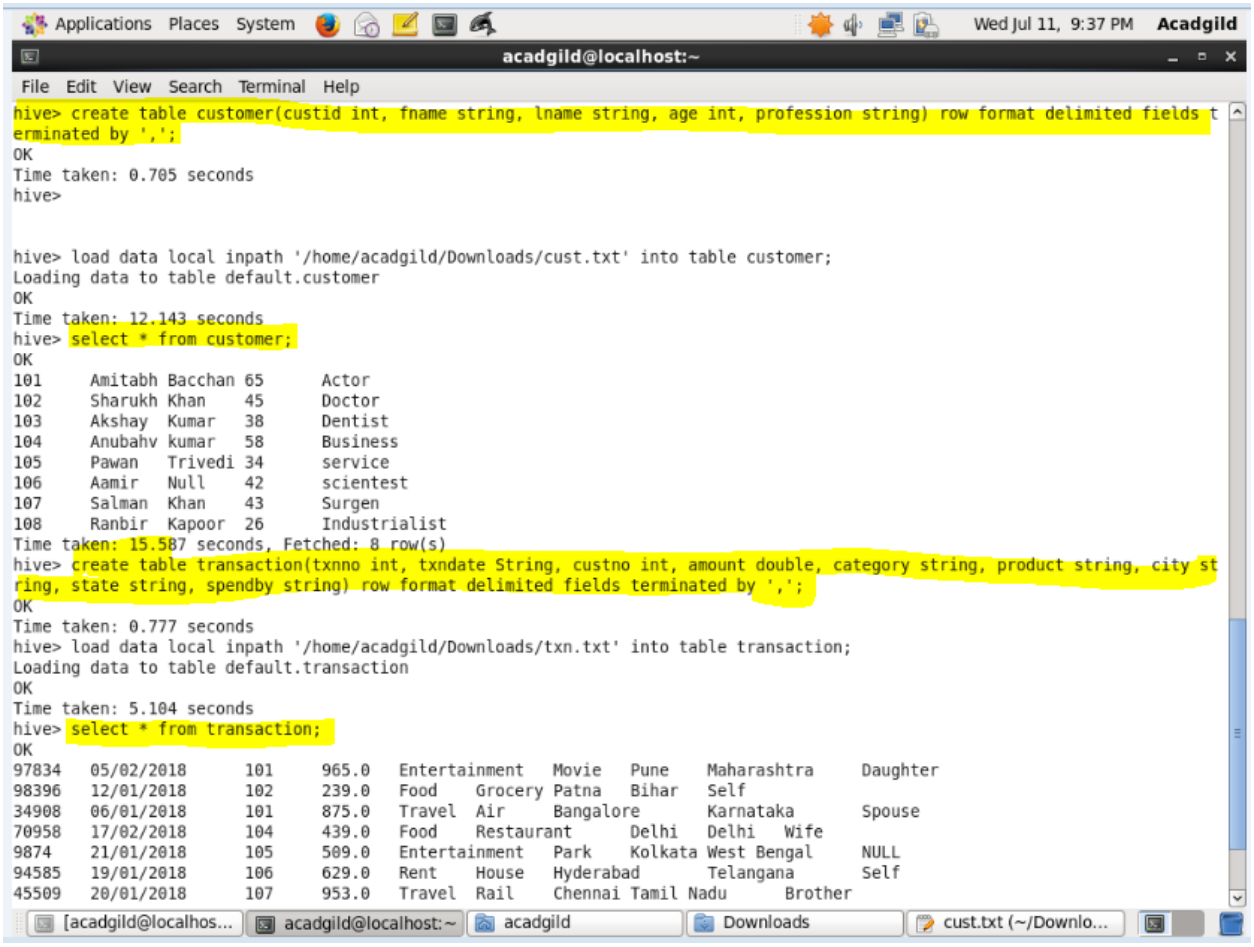
CASE STUDY 2

Case Study Description

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



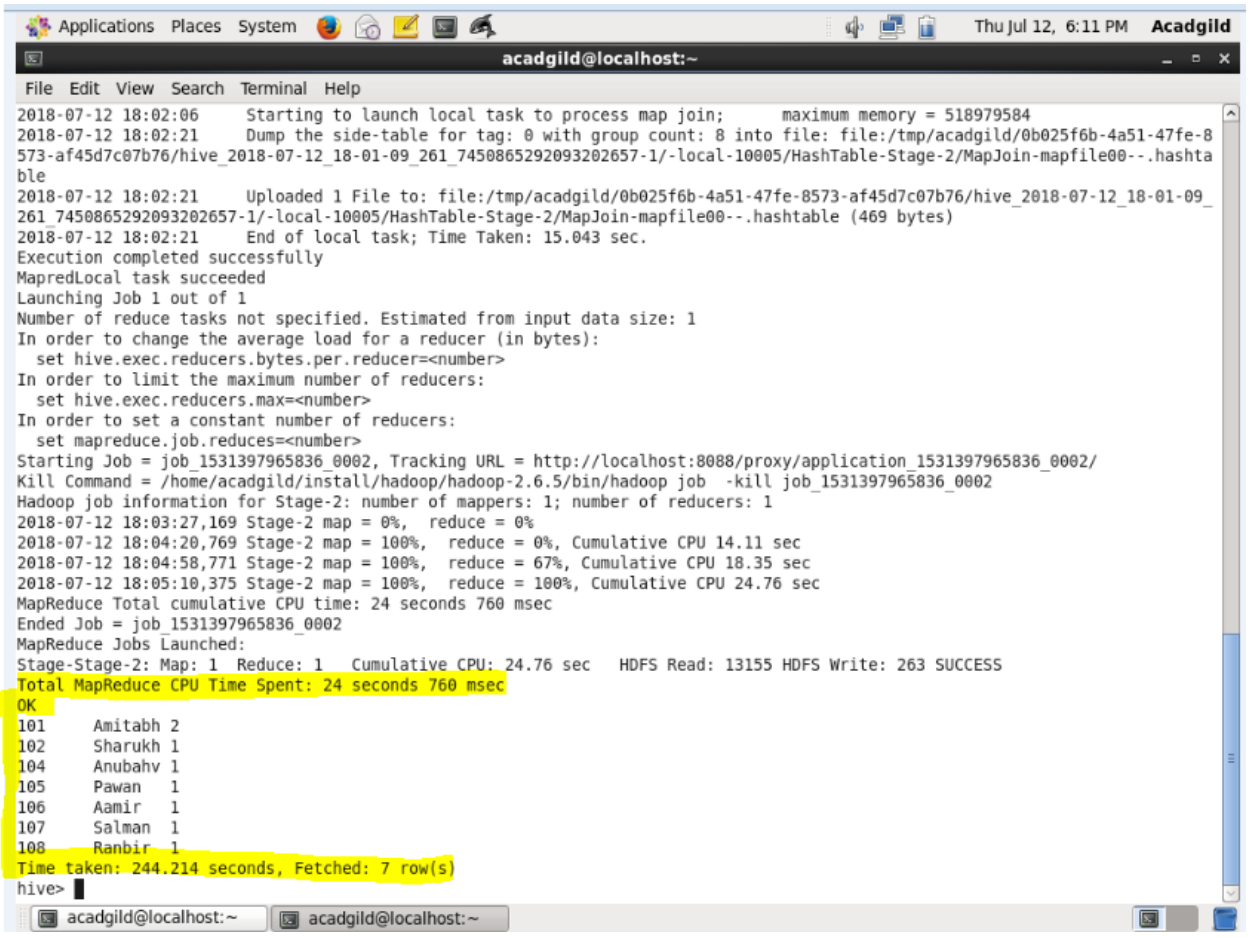
CASE STUDY 2

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

```
hive> select count(*),custno from transaction group by custno;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execu
tion engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180712175508_85307341-225b-487e-9c1f-9be6e76eb1e7
Total jobs = 1
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_1531397965836_0001, Tracking URL = http://localhost:8088/proxy/application_1531397965836_0001/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1531397965836_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-07-12 17:57:23,974 Stage-1 map = 0%,  reduce = 0%
2018-07-12 17:58:11,260 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 7.25 sec
2018-07-12 17:58:52,025 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 10.42 sec
2018-07-12 17:59:05,883 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 18.93 sec
MapReduce Total cumulative CPU time: 18 seconds 930 msec
Ended Job = job_1531397965836_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 18.93 sec   HDFS Read: 9856 HDFS Write: 213 SUCCESS
Total MapReduce CPU Time Spent: 18 seconds 930 msec
OK
2       101
1       102
1       104
1       105
1       106
1       107
1       108
Time taken: 241.925 seconds, Fetched: 7 row(s)
acadgild@localhost:~   acadgild@localhost:~
```

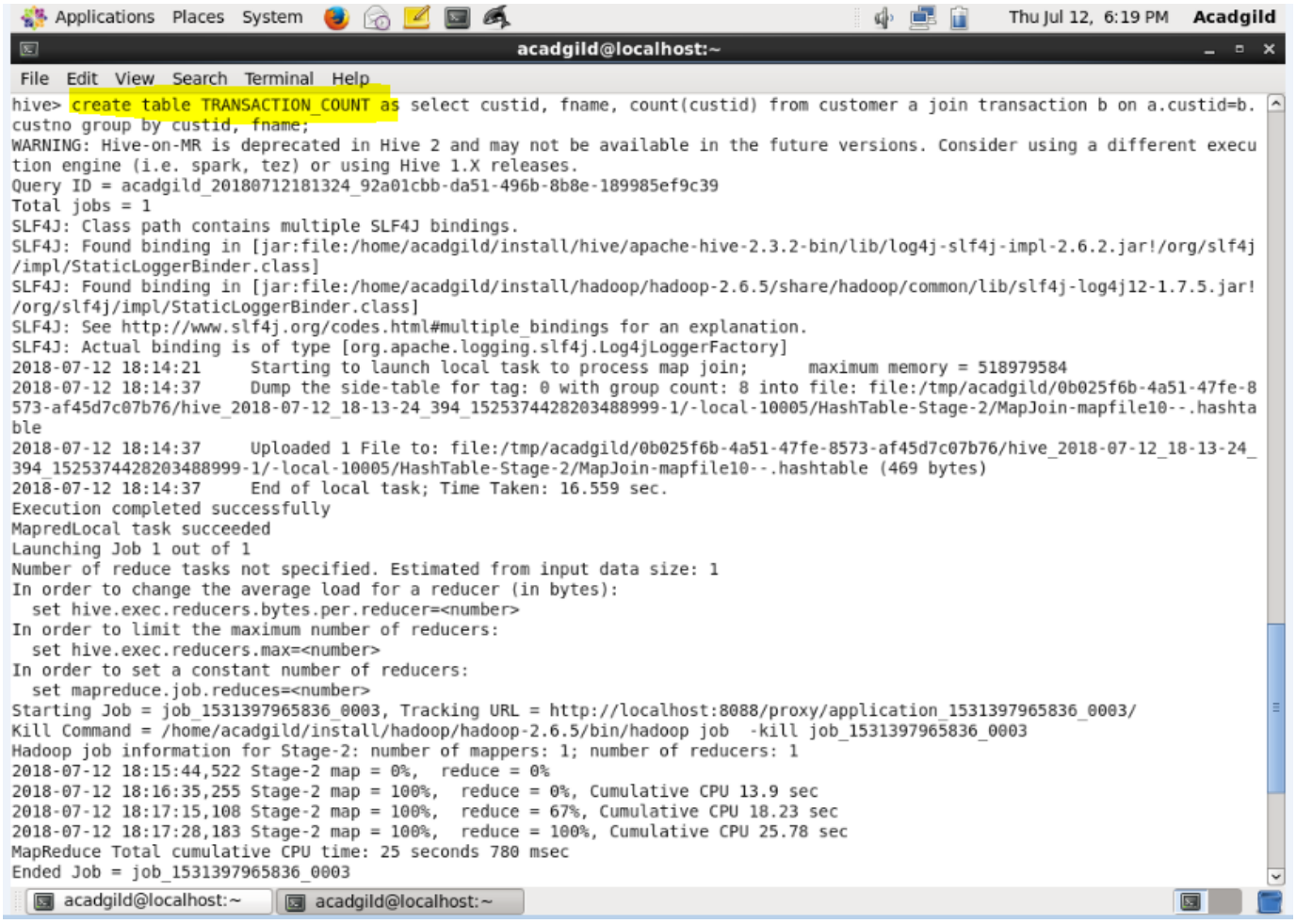
```
Applications  Places  System                                  Thu Jul 12, 6:11 PM  Acadgild
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> select a.custid, a.fname, count(custid) as total_count from customer a inner join transaction b on a.custid = b.custno
group by a.custid,a.fname;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execu
tion engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180712180109_e5e1d901-5976-4f52-803e-1775c88010e5
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]
2018-07-12 18:02:06     Starting to launch local task to process map join;      maximum memory = 518979584
2018-07-12 18:02:21     Dump the side-table for tag: 0 with group count: 8 into file: file:/tmp/acadgild/0b025f6b-4a51-47fe-8
573-af45d7c07b76/hive_2018-07-12_18-01-09_261_7450865292093202657-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile00--.hashta
ble
2018-07-12 18:02:21     Uploaded 1 File to: file:/tmp/acadgild/0b025f6b-4a51-47fe-8573-af45d7c07b76/hive_2018-07-12_18-01-09_
261_7450865292093202657-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile00--.hashtable (469 bytes)
2018-07-12 18:02:21     End of local task; Time Taken: 15.043 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_1531397965836_0002, Tracking URL = http://localhost:8088/proxy/application_1531397965836_0002/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1531397965836_0002
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-07-12 18:03:27,169 Stage-2 map = 0%,  reduce = 0%
2018-07-12 18:04:20,769 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 14.11 sec
2018-07-12 18:04:58,771 Stage-2 map = 100%,  reduce = 67%, Cumulative CPU 18.35 sec
2018-07-12 18:05:10,375 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 24.76 sec
MapReduce Total cumulative CPU time: 24 seconds 760 msec
Ended Job = job_1531397965836_0002
acadgild@localhost:~   acadgild@localhost:~
```

CASE STUDY 2


Applications Places System
Thu Jul 12, 6:11 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Help
2018-07-12 18:02:06 Starting to launch local task to process map join; maximum memory = 518979584
2018-07-12 18:02:21 Dump the side-table for tag: 0 with group count: 8 into file: file:/tmp/acadgild/0b025f6b-4a51-47fe-8573-af45d7c07b76/hive_2018-07-12_18-01-09_261_7450865292093202657-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile00--.hashtable
2018-07-12 18:02:21 Uploaded 1 File to: file:/tmp/acadgild/0b025f6b-4a51-47fe-8573-af45d7c07b76/hive_2018-07-12_18-01-09_261_7450865292093202657-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile00--.hashtable (469 bytes)
2018-07-12 18:02:21 End of local task; Time Taken: 15.043 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_1531397965836_0002, Tracking URL = http://localhost:8088/proxy/application_1531397965836_0002/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job -kill job_1531397965836_0002
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-07-12 18:03:27,169 Stage-2 map = 0%, reduce = 0%
2018-07-12 18:04:20,769 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 14.11 sec
2018-07-12 18:04:58,771 Stage-2 map = 100%, reduce = 67%, Cumulative CPU 18.35 sec
2018-07-12 18:05:10,375 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 24.76 sec
MapReduce Total cumulative CPU time: 24 seconds 760 msec
Ended Job = job_1531397965836_0002
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 24.76 sec HDFS Read: 13155 HDFS Write: 263 SUCCESS
Total MapReduce CPU Time Spent: 24 seconds 760 msec
OK
101 Amitabh 2
102 Sharukh 1
104 Anubahv 1
105 Pawan 1
106 Aamir 1
107 Salman 1
108 Ranbir 1
Time taken: 244.214 seconds, Fetched: 7 row(s)
hive>
acadgild@localhost:~
acadgild@localhost:~

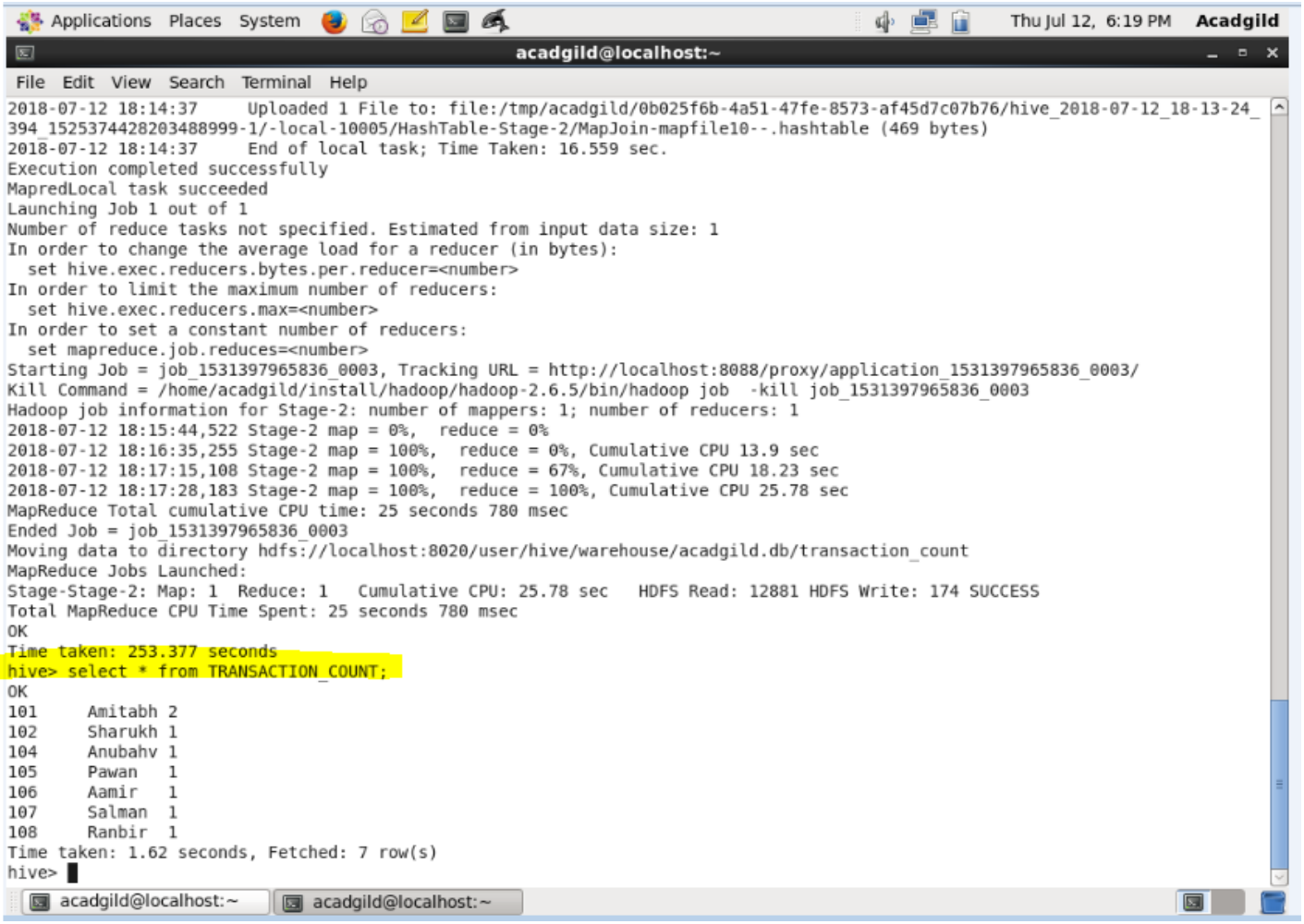
CASE STUDY 2

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).

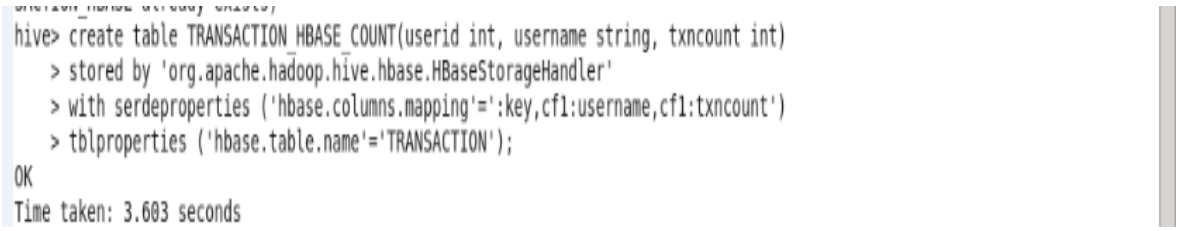


CASE STUDY 2

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).



4) Now let's make the TRANSACTIONS_COUNT table Hbase complaint. In the sense, use Ser Des And Storage 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)



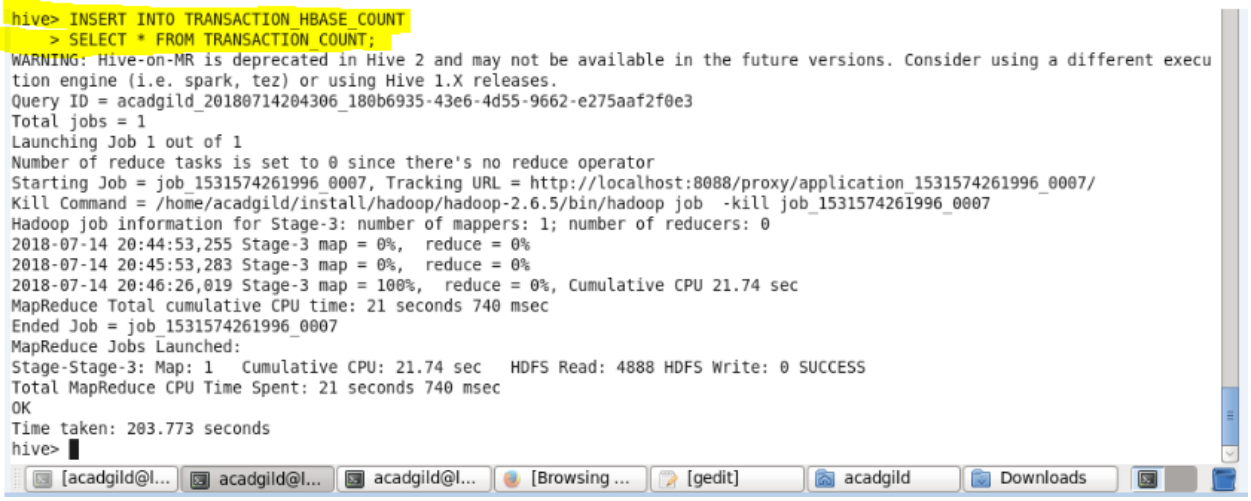
CASE STUDY 2

```
hbase(main):006:0> list
TABLE
TRANSACTION
TRANSACTIONS
2 row(s) in 8.9210 seconds

=> ["TRANSACTION", "TRANSACTIONS"]
hbase(main):007:0> scan 'TRANSACTION'
ROW                                   COLUMN+CELL
0 row(s) in 2.3330 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).

```
hive> INSERT INTO TRANSACTION_HBASE_COUNT
    > SELECT * FROM TRANSACTION_COUNT;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execu
tion engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180714204306_180b6935-43e6-4d55-9662-e275aaf2f0e3
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_1531574261996_0007, Tracking URL = http://localhost:8088/proxy/application_1531574261996_0007/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1531574261996_0007
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 0
2018-07-14 20:44:53,255 Stage-3 map = 0%,  reduce = 0%
2018-07-14 20:45:53,283 Stage-3 map = 0%,  reduce = 0%
2018-07-14 20:46:26,019 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 21.74 sec
MapReduce Total cumulative CPU time: 21 seconds 740 msec
Ended Job = job_1531574261996_0007
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1   Cumulative CPU: 21.74 sec   HDFS Read: 4888 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 21 seconds 740 msec
OK
Time taken: 203.773 seconds
hive>
```




```
hbase(main):008:0> scan 'TRANSACTION'
ROW                                   COLUMN+CELL
 101                                  column=cf1:txncount, timestamp=1531581383815, value=2
 101                                  column=cf1:username, timestamp=1531581383815, value=Amitabh
 102                                  column=cf1:txncount, timestamp=1531581383815, value=1
 102                                  column=cf1:username, timestamp=1531581383815, value=Sharukh
 104                                  column=cf1:txncount, timestamp=1531581383815, value=1
 104                                  column=cf1:username, timestamp=1531581383815, value=Anubahv
 105                                  column=cf1:txncount, timestamp=1531581383815, value=1
 105                                  column=cf1:username, timestamp=1531581383815, value=Pawan
 106                                  column=cf1:txncount, timestamp=1531581383815, value=1
 106                                  column=cf1:username, timestamp=1531581383815, value=Aamir
 107                                  column=cf1:txncount, timestamp=1531581383815, value=1
 107                                  column=cf1:username, timestamp=1531581383815, value=Salman
 108                                  column=cf1:txncount, timestamp=1531581383815, value=1
 108                                  column=cf1:username, timestamp=1531581383815, value=Ranbir
7 row(s) in 8.8810 seconds

hbase(main):009:0>
```

CASE STUDY 2

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