

Case Study 2

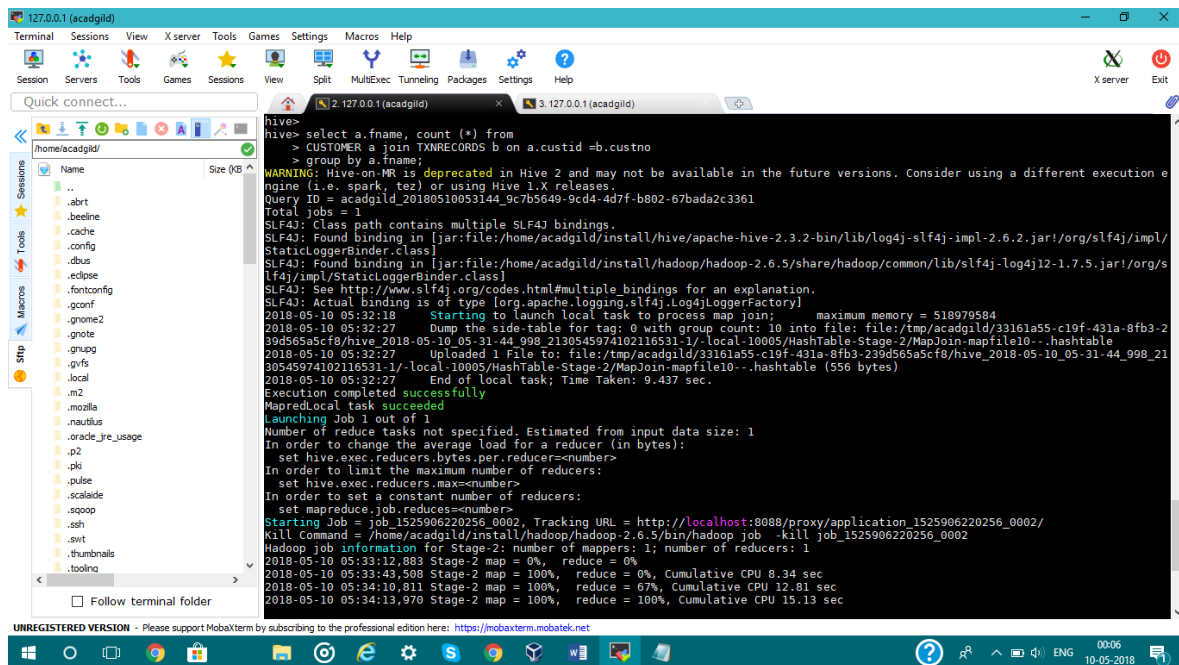
Task1

Find out the number of transaction done by each customer

Command

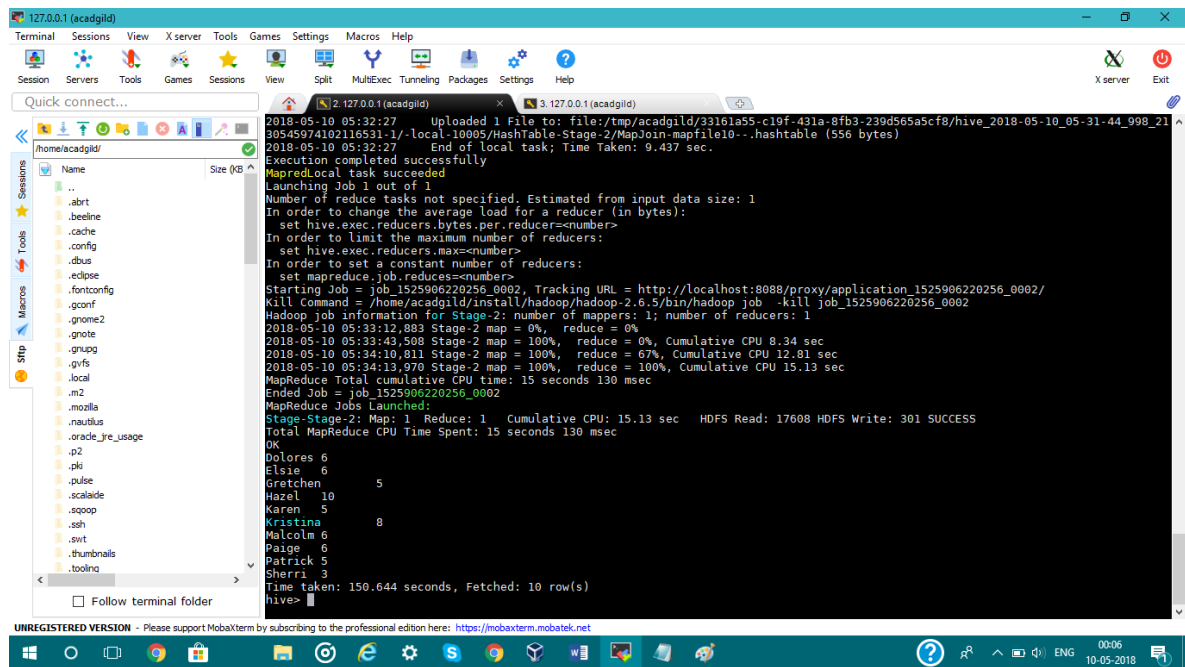
Select a.fname, count(*) from Customers a join Transactions b on a.custid=b.custno group by a.fname

Screenshot



```
hive> select a.fname, count(*) from
> CUSTOMER a join TXNRECORDS b on a.custid=b.custno
> group by a.fname;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180510053144_9c7b5649-9cd4-4d7f-b802-67bada2c3361
Total jobs = 1
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/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.apache.logging.slf4j.Log4jLoggerFactory]
2018-05-10 05:32:18 Starting to launch local task to process map join; maximum memory = 518979584
2018-05-10 05:32:27 Dump the side-table for tag: 0 with group count: 10 into file: file:/tmp/acadgild/33161a55-c19f-431a-8fb3-239d565a5cf8/hive_2018-05-10_05-31-44_998_2130545974102116531-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile10-.hashtable
2018-05-10 05:32:27 Uploaded 1 File to: file:/tmp/acadgild/33161a55-c19f-431a-8fb3-239d565a5cf8/hive_2018-05-10_05-31-44_998_2130545974102116531-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile10-.hashtable (556 bytes)
2018-05-10 05:32:27 End of local task; Time Taken: 9.437 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 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_1525906220256_0002, Tracking URL = http://localhost:8088/proxy/application_1525906220256_0002/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job -kill job_1525906220256_0002
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-05-10 05:33:12,883 Stage-2 map = 0%, reduce = 0%
2018-05-10 05:33:43,508 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 8.34 sec
2018-05-10 05:34:10,811 Stage-2 map = 100%, reduce = 67%, Cumulative CPU 12.61 sec
2018-05-10 05:34:13,970 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 15.13 sec
```

Output



Task2

Create a new table called TRANSACTIONS_COUNT. This table should have

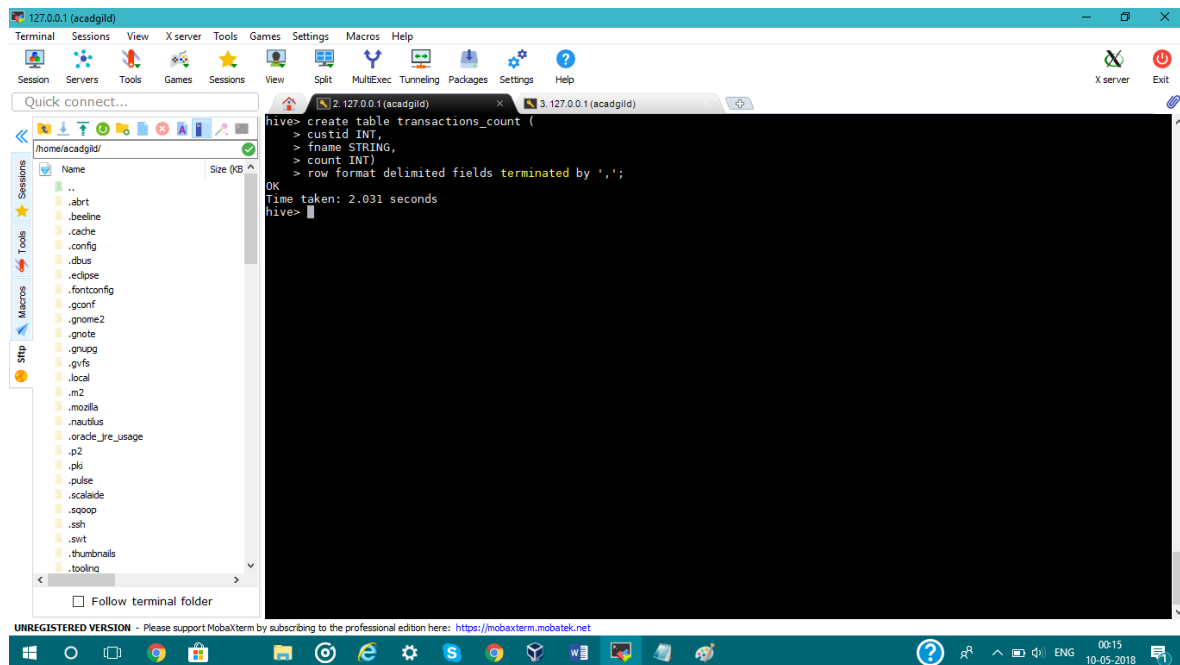
3 fields - custid, fname and count.

Command

create table transactions_count (custid INT, fname STRING, count INT)

row format delimited fields terminated by ',';

Screenshot



Task3

Now write a hive query in such a way that the query populates the data obtained in Step 1 above and populate the table in step 2 above.

Command

Insert overwrite table transactions_count Select a.custid, a.fname, count(*) from Customers a join Transactions b on a.custid=b.custno group by a.custid, a.fname

Screenshot

The screenshot shows a MobaXterm window with a terminal session. The terminal displays the execution of a Hive SQL query. The query is: `hive> INSERT OVERWRITE TABLE transactions_count select a.custid,a.fname, count (*) from CUSTOMER a join TXNRECORDS b on a.custid=b.custno group by a.custid,a.fname;`. The output shows a warning about Hive-on-MR being deprecated, followed by SLF4J bindings, a successful execution of a local task, and the launch of a MapReduce job. The job information for Stage-2 shows 1 mapper and 1 reducer. The final output of the query is a table with 10 rows of customer data.

```
hive> INSERT OVERWRITE TABLE transactions_count
> select a.custid,a.fname, count (*) from
> CUSTOMER a join TXNRECORDS b on a.custid=b.custno
> group by a.custid,a.fname;

WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180510055818_bd83f183-5b2c-464b-9fb4-e2bd0baa2966
Total jobs = 1
2018-05-10 05:58:52 Starting to launch local task to process map join; maximum memory = 518979584
2018-05-10 05:59:01 Dump the side-table for tag: 0 with group count: 10 into file: file:/tmp/acadgild/33161a55-c19f-431a-8fb3-23d565a5c8f8/hive_2018-05-10_05-58-18_132_3256519282249978397-1/-local-10003/HashTable-Stage-2/MapJoin-mapfile20-..hashtable
2018-05-10 05:59:02 Uploaded 1 file to: file:/tmp/acadgild/33161a55-c19f-431a-8fb3-23d565a5c8f8/hive_2018-05-10_05-58-18_132_3256519282249978397-1/-local-10003/HashTable-Stage-2/MapJoin-mapfile20-..hashtable (556 bytes)
2018-05-10 05:59:02 End of local task; Time Taken: 9.922 sec.
Execution completed successfully
MapReduceLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 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_1525906220256_0003, Tracking URL = http://localhost:8088/proxy/application_1525906220256_0003/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job -kill job_1525906220256_0003
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-05-10 05:59:44,317 Stage-2 map = 0%, reduce = 0%
2018-05-10 06:00:25,762 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 9.67 sec
2018-05-10 06:00:56,543 Stage-2 map = 100%, reduce = 67%, Cumulative CPU 13.77 sec
2018-05-10 06:01:08,959 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 21.48 sec
MapReduce Total cumulative CPU time: 21 seconds 480 msec
Ended Job = job_1525906220256_0003
Loading data to table simpledb.transactions_count
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 21.48 sec HDFS Read: 18731 HDFS Write: 258 SUCCESS
Total MapReduce CPU Time Spent: 21 seconds 480 msec
Time taken: 179.537 seconds
hive> select * from transactions_count;
OK
4000001 Kristina 8
4000002 Paige 6
4000003 Sherri 3
4000004 Gretchen 5
4000005 Karen 5
4000006 Patrick 5
4000007 Elsie 6
4000008 Hazel 10
4000009 Malcolm 6
4000010 Dolores 6
Time taken: 1.698 seconds, Fetched: 10 row(s)
hive>
```

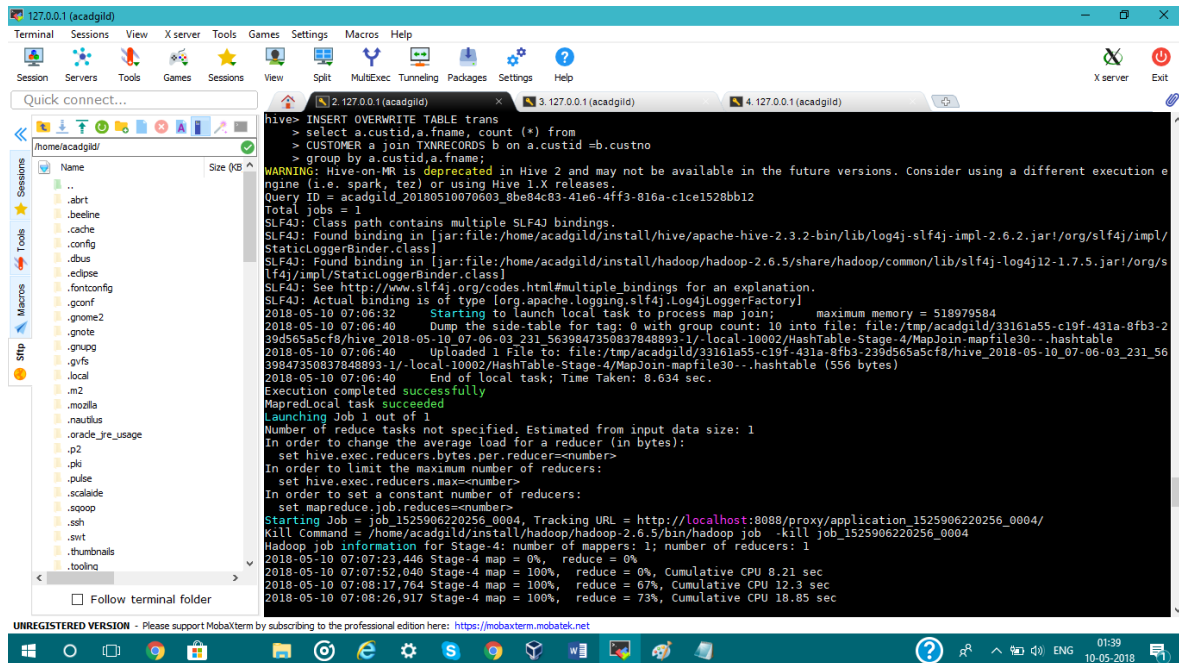
Output

The screenshot shows a MobaXterm window with a terminal session. The terminal displays the output of a Hive SQL query. The query is: `hive> select * from transactions_count;`. The output shows a table with 10 rows of customer data, including customer ID, name, and a count. The final output of the query is a table with 10 rows of customer data.

```
hive> select * from transactions_count;
OK
4000001 Kristina 8
4000002 Paige 6
4000003 Sherri 3
4000004 Gretchen 5
4000005 Karen 5
4000006 Patrick 5
4000007 Elsie 6
4000008 Hazel 10
4000009 Malcolm 6
4000010 Dolores 6
Time taken: 1.698 seconds, Fetched: 10 row(s)
hive>
```


Insert overwrite table trans Select a.custid, a.fname, count(*) from Customers a join Transactions b on a.custid=b.custno group by a.custid, a.fname

Screenshot

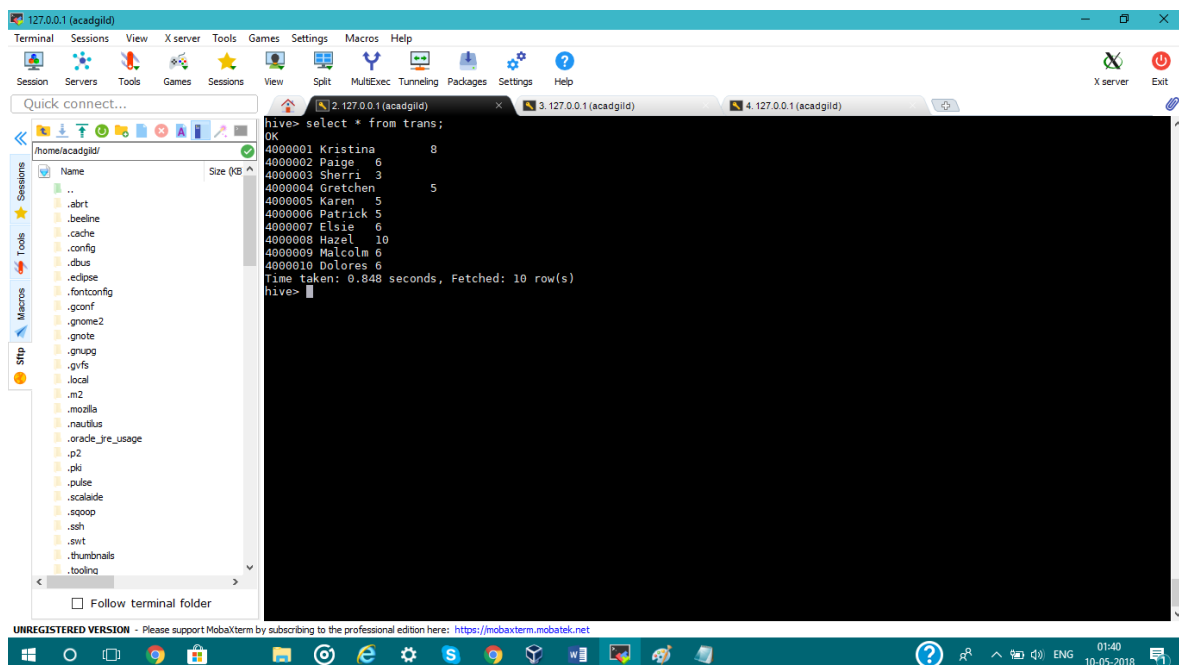


The screenshot shows a terminal window with the following content:

```
hive> INSERT OVERWRITE TABLE trans
> select a.custid,a.fname, count(*) from
> CUSTOMER a join TXNRECORDS b on a.custid=b.custno
> group by a.custid,a.fname;

WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180510070603_8be84c83-41e8-4ff3-816a-c1ce1528bb12
Total jobs = 1
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/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.apache.logging.slf4j.Log4jLoggerFactory]
2018-05-10 07:06:32 Starting to launch local task to process map join; maximum memory = 518979584
2018-05-10 07:06:40 Dump the side-table for tag: 0 with group count: 10 into file: file:/tmp/acadgild/33161a55-c19f-431a-8fb3-239d565a5cf8/hive_2018-05-10_07-06-03_231_5639847350837848893-1/-local-10002/HashTable-Stage-4/MapJoin-mapfile30-..hashtable
2018-05-10 07:06:40 Uploaded 1 File to: file:/tmp/acadgild/33161a55-c19f-431a-8fb3-239d565a5cf8/hive_2018-05-10_07-06-03_231_5639847350837848893-1/-local-10002/HashTable-Stage-4/MapJoin-mapfile30-..hashtable (556 bytes)
2018-05-10 07:06:40 End of local task; Time Taken: 8.634 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 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_1525906220256_0004, Tracking URL = http://localhost:8088/proxy/application/1525906220256_0004/
Kill command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job -kill job_1525906220256_0004
Hadoop job information for Stage-4: number of mappers: 1; number of reducers: 1
2018-05-10 07:07:23,446 Stage-4 map = 0%, reduce = 0%
2018-05-10 07:07:52,940 Stage-4 map = 100%, reduce = 0%, Cumulative CPU 8.21 sec
2018-05-10 07:08:17,764 Stage-4 map = 100%, reduce = 67%, Cumulative CPU 12.3 sec
2018-05-10 07:08:26,917 Stage-4 map = 100%, reduce = 73%, Cumulative CPU 18.85 sec
```

Output



The screenshot shows a terminal window with the following content:

```
hive> select * from trans;
4000001 Kristina      8
4000002 Paige        6
4000003 Sherri       3
4000004 Gretchen     5
4000005 Karen        5
4000006 Patrick      5
4000007 Elsie        6
4000008 Hazel       10
4000009 Malcolm      6
4000010 Dolores      6
Time taken: 0.848 seconds, Fetched: 10 row(s)
hive>
```

Task6

Now from the Hbase level, write the Hbase java API code to access and scan the TRANSACTIONS table data from java level

Code

//importing all the packages

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.HColumnDescriptor;
import org.apache.hadoop.hbase.HTableDescriptor;
import org.apache.hadoop.hbase.KeyValue;
import org.apache.hadoop.hbase.MasterNotRunningException;
import org.apache.hadoop.hbase.ZooKeeperConnectionException;
import org.apache.hadoop.hbase.client.HBaseAdmin;
import org.apache.hadoop.hbase.client.HTable;
import org.apache.hadoop.hbase.client.Result;
import org.apache.hadoop.hbase.client.ResultScanner;
import org.apache.hadoop.hbase.client.Scan;
import org.apache.hadoop.hbase.util.Bytes;
```

```
public class HbaseTable {                                //defining a class
    private static Configuration conf = null; //declaring configurations
```

```

static {
    conf = HBaseConfiguration.create();//initialising configuration
}

//scanning a HBase table

public static void getAllRecord (String tableName) {
    try{
        HTable table = new HTable(conf, tableName);
        Scan s = new Scan();
        ResultScanner ss = table.getScanner(s);
        for(Result r:ss){
            for(KeyValue kv : r.raw()){
                System.out.print(new String(kv.getRow()) + " ");    //for
fetching rowkey
                System.out.print(new String(kv.getFamily()) + ":");    //for
fetching column family
                System.out.print(new String(kv.getQualifier()) + " ");    //for
fetching columns
                System.out.print(kv.getTimestamp() + " ");
                System.out.println(new String(kv.getValue()));
            }
        }
    } catch (IOException e){
        e.printStackTrace();
    }
}

public static void main(String[] args) {

```



```

try {
    String tablename = "transactions";           //initialising the name of
the table in HBase

    String[] families = { "custid", "fname","count" };//initialising the
column families

    System.out.println("=====show all record=====");
    HbaseTable.getAllRecord(tablename);
} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

Screenshot

```

<terminated> HbaseTable (1) [Java Application] /usr/java/jdk1.8.0_151/bin/java (May 26, 2018, 7:29:53 PM)
=====show all record=====
log4j:WARN No appenders could be found for logger (org.apache.hadoop.security.Groups).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
4000001 txndetails:count 1525916309726 8
4000001 txndetails:fname 1525916309726 Kristina
4000002 txndetails:count 1525916309726 6
4000002 txndetails:fname 1525916309726 Paige
4000003 txndetails:count 1525916309726 3
4000003 txndetails:fname 1525916309726 Sherri
4000004 txndetails:count 1525916309726 5
4000004 txndetails:fname 1525916309726 Gretchen
4000005 txndetails:count 1525916309726 5
4000005 txndetails:fname 1525916309726 Karen
4000006 txndetails:count 1525916309726 5
4000006 txndetails:fname 1525916309726 Patrick
4000007 txndetails:count 1525916309726 6
4000007 txndetails:fname 1525916309726 Elsie
4000008 txndetails:count 1525916309726 10
4000008 txndetails:fname 1525916309726 Hazel
4000009 txndetails:count 1525916309726 6
4000009 txndetails:fname 1525916309726 Malcolm
4000010 txndetails:count 1525916309726 6
4000010 txndetails:fname 1525916309726 Dolores

```