Run hive workflow using oozie:

1. Find the oozie examples file directory:

**find / -name "\*oozie\*examples\*.tar.gz"**

1. Copy and extract the directory:

**cp /usr/hdp/2.6.1.0-129/oozie/doc/oozie-examples.tar.gz .**

**tar xzf oozie-examples.tar.gz**

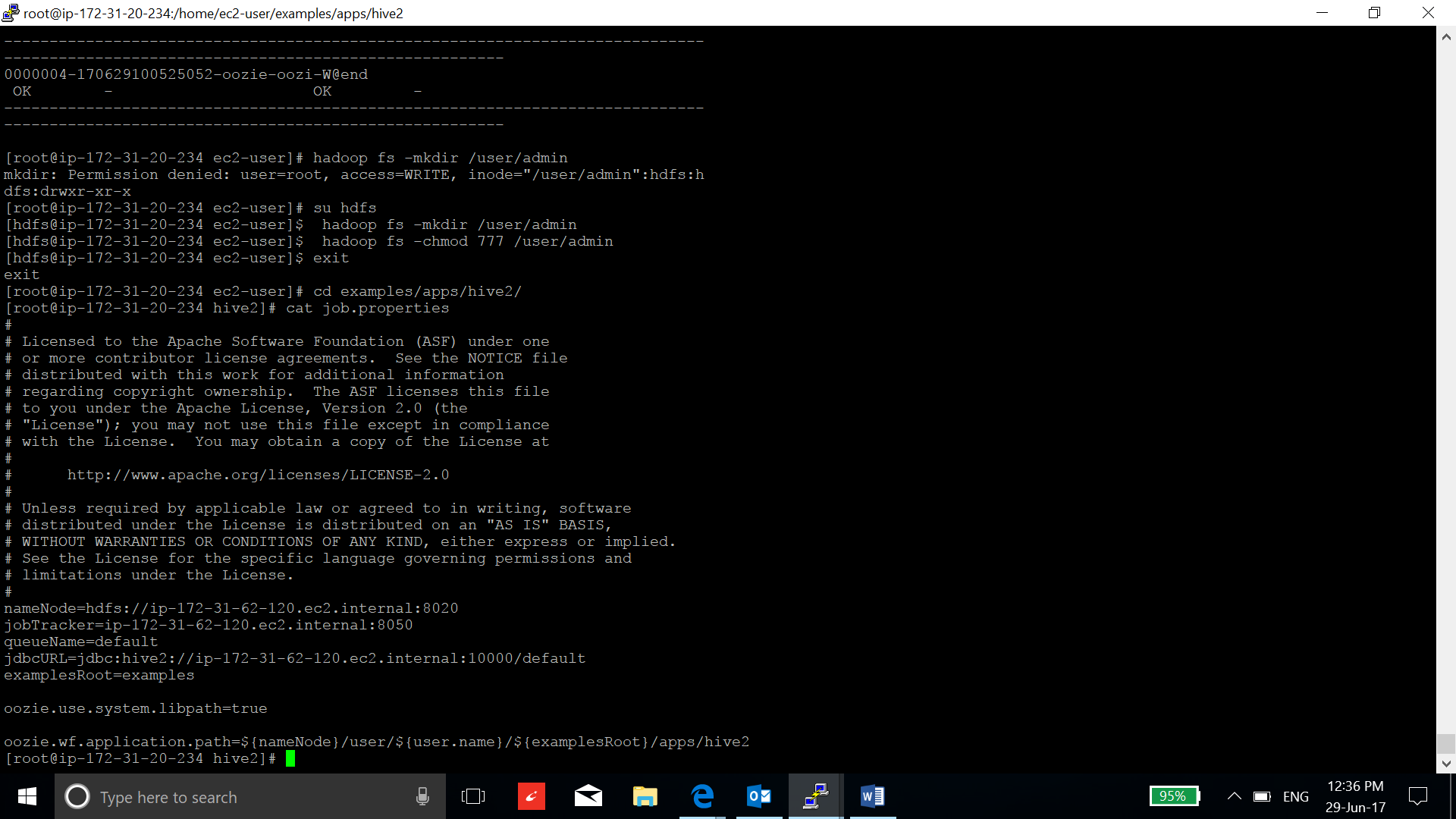
1. Remove the workflow.xml.security file for the demo purpose:

**sudo rm -f /home/ec2-user/examples/apps/hive2/workflow.xml.security**

**sudo rm -f /home/ec2-user/examples/apps/hive2/job.properties.security**

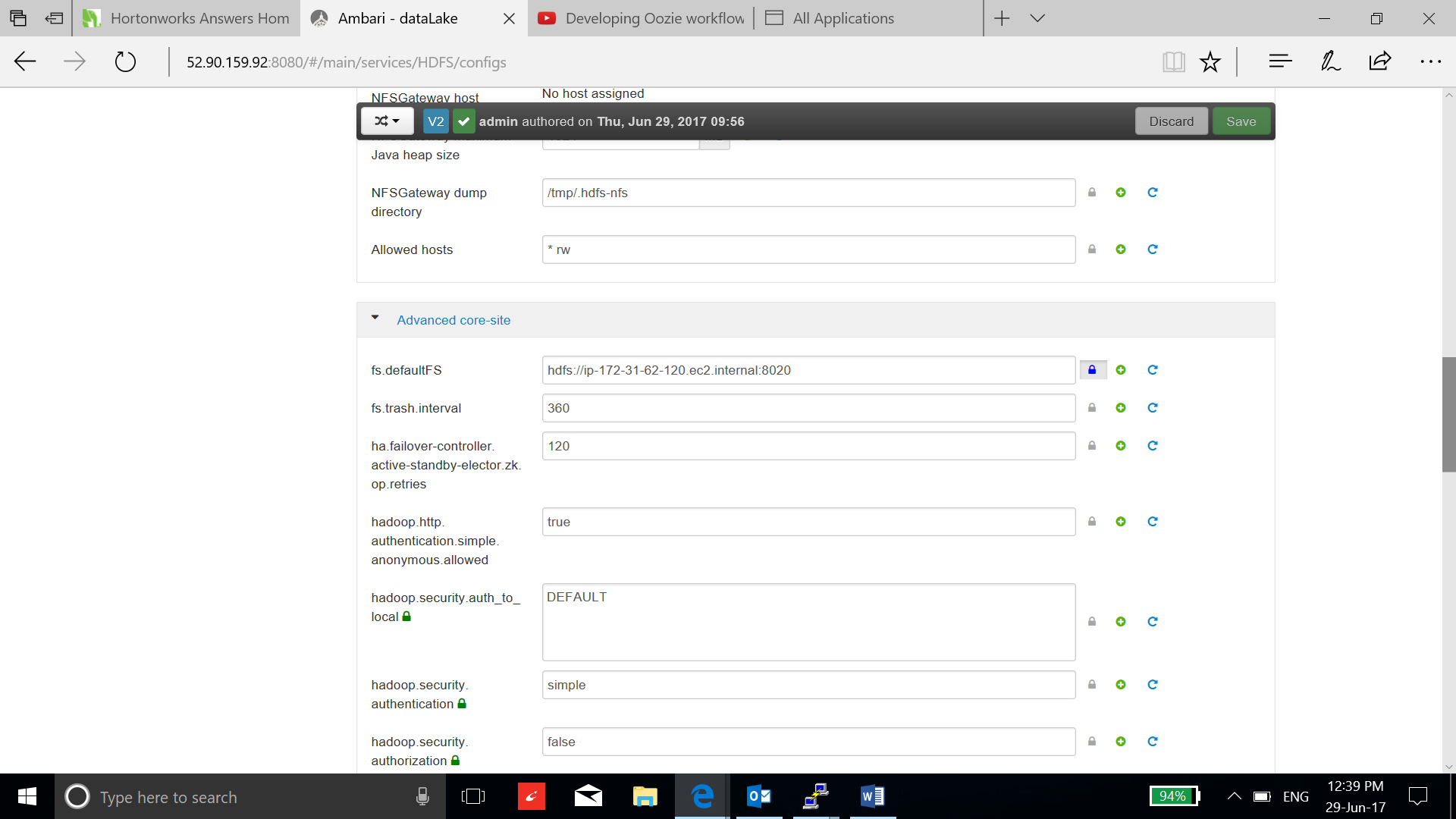
1. Go to the job properties and change the name-node, job-tracker address and jdbcURL.

**vi /home/ec2-user/examples/apps/hive2/job.properties**

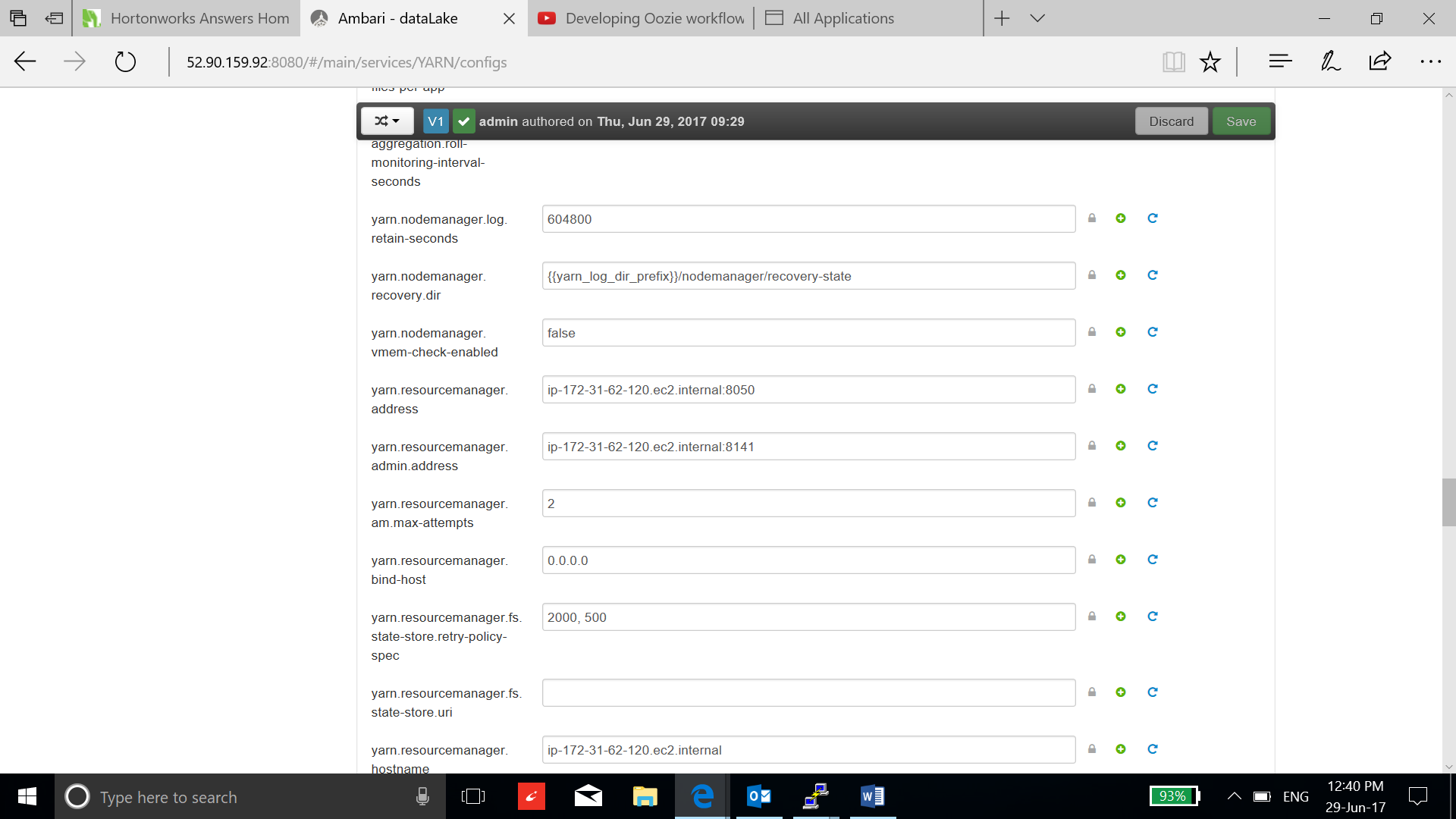


1. Find the namenode address, job tracker address and jdbc hive2 server address:

**vi /etc/hadoop/conf/core-site.xml**



**vi /etc/Hadoop/conf/yarn-site.xml**

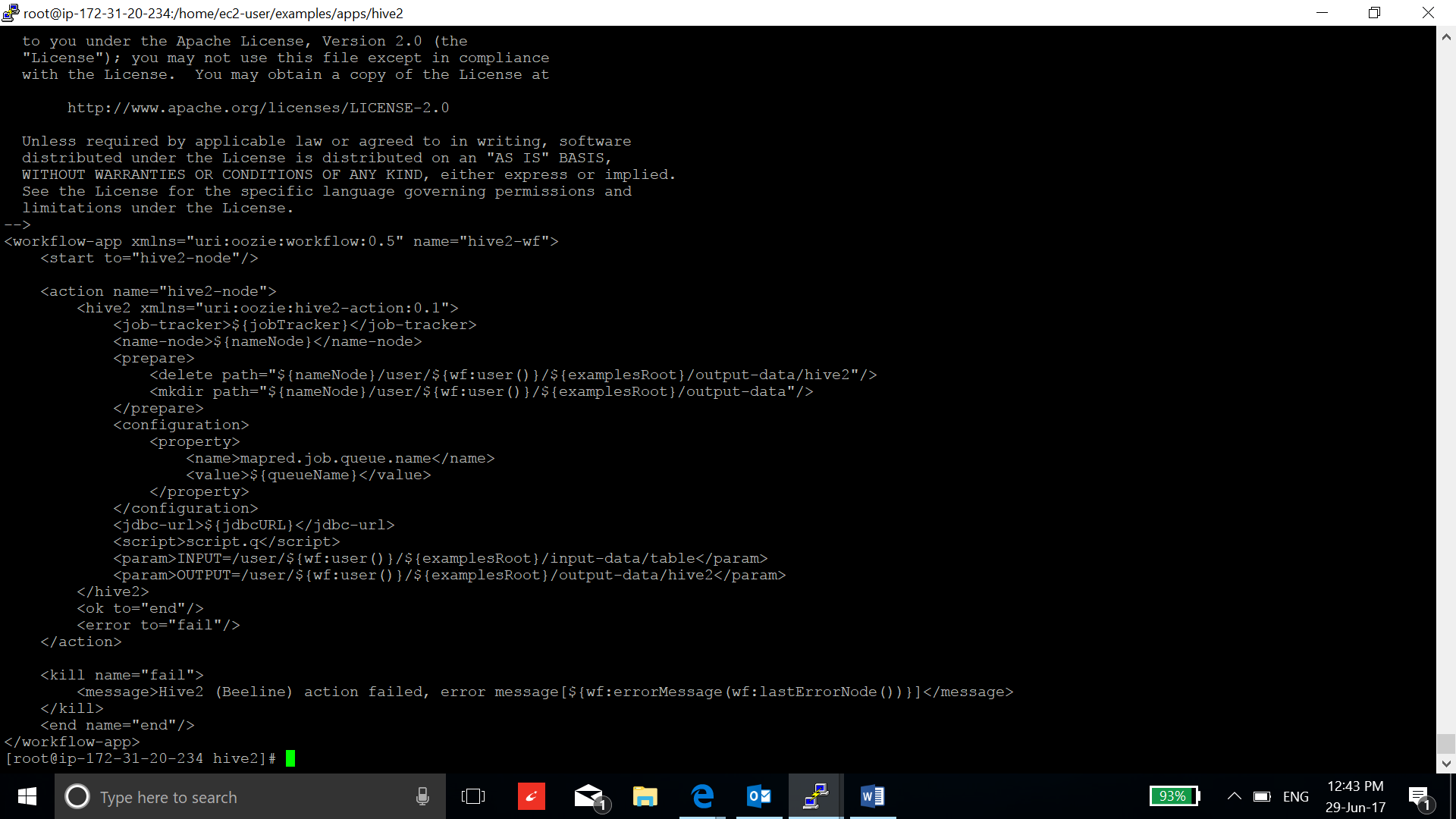


**To get the hive2 server address, go to ambari/hive/config/advanced, search “hiveserver”, It will give you hiveserver2 port number and you can get the ip address by selecting hiveserver2 in summary section.**

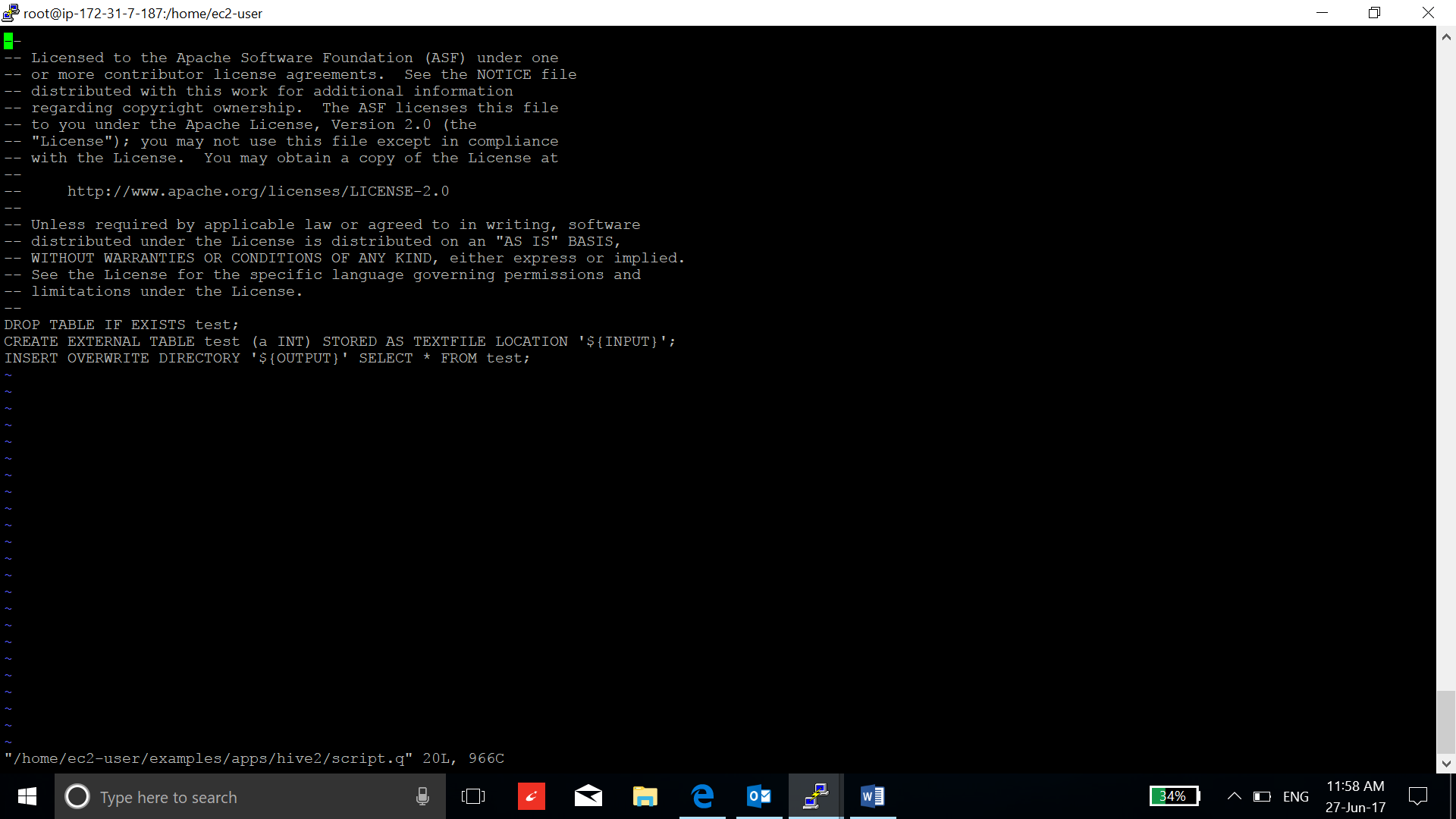
1. Now configure the workflow.xml according to your workflow specification.

General Configurations:

* Change the name tag and but not necessary.
* Change the job tracker and name node address in the job properties.
* Change the property tag to delete the already existing directories if any, because HDFS gives the error if there is already a directory.
* Add properties tag according to the job requirement.
* Change the JDBC url in job properties to connect to the hive2 server.
* Write the hive script name and put it with job properties file.
* Write the input and output path for the data directories.
* Set the end and ok tag according to the requirement.



1. Write the Script:



1. Now transfer the hive2 files to the HDFS’s directory /user/ec2-user/examples/app to run the job:

**su hdfs**

**hadoop fs -mkdir /user/root**

**hadoop fs -put examples /user/root**

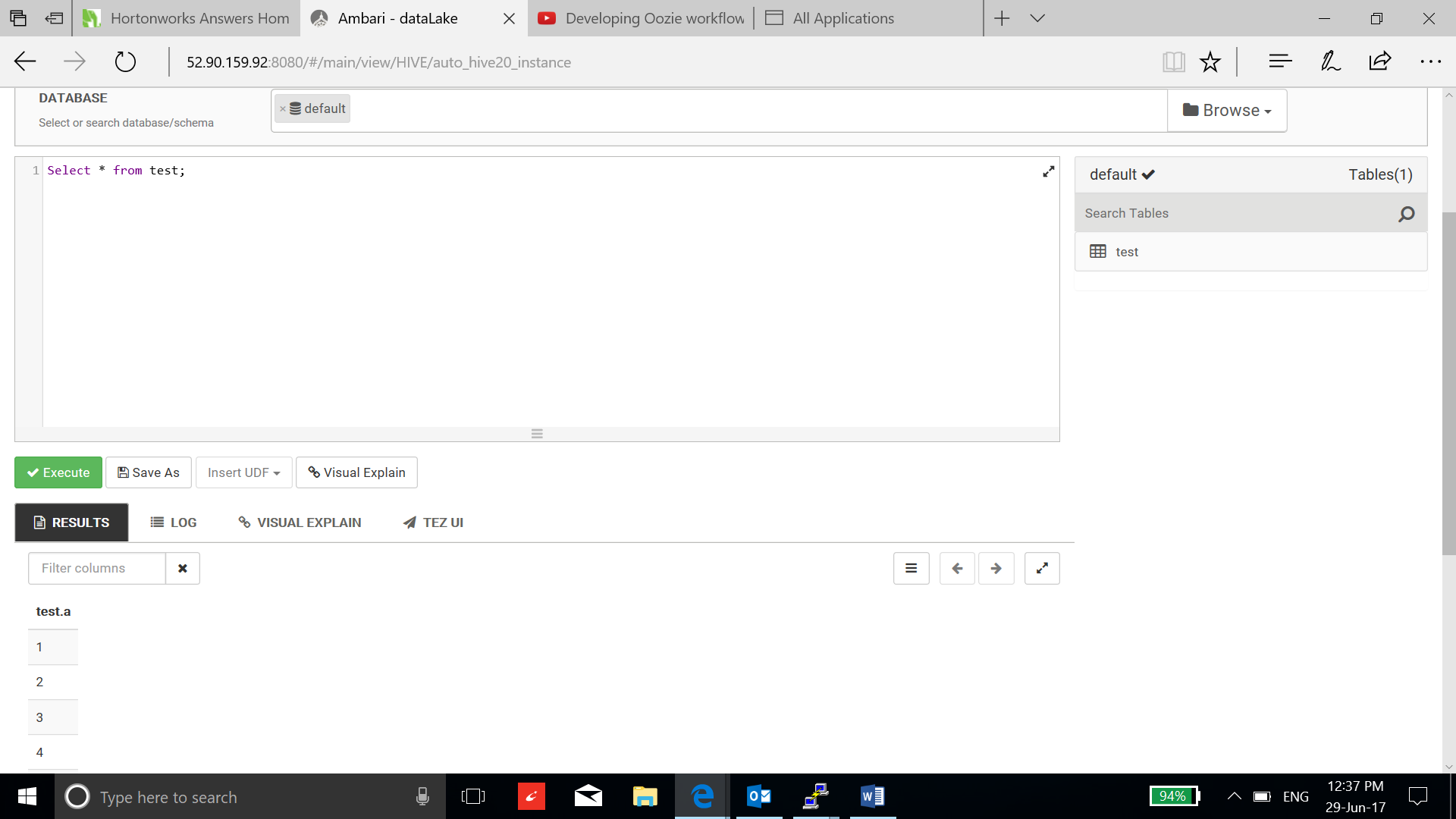
1. Run the workflow:

**oozie job -oozie http://ip-172-31-62-120.ec2.internal:11000/oozie -config /home/ec2-user/examples/apps/hive2/job.properties -run**

Oozie base url can be seen in the config tab of oozie in Ambari.

1. Check the status of the job:

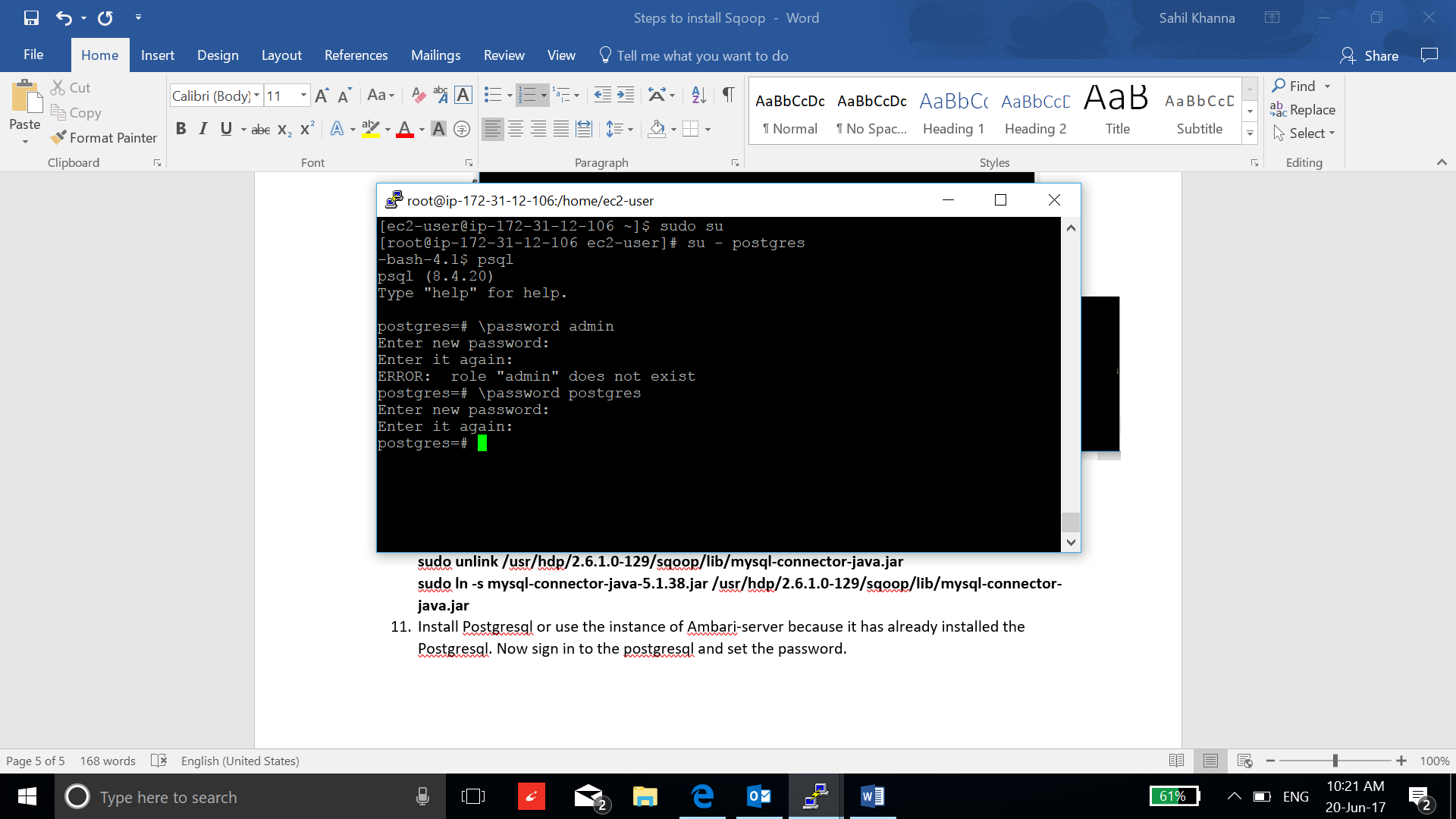
Go to hive2 view in ambari and check the table.



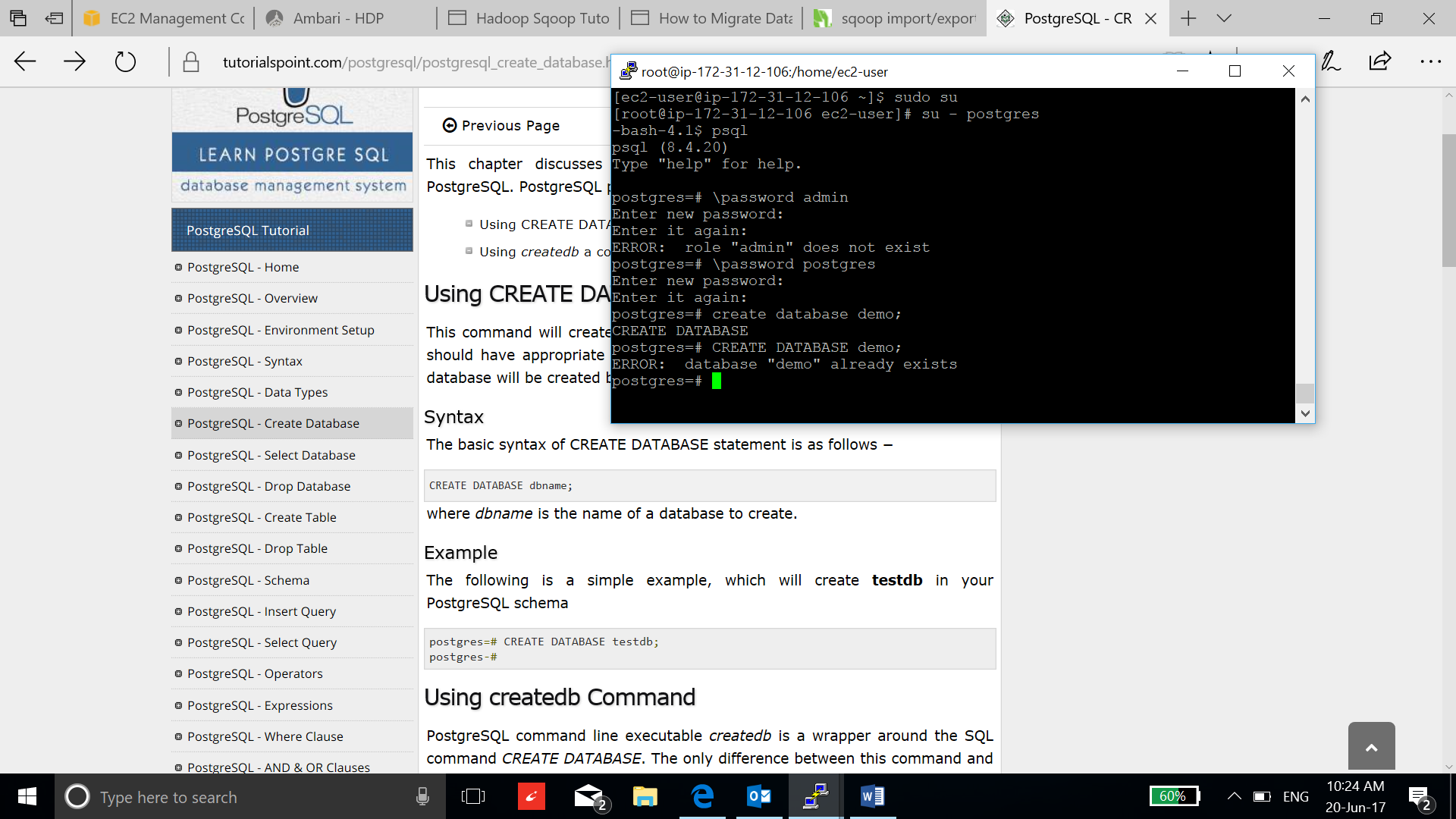
Run Sqoop workflow using Oozie:

1. Make a table in a postgresql to transport in HDFS:

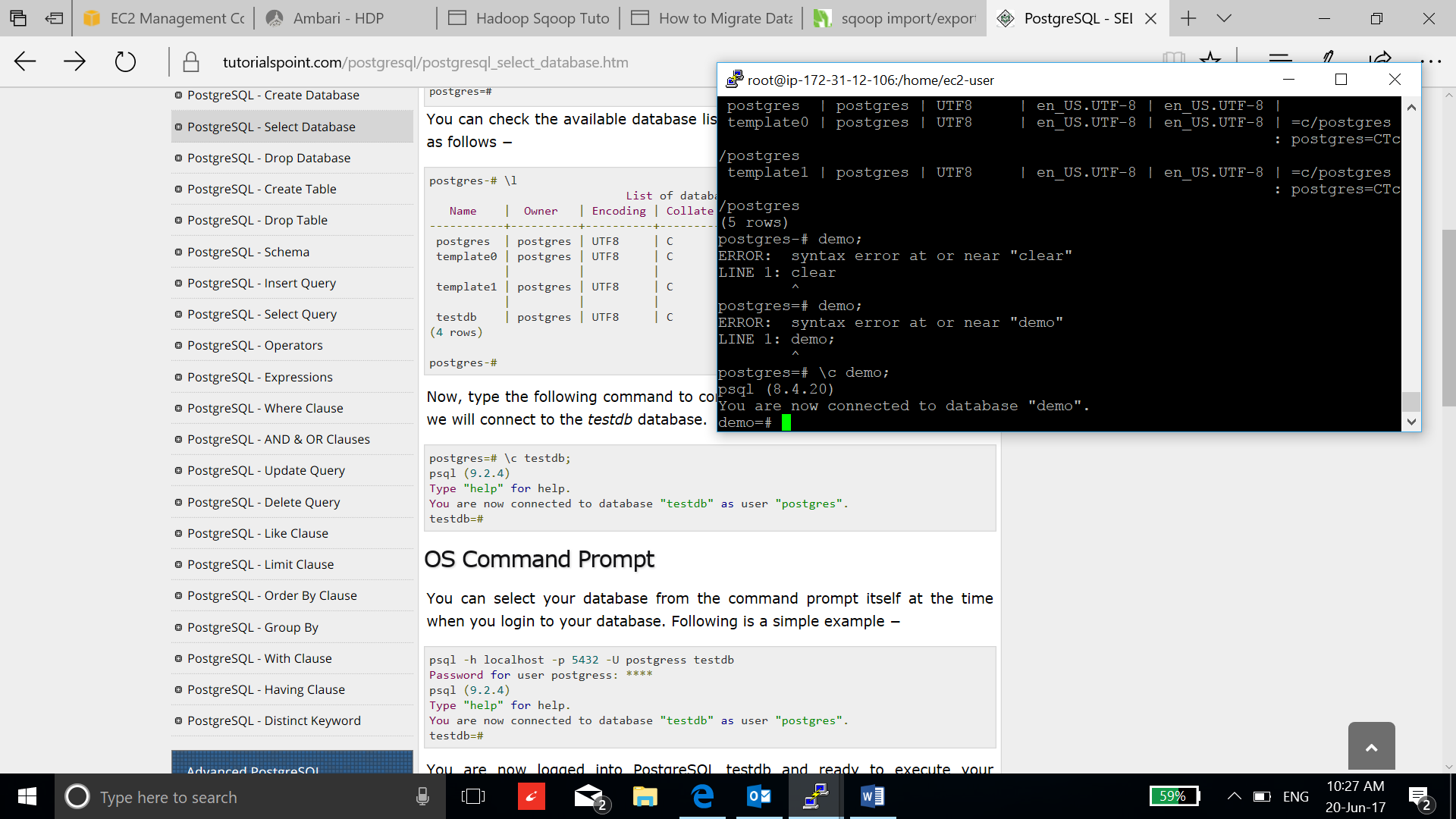
* First sign in to the postgresql and set the password.



* Now create database “demo”:

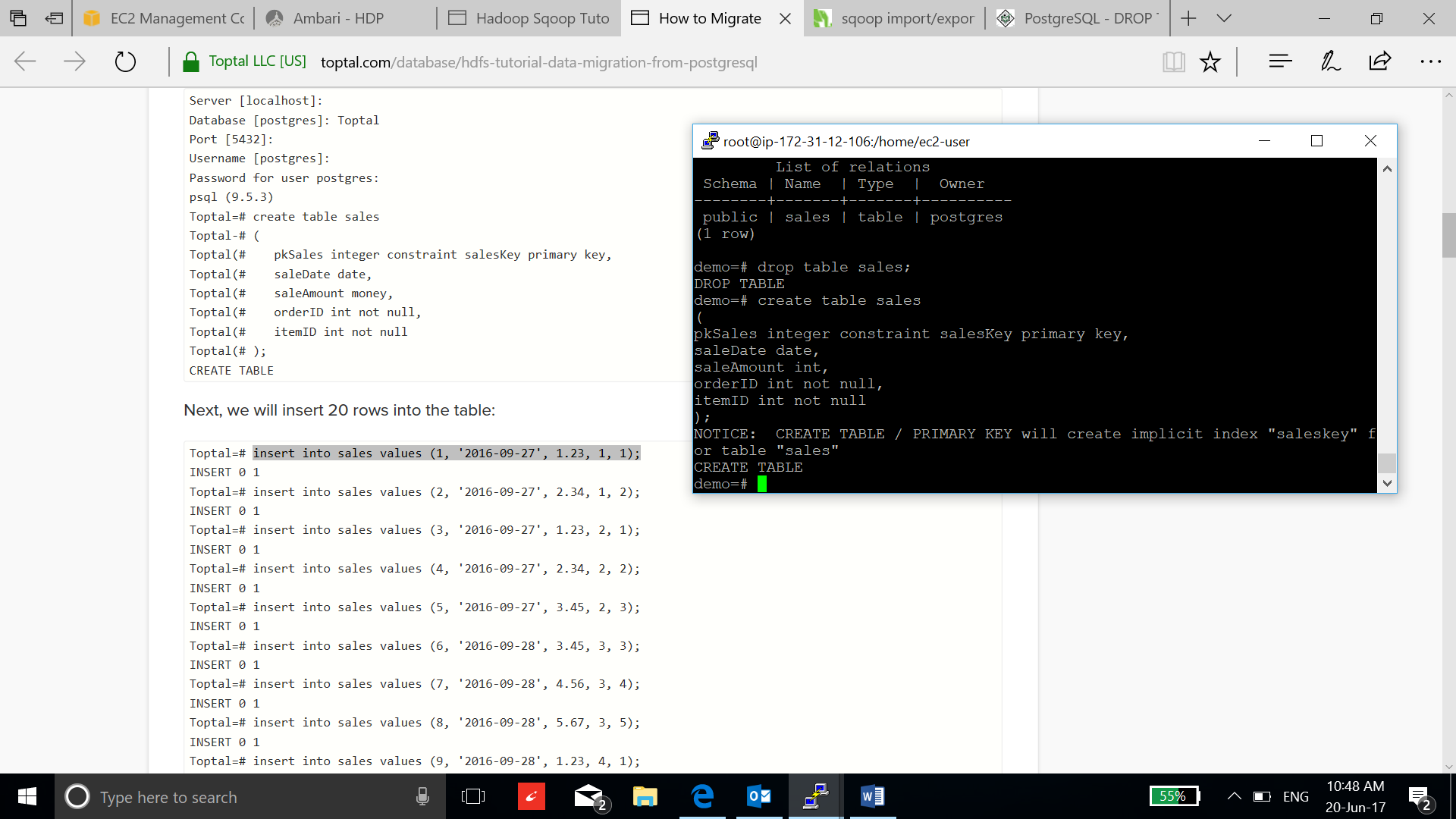


* Connect to the database “demo”:



* Now create a table to transfer HDFS:

**create table sales ( pkSales integer constraint salesKey primary key, saleDate date, saleAmount int, orderID int not null, itemID int not null);**



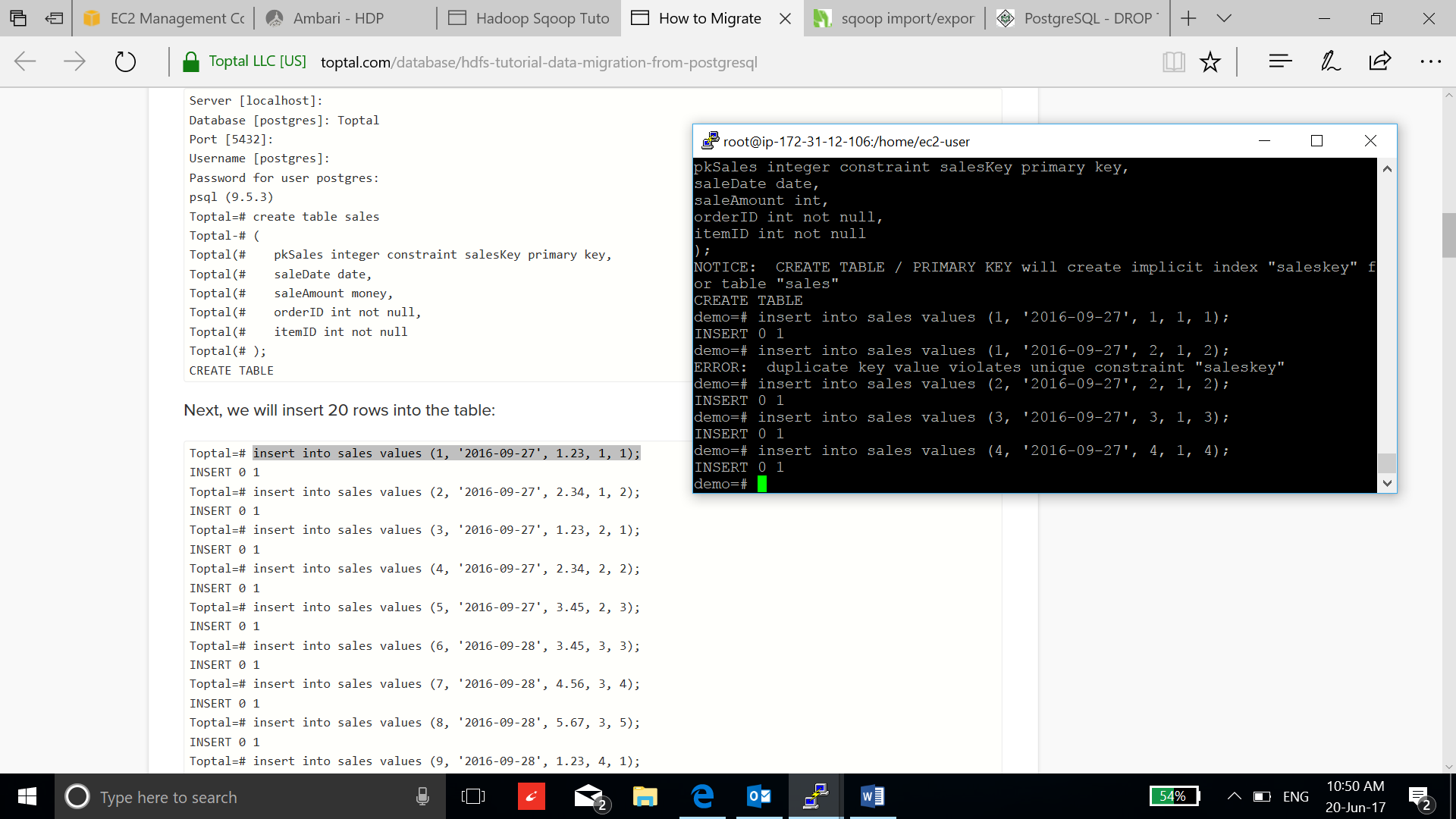
* Insert the data into the table:

**insert into sales values (1, '2016-09-27', 1, 1, 1);**

**insert into sales values (2, '2016-09-27', 2, 1, 2);**

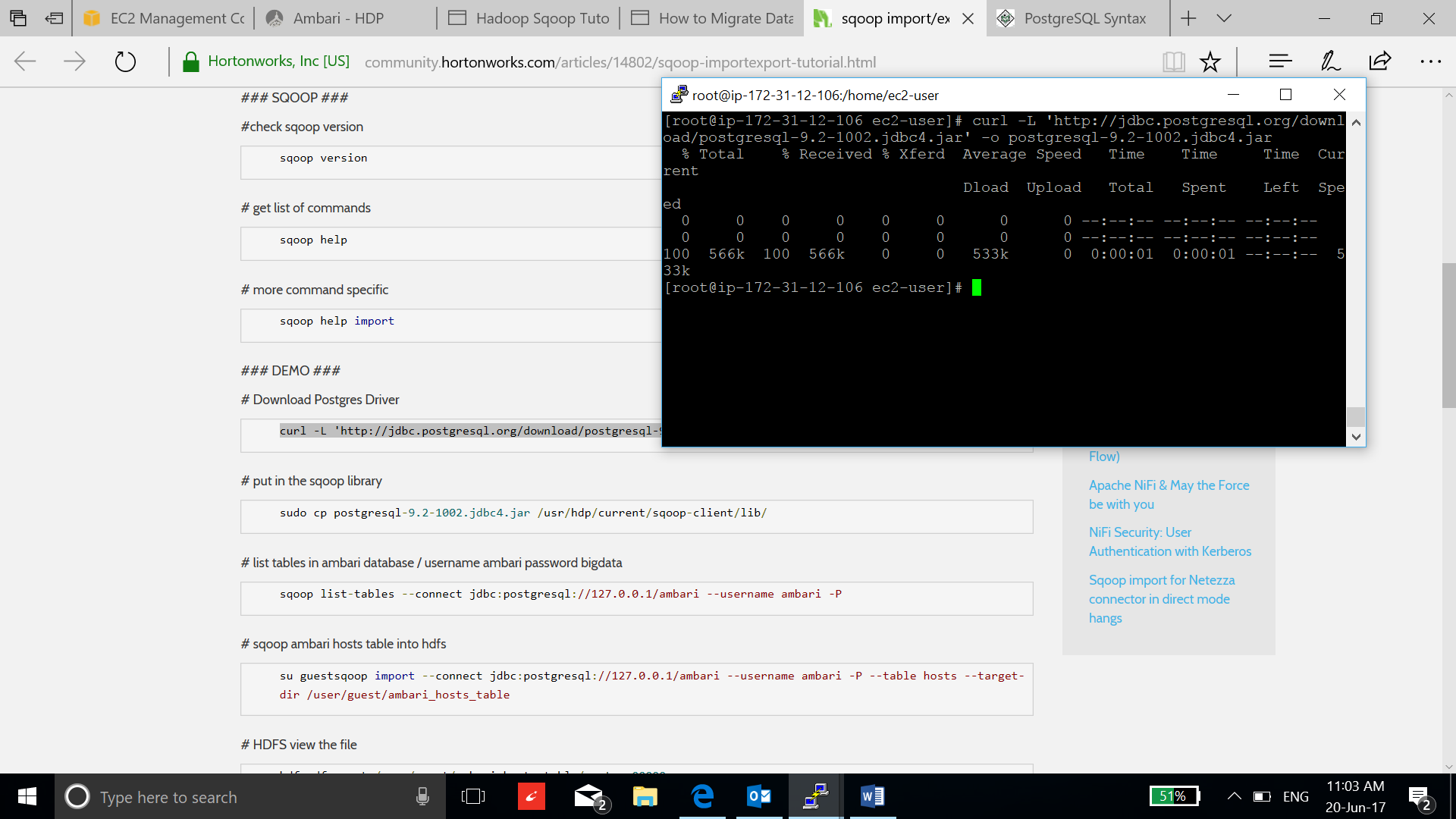
**insert into sales values (3, '2016-09-27', 3, 1, 3);**

**insert into sales values (4, '2016-09-27', 4, 1, 4);**



* Now download the postgres driver:

**curl -L 'http://jdbc.postgresql.org/download/postgresql-9.2-1002.jdbc4.jar' -o postgresql-9.2-1002.jdbc4.jar**



* Copy the drivers in the sqoop library and oozie library in HDFS:

**sudo cp postgresql-9.2-1002.jdbc4.jar /usr/hdp/current/sqoop-client/lib/**

**su hdfs**

**hadoop fs -chown root /user/oozie/share/lib/lib\_20170629100500/sqoop**

**hadoop fs -chown root /user/oozie/share/lib/lib\_20170629100500**

**hadoop fs -chown root /user/oozie/share/lib/**

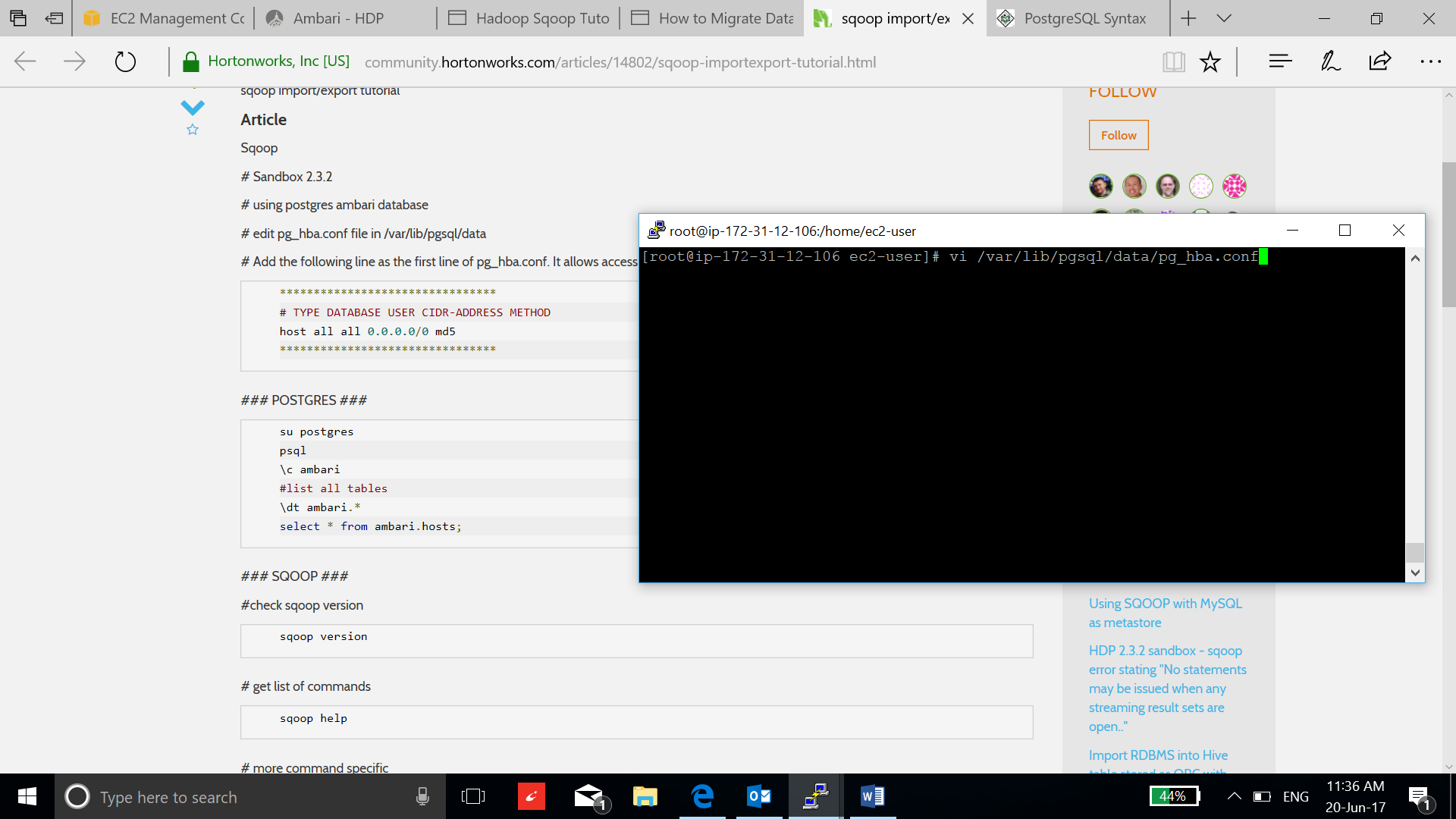
**hadoop fs -chown root /user/oozie/share**

**hadoop fs -chown root /user/oozie**

**exit**

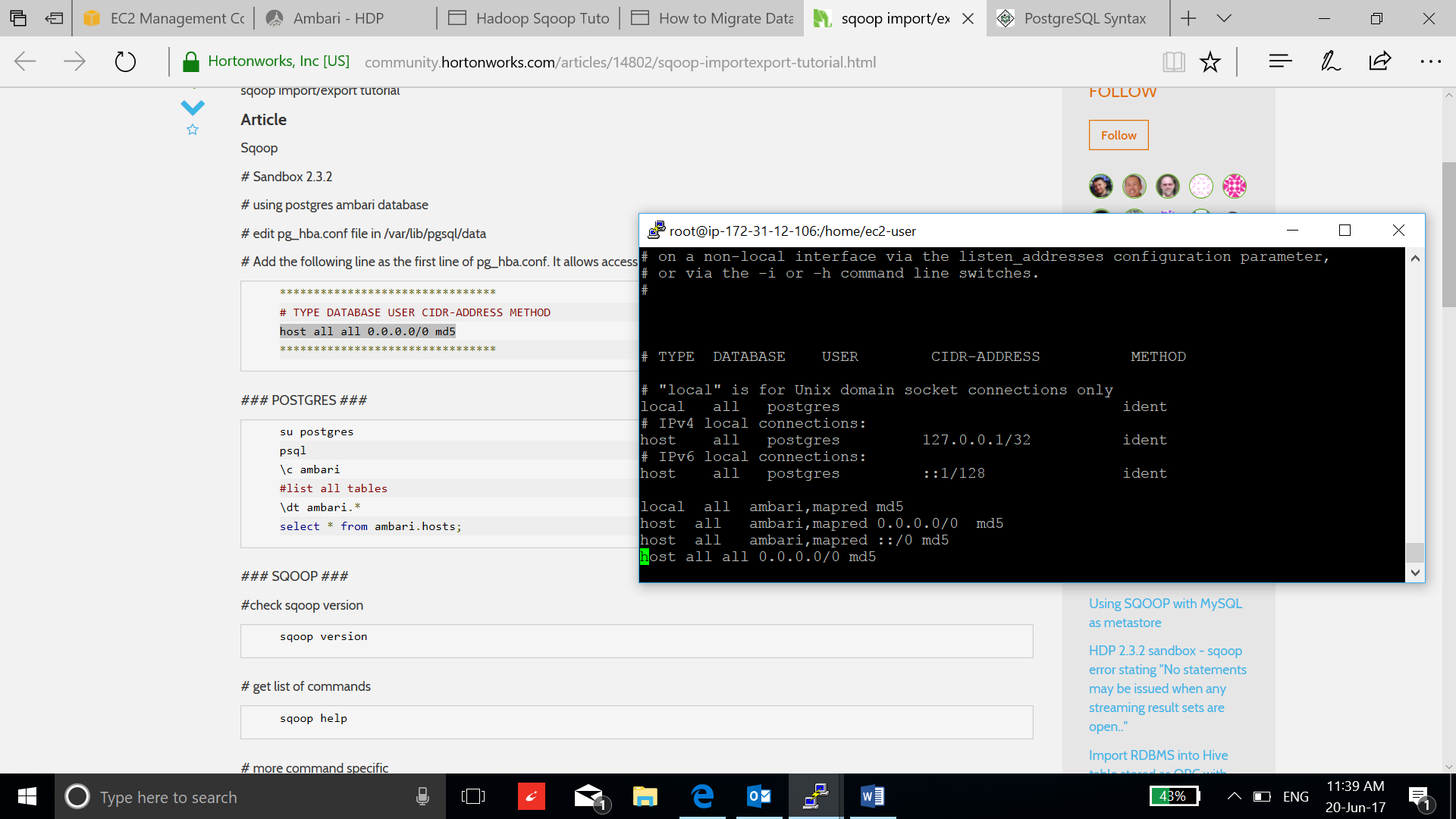
**hadoop fs -put postgresql-9.2-1002.jdbc4.jar /user/oozie/share/lib/lib\_20170629100500/sqoop/.**

* Open security connection to the postgresql database by editing pg\_hba.conf file:



* Add the host configuration line in the starting of the file:

**host all all 0.0.0.0/0 md5**



* Restart the postgresql:

**/etc/init.d/postgresql restart**

1. Now copy the examples of the oozie in some directory at local machine and at HDFS:

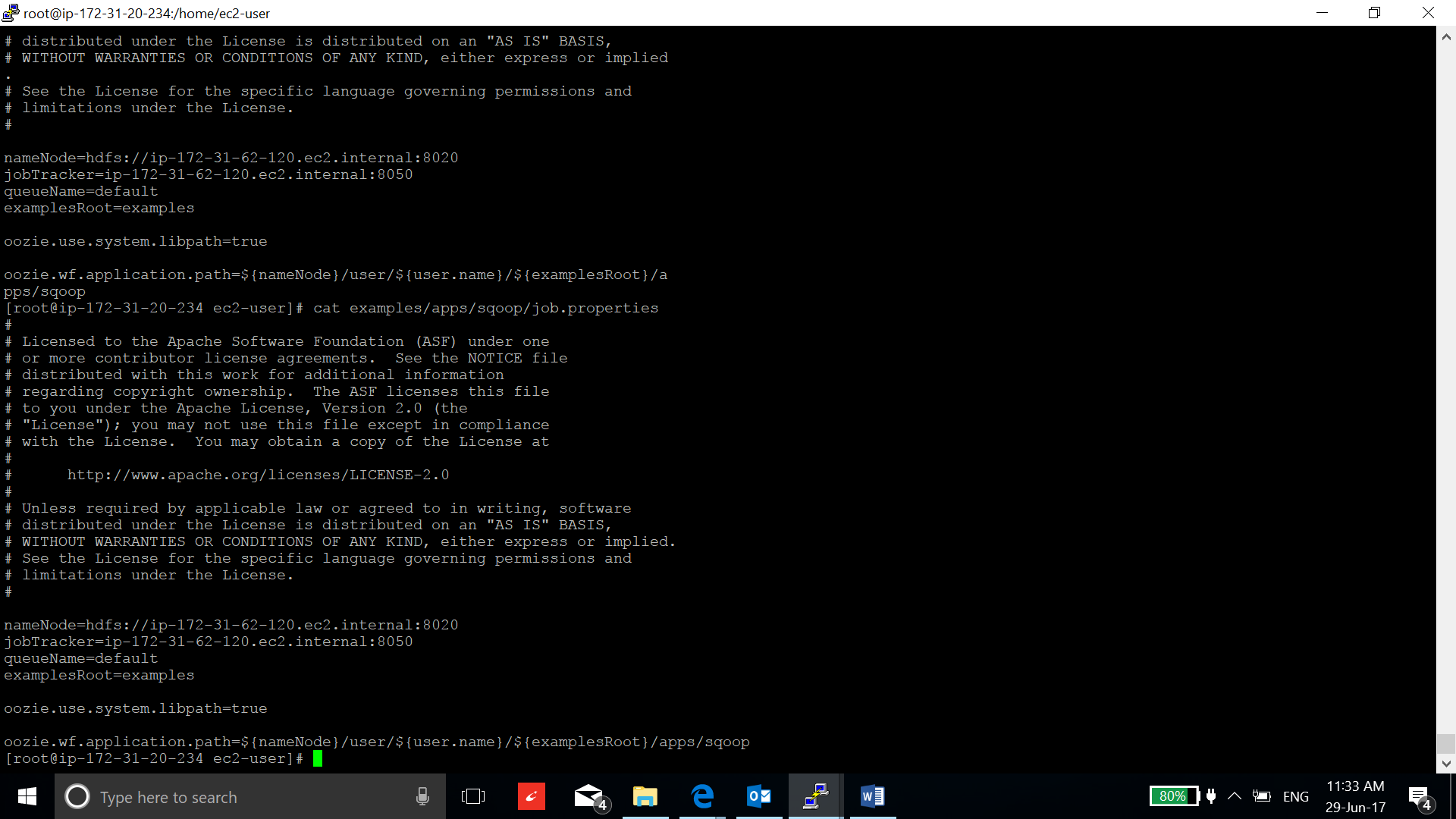
**find / -name "\*oozie\*examples\*.tar.gz"**

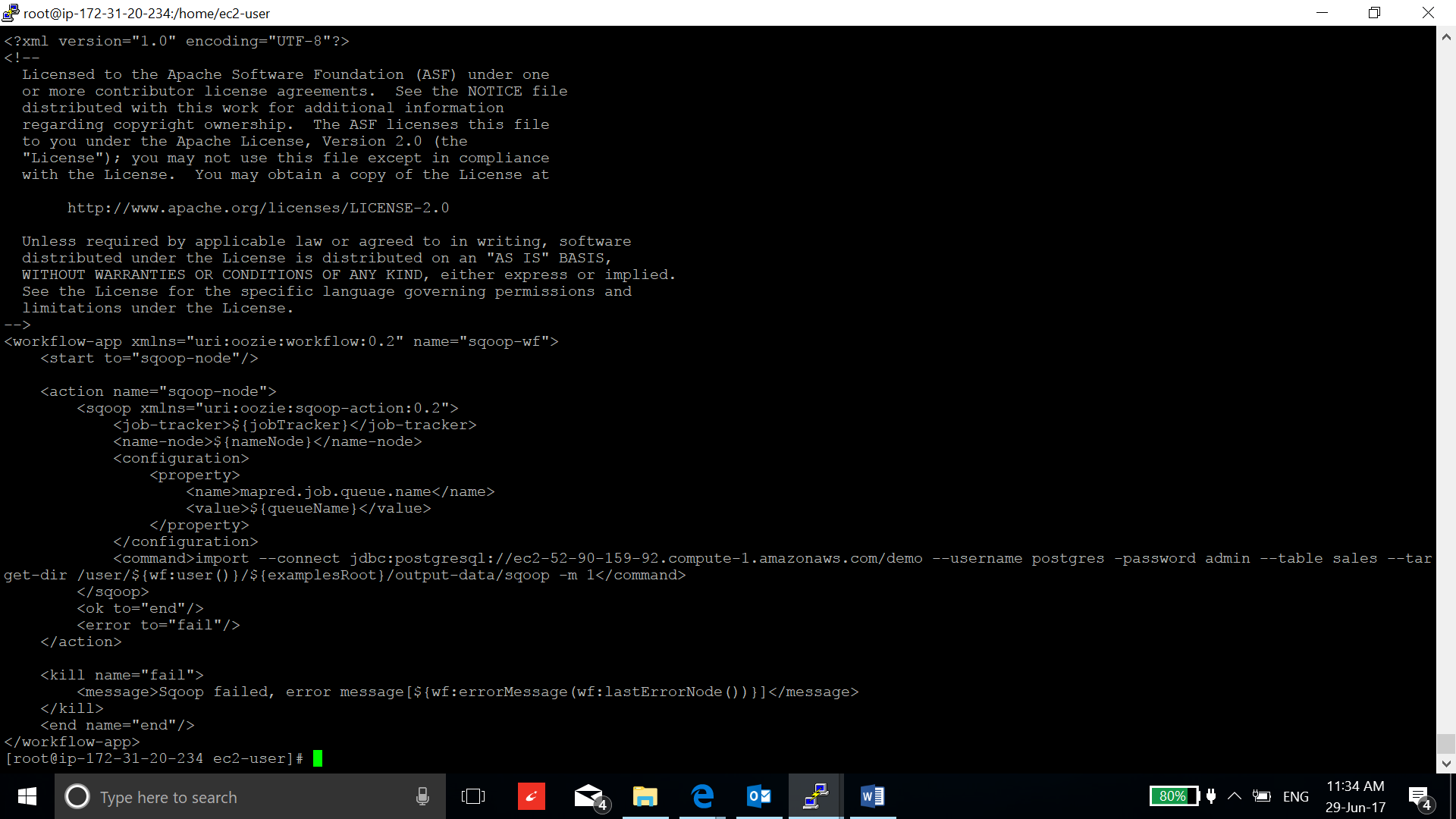
**cp /usr/hdp/2.6.1.0-129/oozie/doc/oozie-examples.tar.gz .**

**tar xzf oozie-examples.tar.gz**

**Hadoop fs -mkdir /user/root**

1. Change the job-properties file and workflow.xml according to the following snapshot.



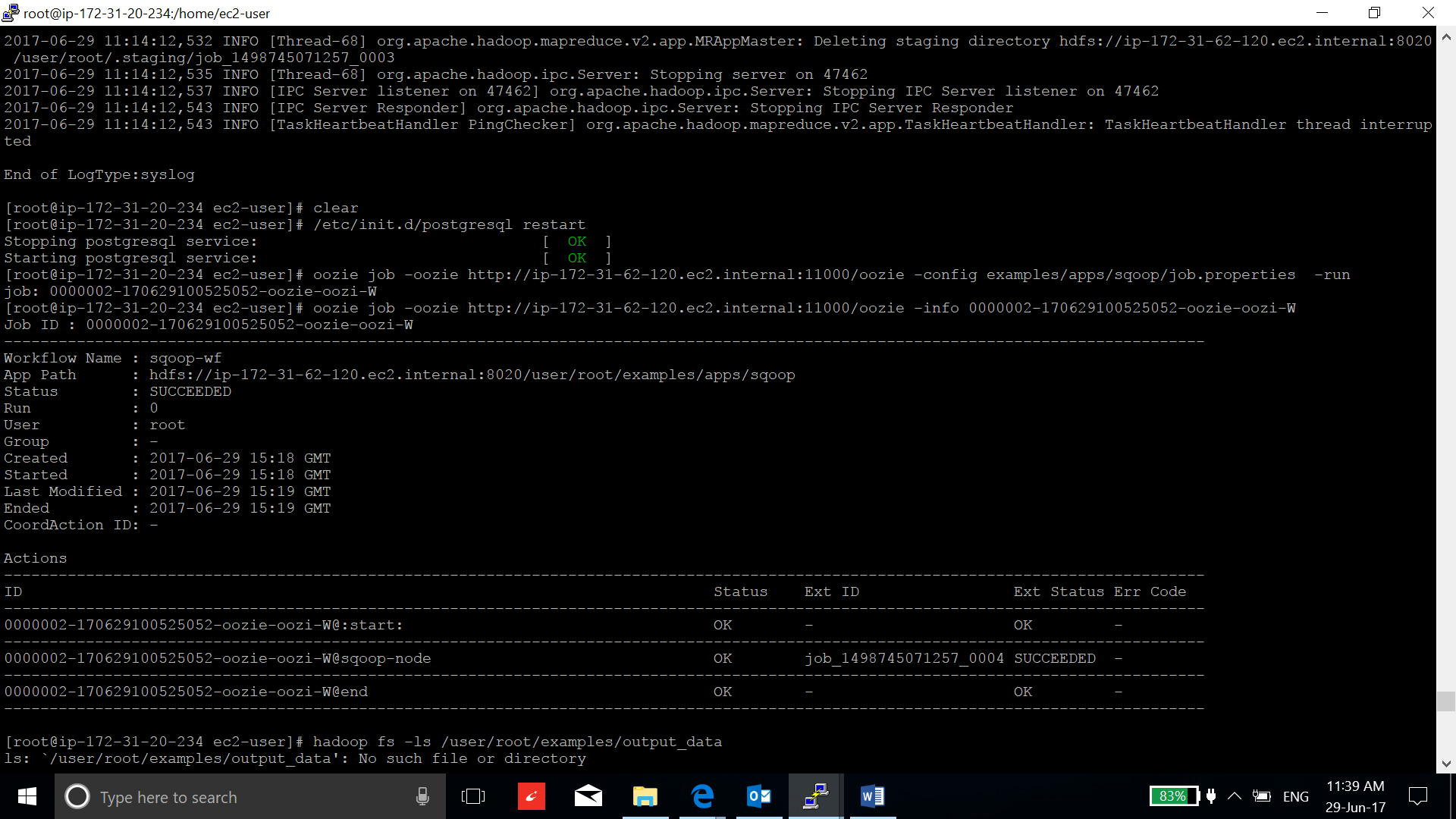


1. Now transfer the examples directory to the oozie.wf.application.path:

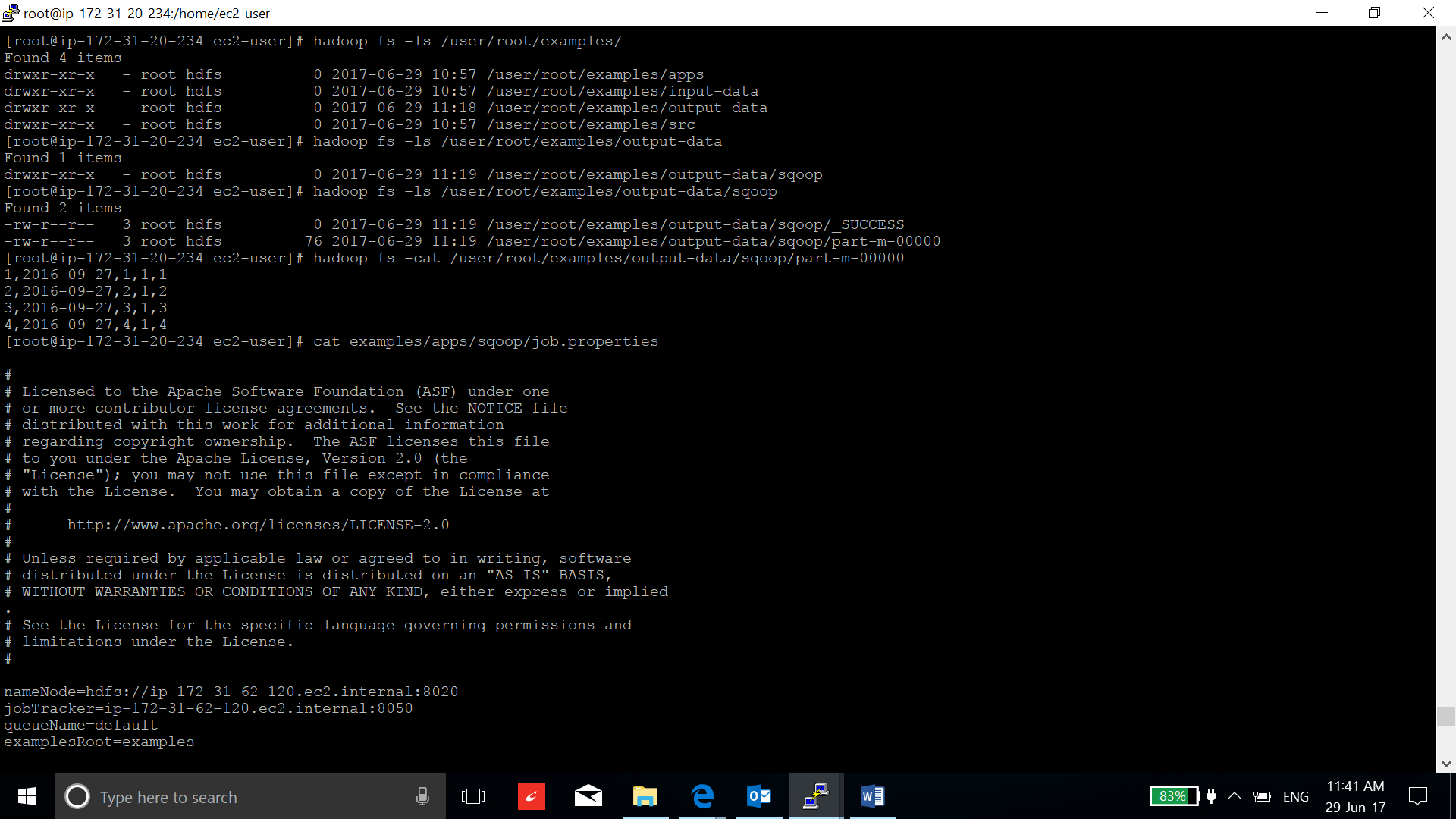
**hadoop fs -put examples /user/root/**

1. Run the job and check the status till job succeeded:

**oozie job -oozie http://ip-172-31-62-120.ec2.internal:11000/oozie -config examples/apps/sqoop/job.properties -run**

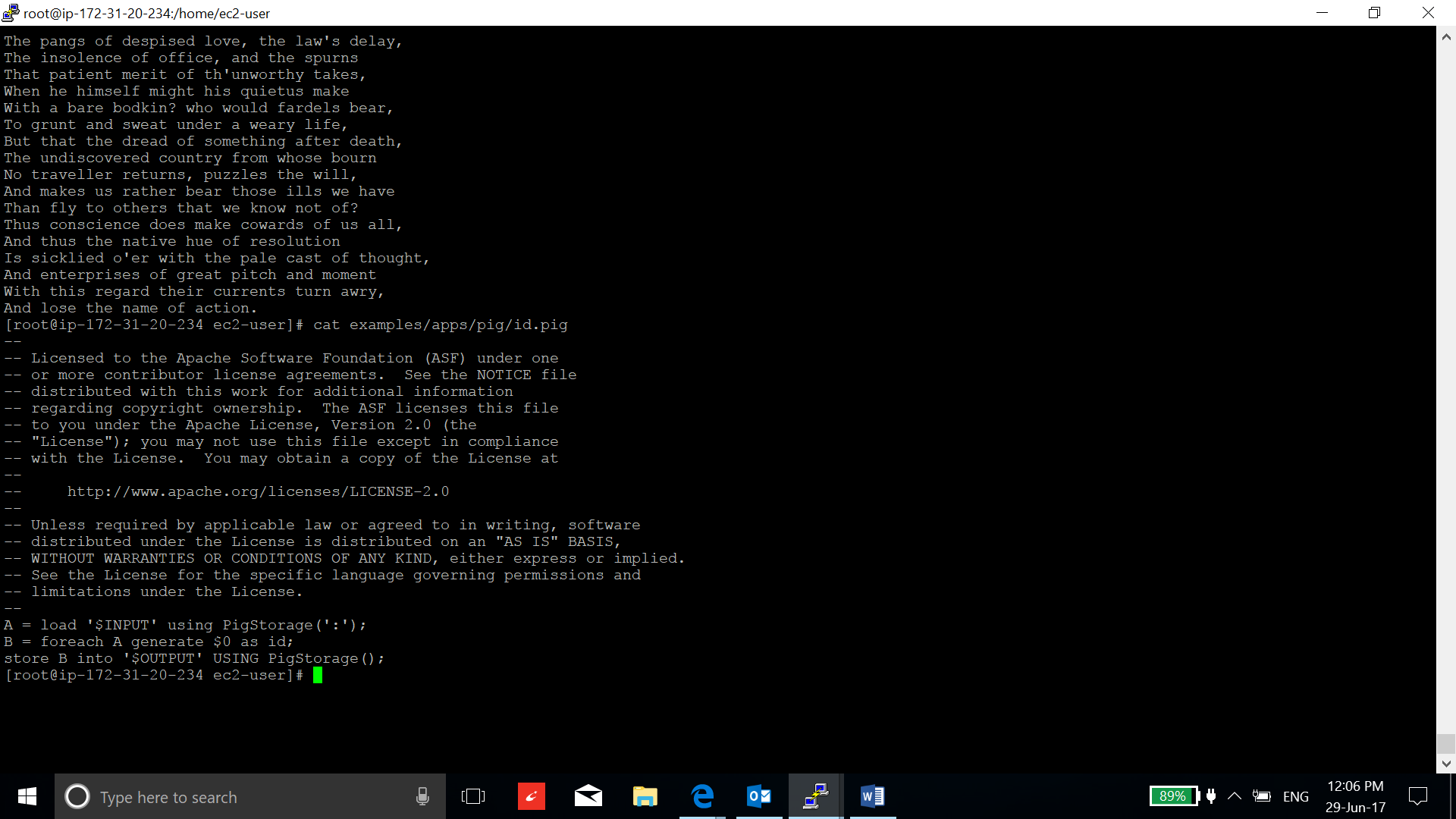


1. Verify the workflow by checking the file in the output directory at HDFS:

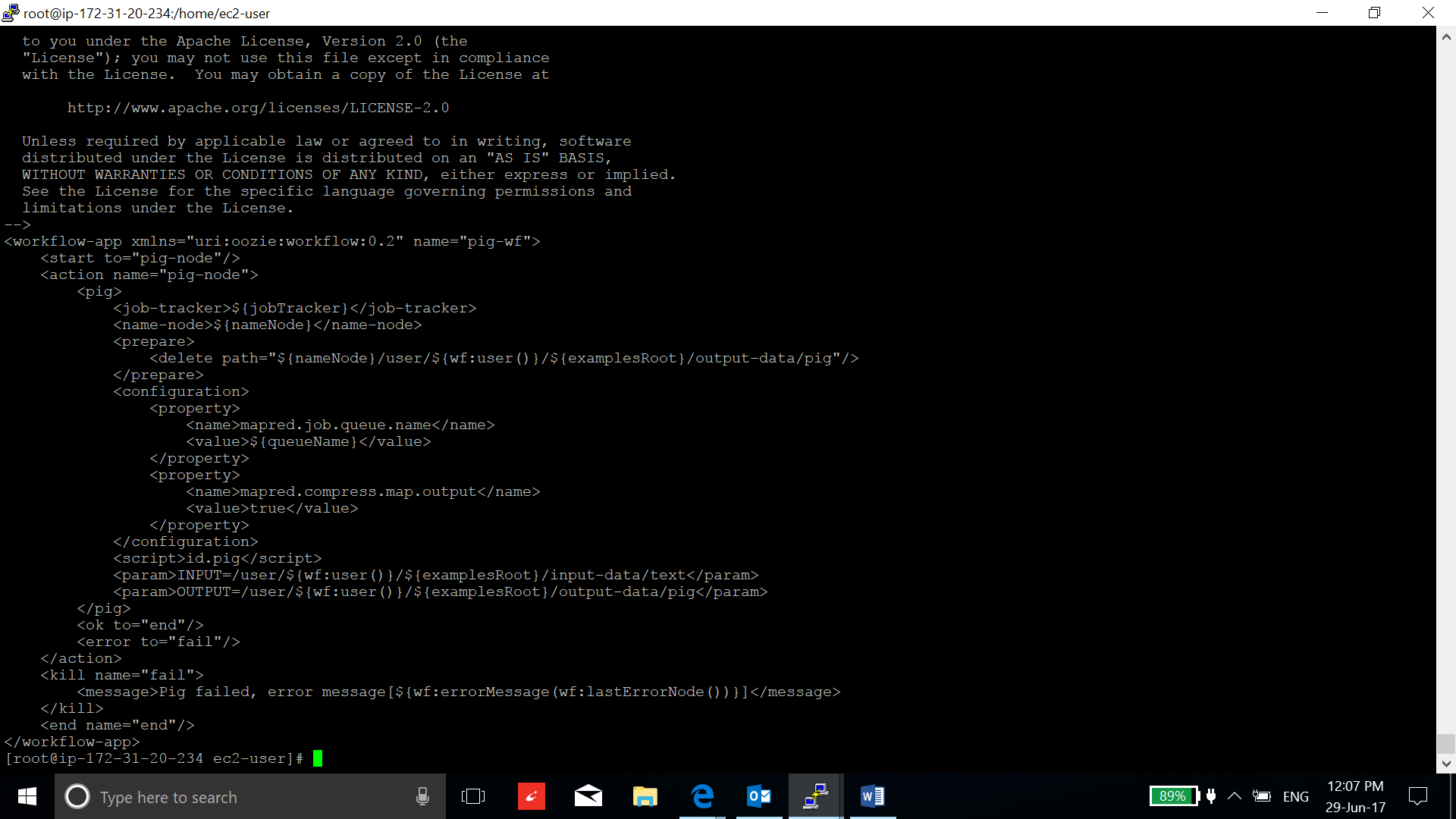


Run PIG workflow using Oozie:

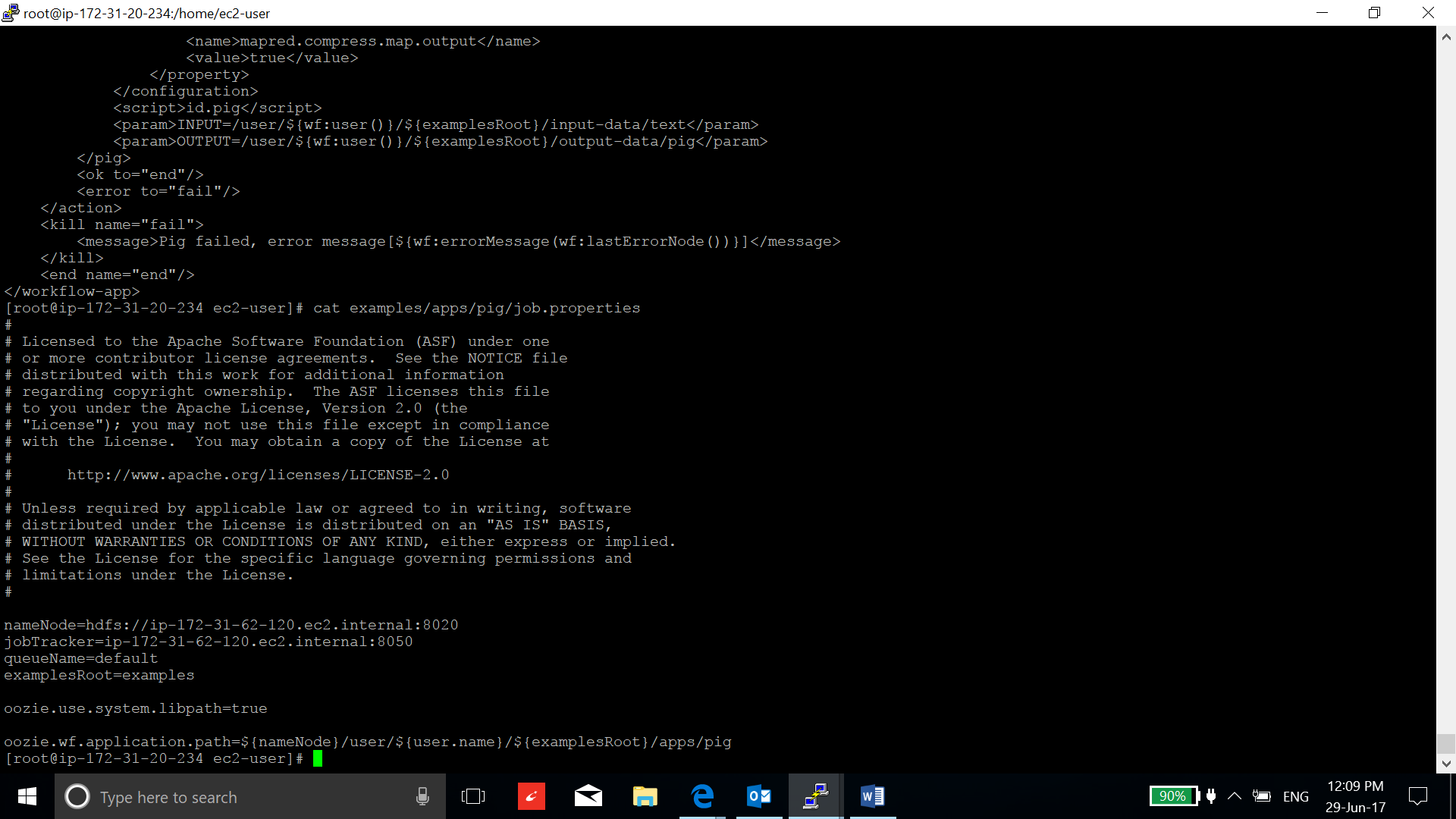
1. I am using the oozie’s examples directory to test the PIG overflow.
2. Edit the pig script, job-properties and workflow.xml file according to your task:



The input path of the data and output path to store the data is mentioned in workflow.xml under script tag.

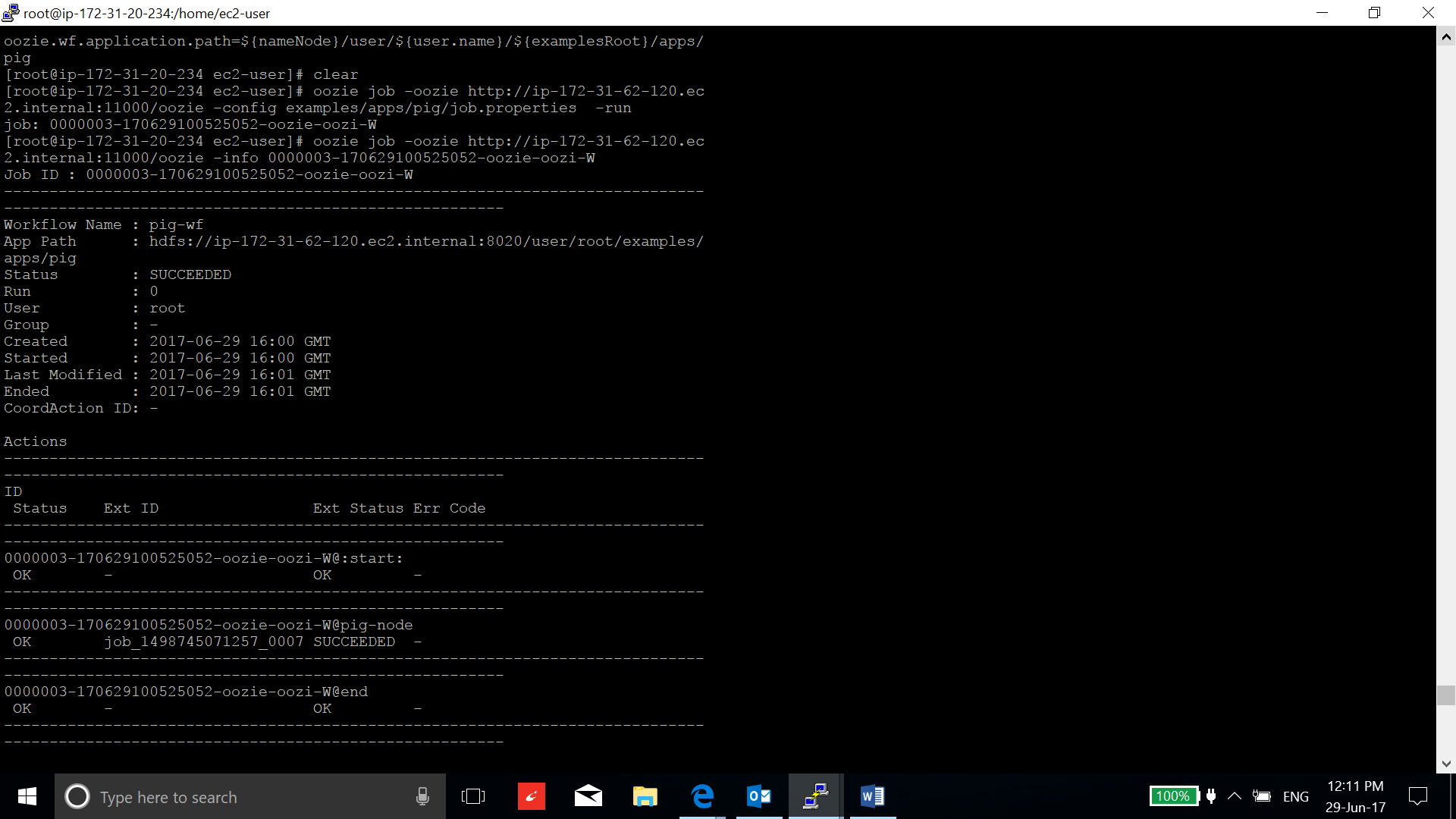


Change the input and output data directories. Put the script in the oozie.wf.application.path mentioned in job.properties.



Change the address according to the configuration of your instances.

1. Run the workflow:



1. Verify the result of the script:

