysis stats: [numFiles=1, totalSize=448]
OK
Time taken: 1.65 seconds
kiran@Acadgild:~$
```

Checking the data



```
hive> use bank;
OK
Time taken: 0.665 seconds
hive> select * from email_analysis;
OK
101     2012-02-10      Hi All Welcome  NO      NO
101     2012-02-10      Hi All Welcome  NO      NO
102     2012-02-08      Hi All Welcome  YES     NO
102     2012-02-11      Hi All Welcome  NO      YES
103     2012-01-14      Hi All Welcome  YES     NO
104     2012-05-09      Hi All Welcome  YES     NO
101     2012-02-10      Hi All Welcome  NO      NO
101     2012-02-10      Hi All Welcome  YES     NO
101     2012-02-10      Hi All Welcome  NO      NO
101     2012-02-10      Hi All Welcome  NO      NO
101     2014-02-10      Hi All Welcome  NO      YES
101     2012-02-10      Hi All Welcome  NO      NO
Time taken: 0.7 seconds, Fetched: 12 row(s)
hive>
```

1. Which is the longest running email?

```
SELECT id FROM
(
SELECT id, RANK() OVER (ORDER BY datediff(closed_date, opened_date) DESC) AS rank
FROM
(
SELECT id,
MIN(IF(opened="YES",reporting_date,NULL)) AS opened_date,
MIN(IF(closed="YES",reporting_date,NULL)) AS closed_date
FROM email_analysis
GROUP BY id
) inner_1
WHERE opened_date IS NOT NULL AND closed_date IS NOT NULL
) inner_2
WHERE rank = 1;
```

2. Find out the list of emails which were unanswered.

```
SELECT id
FROM
(
SELECT id,
MIN(IF(opened="YES",reporting_date,NULL)) AS opened_date,
MIN(IF(closed="YES",reporting_date,NULL)) AS closed_date
FROM email_analysis
GROUP BY id
) inne
WHERE opened_date IS NULL AND closed_date IS NOT NULL;
```