Knowledge discovery and Management

CS560

Report on LAB-4 tutorial

By : Vidhi Shah

16178723

Complete the following tasks:

1. Cloudera / Mahout / Solr: Implement a Mahout application (e.g., classification/Recommedation/Association Rule Mining) using your own data and store your output to Solr.

Answer:

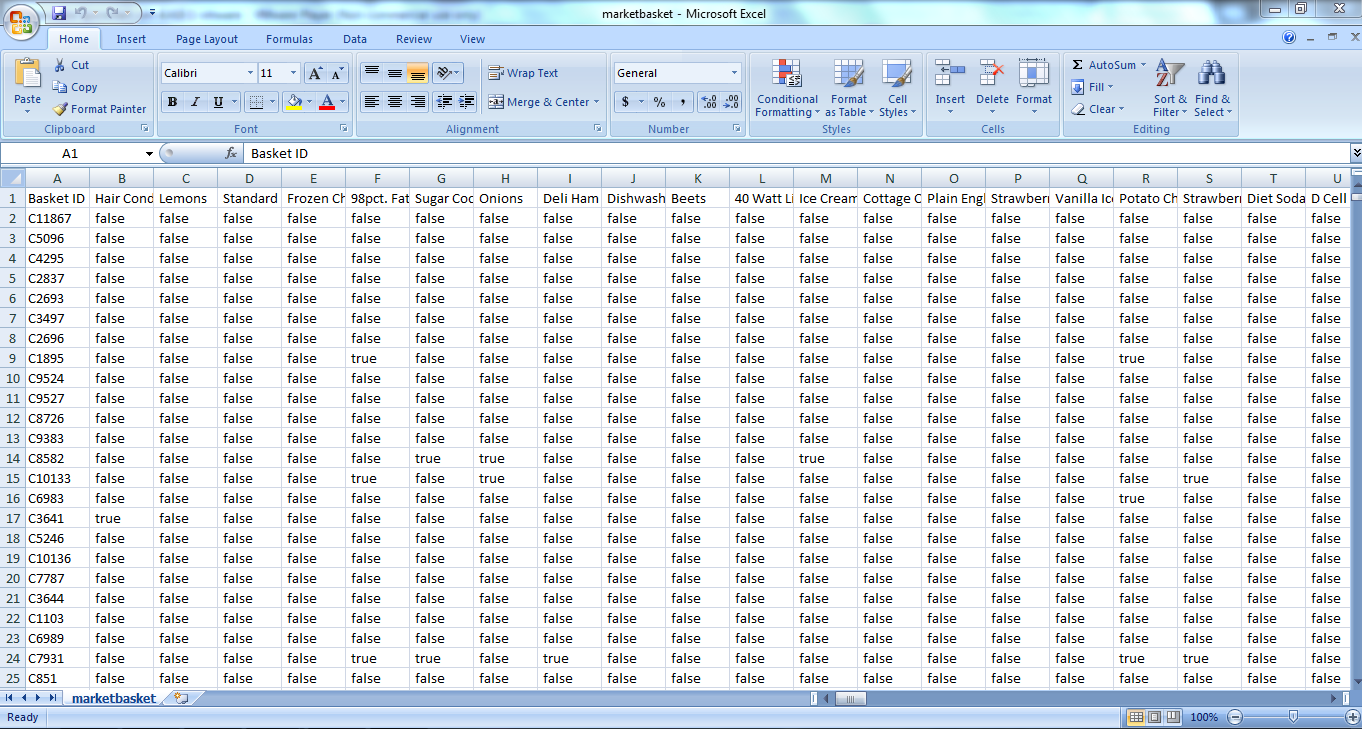
In my work, I have used Recommendation rule mining is used for the data mining. I have gone through the following steps:

1. **Implementing Association With Mahout Frequent Pattern Mining :**

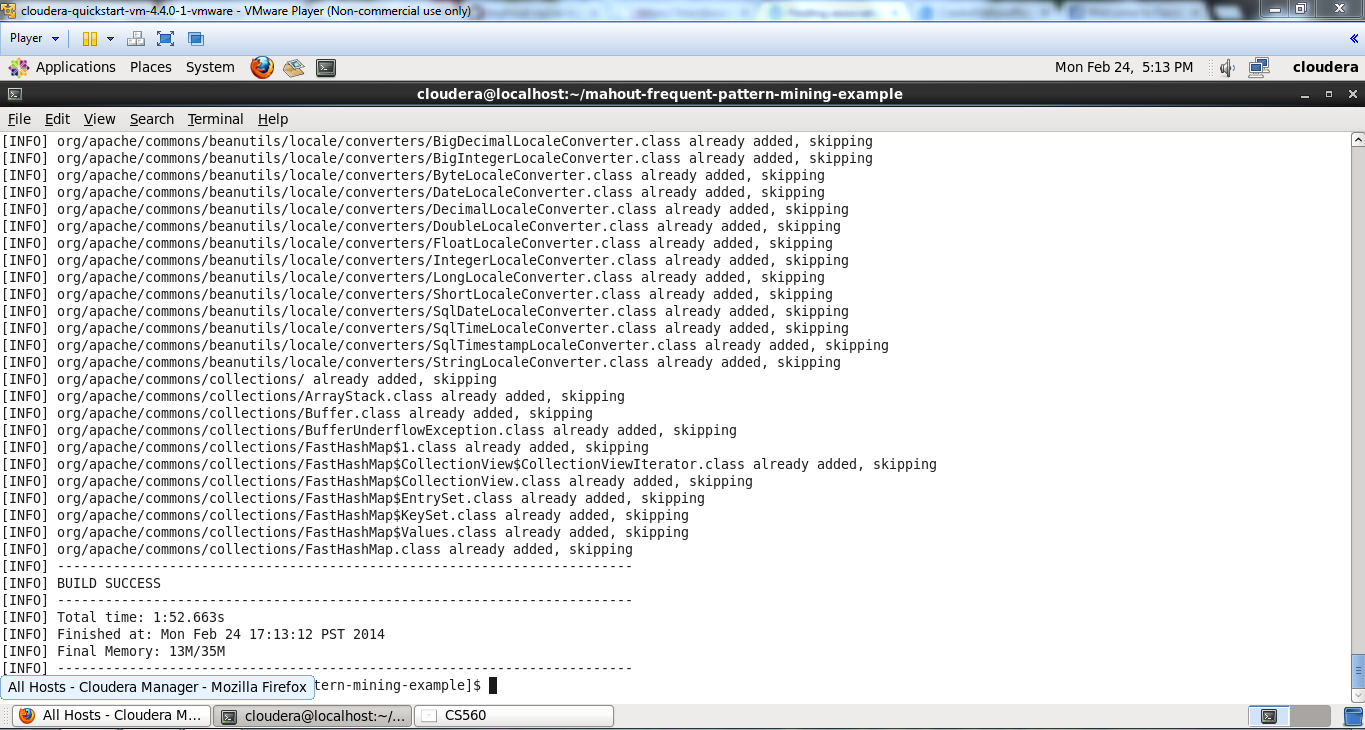
The below screen shot shows the downloading of mahout-frequent-pattern-mining-example source code.



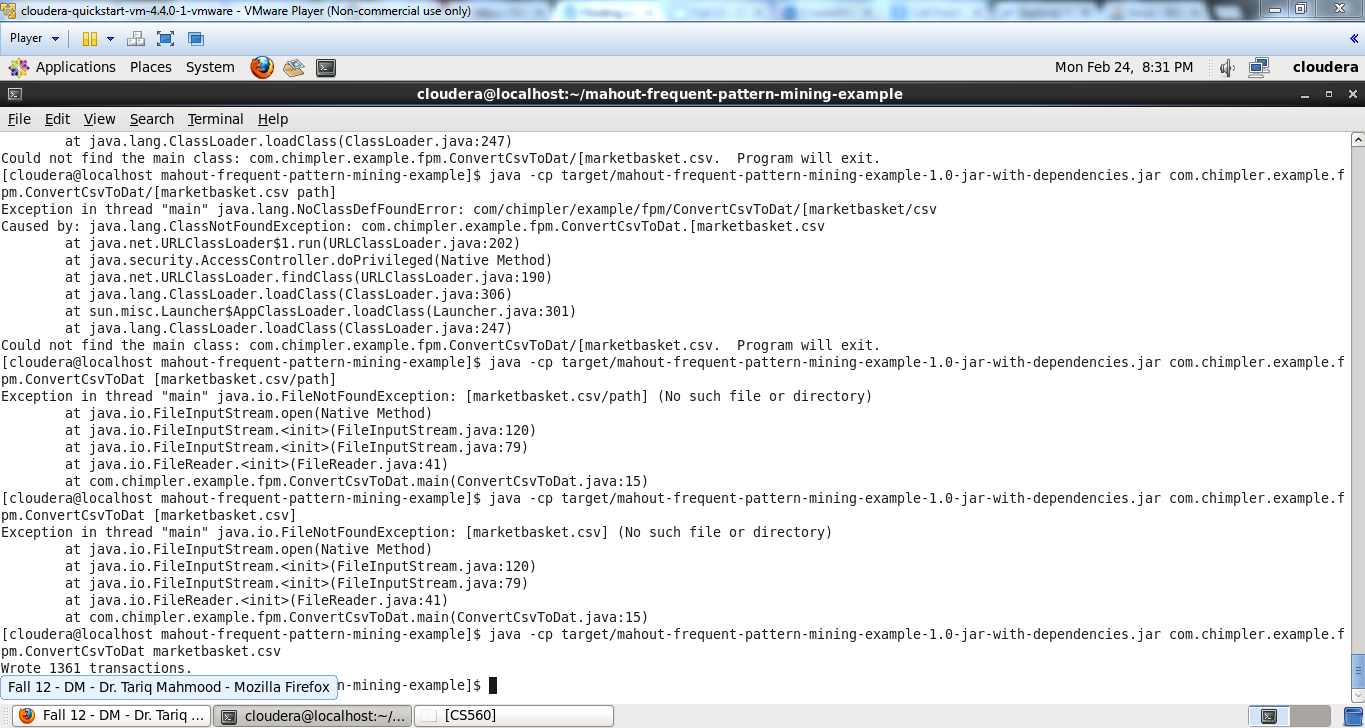
The below screen shot shows the format of the csv file.



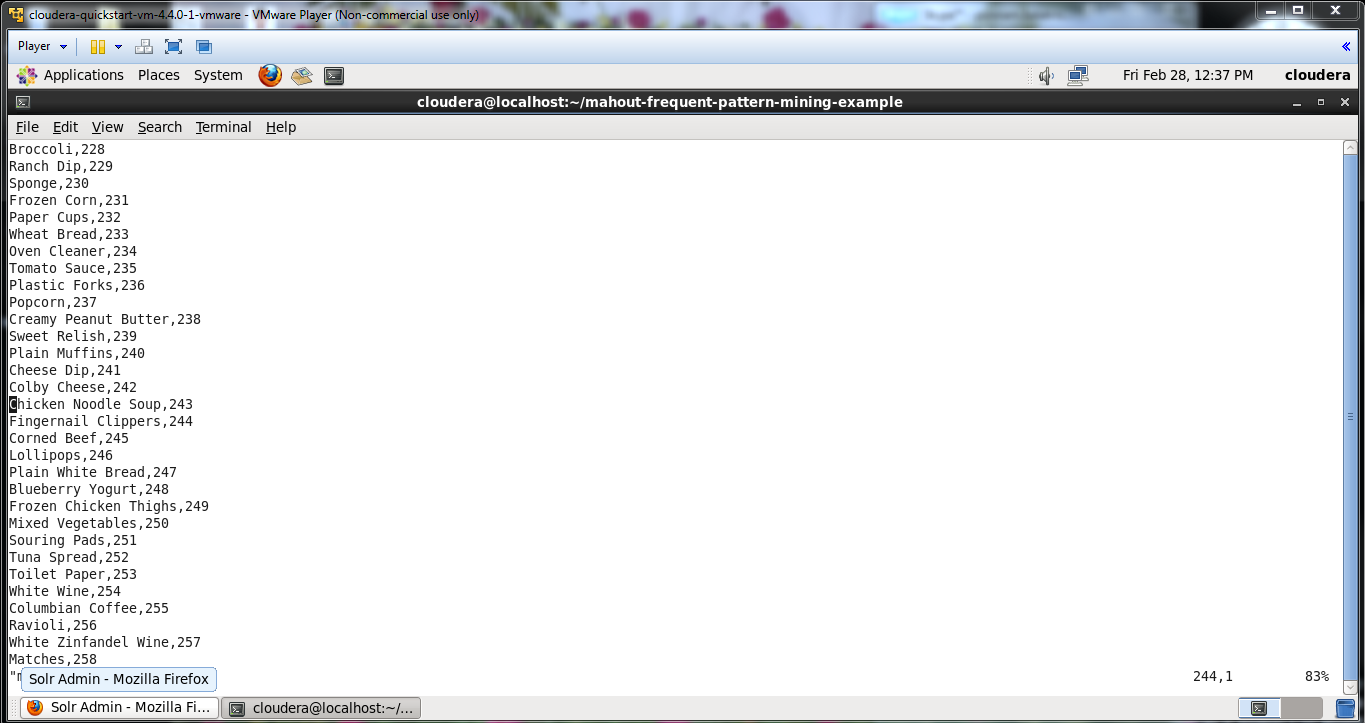
This screen shot shows the compilng of project.



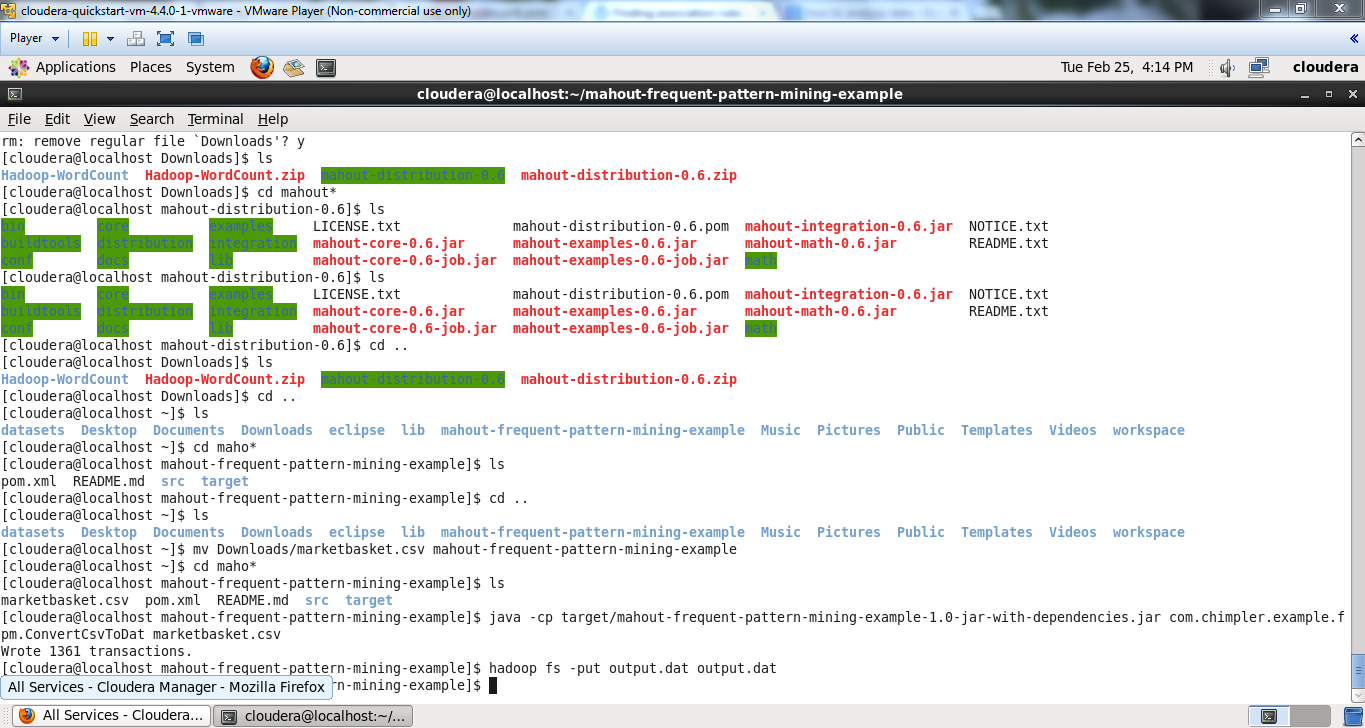
This screen shot shows running of project



The screen shot shows the data in the mapping file.



The below screen shows sending output file to the Hadoop file system.



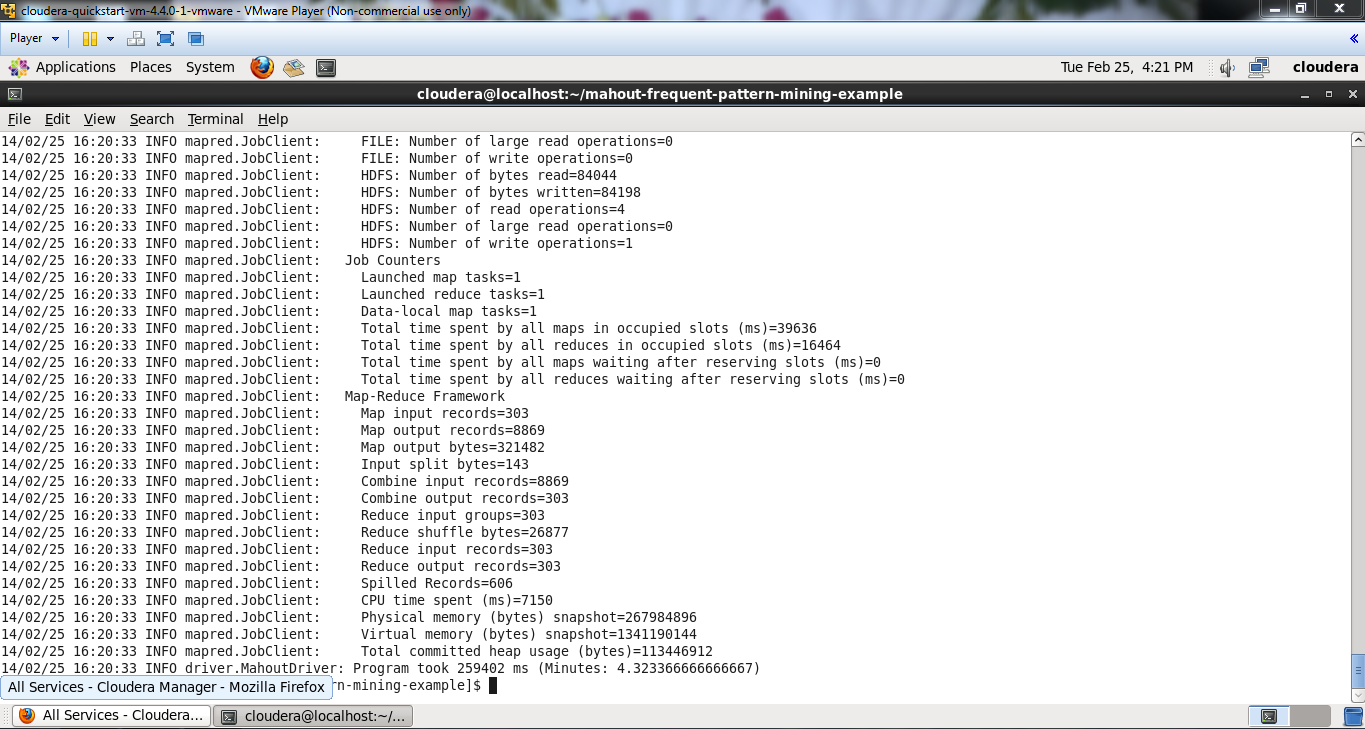
Then run the Mahout by using the command

“**mahout fpg -i output.dat -o patterns -k 10 -method mapreduce -s 2**”

-k 10 means that for each item i, we find the top 10 association rules: the sets of items including the item i which occurs in the biggest number of transactions.

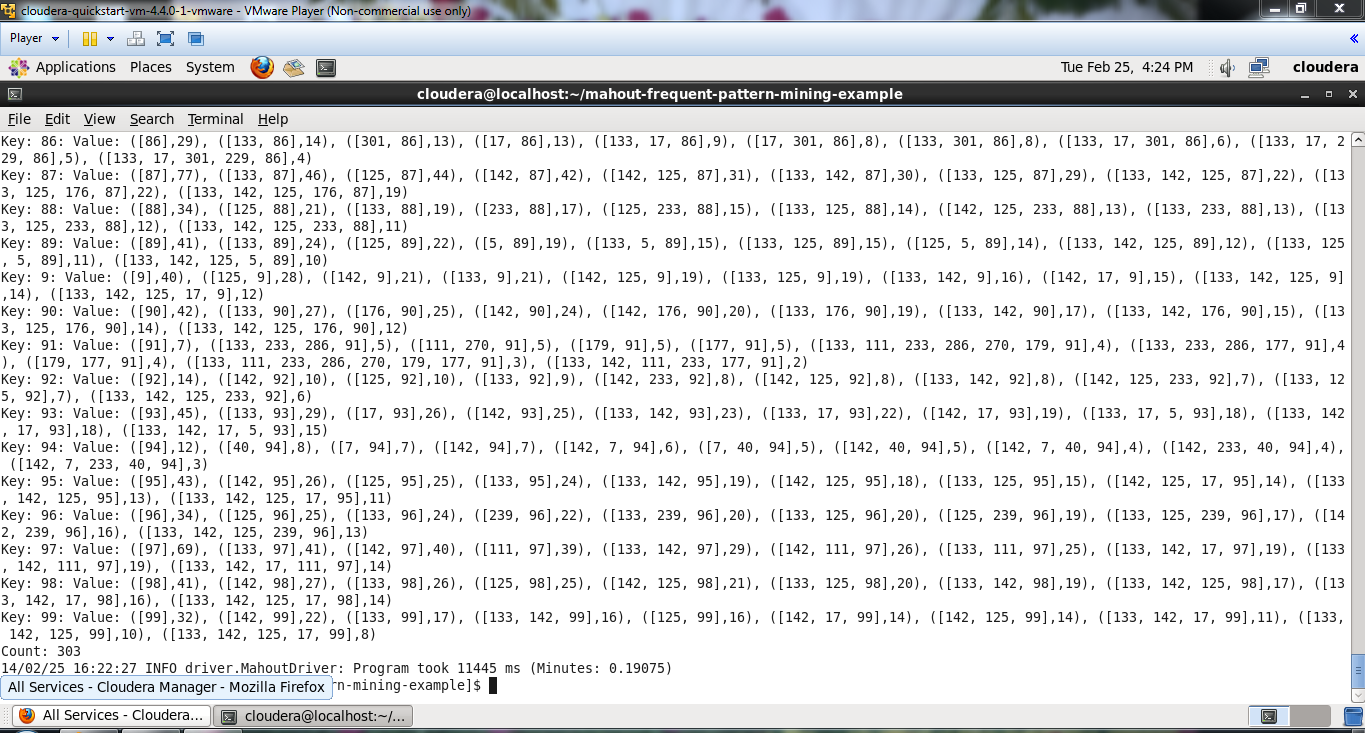
-s 2 means that we only consider the sets of items which appear in more than 2 transactions.

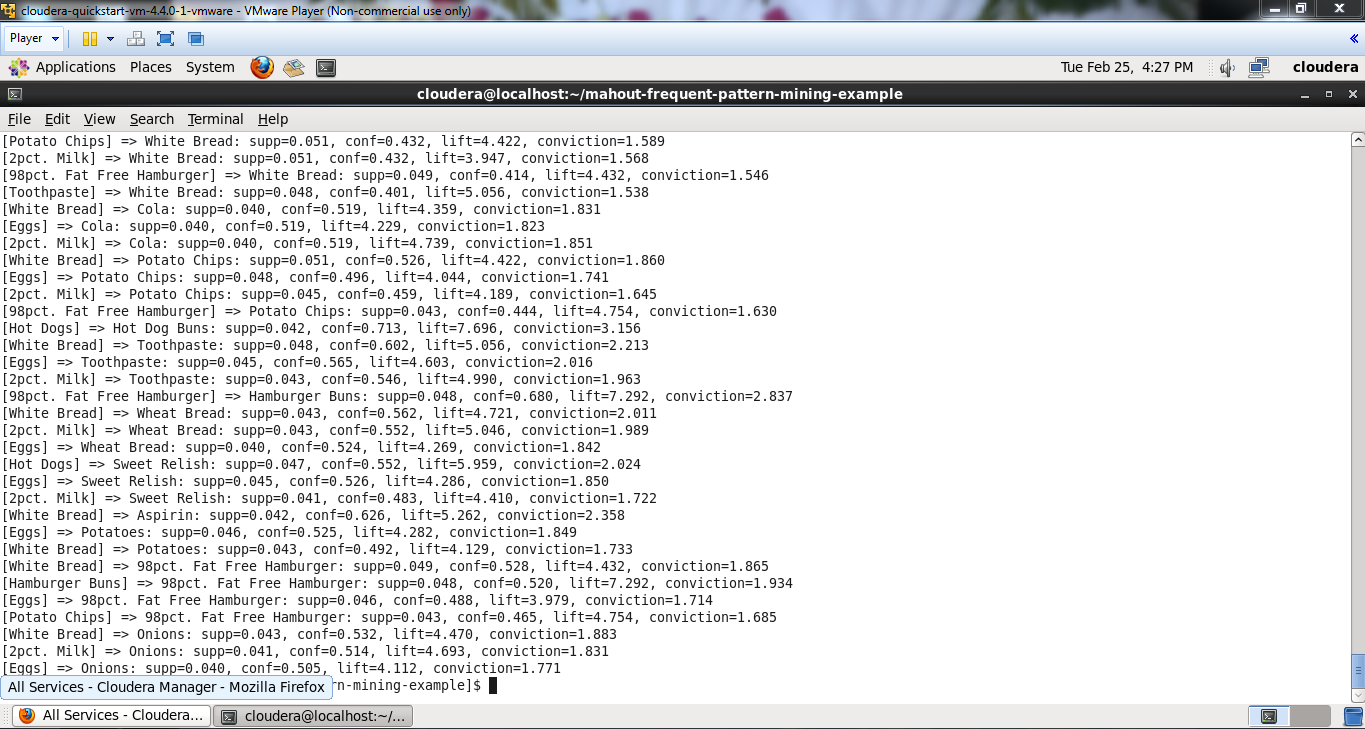
The below screen shows running of mahout command.



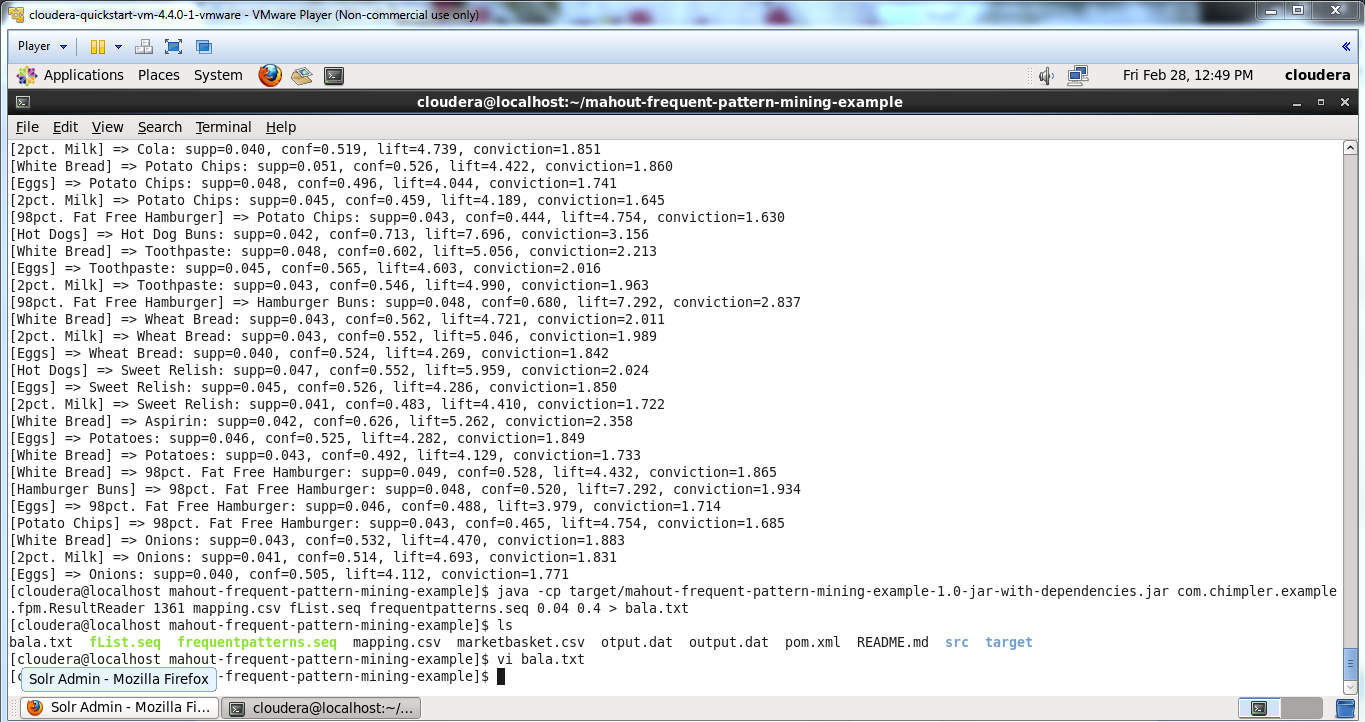
After running mahout it creates several files and directories in that folder

They are





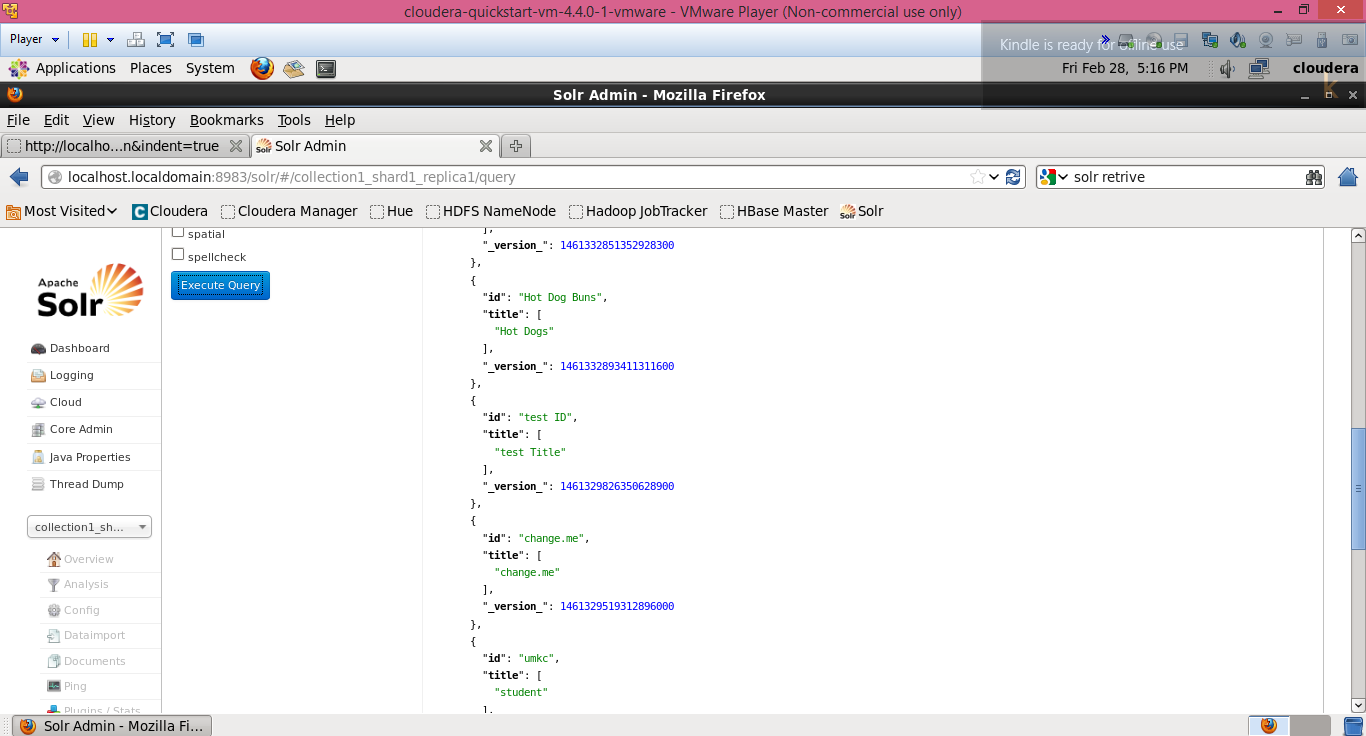
The below screen shows the output in the text file.



Now establish the restful service and connect to the solr to display this output in the solr.

Push the data to the Solr.

The below screen shows the all data which is inserted by querying \*

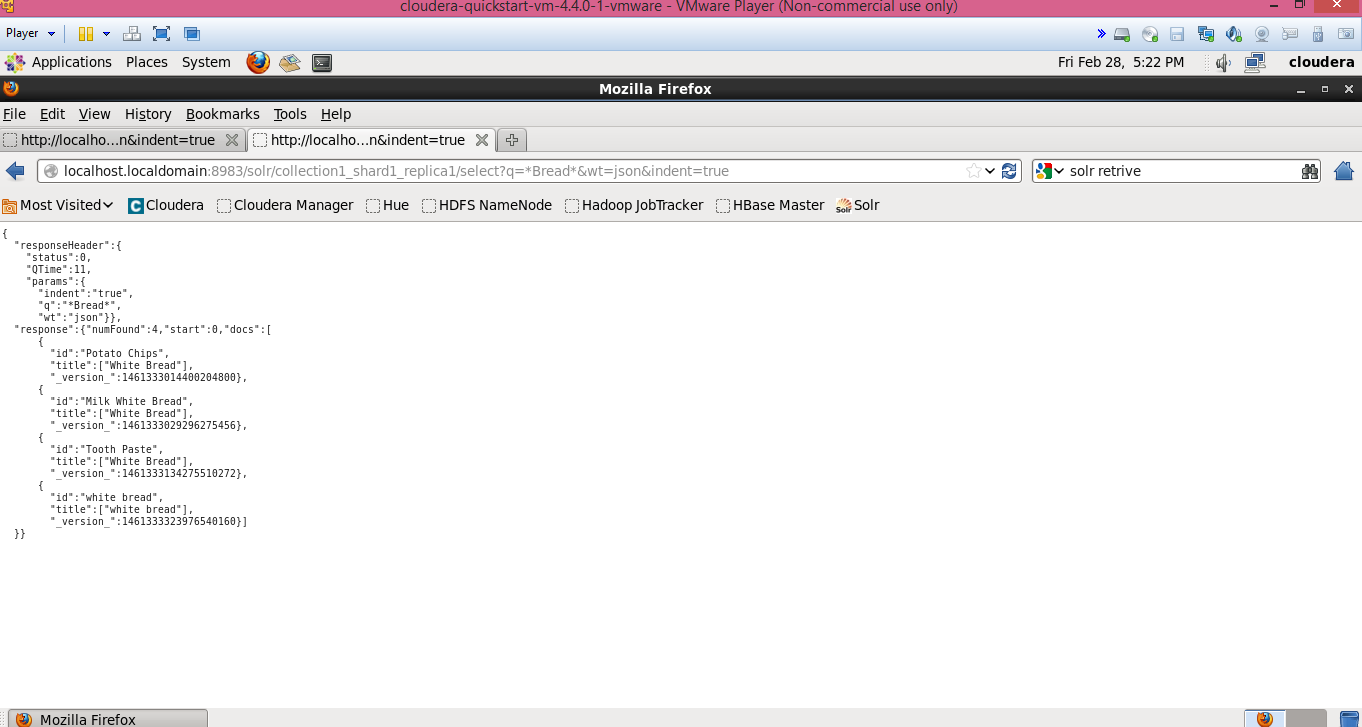


The URI of these data in the solr is:

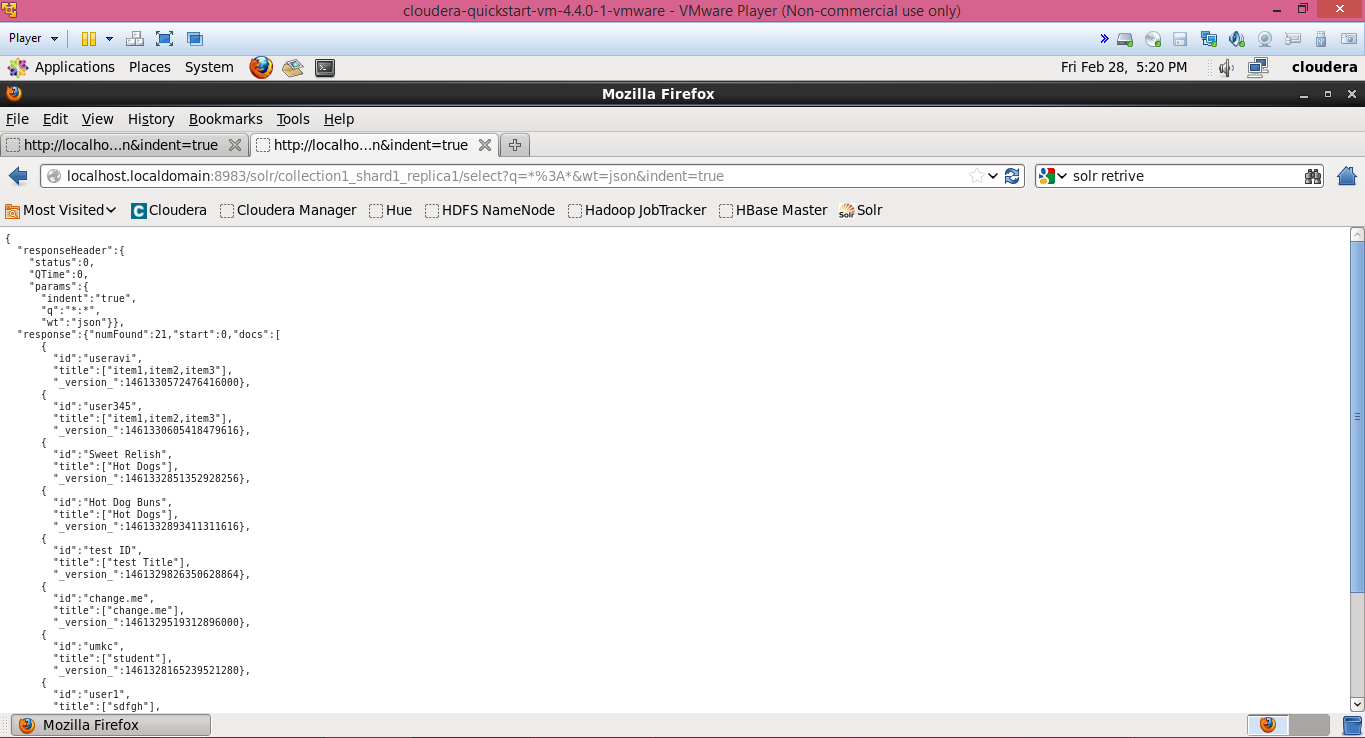
[**http://localhost.localdomain:8983/solr/collection1\_shard1\_replica1/select?q=\*Bread\*&wt=json&indent=true**](http://localhost.localdomain:8983/solr/collection1_shard1_replica1/select?q=*Bread*&wt=json&indent=true)

Where the query is given between \* \*

The below screen shows the data of the single item.

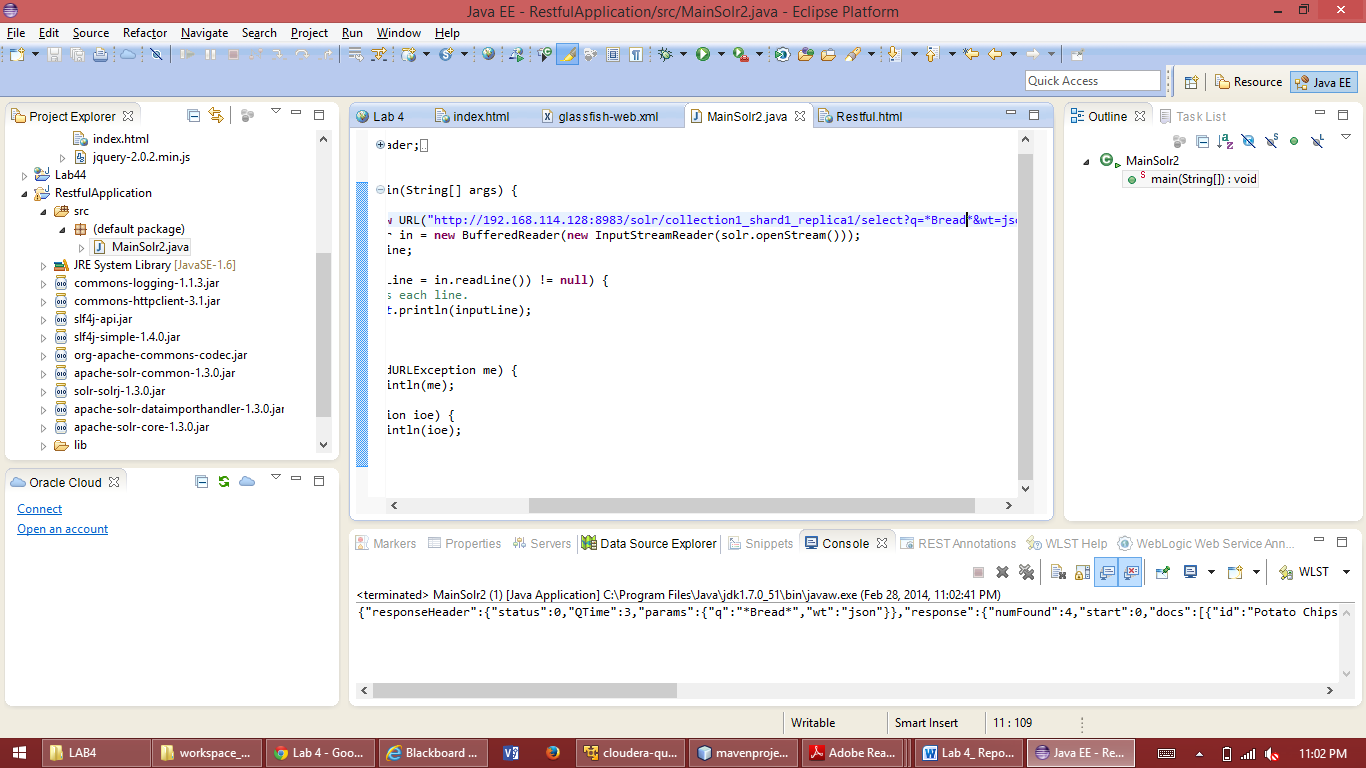


The below screen shows the uri and data of all items.

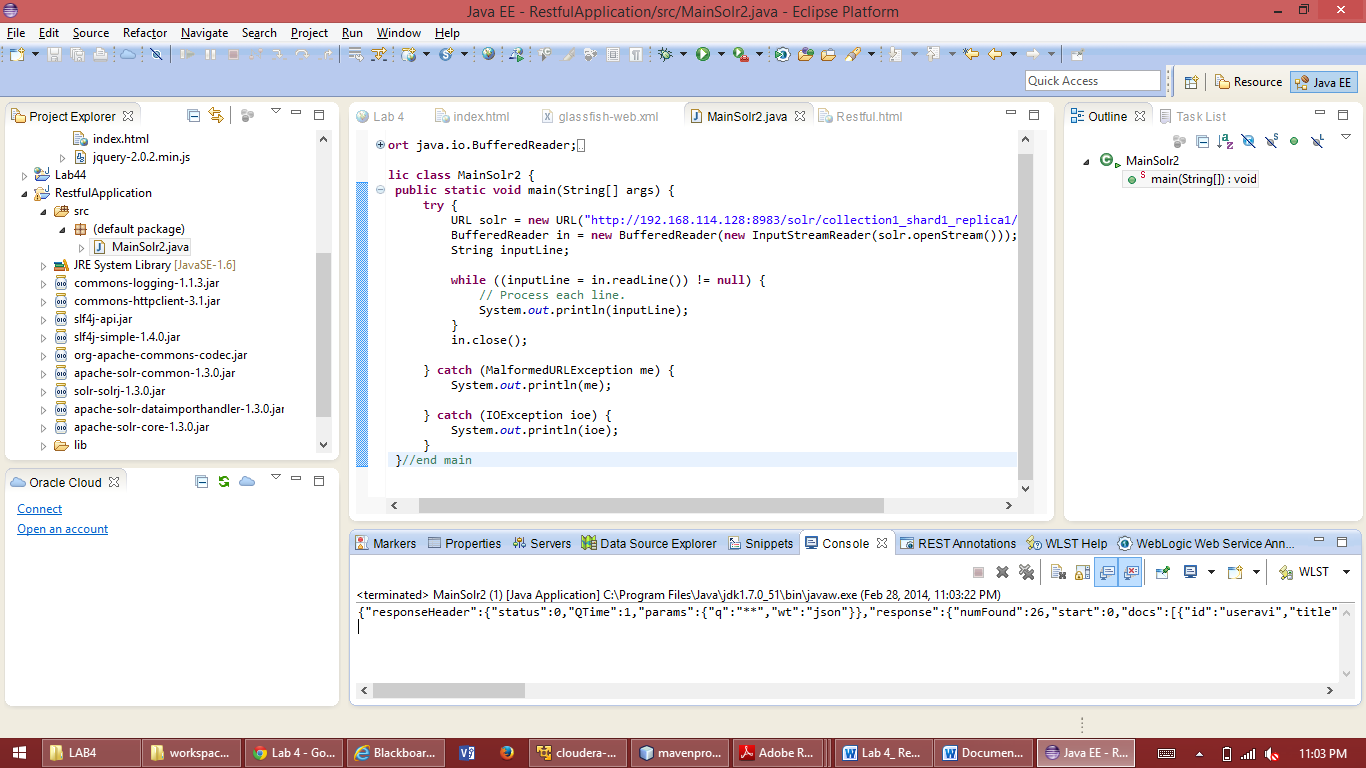


1. **Mobile Client Application :**

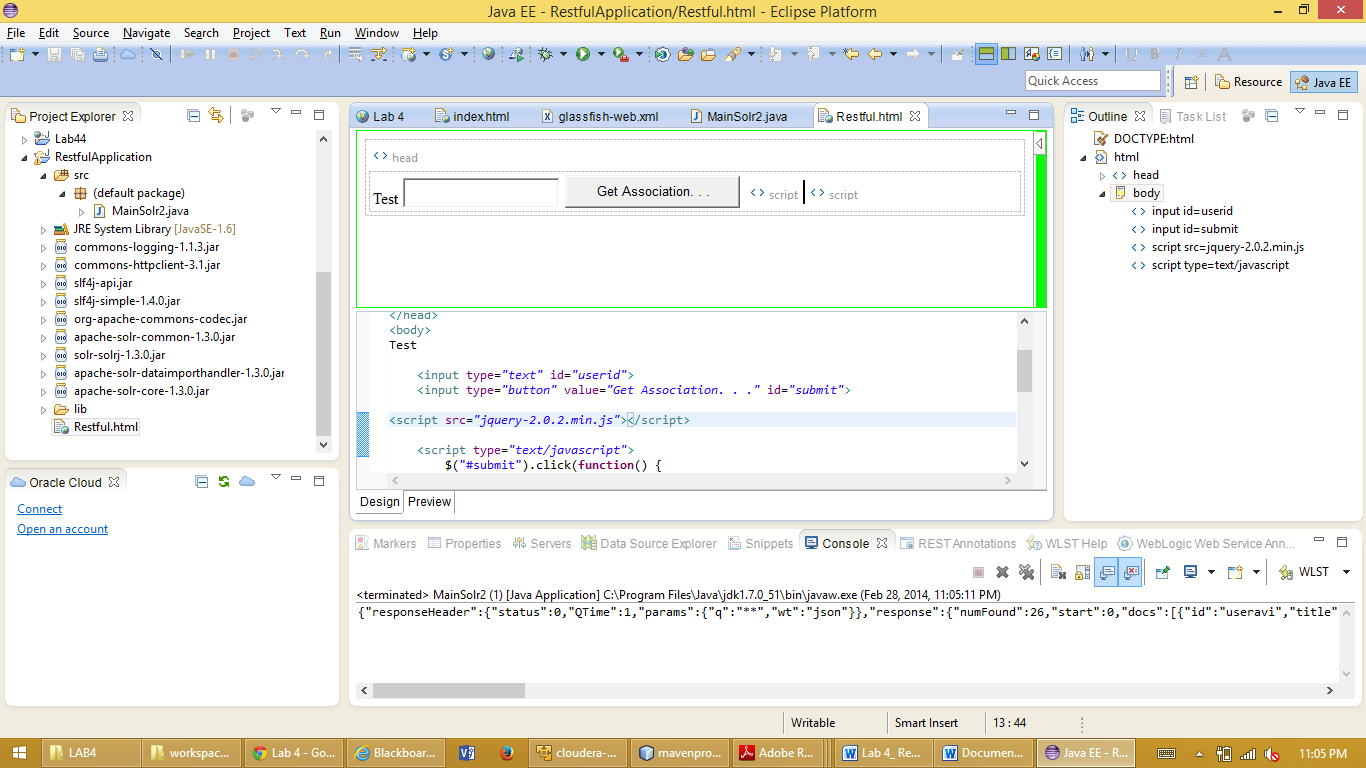
Create a Restful service to access data from the solr. For Restful service we can call the URL in java code. This URL access the solr connection and gets the data given.



{"id":"Sweet Relish","title":["Hot Dogs"],"\_version\_":1461332851352928256},{"id":"Hot Dog Buns","title":["Hot Dogs"],"\_version\_":1461332893411311616},{"id":"testID","title":["test Title"],"\_version\_":1461329826350628864},{"id":"change.me","title":["change.me"],"\_version\_":1461329519312896000},{"id":"umkc","title":["student"],"\_version\_":1461328165239521280},{"id":"Potato Chips","title":["White Bread"],"\_version\_":1461333014400204800},{"id":"Milk White Bread","title":["White Bread"],"\_version\_":1461333029296275456},{"id":"ToothPaste","title":["White Bread"],"\_version\_":1461333134275510272}]}}



The below screen shows the Getting Association through URL from app



1. **Implementing Association With Mahout Frequent Pattern Mining :**

Association rule finds the relation between variables in database.

First step is getting the source code from git hub. We can get the source code from git hub using the command

“**git clone** [**https://github.com/fredang/mahout-frequent-pattern-mining-example.git**](https://github.com/fredang/mahout-frequent-pattern-mining-example.git)”

The above command will download the source code of mahout-frequent-pattern-mining-example.

The below screen shot shows the downloading of mahout-frequent-pattern-mining-example source code.



This download creates a folder mahout-frequent-pattern-mining-example folder in the local. This contains the pom.xml file and source, target folders.

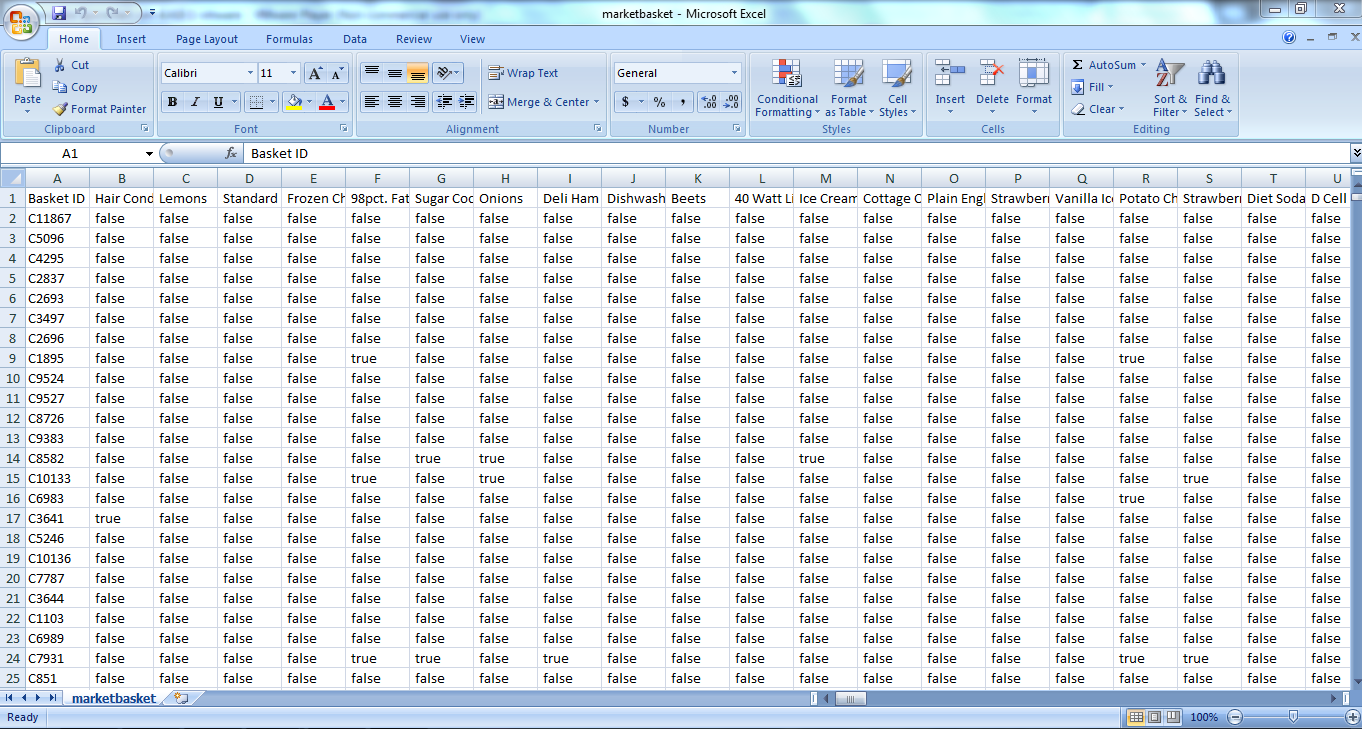
Then need to compile the project. The project can be compiled using the command

“**mvn clean package assembly:single**”

Then we need to write the dataset of items. The example we are using is a grocery store. So we need to mention the items, ItemID in the csv file because the code we are using is passing csv file as an argument to the source code.

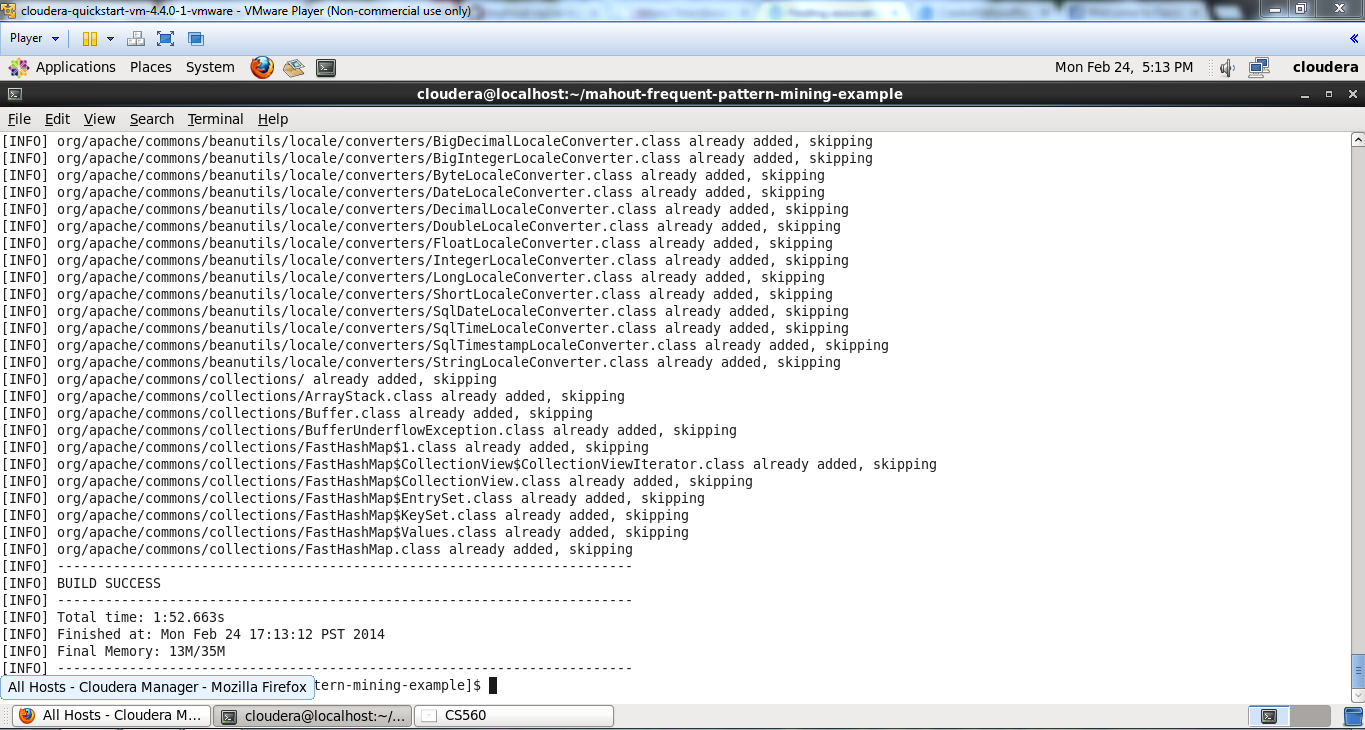
In this file we mention the all items in the grocery store and their ID with all the data

The below screen shot shows the format of the csv file.



The below screen shot shows the compiling of the source code of mahout-frequent-patern-mining-example. This project is compiled using the maven.

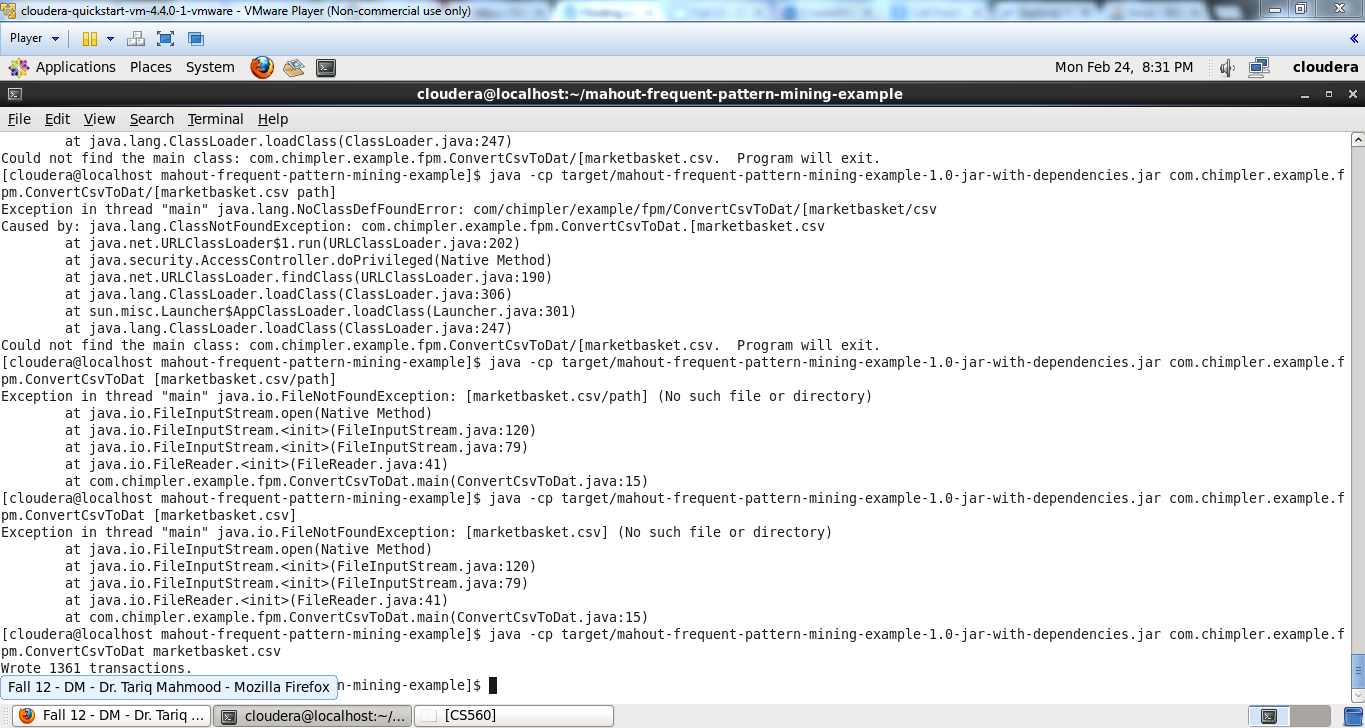
This screen shot shows the compilng of project.



Then run the project. Project can be run by using the command

“**java -cp target/mahout-frequent-pattern-mining-example-1.0-jar-with-dependencies.jar com.chimpler.example.fpm.ConvertCsvToDat marketbasket.csv**”

This screen shot shows running of project



This write all transactions to output file.

After running the project it creates two files output.dat and mapping.csv in the folder mahout-frequent-pattern-mining-example.

Output.dat contains the transaction data in new data format.

Mapping.csv contains the mapping between the item name and the item id.

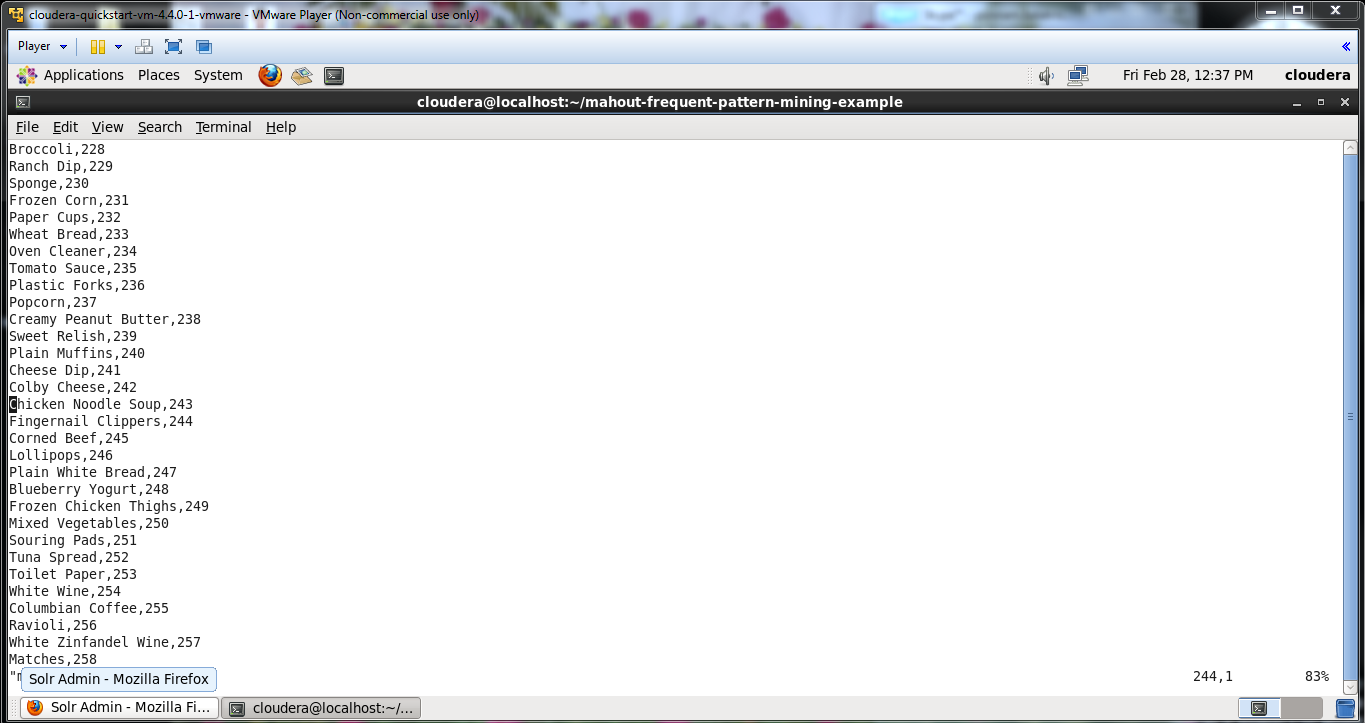
To run mahout frequent pattern mining we need to copy the file Output.dat which is created in mahout-frequent-pattern-mining-example folder to Hadoop File System (HDFS).

The copying is done by using the command “**hadoop fs -put output.dat output.dat**”.

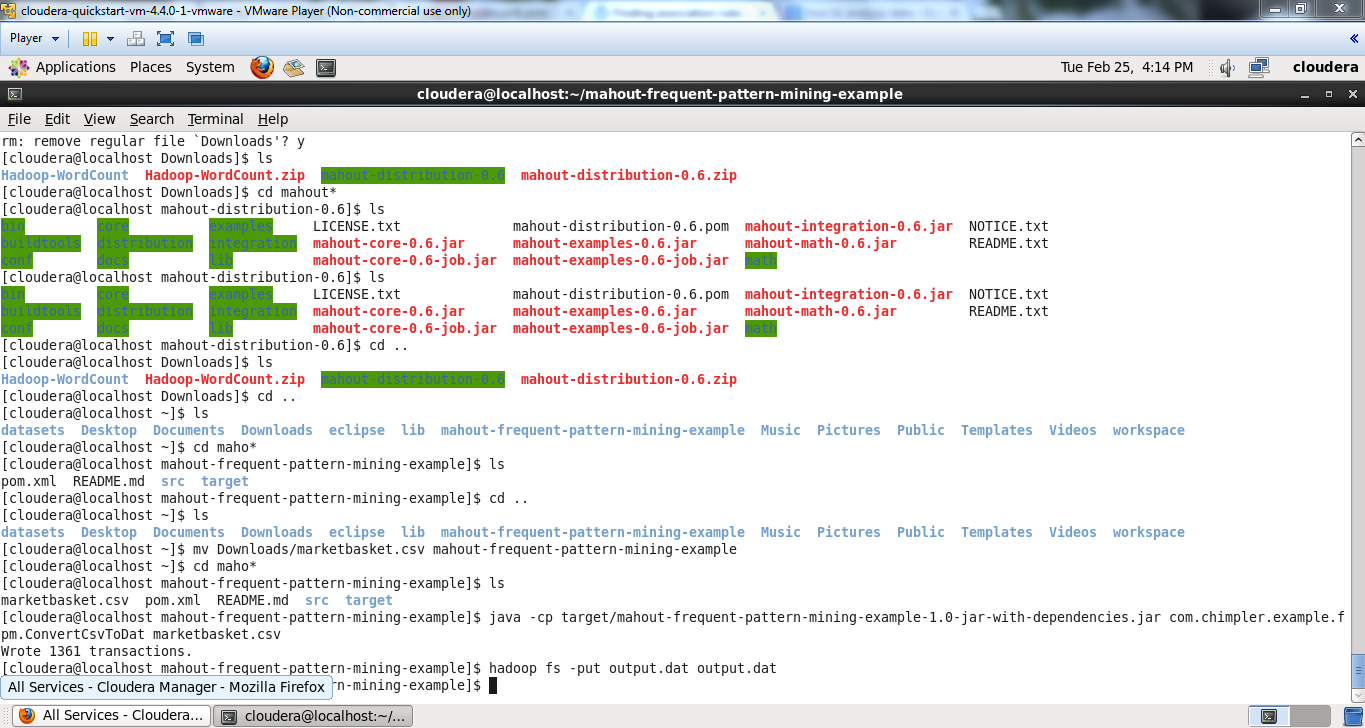
Then we need to run the mahout.

In the mapping file it counts the word count how many times the word is there. Thus the mapping file in mahout-frequent-pattern-mining-example folder contains the word count of all the words.

The screen shot shows the data in the mapping file.



The below screen shows sending output file to the Hadoop file system.



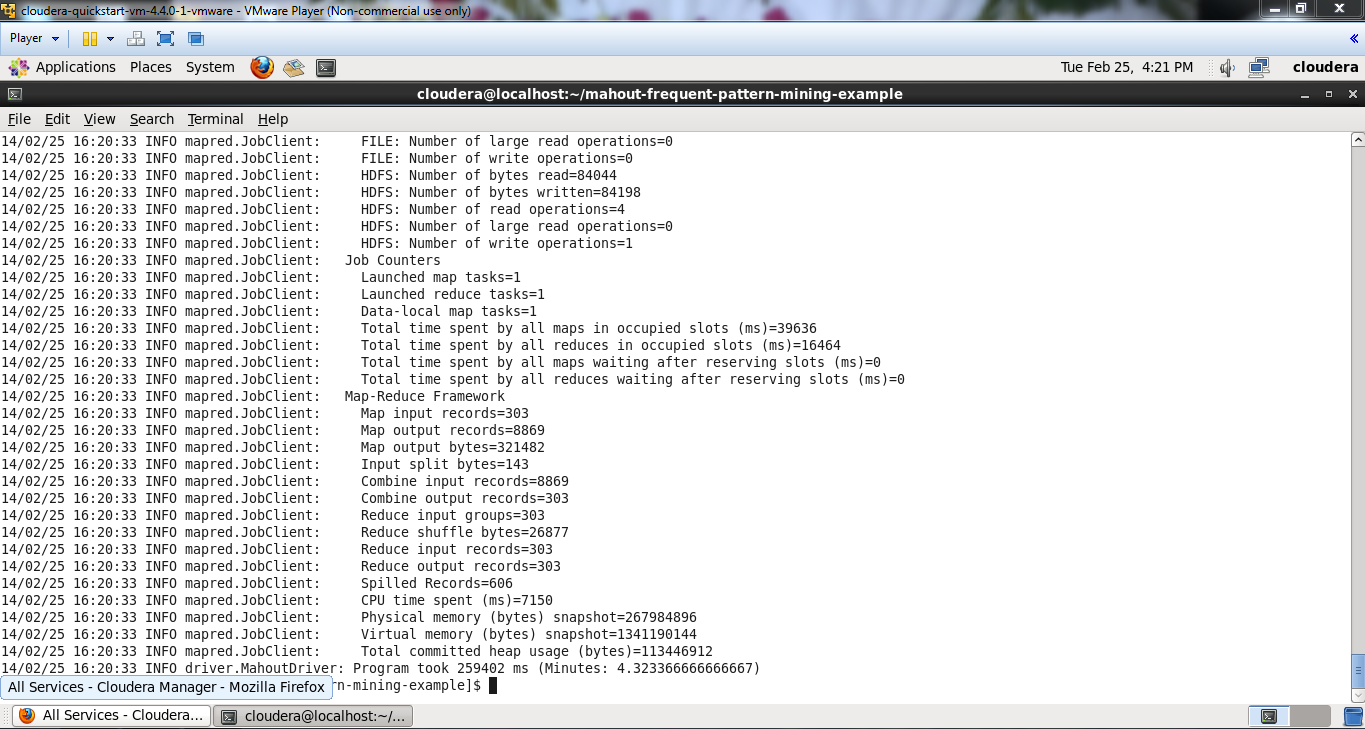
Then run the Mahout by using the command

“**mahout fpg -i output.dat -o patterns -k 10 -method mapreduce -s 2**”

-k 10 means that for each item i, we find the top 10 association rules: the sets of items including the item i which occurs in the biggest number of transactions.

-s 2 means that we only consider the sets of items which appear in more than 2 transactions.

The below screen shows running of mahout command.



After running mahout it creates several files and directories in that folder

They are

fList: it is sequence file which associates to an item the number of transactions.

Frequentpatterns: it is the sequence file which contains for each item.

Then we can also see the raw result by using the command

“**mahout seqdumper -i patterns/frequentpatterns/part-r-00000**”

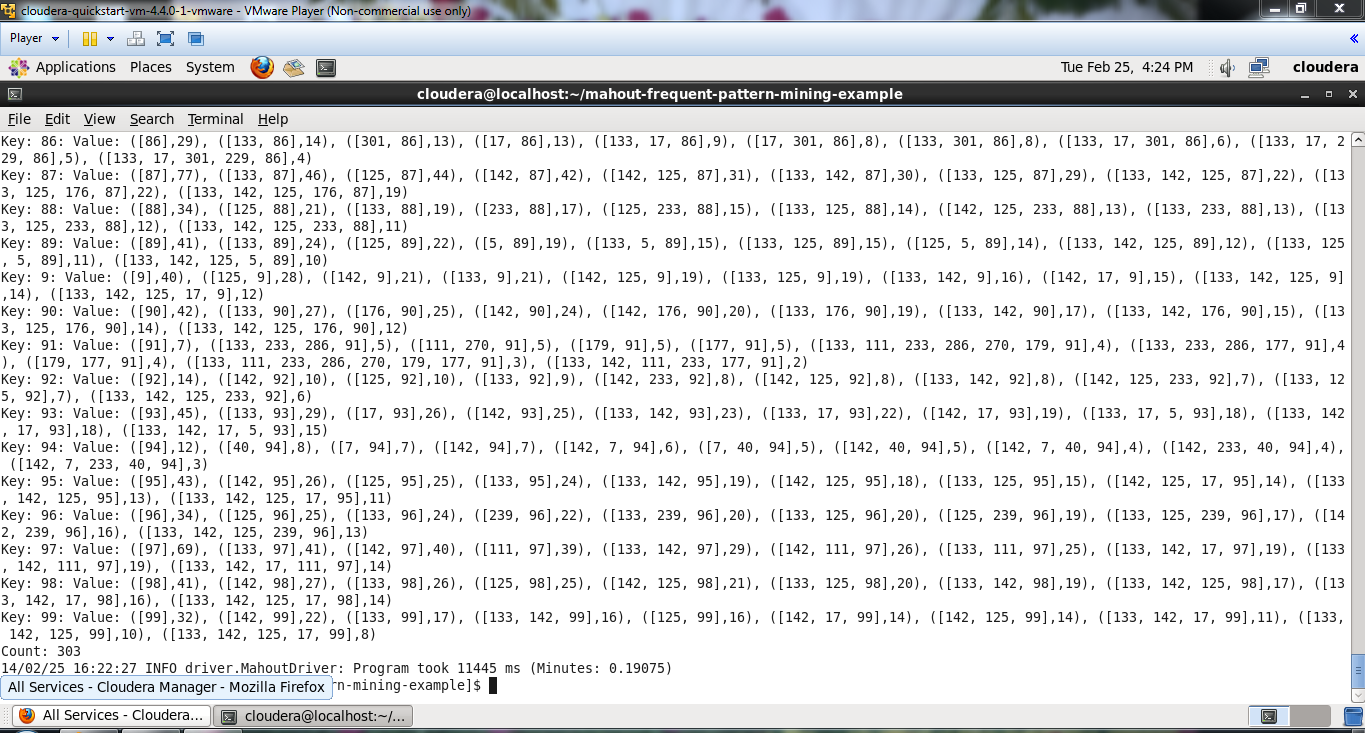
And we can also need to copy the files flist and frequent patterns from Hadoop File System to local such that we can analyze the data.

Those copies of files can be done using the command

“**hadoop fs -getmerge patterns/frequentpatterns frequentpatterns.seq**”

“**hadoop fs -get patterns/fList fList.seq**”

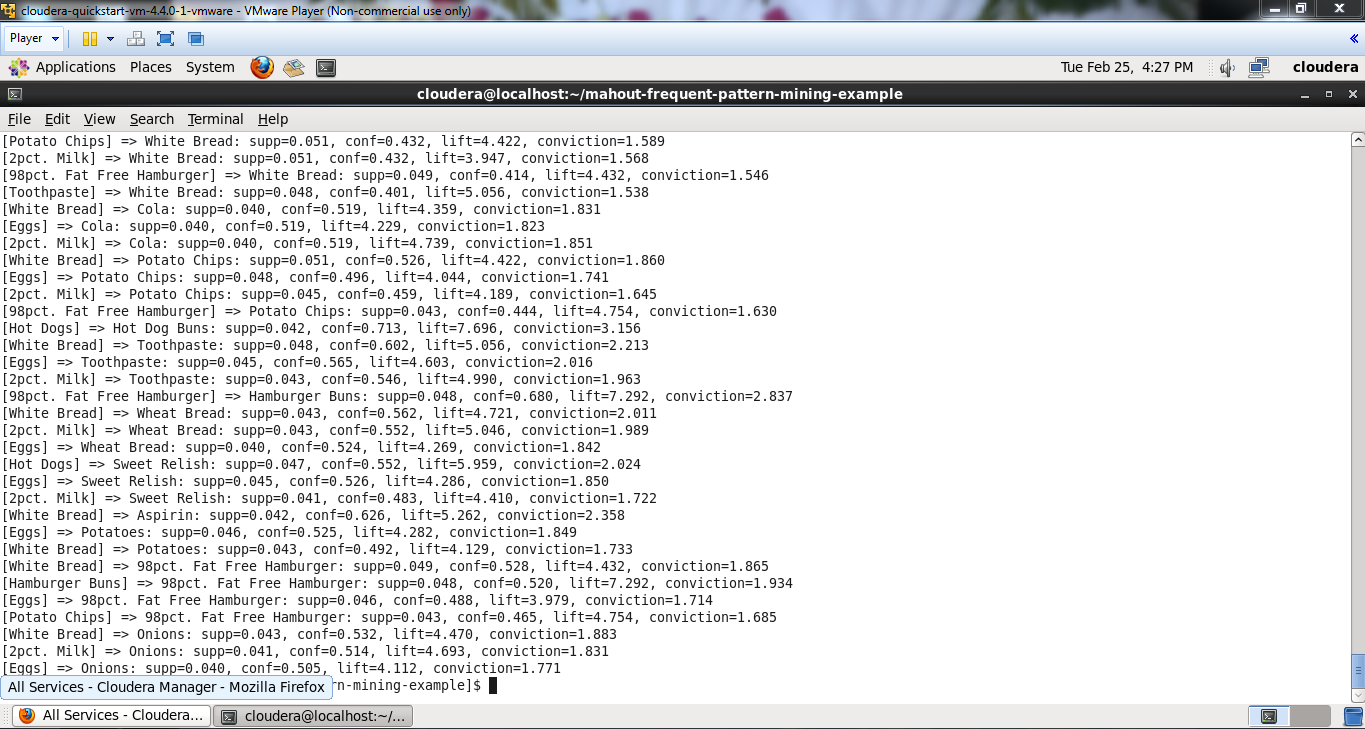
The below Screen shows the raw data of result after running the mahout.



Then run the program.This program can be run by using the command

“**java -cp target/mahout-frequent-pattern-mining-example-1.0-jar-with-dependencies.jar com.chimpler.example.fpm.ResultReader 1361 mapping.csv fList.seq frequentpatterns.seq 0.04 0.4**”

The below screen shows the output after running the command

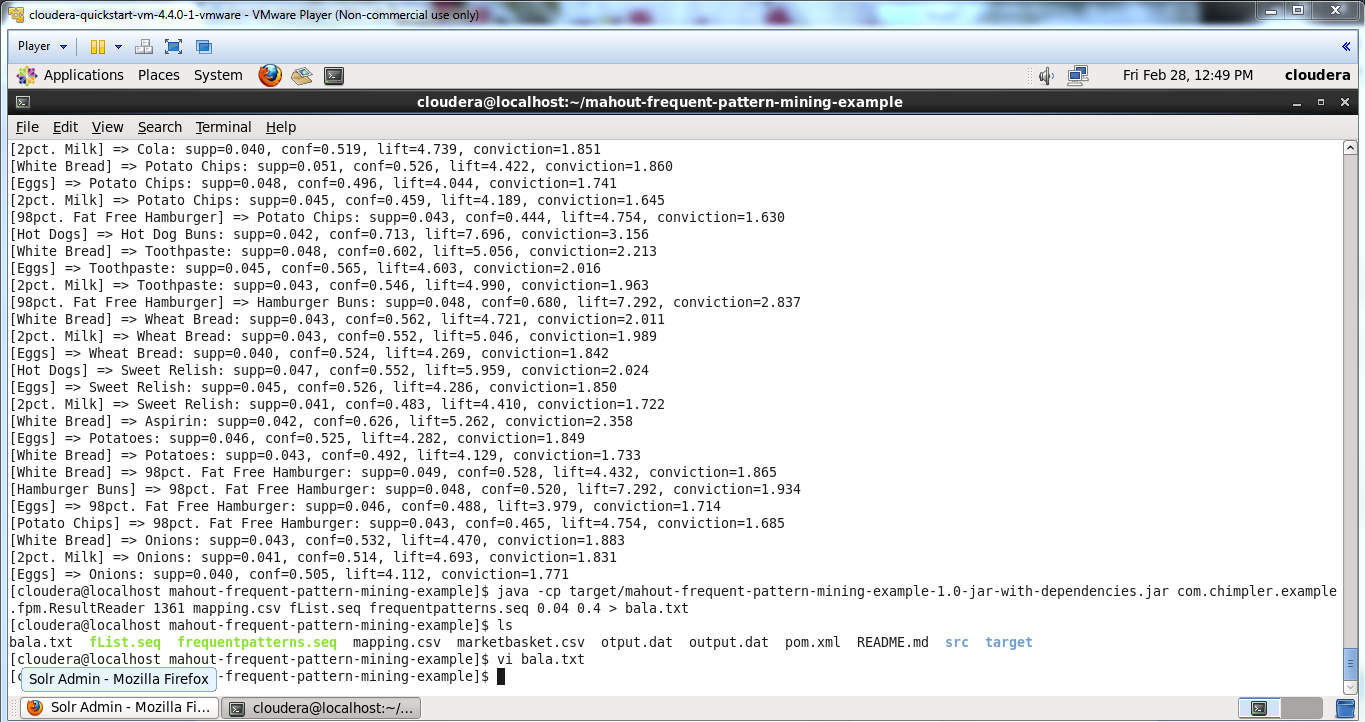


Then copy the output directly to the text file by using the command

“**java -cp target/mahout-frequent-pattern-mining-example-1.0-jar-with-dependencies.jar com.chimpler.example.fpm.ResultReader 1361 mapping.csv fList.seq frequentpatterns.seq 0.04 0.4 > bala.txt**”

This creates the text file and copy the output into this file.

The below screen shows the output in the text file.



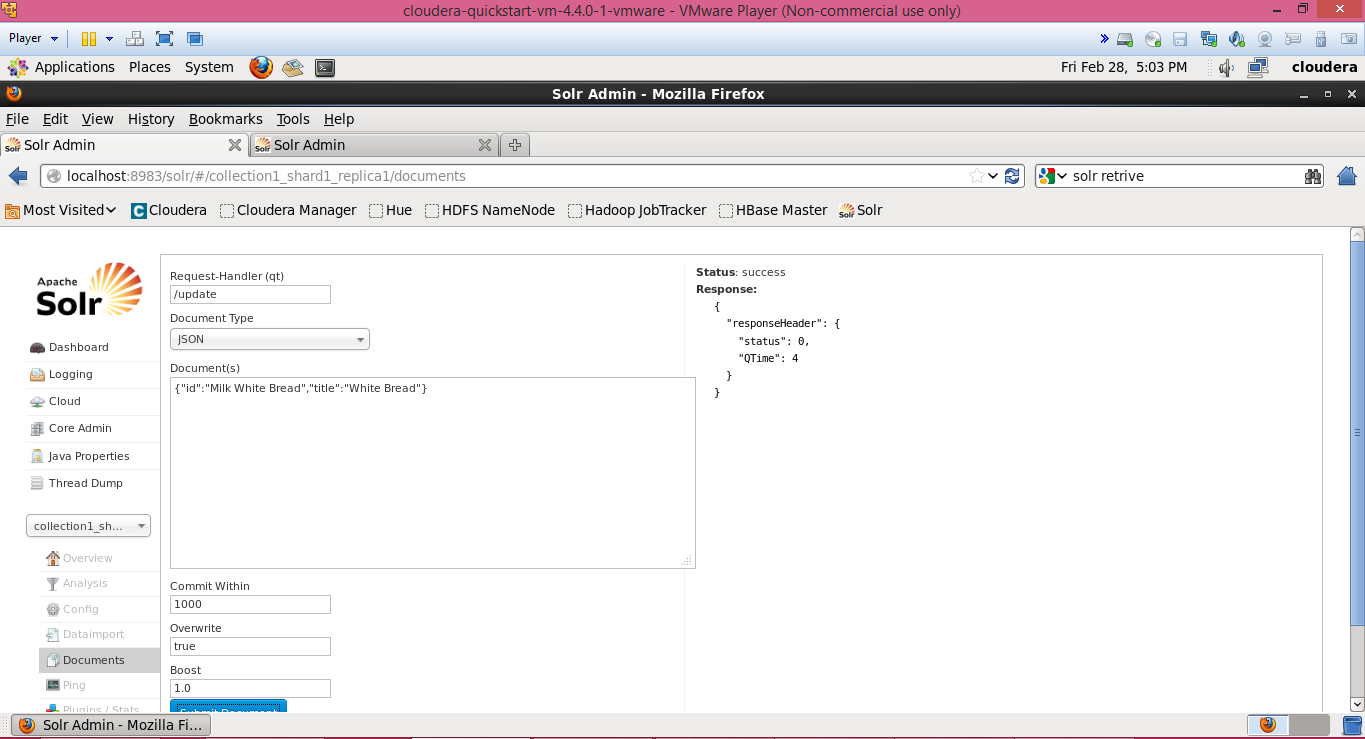
Now establish the restful service and connect to the solr to display this output in the solr.

Push the data to the Solr.

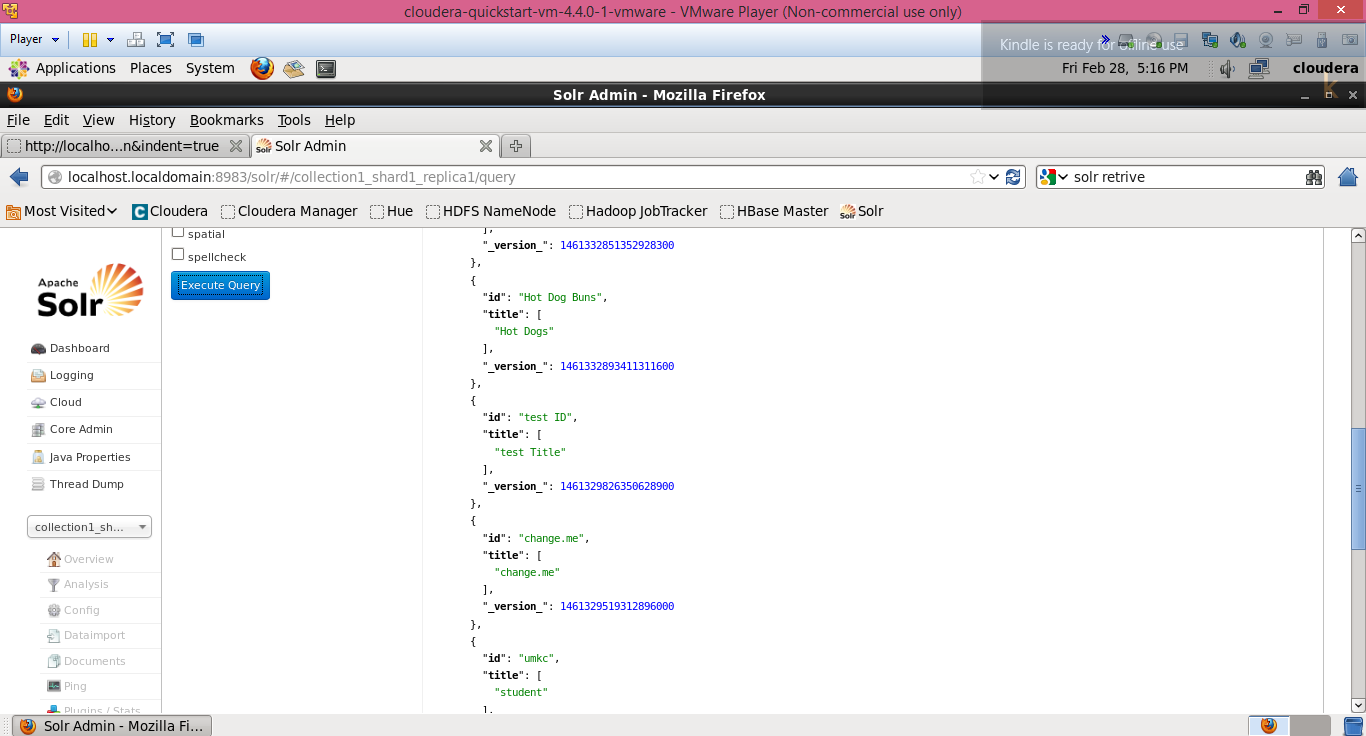
Then it shows the success message after pushing data to the solr. Then based on the query we can extract the data. If we give \* it gives all the data which is inserted as json object.

Here we can see the no of items used and the association what are the combination of items many of users are

The below screen shows the pushing the data into solr.



The below screen shows the all data which is inserted by querying \*

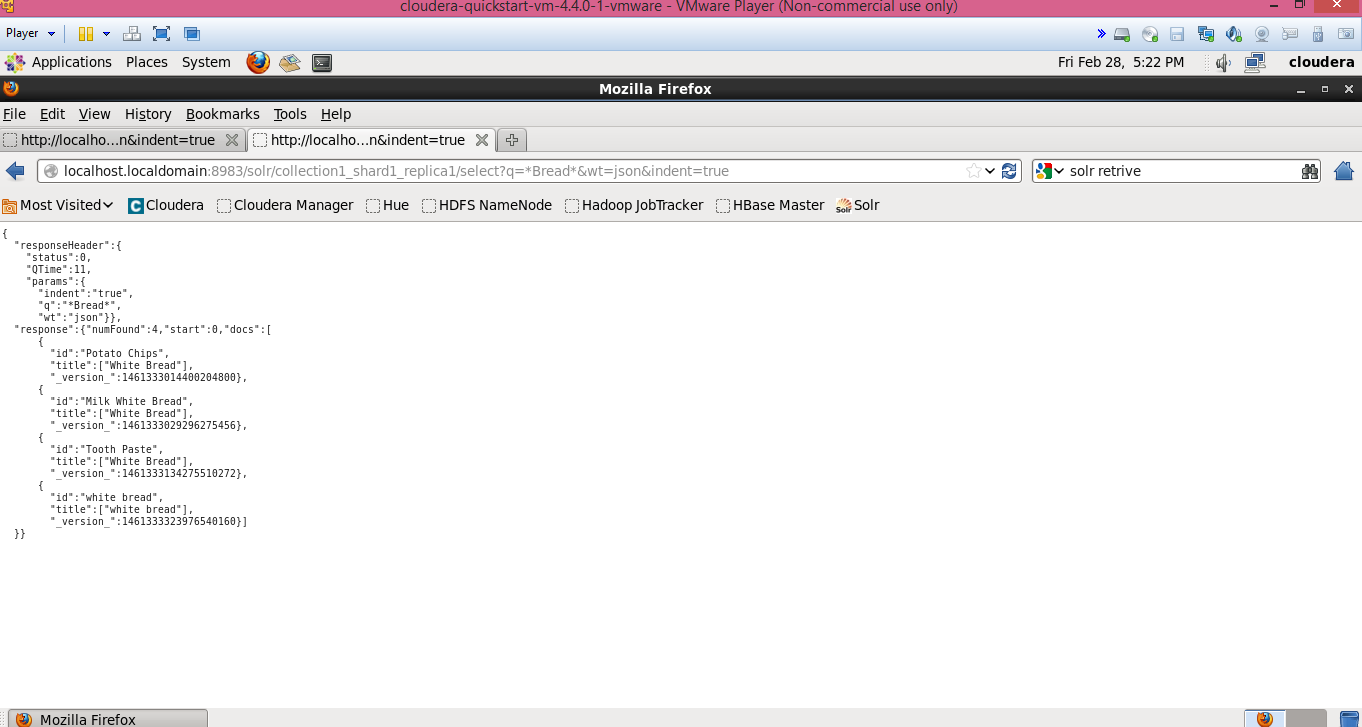


The URI of these data in the solr is:

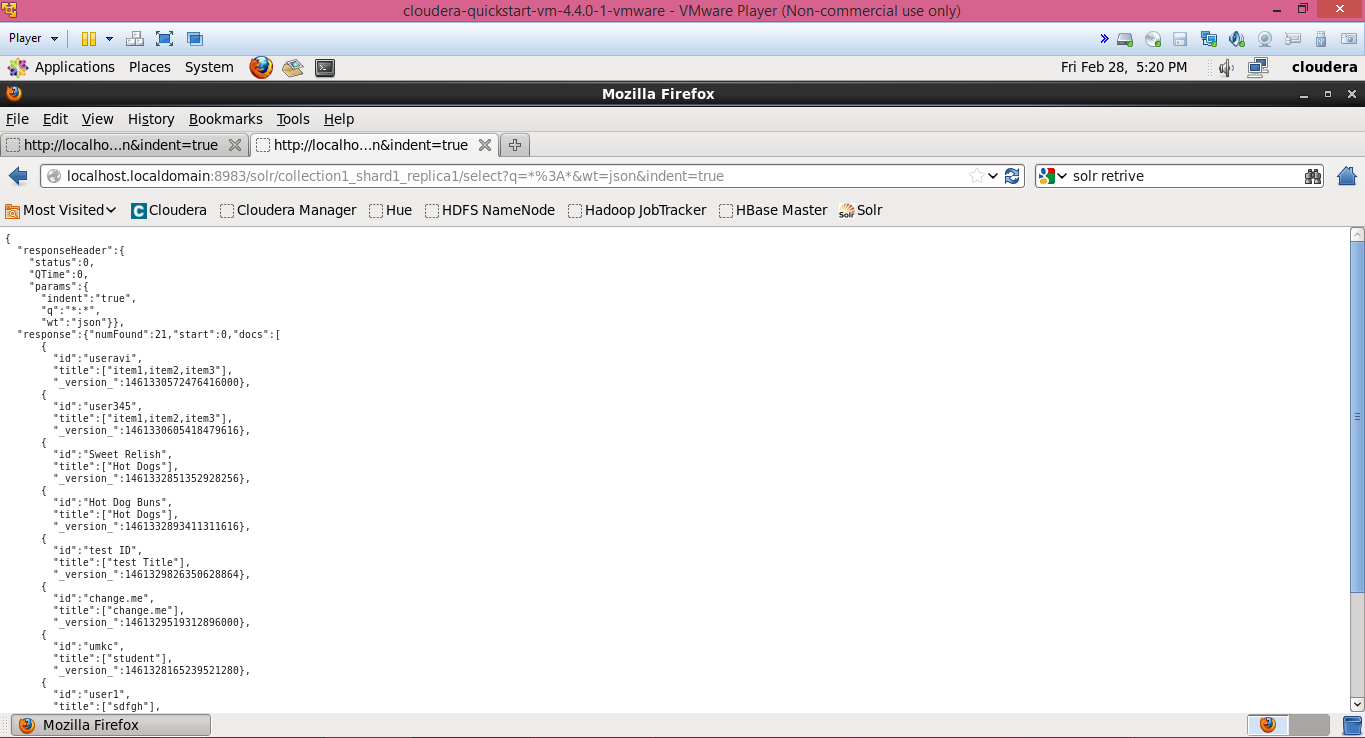
[**http://localhost.localdomain:8983/solr/collection1\_shard1\_replica1/select?q=\*Bread\*&wt=json&indent=true**](http://localhost.localdomain:8983/solr/collection1_shard1_replica1/select?q=*Bread*&wt=json&indent=true)

Where the query is given between \* \*

The below screen shows the data of the single item.

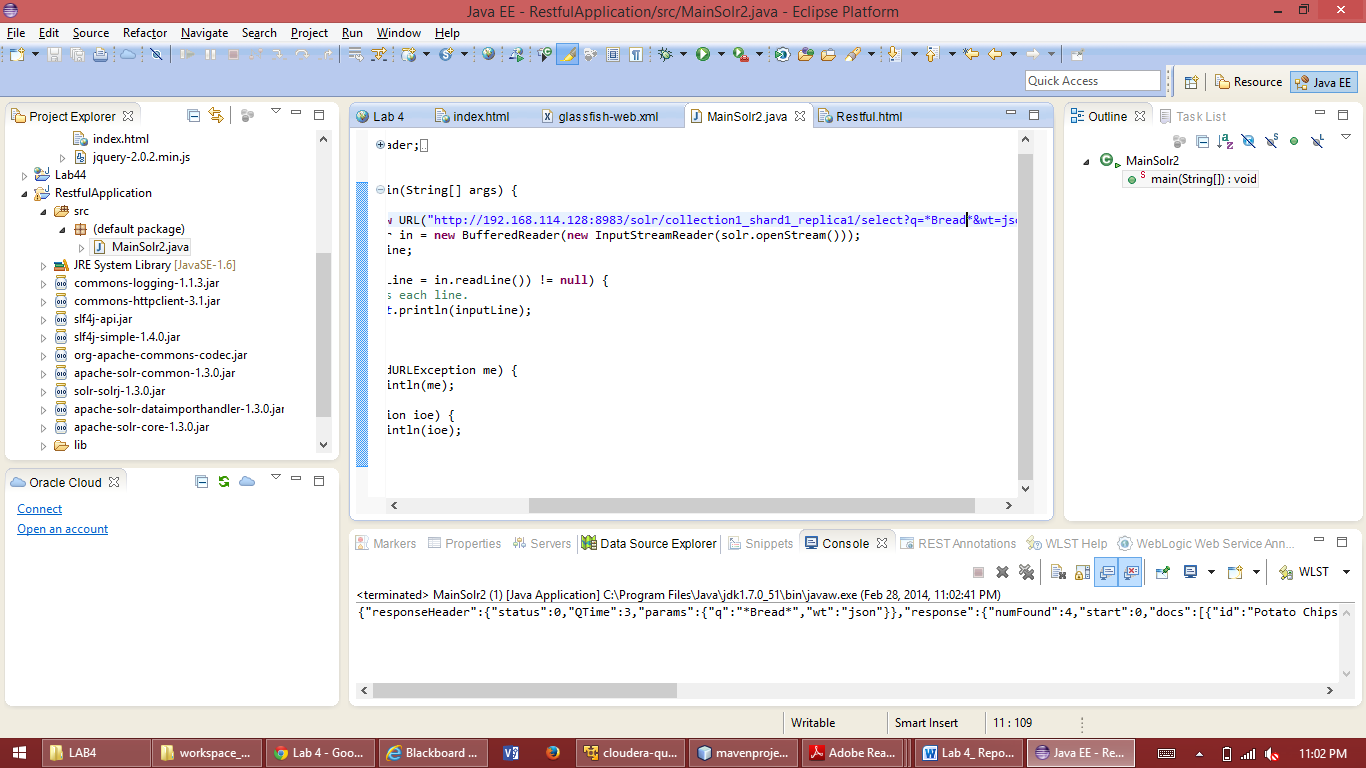


The below screen shows the uri and data of all items.



1. **Mobile Client Application :**

Create a Restful service to access data from the solr. For Restful service we can call the URL in java code. This URL access the solr connection and gets the data given.



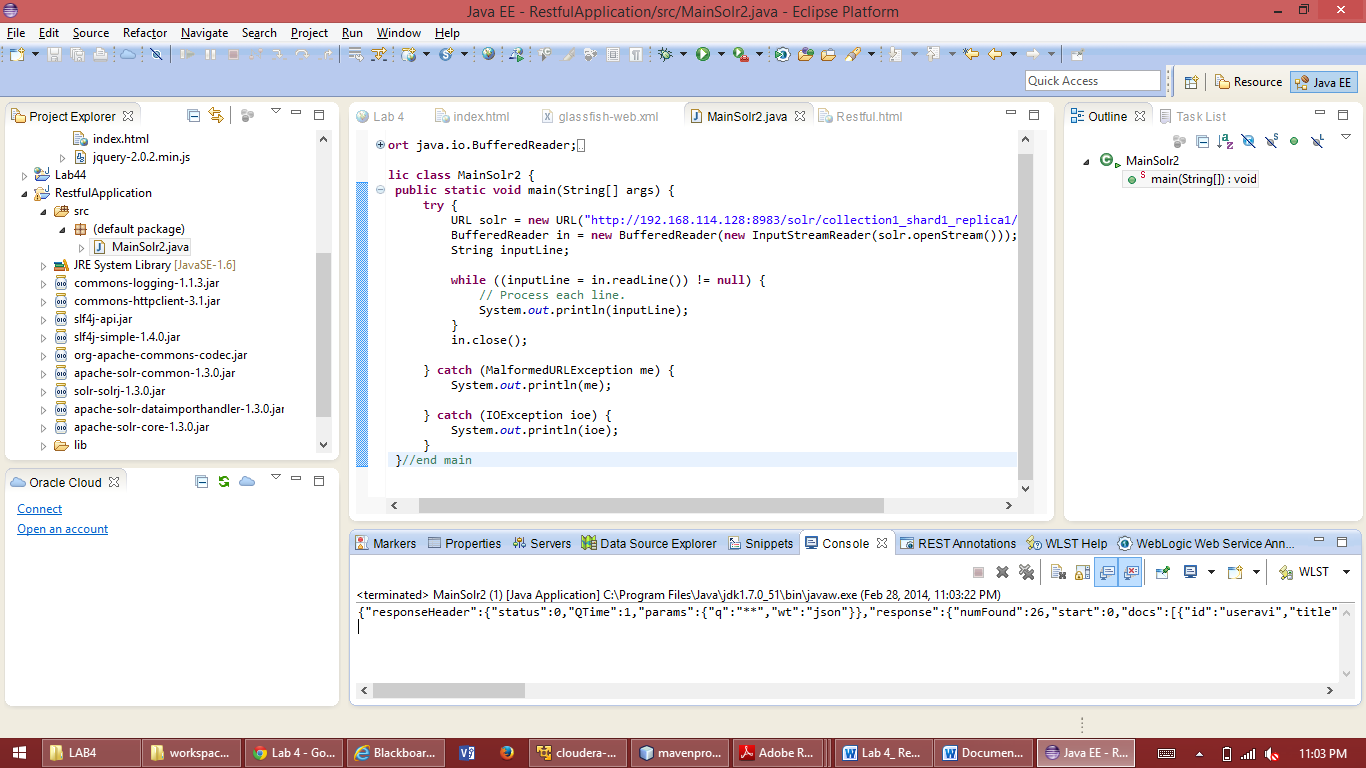
Here in the above screen the URl gets access to the solr and displays the output in the console. Here if we get give particular item it displays only the number of that particular items.

If we press \* it gives the whole items and their count as output.

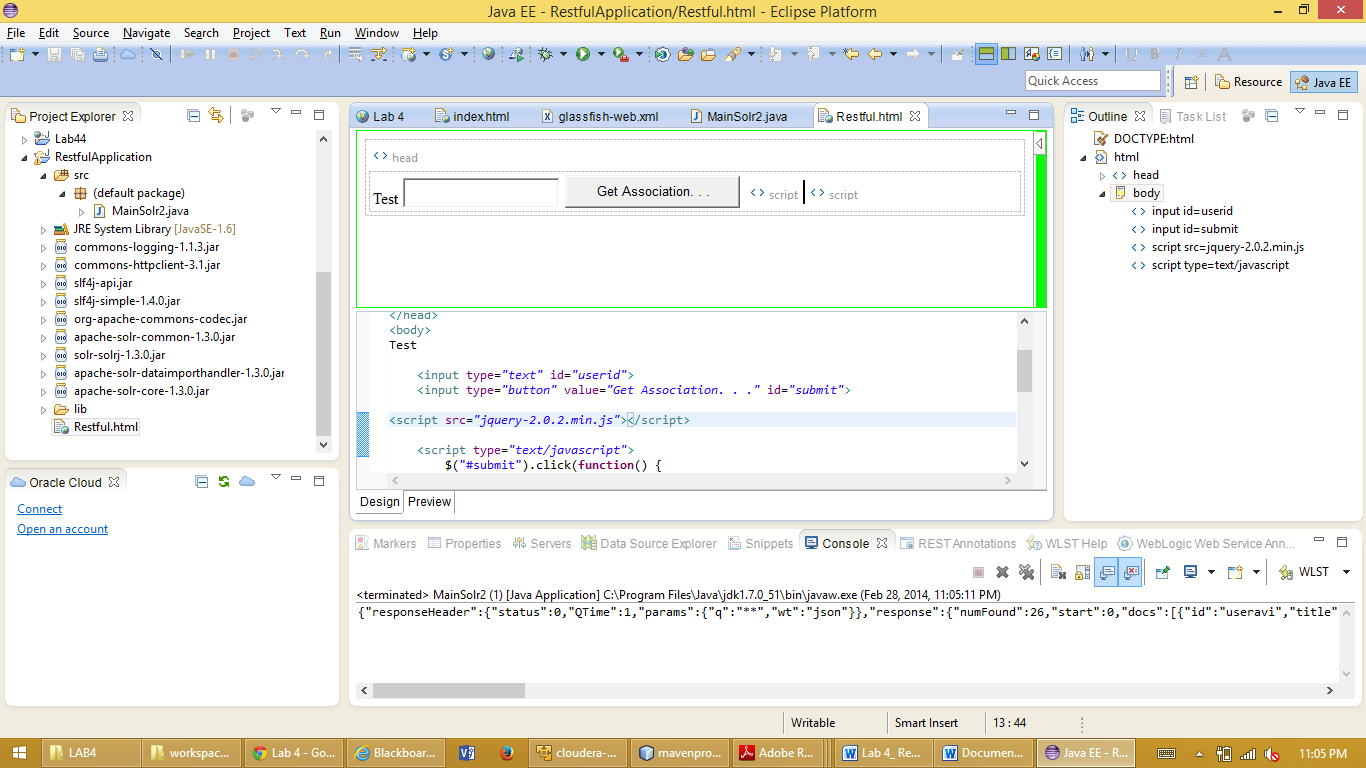
If we need all items then in the console it displays as

{"id":"Sweet Relish","title":["Hot Dogs"],"\_version\_":1461332851352928256},{"id":"Hot Dog Buns","title":["Hot Dogs"],"\_version\_":1461332893411311616},{"id":"testID","title":["test Title"],"\_version\_":1461329826350628864},{"id":"change.me","title":["change.me"],"\_version\_":1461329519312896000},{"id":"umkc","title":["student"],"\_version\_":1461328165239521280},{"id":"Potato Chips","title":["White Bread"],"\_version\_":1461333014400204800},{"id":"Milk White Bread","title":["White Bread"],"\_version\_":1461333029296275456},{"id":"ToothPaste","title":["White Bread"],"\_version\_":1461333134275510272}]}}

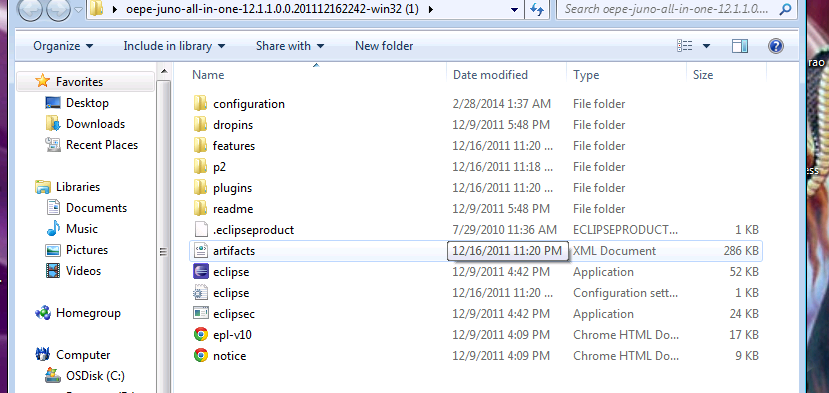
The Below screen displays all the Items in the Json



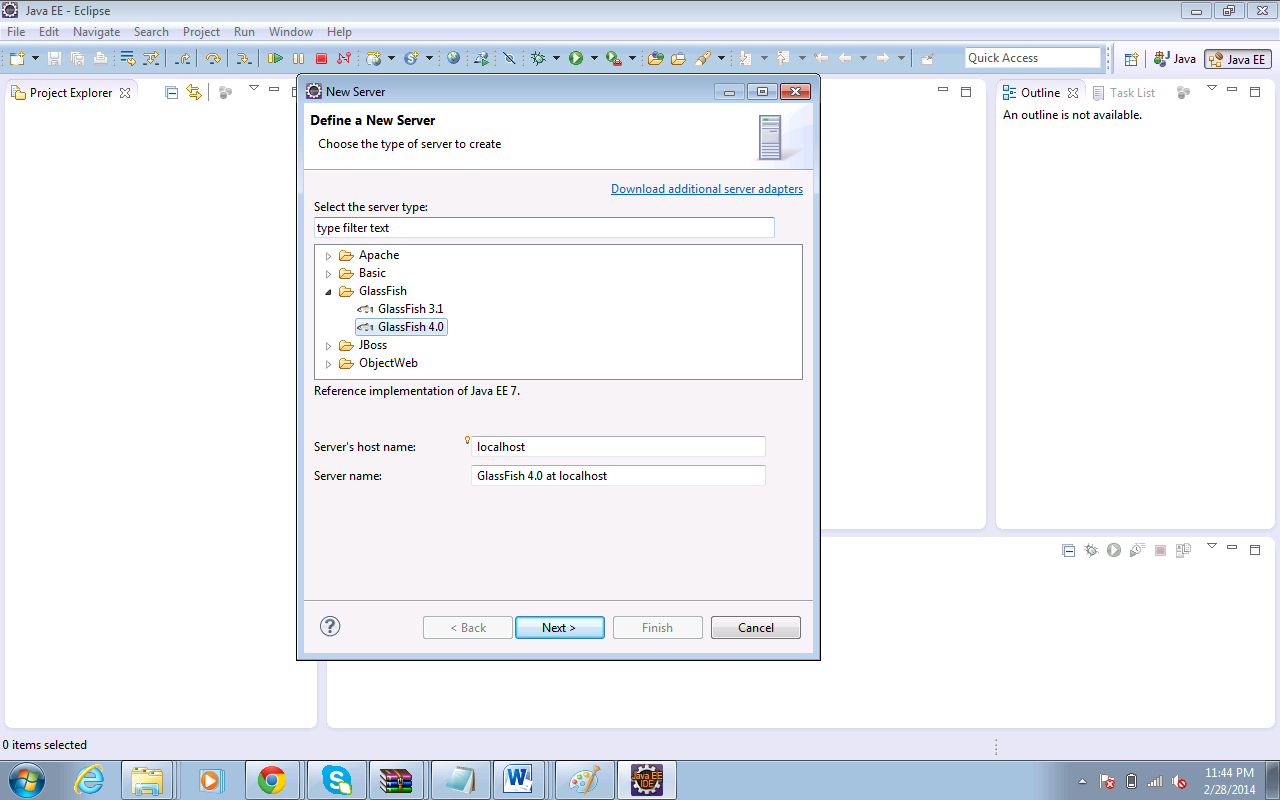
The below screen shows the Getting Association through URL from app



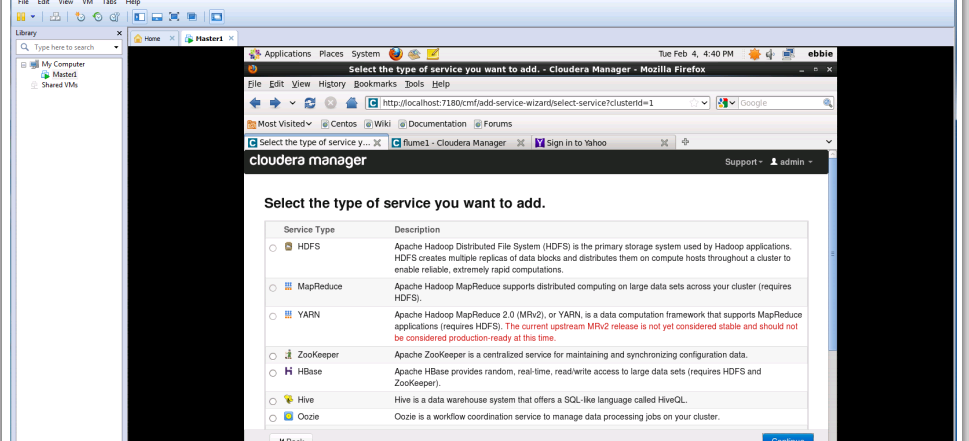
Installed OEPE Juno in my local machine



Created glassfish server



Implemented solr and hdfs services in cloudera manager



2. Mobile Application: Implement a mobile application to use the information through Solr REST Web Services

3. Write a short report on your work (including screenshots).   
4. Post all your work (source code and report) to Lab 4 directory of your GitHub site. And post your GitHub Lab 4 link to the following site