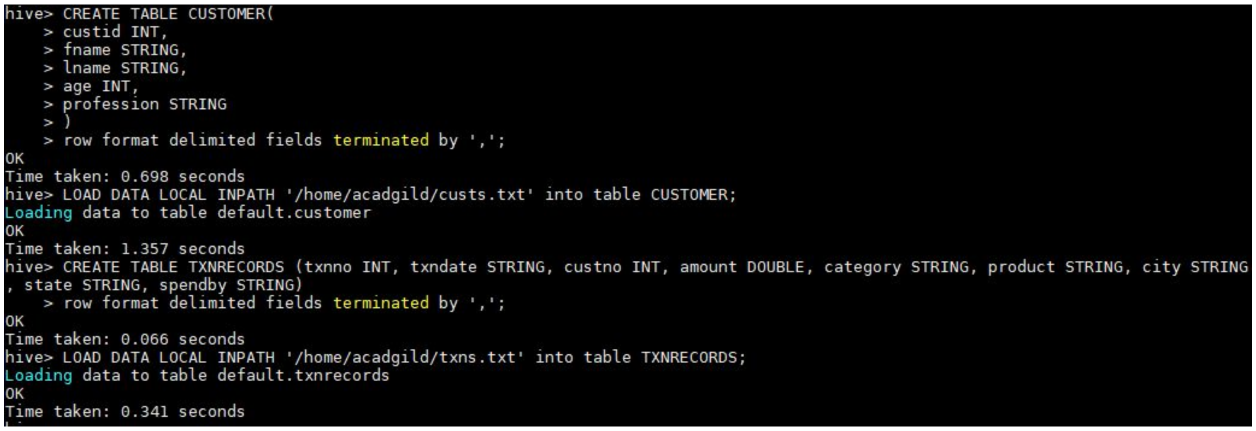
Case Study 2:

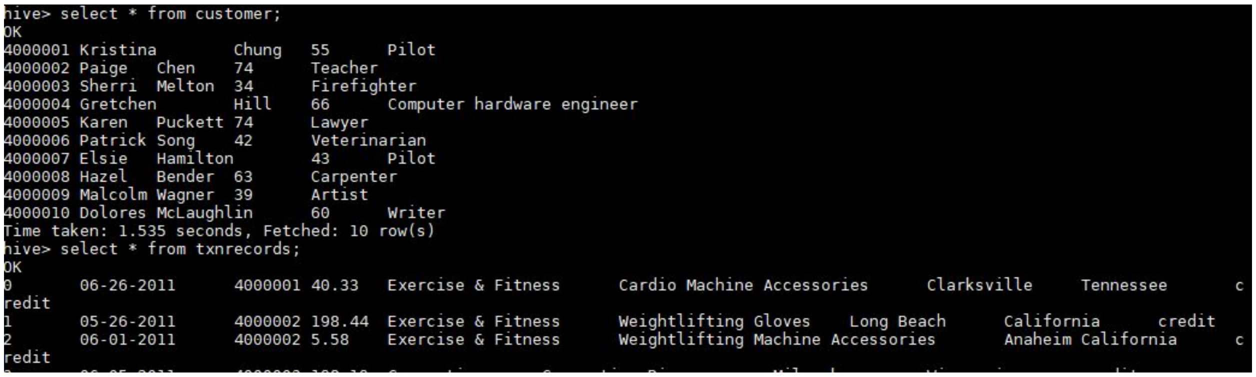
Case Study Description:

Let us take up the CUSTOMER and TRANSACTIONS table we have created in the Lets Do Together section. Let us solve the following use cases using these tables :-

As part of the case study, Creating 2 tables namely customer and transaction using hive and loading the data from a txt file. After doing datanode format all the data is cleared out.



```
hive> CREATE TABLE CUSTOMER(
    > custid INT,
    > fname STRING,
    > lname STRING,
    > age INT,
    > profession STRING
    > )
    > row format delimited fields terminated by ',';
OK
Time taken: 0.698 seconds
hive> LOAD DATA LOCAL INPATH '/home/acadgild/custs.txt' into table CUSTOMER;
Loading data to table default.customer
OK
Time taken: 1.357 seconds
hive> CREATE TABLE TXNRECORDS (txnno INT, txndate STRING, custno INT, amount DOUBLE, category STRING, product STRING, city STRING
, state STRING, spendby STRING)
    > row format delimited fields terminated by ',';
OK
Time taken: 0.066 seconds
hive> LOAD DATA LOCAL INPATH '/home/acadgild/txns.txt' into table TXNRECORDS;
Loading data to table default.txnrecords
OK
Time taken: 0.341 seconds
```



```
hive> select * from customer;
OK
4000001 Kristina        Chung   55      Pilot
4000002 Paige   Chen    74      Teacher
4000003 Sherri  Melton  34      Firefighter
4000004 Gretchen        Hill    66      Computer hardware engineer
4000005 Karen   Puckett 74      Lawyer
4000006 Patrick Song    42      Veterinarian
4000007 Elsie   Hamilton        43      Pilot
4000008 Hazel   Bender  63      Carpenter
4000009 Malcolm Wagner  39      Artist
4000010 Dolores McLaughlin      60      Writer
Time taken: 1.535 seconds, Fetched: 10 row(s)
hive> select * from txnrecords;
OK
0       06-26-2011      4000001 40.33   Exercise & Fitness      Cardio Machine Accessories      Clarksville     Tennessee       c
redit
1       05-26-2011      4000002 198.44  Exercise & Fitness      Weightlifting Gloves    Long Beach      California      credit
2       06-01-2011      4000002 5.58    Exercise & Fitness      Weightlifting Machine Accessories       Anaheim California      c
redit
```

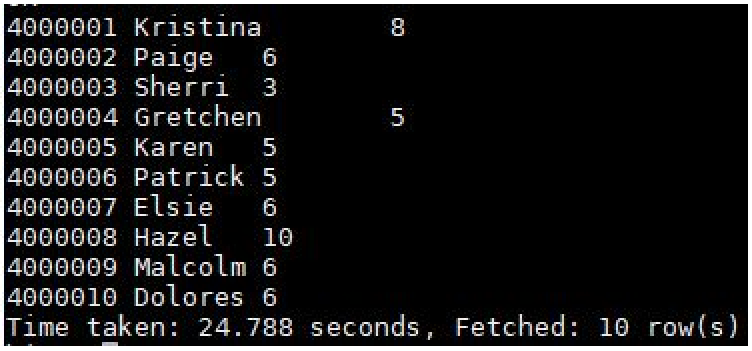
1. Find out the number of transaction done by each customer (These should be take up in module 8 itself)

Query:

----

select a.custid, a.fname, count(b.custno) from CUSTOMER a join TXNRECORDS b on a.custid = b.custno group by a.custid, a.fname;

Output:

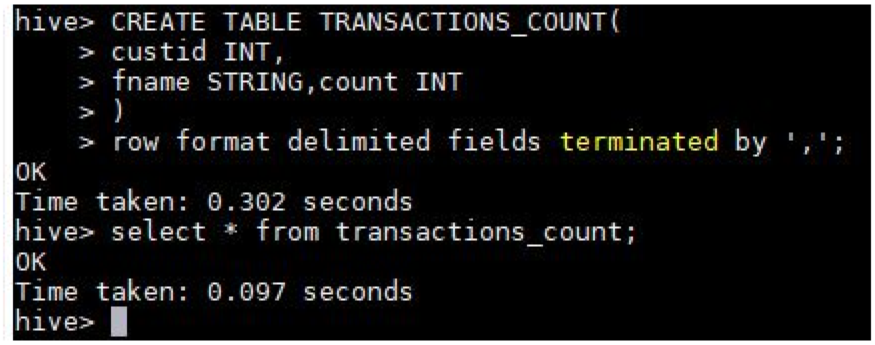


```
4000001 Kristina        8
4000002 Paige   6
4000003 Sherri  3
4000004 Gretchen        5
4000005 Karen   5
4000006 Patrick 5
4000007 Elsie   6
4000008 Hazel   10
4000009 Malcolm 6
4000010 Dolores 6
Time taken: 24.788 seconds, Fetched: 10 row(s)
```

2. Create a new table called TRANSACTIONS_COUNT. This table should have 3 fields - custid, fname and count. (Again to be done in module 8)

CREATE TABLE TRANSACTIONS_COUNT(custid INT, fname STRING,count INT) row format delimited fields terminated by ',';

Output:



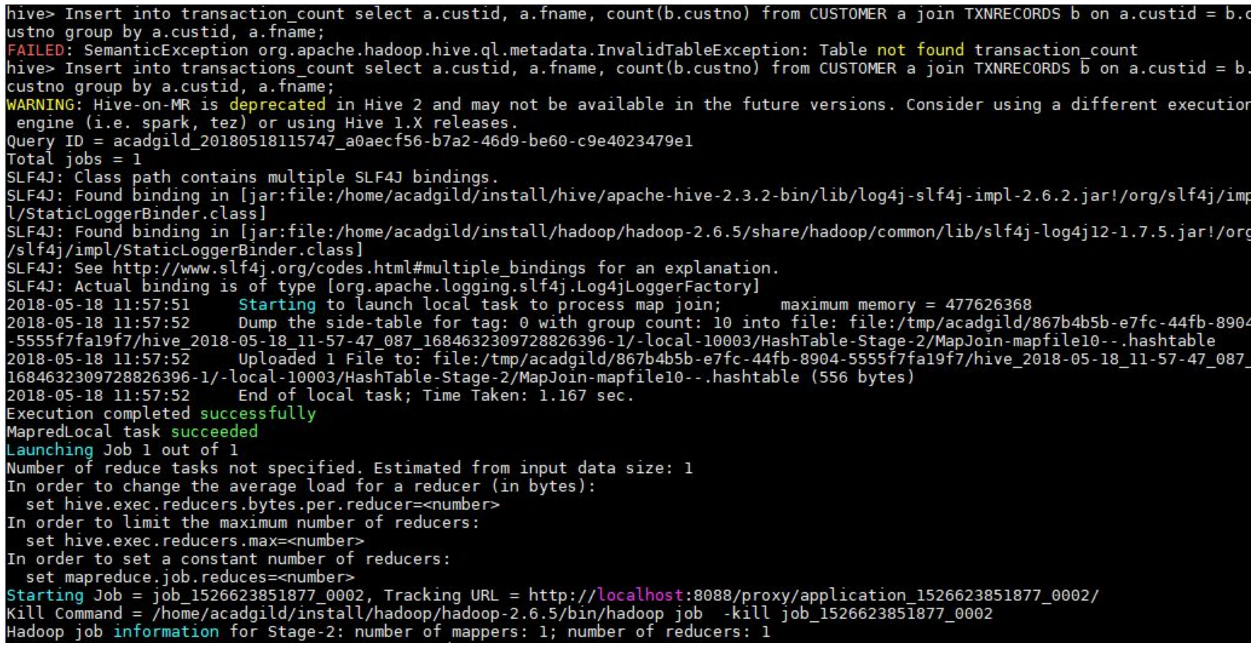
```
hive> CREATE TABLE TRANSACTIONS_COUNT(
    > custid INT,
    > fname STRING,count INT
    > )
    > row format delimited fields terminated by ',';
OK
Time taken: 0.302 seconds
hive> select * from transactions_count;
OK
Time taken: 0.097 seconds
hive>
```

3. 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. (This has to be done in module 9).

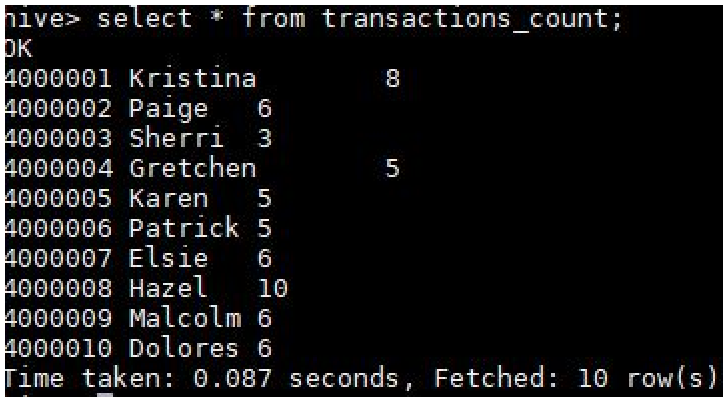
Query:

Insert into transaction_count select a.custid, a.fname, count(b.custno) from CUSTOMER a join TXNRECORDS b on a.custid = b.custno group by a.custid, a.fname;

Output:



```
hive> Insert into transaction_count select a.custid, a.fname, count(b.custno) from CUSTOMER a join TXNRECORDS b on a.custid = b.c
ustno group by a.custid, a.fname;
FAILED: SemanticException org.apache.hadoop.hive.ql.metadata.InvalidTableException: Table not found transaction_count
hive> Insert into transactions_count select a.custid, a.fname, count(b.custno) 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 executio
 engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180518115747_a0aecf56-b7a2-46d9-be60-c9e4023479e1
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/imp
l/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]
2018-05-18 11:57:51     Starting to launch local task to process map join;      maximum memory = 477626368
2018-05-18 11:57:52     Dump the side-table for tag: 0 with group count: 10 into file: file:/tmp/acadgild/867b4b5b-e7fc-44fb-8904
-5555f7fa19f7/hive_2018-05-18_11-57-47_087_1684632309728826396-1/-local-10003/HashTable-Stage-2/MapJoin-mapfile10--.hashtable
2018-05-18 11:57:52     Uploaded 1 File to: file:/tmp/acadgild/867b4b5b-e7fc-44fb-8904-5555f7fa19f7/hive_2018-05-18_11-57-47_087_
1684632309728826396-1/-local-10003/HashTable-Stage-2/MapJoin-mapfile10--.hashtable (556 bytes)
2018-05-18 11:57:52     End of local task; Time Taken: 1.167 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_1526623851877_0002, Tracking URL = http://localhost:8088/proxy/application_1526623851877_0002/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1526623851877_0002
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
```



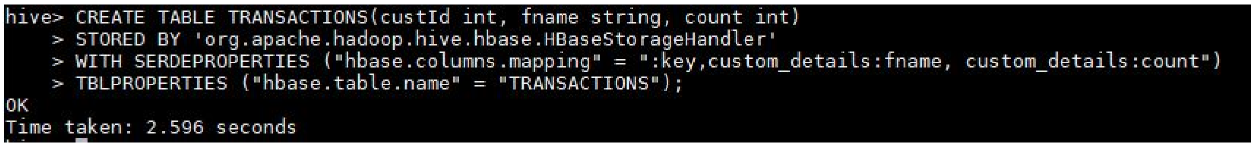
```
hive> select * from transactions_count;
OK
4000001 Kristina        8
4000002 Paige   6
4000003 Sherri  3
4000004 Gretchen        5
4000005 Karen   5
4000006 Patrick 5
4000007 Elsie   6
4000008 Hazel   10
4000009 Malcolm 6
4000010 Dolores 6
Time taken: 0.087 seconds, Fetched: 10 row(s)
```

4. Now let's make the TRANSACTIONS_COUNT table Hbase complaint. In the sense, use Ser Des And Storate handler features of hive to change the TRANSACTIONS_COUNT table to be able to create a TRANSACTIONS table in Hbase. (This has to be done in module 10)

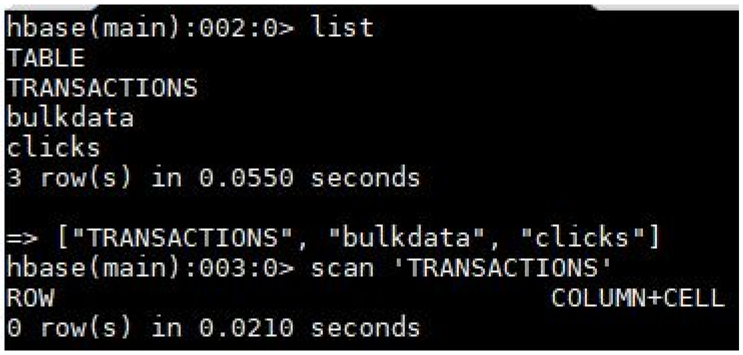
Query:

CREATE TABLE TRANSACTIONS(custId int, fname string, count int)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" =
":key,custom_details:fname, custom_details:count")
TBLPROPERTIES ("hbase.table.name" = "TRANSACTIONS");

Output:

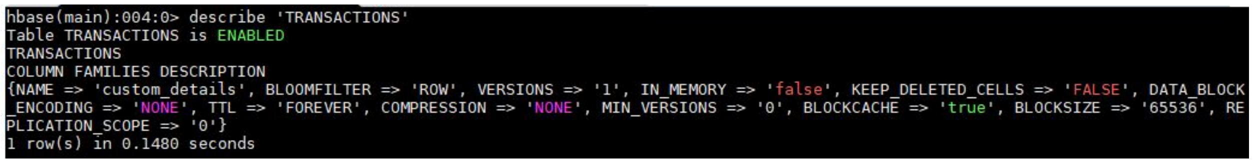


```
hive> CREATE TABLE TRANSACTIONS(custId int, fname string, count int)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,custom_details:fname, custom_details:count")
    > TBLPROPERTIES ("hbase.table.name" = "TRANSACTIONS");
OK
Time taken: 2.596 seconds
```



```
hbase(main):002:0> list
TABLE
TRANSACTIONS
bulkdata
clicks
3 row(s) in 0.0550 seconds

=> ["TRANSACTIONS", "bulkdata", "clicks"]
hbase(main):003:0> scan 'TRANSACTIONS'
ROW                           COLUMN+CELL
0 row(s) in 0.0210 seconds
```



```
hbase(main):004:0> describe 'TRANSACTIONS'
Table TRANSACTIONS is ENABLED
TRANSACTIONS
COLUMN FAMILIES DESCRIPTION
{NAME => 'custom_details', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK
_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', RE
PLICATION_SCOPE => '0'}
1 row(s) in 0.1480 seconds
```

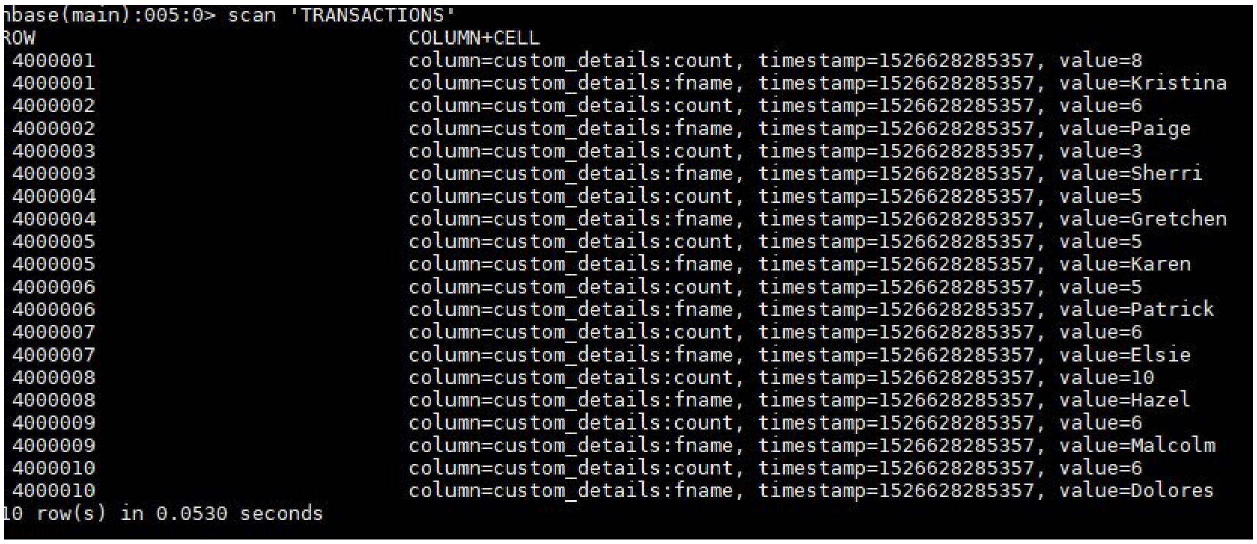
5. Now insert the data in TRANSACTIONS_COUNT table using the query in step 3 again, this should populate the Hbase

TRANSACTIONS table automatically (This has to be done in module 10)

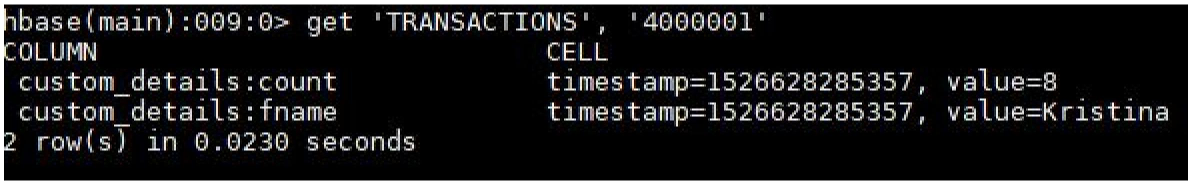
Query:

INSERT OVERWRITE table Transactions **select a.custid, a.fname, count(b.custno) from CUSTOMER a join TXNRECORDS b on a.custid = b.custno group by a.custid, a.fname;**

Output:



```
hbase(main):005:0> scan 'TRANSACTIONS'
ROW                              COLUMN+CELL
 4000001                         column=custom_details:count, timestamp=1526628285357, value=8
 4000001                         column=custom_details:fname, timestamp=1526628285357, value=Kristina
 4000002                         column=custom_details:count, timestamp=1526628285357, value=6
 4000002                         column=custom_details:fname, timestamp=1526628285357, value=Paige
 4000003                         column=custom_details:count, timestamp=1526628285357, value=3
 4000003                         column=custom_details:fname, timestamp=1526628285357, value=Sherri
 4000004                         column=custom_details:count, timestamp=1526628285357, value=5
 4000004                         column=custom_details:fname, timestamp=1526628285357, value=Gretchen
 4000005                         column=custom_details:count, timestamp=1526628285357, value=5
 4000005                         column=custom_details:fname, timestamp=1526628285357, value=Karen
 4000006                         column=custom_details:count, timestamp=1526628285357, value=5
 4000006                         column=custom_details:fname, timestamp=1526628285357, value=Patrick
 4000007                         column=custom_details:count, timestamp=1526628285357, value=6
 4000007                         column=custom_details:fname, timestamp=1526628285357, value=Elsie
 4000008                         column=custom_details:count, timestamp=1526628285357, value=10
 4000008                         column=custom_details:fname, timestamp=1526628285357, value=Hazel
 4000009                         column=custom_details:count, timestamp=1526628285357, value=6
 4000009                         column=custom_details:fname, timestamp=1526628285357, value=Malcolm
 4000010                         column=custom_details:count, timestamp=1526628285357, value=6
 4000010                         column=custom_details:fname, timestamp=1526628285357, value=Dolores
10 row(s) in 0.0530 seconds
```



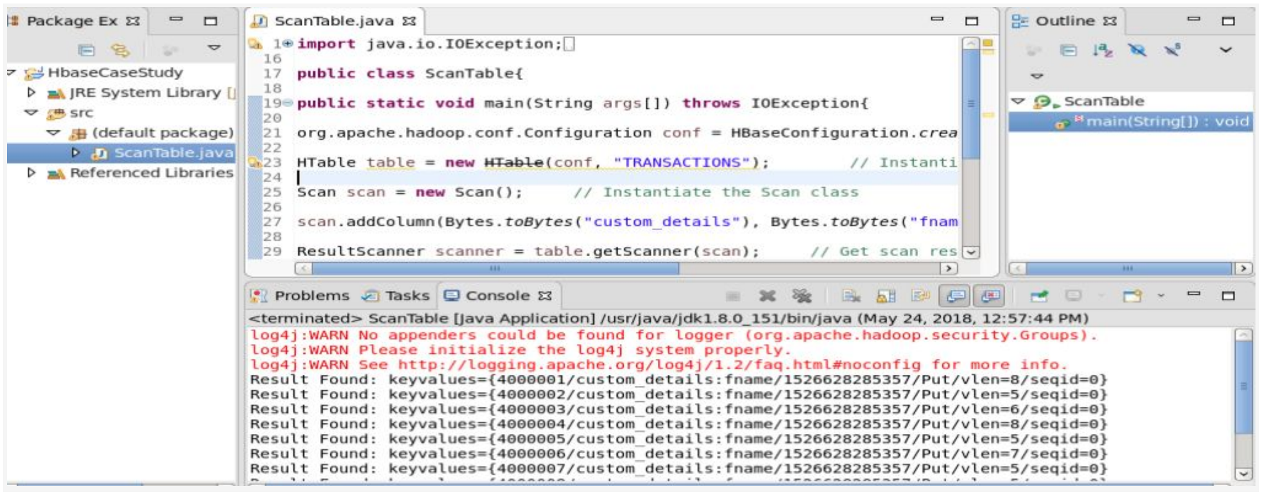
```
hbase(main):009:0> get 'TRANSACTIONS', '4000001'
COLUMN                           CELL
 custom_details:count            timestamp=1526628285357, value=8
 custom_details:fname            timestamp=1526628285357, value=Kristina
2 row(s) in 0.0230 seconds
```

6. **Now from the Hbase level, write the Hbase java API code to access and scan the TRANSACTIONS table data from java level (This should** be done in module 11)

Steps:

Write a java API code and run as Java application in eclipse.
Executed in VM Machine.
The source code is uploaded as separate file.

Output:



```
import java.io.IOException;

public class ScanTable{

public static void main(String args[]) throws IOException{

org.apache.hadoop.conf.Configuration conf = HBaseConfiguration.crea

HTable table = new ~~HTable~~(conf, "TRANSACTIONS");          // Instanti

Scan scan = new Scan();      // Instantiate the Scan class

scan.addColumn(Bytes.toBytes("custom_details"), Bytes.toBytes("fnam

ResultScanner scanner = table.getScanner(scan);      // Get scan res
```

```
<terminated> ScanTable [Java Application] /usr/java/jdk1.8.0_151/bin/java (May 24, 2018, 12:57:44 PM)
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.
Result Found: keyvalues={4000001/custom_details:fname/1526628285357/Put/vlen=8/seqid=0}
Result Found: keyvalues={4000002/custom_details:fname/1526628285357/Put/vlen=5/seqid=0}
Result Found: keyvalues={4000003/custom_details:fname/1526628285357/Put/vlen=6/seqid=0}
Result Found: keyvalues={4000004/custom_details:fname/1526628285357/Put/vlen=8/seqid=0}
Result Found: keyvalues={4000005/custom_details:fname/1526628285357/Put/vlen=5/seqid=0}
Result Found: keyvalues={4000006/custom_details:fname/1526628285357/Put/vlen=7/seqid=0}
Result Found: keyvalues={4000007/custom_details:fname/1526628285357/Put/vlen=5/seqid=0}
```