1. Prerequisites

1.1. Hadoop Cluster Installation

Apache Pig is a platform build on the top of Hadoop. You can refer to our previously published article to install a Hadoop single node cluster on Windows 10.

1.2. 7zip

7zip is needed to extract .tar.gz archives we will be downloading in this guide.

2. Downloading Apache Pig

To download the Apache Pig, you should go to the following link:

- https://downloads.apache.org/pig/

Pig Releases

Please make sure you're downloading from a nearby mirror site, not from www.apache.org.

Older releases are available from the archives.

Name	Last modified	Size	Description
Parent Directory		-	
latest/	2018-05-04 17:41	-	
pig-0.16.0/	2018-05-04 17:38	-	
pig-0.17.0/	2018-05-04 17:41	-	
KEYS	2017-06-19 08:12	11K	

Figure 1 — Apache Pig releases directory

If you are looking for the latest version, navigate to “latest” directory, then download the pig-x.xx.x.tar.gz file.

Index of /pig/latest

Name	Last modified	Size	Description
Parent Directory		-	
RELEASE_NOTES.txt	2017-06-16 18:10	1.9K	
pig-0.17.0-src.tar.gz	2017-06-16 18:11	15M	
pig-0.17.0-src.tar.gz.asc	2017-06-16 18:11	488	
pig-0.17.0-src.tar.gz.md5	2017-06-16 18:11	56	
pig-0.17.0.tar.gz	2017-06-16 18:10	220M	
pig-0.17.0.tar.gz.asc	2017-06-16 18:11	488	
pig-0.17.0.tar.gz.md5	2017-06-16 18:11	52	

Figure 2 — Download Apache Pig binaries

After the file is downloaded, we should extract it twice using 7zip *(using 7zip: the first time we extract the .tar.gz file, the second time we extract the .tar file).* We will extract the Pig folder into "E:\hadoop-env" directory as used in the previous articles.

3. Setting Environment Variables

After extracting Derby and Hive archives, we should go to Control Panel > System and Security > System. Then Click on "Advanced system settings".

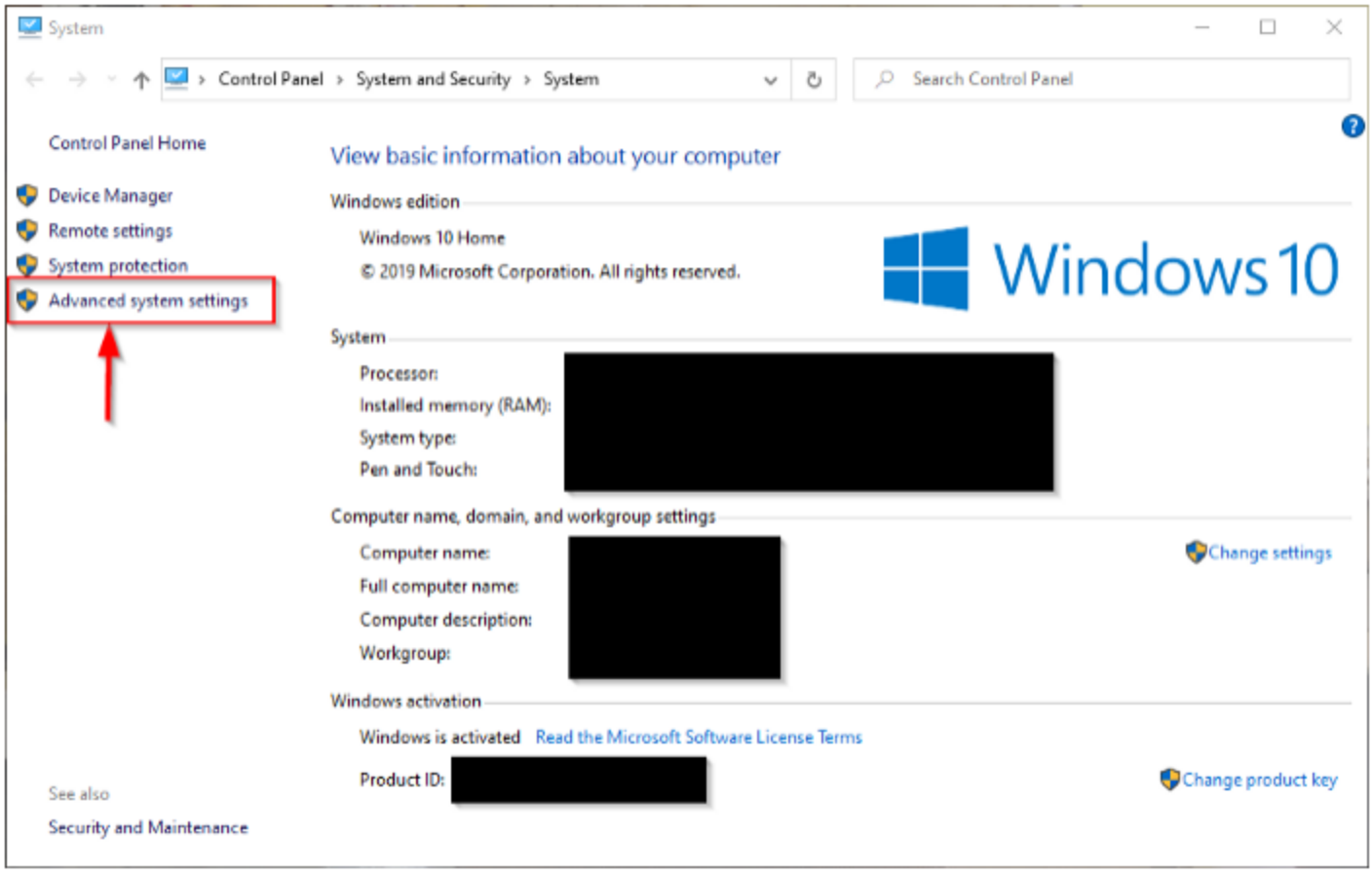


Figure 3 — Advanced system settings

In the advanced system settings dialog, click on “Environment variables” button.

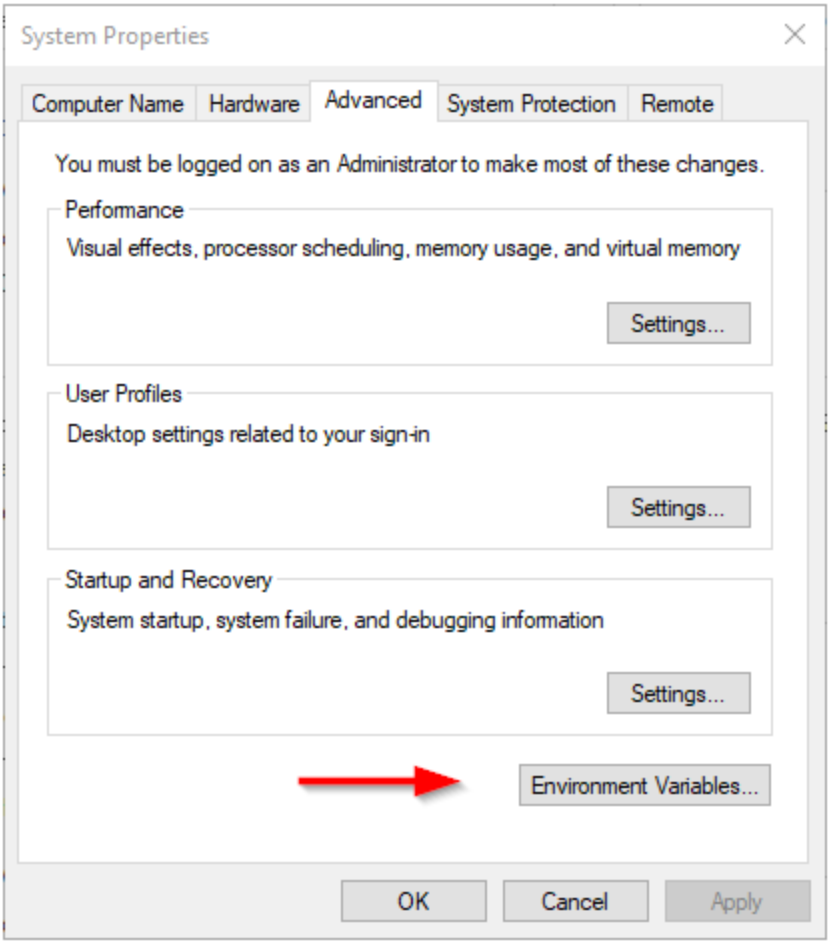


Figure 4 — Opening environment variables editor

Now we should add the following user variables:

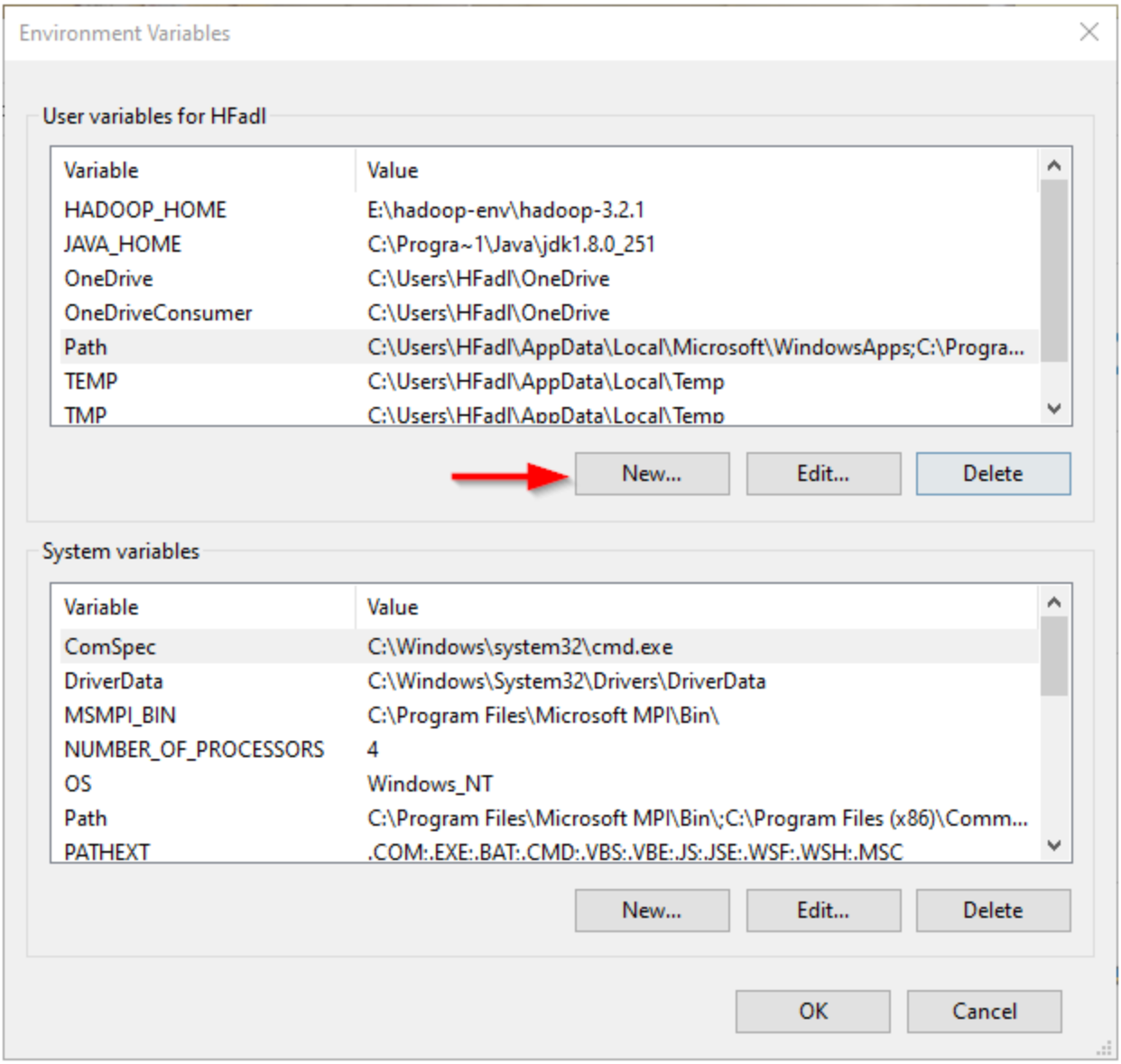


Figure 5 — Adding user variables

- PIG_HOME: “E:\hadoop-env\pig-0.17.0”

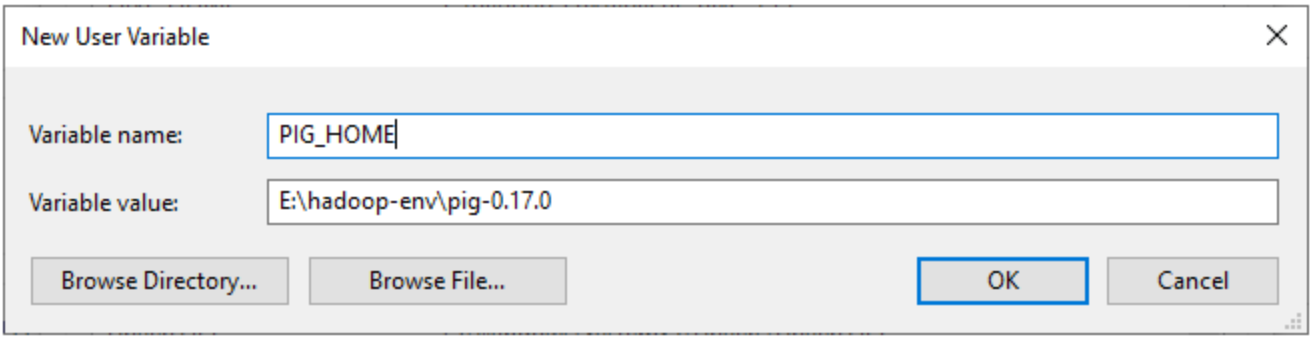


Figure 6 — Adding PIG_HOME variable

Now, we should edit the Path user variable to add the following paths:

- %PIG_HOME%\bin

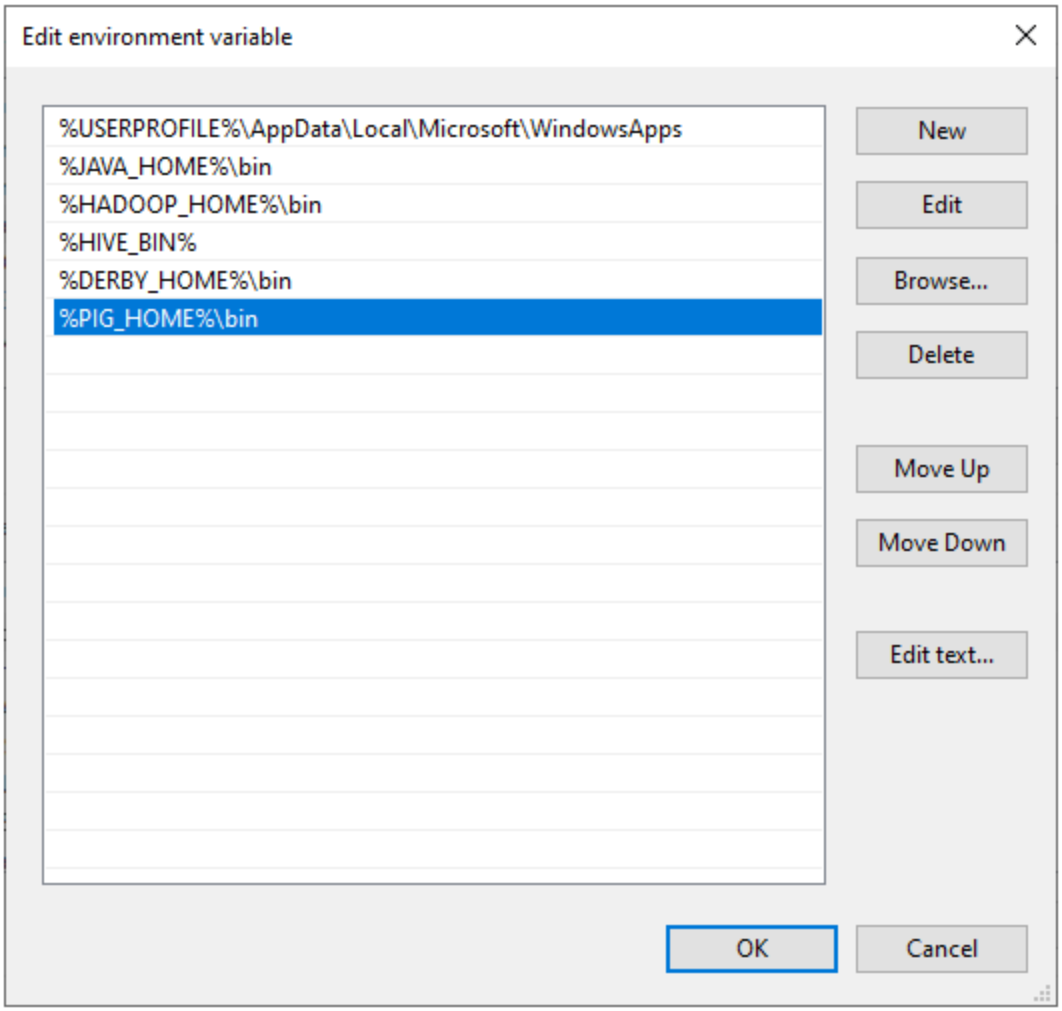


Figure 7 — Editing Path variable

4. Starting Apache Pig

After setting environment variables, let's try to run Apache Pig.

***Note**: Hadoop Services must be running*

Open a command prompt as administrator, and execute the following command

pig -version

You will receive the following exception:

'E:\hadoop-env\hadoop-3.2.1\bin\hadoop-config.cmd' is not recognized as an internal or external command,
operable program or batch file.
'-Xmx1000M' is not recognized as an internal or external command,
operable program or batch file.

```
E:\>pig -version
'E:\hadoop-env\hadoop-3.2.1\bin\hadoop-config.cmd' is not recognized as an internal or external command,
operable program or batch file.
'-Xmx1000M' is not recognized as an internal or external command,
operable program or batch file.
```

Figure 8 — Pig exception

To fix this error, we should edit the pig.cmd file located in the “pig-0.17.0\bin” directory by changing the HADOOP_BIN_PATH value from “%HADOOP_HOME%\bin” to “%HADOOP_HOME% \libexec”.

Now, let's try to run the “pig -version” command again:

```
E:\>pig -version
Apache Pig version 0.17.0 (r1797386)
compiled Jun 02 2017, 15:41:58
```

Figure 9 — Pig installation validated

The simplest way to write PigLatin statements is using Grunt shell which is an interactive tool where we write a statement and get the desired output. There are two modes to involve Grunt Shell:

1. Local: All scripts are executed on a single machine without requiring Hadoop. (command: pig -x local)
2. MapReduce: Scripts are executed on a Hadoop cluster (command: pig -x MapReduce)

Since we have installed Apache Hadoop 3.2.1 which is not compatible with Pig 0.17.0, we will try to run Pig using local mode.

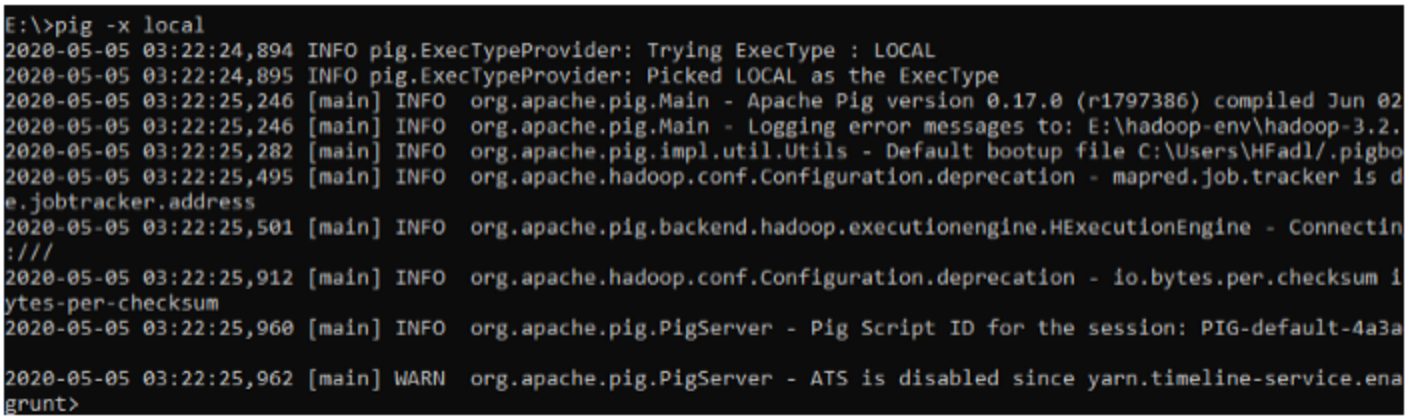


Figure 10 — Starting Grunt Shell in local mode

Apache PIG CASE STUDY:

1. Download the dataset containing the Agriculture related data about crops in various regions and their area and produce. The link for dataset –https://www.kaggle.com/abhinand05/crop-production-in-india The dataset contains 7 columns namely as follows.

State_Name : chararray ;

District_Name : chararray ;

Crop_Year : int ;

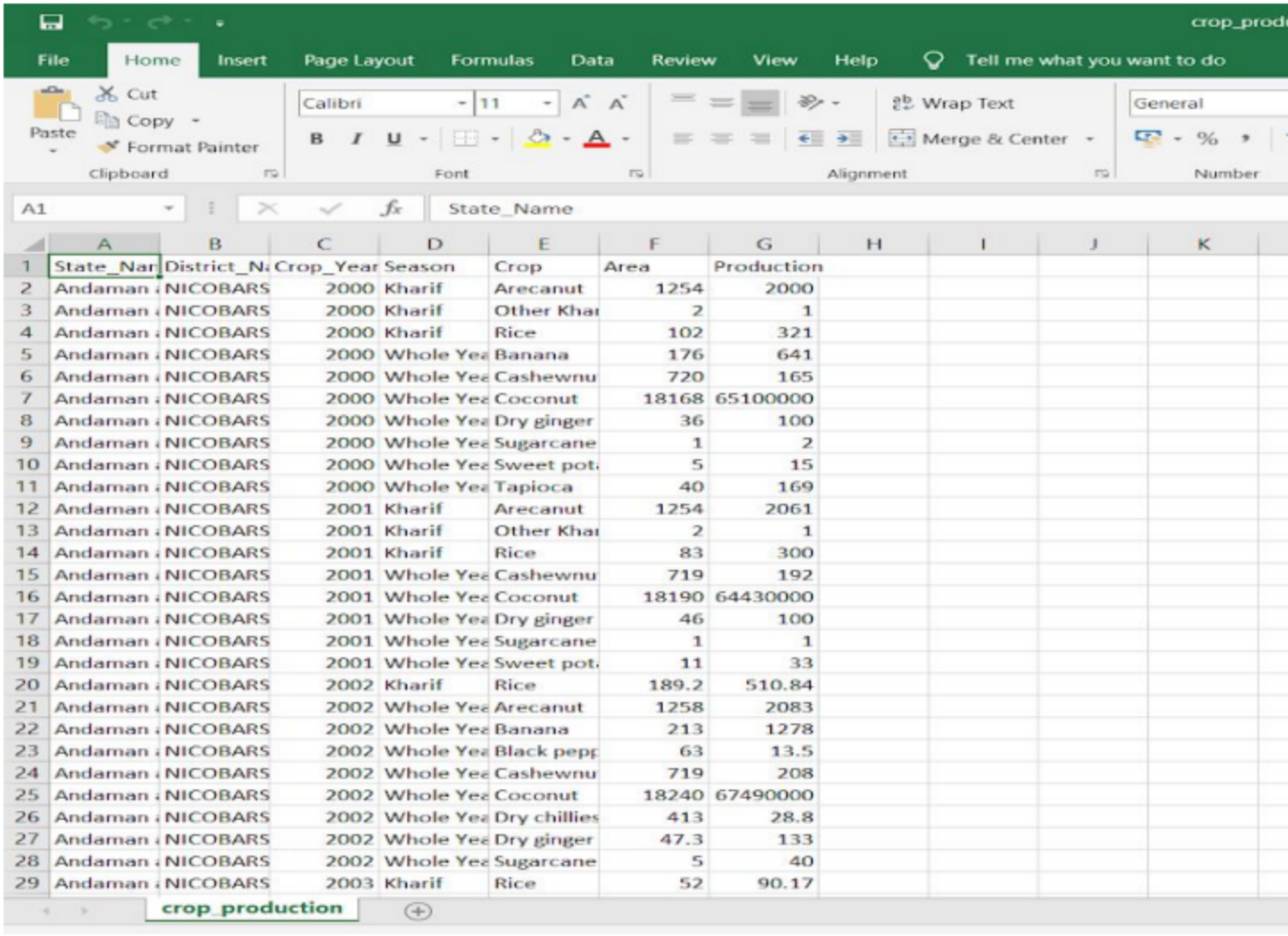
Season : chararray ;

Crop : chararray ;

Area : int ;

Production : int

No of rows: 246092
No of columns: 7



State_Name	District_Name	Crop_Year	Season	Crop	Area	Production
Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Arecanut	1254	2000
Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Other Kharif pulses	2	1
Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Rice	102	321
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Banana	176	641
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Cashewnut	720	165
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Coconut	18168	65100000
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Dry ginger	36	100
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Sugarcane	1	2
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Sweet potato	5	15
Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Tapioca	40	169
Andaman and Nicobar Islands	NICOBARS	2001	Kharif	Arecanut	1254	2061
Andaman and Nicobar Islands	NICOBARS	2001	Kharif	Other Kharif pulses	2	1
Andaman and Nicobar Islands	NICOBARS	2001	Kharif	Rice	83	300
Andaman and Nicobar Islands	NICOBARS	2001	Whole Year	Cashewnut	719	192
Andaman and Nicobar Islands	NICOBARS	2001	Whole Year	Coconut	18190	64430000
Andaman and Nicobar Islands	NICOBARS	2001	Whole Year	Dry ginger	46	100
Andaman and Nicobar Islands	NICOBARS	2001	Whole Year	Sugarcane	1	1
Andaman and Nicobar Islands	NICOBARS	2001	Whole Year	Sweet potato	11	33
Andaman and Nicobar Islands	NICOBARS	2002	Kharif	Rice	189.2	510.84
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Arecanut	1258	2083
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Banana	213	1278
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Black pepper	63	13.5
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Cashewnut	719	208
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Coconut	18240	67490000
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Dry chillies	413	28.8
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Dry ginger	47.3	133
Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Sugarcane	5	40
Andaman and Nicobar Islands	NICOBARS	2003	Kharif	Rice	52	90.17

2. Enter pig local mode using
grunt > pig -x local

```
C:\WINDOWS\system32>pig -x local
2020-11-06 00:22:54,632 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2020-11-06 00:22:54,633 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
```

3. Load the dataset in the local mode
grunt > agriculture= LOAD 'F:/csv files/crop_production.csv' using PigStorage (',')

as (State_Name:chararray , District_Name:chararray , Crop_Year:int ,

Season:chararray , Crop:chararray , Area:int , Production:int) ;

```
grunt> agriculture= load 'F:/csv files/crop_production.csv'using PigStorage (',') as (State_Name:chararray,District_Name:chararray,Crop_Year:i
2020-11-06 00:26:13,191 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.
grunt>
```

4. Dump and describe the data set agriculture using
grunt > dump agriculture;

grunt > describe agriculture;

```
Administrator: Command Prompt - pig  -x local
(West Bengal,PURULIA,2014,Kharif     ,Moth,16,14)
(West Bengal,PURULIA,2014,Kharif     ,Niger seed,204,74)
(West Bengal,PURULIA,2014,Kharif     ,Other Kharif pulses,79,39)
(West Bengal,PURULIA,2014,Kharif     ,Sannhamp,171,727)
(West Bengal,PURULIA,2014,Kharif     ,Soyabean,18,7)
(West Bengal,PURULIA,2014,Kharif     ,Sunflower,46,42)
(West Bengal,PURULIA,2014,Kharif     ,Urad,11493,3287)
(West Bengal,PURULIA,2014,Rabi       ,Arhar/Tur,671,723)
(West Bengal,PURULIA,2014,Rabi       ,Gram,198,203)
(West Bengal,PURULIA,2014,Rabi       ,Groundnut,30,25)
(West Bengal,PURULIA,2014,Rabi       ,Horse-gram,660,332)
(West Bengal,PURULIA,2014,Rabi       ,Khesari,146,126)
(West Bengal,PURULIA,2014,Rabi       ,Linseed,160,51)
(West Bengal,PURULIA,2014,Rabi       ,Masoor,31,19)
(West Bengal,PURULIA,2014,Rabi       ,Moong(Green Gram),64,40)
(West Bengal,PURULIA,2014,Rabi       ,Peas & beans (Pulses),12,12)
(West Bengal,PURULIA,2014,Rabi       ,Potato,477,9995)
(West Bengal,PURULIA,2014,Rabi       ,Rapeseed &Mustard,1885,1508)
(West Bengal,PURULIA,2014,Rabi       ,Safflower,54,37)
(West Bengal,PURULIA,2014,Rabi       ,Urad,220,113)
(West Bengal,PURULIA,2014,Rabi       ,Wheat,1622,3663)
(West Bengal,PURULIA,2014,Summer     ,Maize,325,2039)
(West Bengal,PURULIA,2014,Summer     ,Rice,306,801)
(West Bengal,PURULIA,2014,Summer     ,Sesamum,627,463)
(West Bengal,PURULIA,2014,Whole Year ,Sugarcane,324,16250)
(West Bengal,PURULIA,2014,Winter     ,Rice,279151,597899)
(West Bengal,PURULIA,2014,Winter     ,Sesamum,175,88)
grunt> describe agriculture;
agriculture: {State_Name: chararray,District_Name: chararray,Crop_Year: int,Season: chararray,Crop: chararray,Area: int,P
grunt>
```

5. Executing the PIG queries in local mode
You can follow these written queries to analyze the dataset using the various functions and operators in PIG. You need to follow all the above steps before proceeding.

Query 1: Grouping All Records State wise.
This command will group all the records by the column State_Name.
grunt > statewisecrop = GROUP agriculture BY State_Name;

grunt > DUMP statewisecrop;

grunt > DESCRIBE statewisecrop;

```
grunt> statewisecrop = GROUP agriculture by State_Name;
grunt> describe statewisecrop;
statewisecrop: {group: chararray,agriculture: {(State_Name: chararray,District_Name: chararray,Crop_Year: int,Season: ch
ararray,Crop: chararray,Area: int,Production: int)}}
grunt>
```

Now store the result of the query in a CSV file for better understanding. We have to mention the name of the object and the path where it needs to be stored.

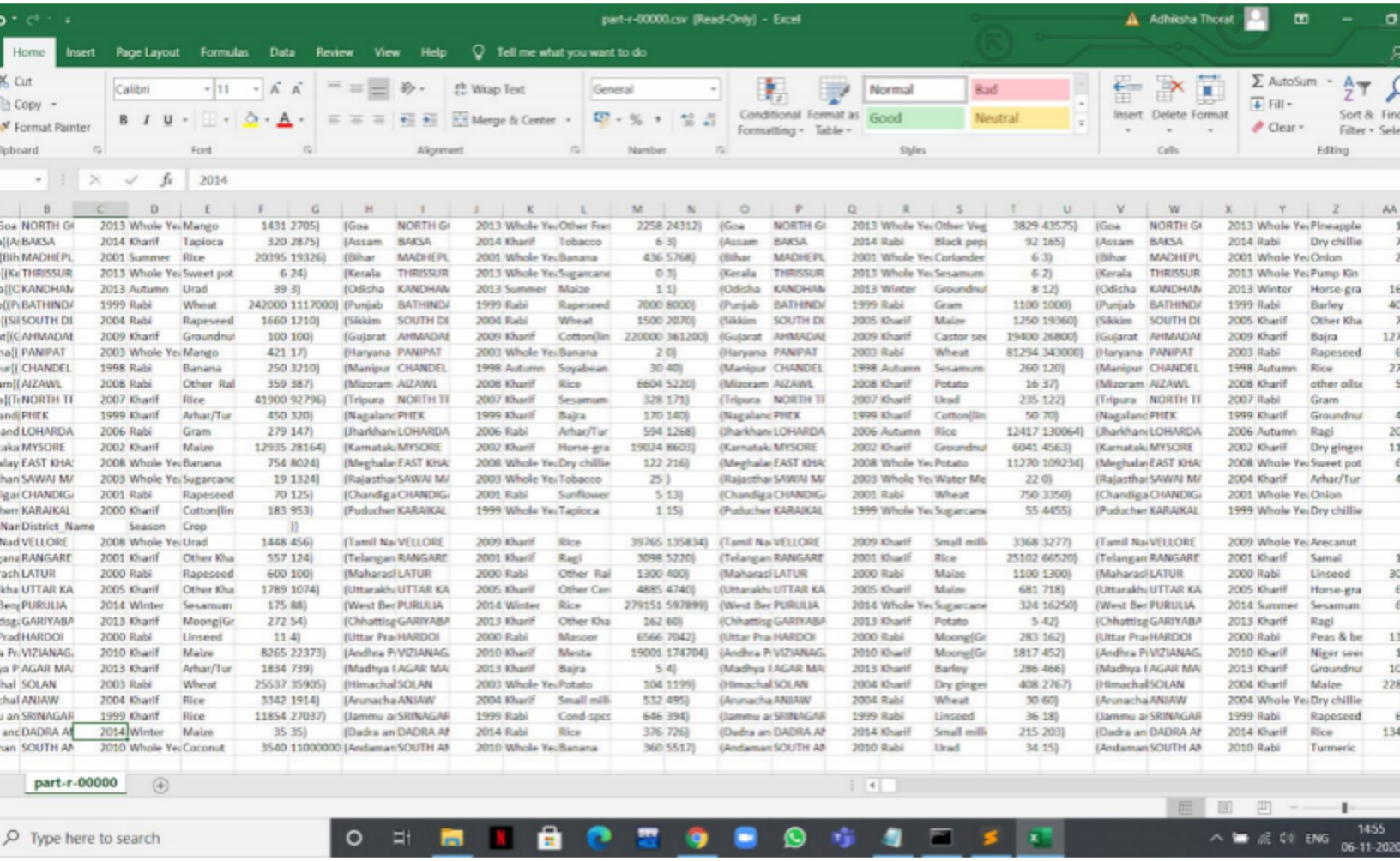
pathname -> 'F:/csv files/statewiseoutput'

grunt > STORE statewisecrop INTO 'F:/csv files/statewiseoutput';

```
grunt> STORE statewisecrop INTO 'F:/csv files/statewiseoutput';
2020-11-06 14:50:53,644 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.by
2020-11-06 14:50:53,720 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY
2020-11-06 14:50:53,818 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.by
2020-11-06 14:50:53,910 [main] INFO  org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPru
 LoadTypeCastInserter, MergeFilter, MergeForEach, NestedLimitOptimizer, PartitionFilterOptimizer, PredicatePushdownOptimizer, PushDownForEachFla
```

```
Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTime      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime   MedianReducet
job_local1720445657_0001        1       1       n/a     n/a     n/a     n/a     n/a     n/a     n/a     agriculture,statewisecrop       GROUP_

Input(s):
Successfully read 246092 records from: "F:/csv files/crop_production.csv"

Output(s):
Successfully stored 34 records in: "F:/csv files/statewiseoutput"

Counters:
Total records written : 34
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local1720445657_0001


2020-11-06 14:51:08,776 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2020-11-06 14:51:08,780 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2020-11-06 14:51:08,791 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2020-11-06 14:51:08,806 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Encountered Warning FIELD_DISCA
2020-11-06 14:51:08,806 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
grunt>
```

The output will be in a file named **'part-r-00000'** which needs to be renamed as **'part-r-00000.csv'** to be opened in the Excel format and to make it readable. You will find this file in the path that we have mentioned in the above query. In my case it was in the path 'F:/csv files/statewiseoutput/'.
The output file will look something like this:



CSV file of output of query 1

Query 2: Generate Total Crop wise Production and Area

In the above query, we need to group by Crop type and then find the SUM of their Productions and Area.

grunt > cropinfo = FOREACH(GROUP agriculture BY Crop)

GENERATE group AS Crop, SUM(agriculture.Area) as AreaPerCrop ,

SUM(agriculture.Production) as ProductionPerCrop;

grunt > DESCRIBE cropinfo;

Administrator: Command Prompt - pig -x local

```
grunt> cropinfo = FOREACH( GROUP agriculture BY Crop )
>> GENERATE group AS Crop, SUM(agriculture.Area) as AreaPerCrop , SUM( agriculture.Production)
grunt> describe cropinfo;
cropinfo: {Crop: chararray,AreaPerCrop: long,ProductionPerCrop: long}
grunt>
```

grunt > STORE cropinfo INTO 'F:/csv files/cropinfooutput';

The output will be in a file named **'part-r-00000'** which needs to be renamed as **'part-r-00000.csv'** to be opened in the Excel format and to make it readable. You can check the csv output by opening the command prompt in administrator mode and running the command as follows.

C:\Users > cat 'F:/csv files/cropinfooutput/part-r-00000.csv'

This will return all the output on the command prompt. You can see that we have three columns in the output.

Output:

Crop ,

AreaperCrop ,

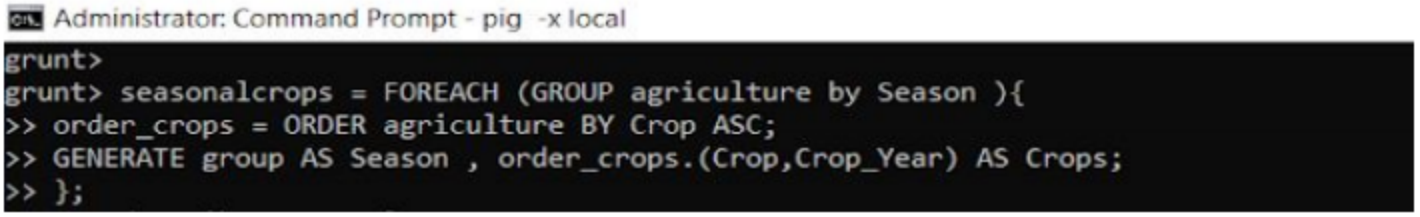
ProductionPerCrop.

Command Prompt

```
C:\Users\Adhiksha>cat 'F:/csv files/cropinfooutput/part-r-00000.cs
Ber     118     0
Tea     77810   135981
Yam     1775    0
Bean    4650    6240
Crop
Gram    118207908       99414201
Jute    14730166        181558166
Moth    15715275        3698783
Pear    2676    0
Ragi    24829204        35131126
Rice    747125124       1605470183
Urad    47575717        22410051
Apple   9       0
Bajra   141140804       129680983
Jowar   137715913       114598257
Kapas   8862    8542
Korra   179309  123713
Lemon   75307   540994
Maize   121746610       273341393
Mango   2380346 12770536
Mesta   1621365 12393542
Onion   6308484 72453027
Paddy   17536469        31702401
Peach   42      0
Plums   809     0
Samai   245261  145175
```

Query 3: The majority of crops are grown in a Season and in which year.
In this query, we need to group the crops by season and order them alphabetically. Also, this will tell us which crops are found in a season and with year.

```
grunt > seasonalcrops = FOREACH (GROUP agriculture by Season ){
        order_crops = ORDER agriculture BY Crop ASC;
        GENERATE group AS Season , order_crops.(Crop) AS Crops;
        };
```



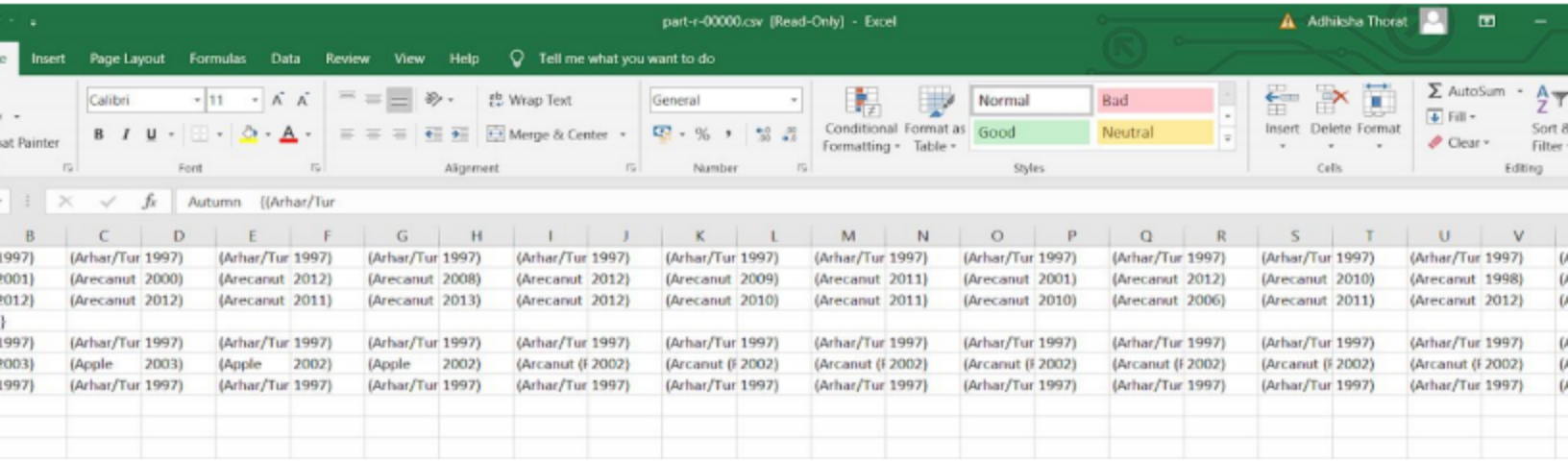
grunt > DESCRIBE seasonalcrops;

grunt > STORE seasonalcrops INTO 'F:/csv files/seasonaloutput;

The output will be in a file named 'part-r-00000' which needs to be renamed as 'part-r-00000.csv' to be opened in the Excel format and to make it readable. You can check the csv output by opening the command prompt in administrator mode and running the command as follows.

C:\Users > cat 'F:/csv files/seasonaloutput/part-r-00000.csv'

You can check the output from the 'part-r-00000.csv' by opening the file. You can see all the distinct seasons in the first row followed by all the crops and their years of production.

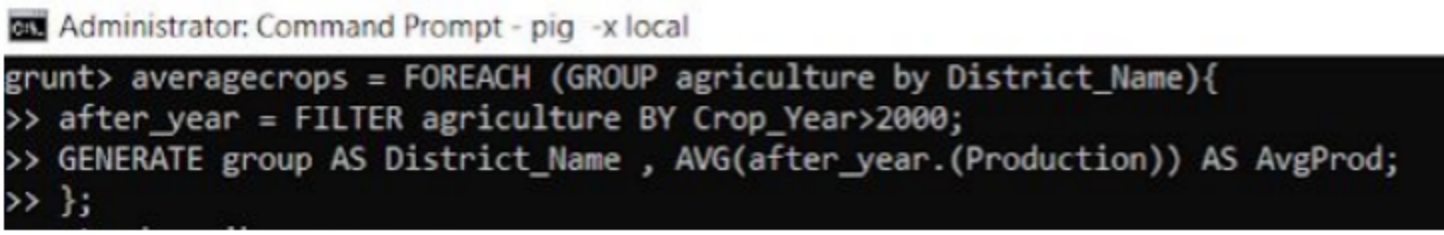


Output of this query number 3

Query 4: Average crop production in each district after the year 2000.

First, we need to group by district name and then find the average of the total crop production but only after the year 2000.

```
grunt > averagecrops = FOREACH (GROUP agriculture by District_Name){
            after_year = FILTER agriculture BY Crop_Year>2000;
            GENERATE group AS District_Name , AVG(after_year.(Production)) AS
            AvgProd;
            };
```

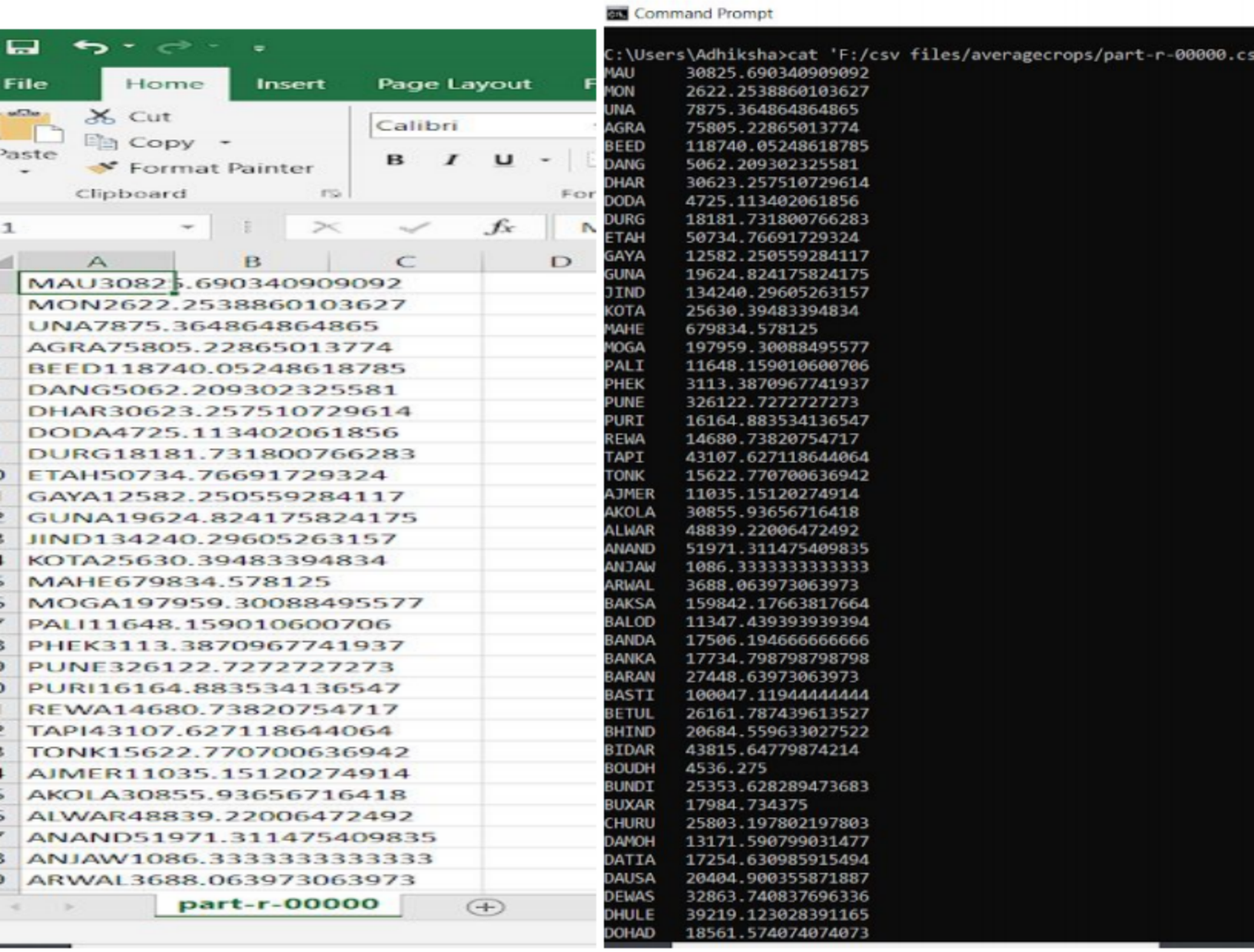



```
grunt > DESCRIBE averagecrops;
```

```
grunt> describe averagecrops;
averagecrops: {District_Name: chararray,AvgProd: double}
```

```
grunt > STORE averagecrops INTO 'F:/csv files/averagecrops;
```

You can check the output from the 'part-r-00000.csv' by opening the file. This file will contain two columns. The first one has all distinct district names and the second one will have the average production of all crops in each district after the year 2000.



Output of the query number 4

Query 5: Highest produced crops and details from each State.
First, we need to group the input by the state name. Then iterate through each grouped record and then find the TOP 1 record with the highest Production from

each state.

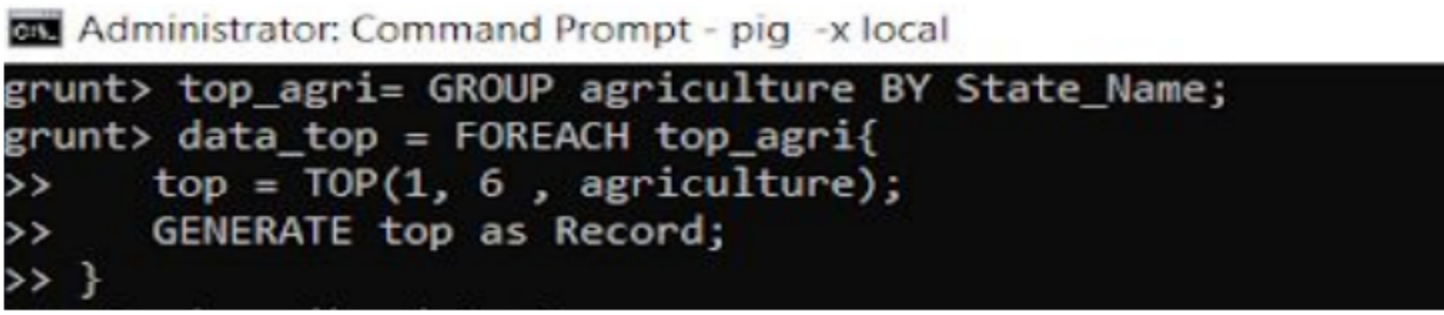
grunt > top_agri= GROUP agriculture BY State_Name;

grunt > data_top = FOREACH top_agri{

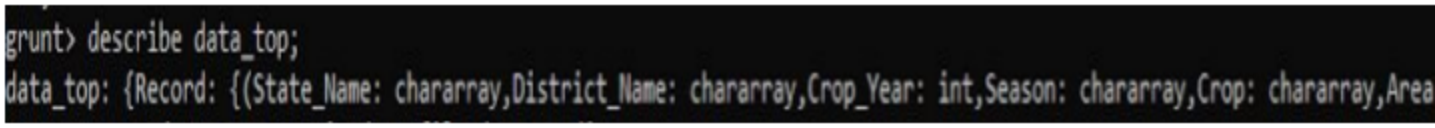
top = TOP(1, 6 , agriculture);

GENERATE top as Record;

}



grunt > DESCRIBE cropinfo;



grunt > STORE averagecrops INTO 'F:/csv files/averagecrops;

You can check the output from the 'part-r-00000.csv' by opening the file. This file contains records from each unique state who are having the highest Production amount. Read above and follow the steps to create 'part-r-00000.csv'. You can check the csv output by opening the command prompt in administrator mode and running the command:

C:\Users > cat 'F:/csv files/top1prod/part-r-00000.csv'

Command Prompt

```
C:\Users\Adhiksha>cat 'F:/csv files/top1prod/part-r-00000.csv'
{(Goa,SOUTH GOA,2007,Whole Year ,Coconut ,14296,71410000)}
{(Assam,NAGAON,1999,Whole Year ,Coconut ,5035,60719000)}
{(Bihar,PASHCHIM CHAMPARAN,2014,Whole Year ,Sugarcane,149624,9614539)}
{(Kerala,MALAPPURAM,2012,Whole Year ,Coconut ,102417,1125000000)}
{(Odisha,GANJAM,2014,Winter     ,Rice,264000,715000)}
{(Punjab,FIROZEPUR,2011,Rabi       ,Wheat,394000,1969000)}
{(Sikkim,SOUTH DISTRICT,2010,Kharif     ,Maize,14330,233145)}
{(Gujarat,SURAT,2005,Whole Year ,Sugarcane,130600,11754000)}
{(Haryana,YAMUNANAGAR,2006,Whole Year ,Sugarcane,42191,2894000)}
{(Manipur,IMPHAL EAST,2004,Kharif     ,Rice,30210,89690)}
{(Mizoram,AIZAWL,2000,Kharif     ,Rice,24666,53997)}
{(Tripura,WEST TRIPURA,2011,Kharif     ,Rice,101031,282578)}
{(Nagaland,DIMAPUR,2005,Whole Year ,Sugarcane,2120,160100)}
{(Jharkhand,PAKUR,2003,Winter     ,Rice,413835,509568)}
{(Karnataka,BELGAUM,2010,Whole Year ,Sugarcane,187884,18027470)}
{(Meghalaya,EAST KHASI HILLS,1998,Whole Year ,Potato,11803,129254)}
{(Rajasthan,GANGANAGAR,2010,Rabi       ,Wheat,236076,1049386)}
{(Chandigarh,CHANDIGARH,1998,Rabi       ,Wheat,880,3960)}
{(Puducherry,PONDICHERRY,2009,Whole Year ,Coconut ,1169,22012000)}
{(State_Name,District_Name,,Season,Crop,,)}
{(Tamil Nadu,COIMBATORE,2011,Whole Year ,Coconut ,82704,1250800000)}
{(Telangana ,KHAMMAM,2003,Whole Year ,Coconut ,1050,11933140)}
{(Maharashtra,SOLAPUR,2014,Whole Year ,Sugarcane,205500,20049700)}
{(Uttarakhand,HARIDWAR,2007,Whole Year ,Sugarcane,74811,4630801)}
{(West Bengal,24 PARAGANAS NORTH,2006,Whole Year ,Coconut ,3338,75095000)}
{(Chhattisgarh,BALRAMPUR,2013,Kharif     ,Sugarcane,44097,2129180)}
{(Uttar Pradesh,KHERI,2014,Kharif     ,Sugarcane,268653,17757963)}
{(Andhra Pradesh,EAST GODAVARI,2014,Whole Year ,Coconut ,46865,780162000)}
{(Madhya Pradesh,NARSINGHPUR,2007,Whole Year ,Sugarcane,31626,1545480)}
{(Himachal Pradesh,KANGRA,2004,Rabi       ,Wheat,94424,170283)}
{(Arunachal Pradesh,CHANGLANG,2008,Kharif     ,Rice,16763,35010)}
{(Jammu and Kashmir ,JAMMU,2004,Rabi       ,Wheat,93162,209390)}
{(Dadra and Nagar Haveli,DADRA AND NAGAR HAVELI,2002,Winter     ,Sugarcane,1537,1229
{(Andaman and Nicobar Islands,NICOBARS,2010,Whole Year ,Coconut ,14560,71300000)}

C:\Users\Adhiksha>
```