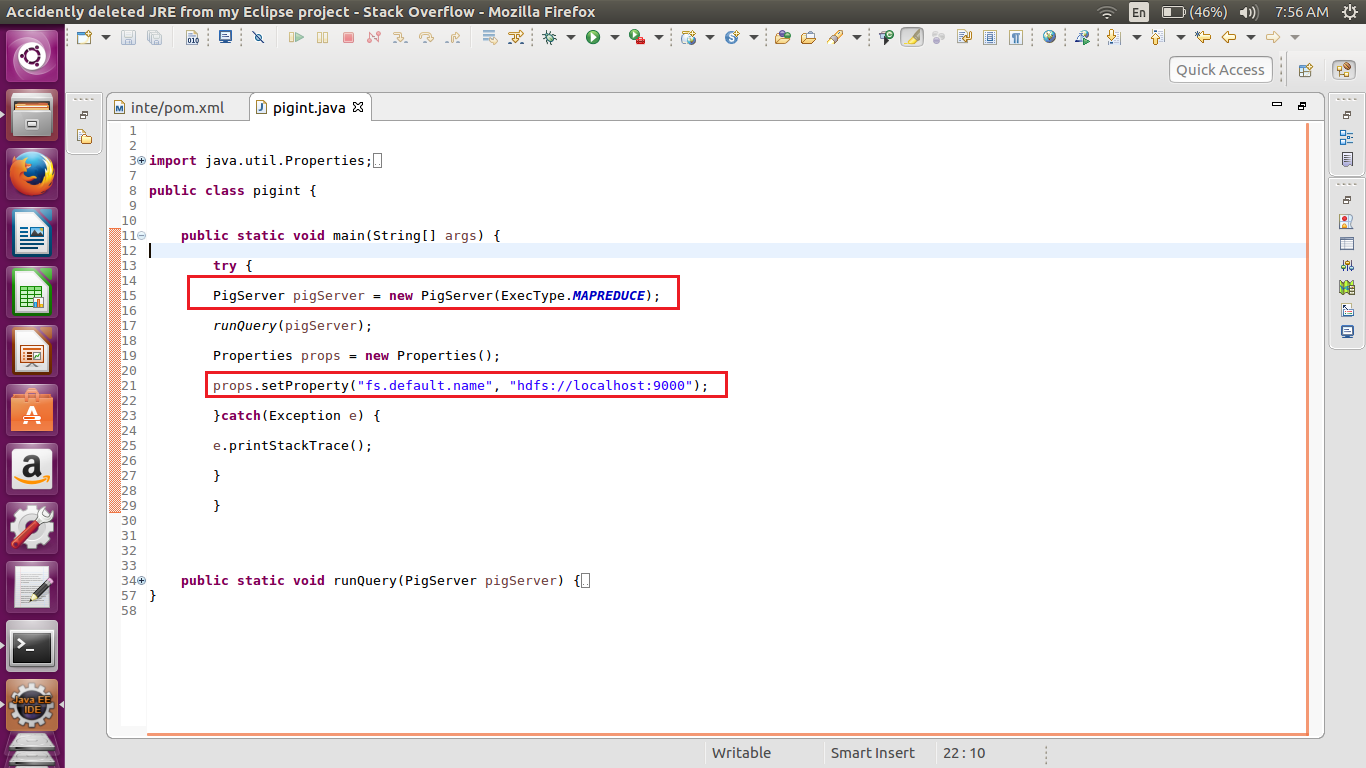
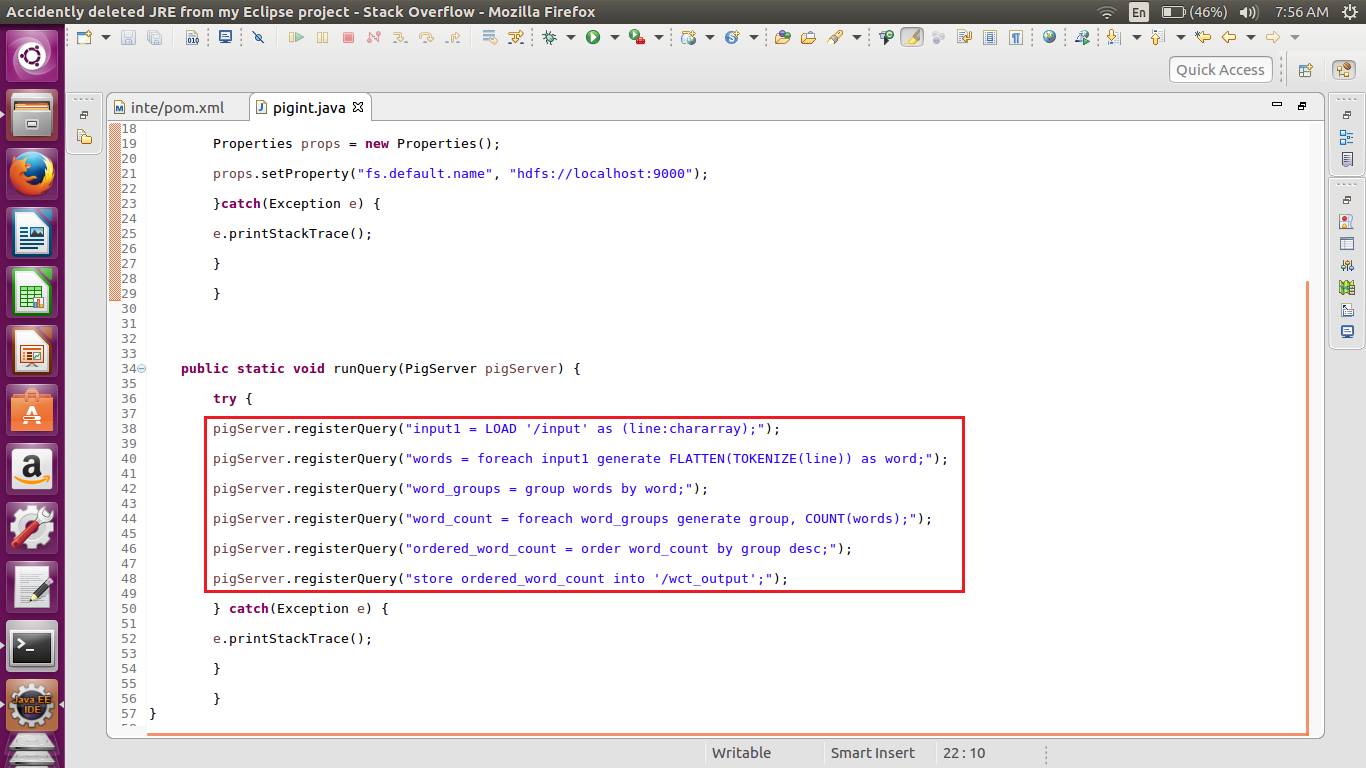
Embed your pig scripts inside a java program by following the steps in the below blog.

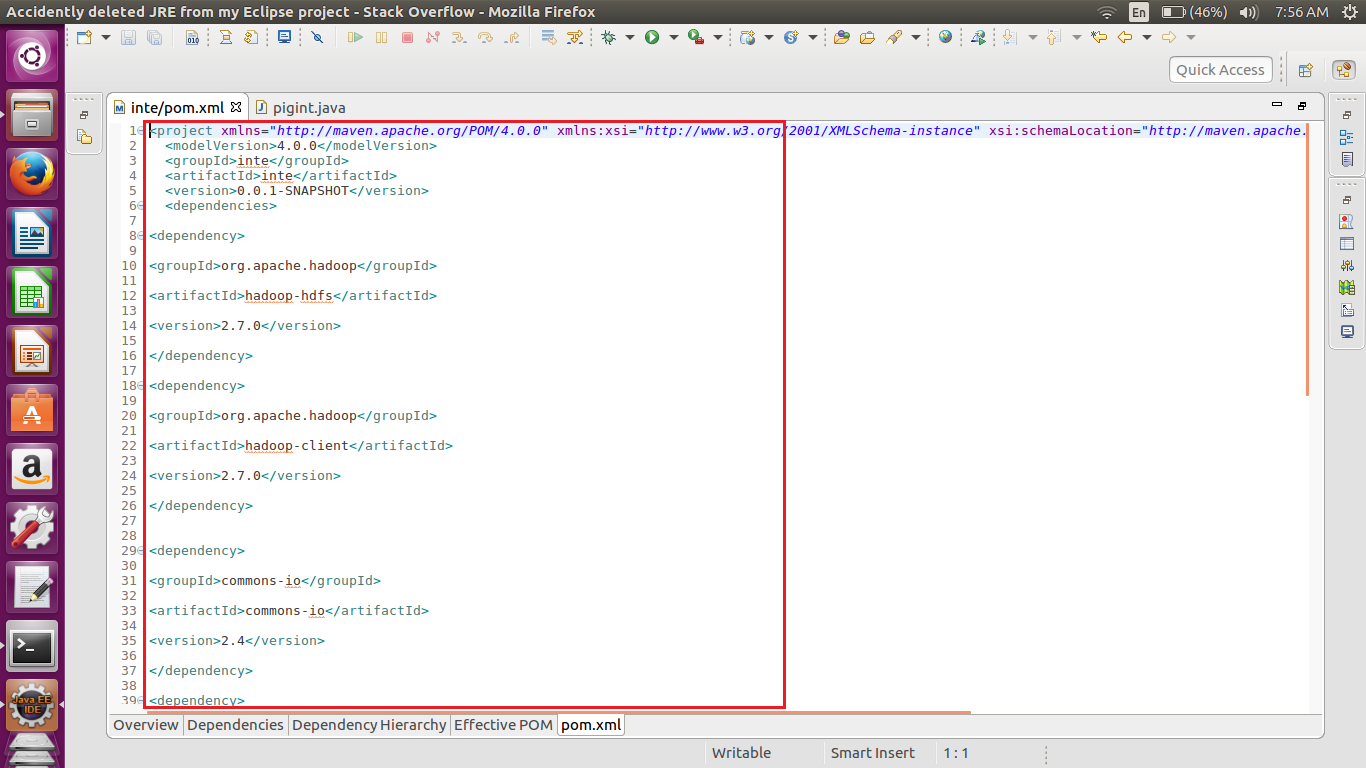




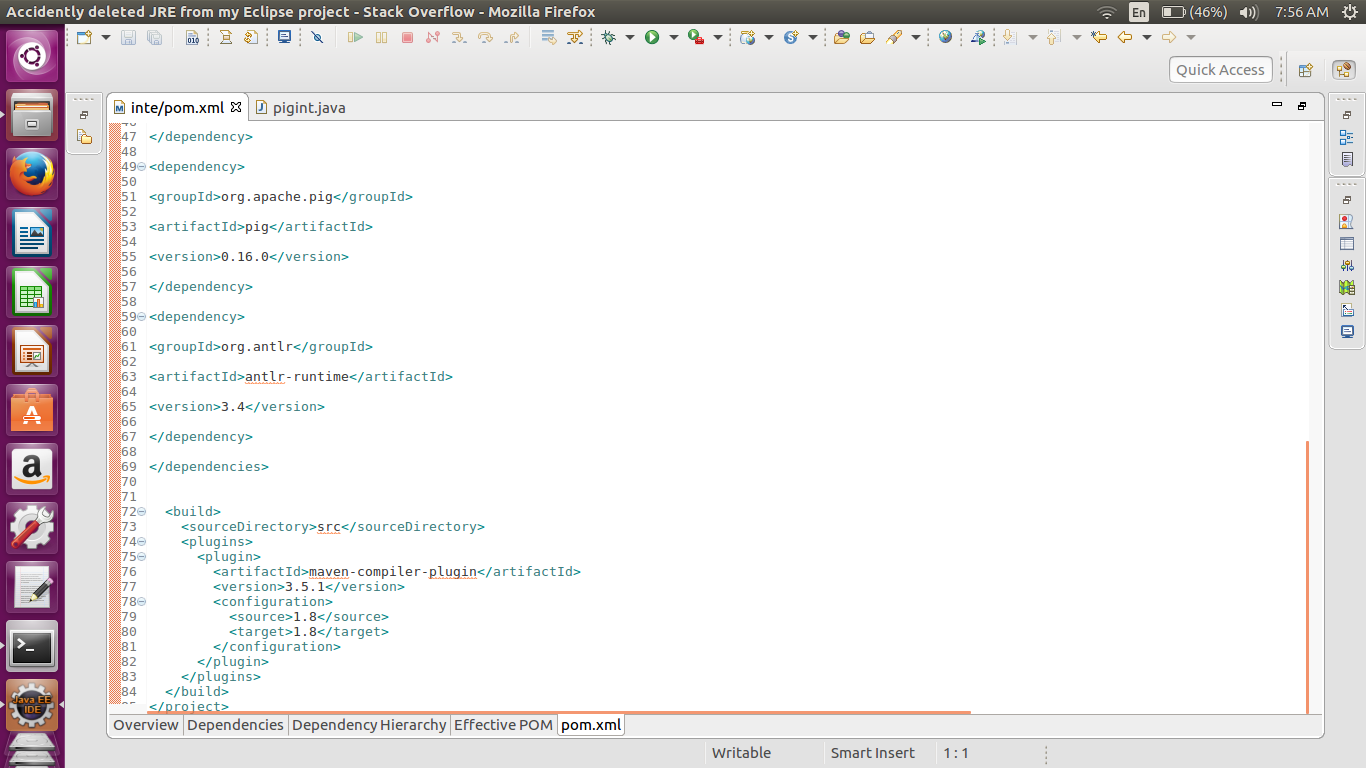
* Pig Server is the main class for embedding Apache Pig as part of your java applications. In this we are executing the pig in the map reduce mode.
* PigServer can take two additional parameters while registering your pig script.
* Params: this is a key/value map passed that can be referenced in your pigscript as $key.
* ParamFiles: takes in filenames that contain the parameters.
* To include a Pig Latin script, we need to use pigServer.registerQuery method provided by the PigServer.
* Using the Properties class we set the path for hdfs . the above code we are using the port number to connect ot hdfs.
* We can similarly use pigServer.registerJar to add a jar for execution of the pig latin script.

For running pig script we have to convert the java project into maven project.

Right Click on the Project–>Configure–>Convert to Maven Project.



We are adding the below dependencies into the maven pom.xml file.



Actual dependencies add in the pom.xml file.

In the dependencies we have to correctly mention the Hadoop and the pig versions. Else the version conflict may happen.

----------------------------------------------------------------------------------------------------------------------------------------------------------------

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>inte</groupId>

<artifactId>inte</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-hdfs</artifactId>

<version>2.7.0</version>

</dependency>

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-client</artifactId>

<version>2.7.0</version>

</dependency>

<dependency>

<groupId>commons-io</groupId>

<artifactId>commons-io</artifactId>

<version>2.4</version>

</dependency>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.16</version>

</dependency>

<dependency>

<groupId>org.apache.pig</groupId>

<artifactId>pig</artifactId>

<version>0.16.0</version>

</dependency>

<dependency>

<groupId>org.antlr</groupId>

<artifactId>antlr-runtime</artifactId>

<version>3.4</version>

</dependency>

</dependencies>

<build>

<sourceDirectory>src</sourceDirectory>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.5.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

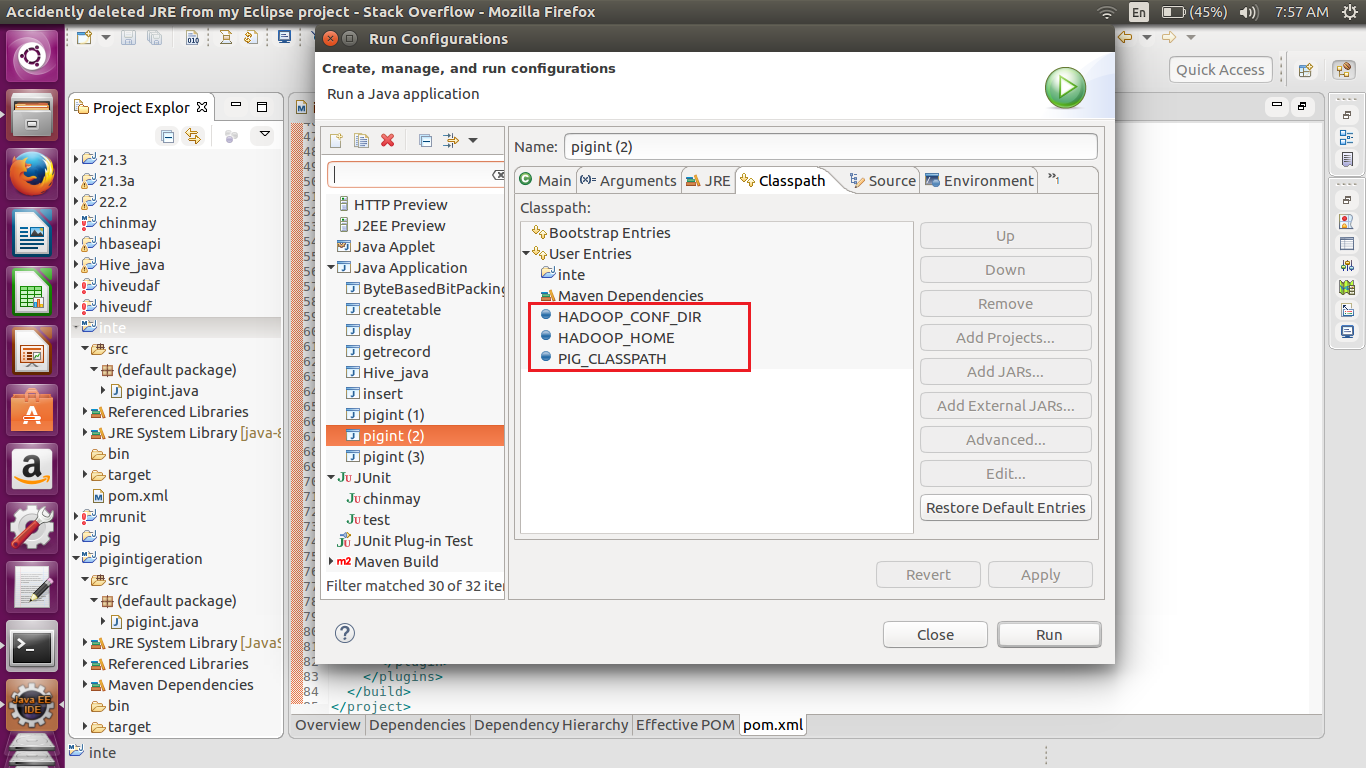
</plugin>

</plugins>

</build>

</project>

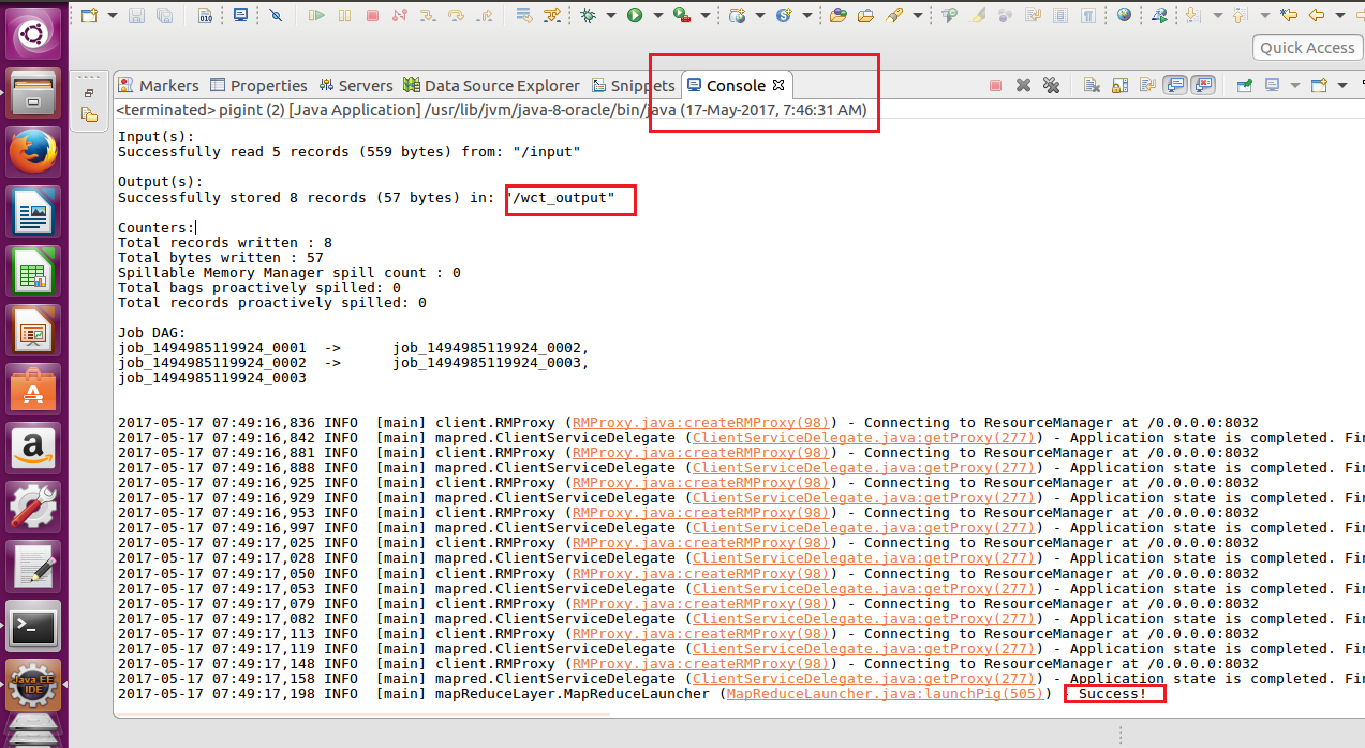
----------------------------------------------------------------------------------------------------------------------------------------------------------------



We also have add the class path in order to execute the Hadoop with different version.

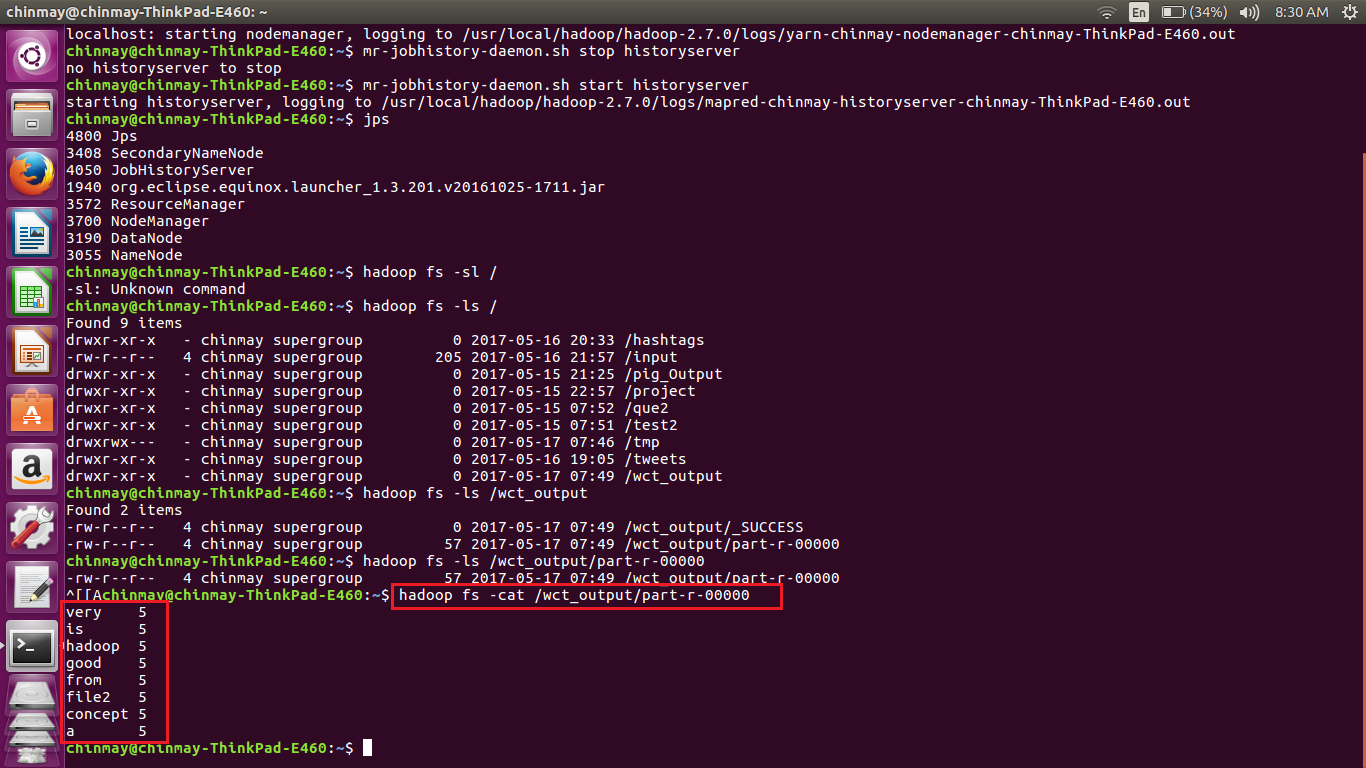
By default the pig executes with the Hadoop 0.20 so change it to 2.7.0 we have to mention the Hadoop home class path.

Also to get the configurations we have to declare the Hadoop configuration class path to get the configurations for the Hadoop.

****

After adding all the necessary jar from the pig directory and executing the project as java application we can run pig script using the java api.

Above we can see the pig job being completed and storing the result in the ‘wct\_output’

****

The input file had the “Hadoop is a good concept from file2 ” repeated five times. So the above gives us the word count as above.

NOTE – ALSO UPLOADED THE POM.XML FILE USED FOR THE MAVEN PROJECT.