

BIG DATA HADOOP AND SPARK DEVELOPMENT

ASSIGNMENT 7

Table of Contents:

| | |
|-----------------------|----|
| 1. Introduction | 1 |
| 2. Objective | 1 |
| 3. Problem Statement | 1 |
| 4. Expected Output | |
| • Task 1 | 4 |
| • Task 2 | 14 |
| ○ Sub task 1 | 17 |
| ○ Sub task 2 | 22 |
| ○ Sub task 3 | 27 |
| ○ Sub task 4 | 33 |
| ○ Sub task 5 | 36 |
| • Task 3 | 39 |
| ○ Problem Statement 1 | 40 |
| ○ Problem Statement 2 | 44 |
| ○ Problem Statement 3 | 47 |
| ○ Problem Statement 4 | 50 |

BIG DATA HADOOP AND SPARK DEVELOPMENT

1. Introduction

In this assignment, the given tasks are performed and Output of the tasks are recorded in the form of Screenshots.

2. Objective

This Assignment consolidates the deeper understanding of the Session – 7 Introduction to APACHE PIG

3. Problem Statement

- Task 1

- Write a program to implement wordcount using Pig

- Task 2

- We have employee_details and employee_expenses files. Use local mode while running Pig and write Pig Latin script to get below results:

employee_details (EmpID, Name, Salary, DepartmentID)

https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_details.txt

employee_expenses (EmpID, Expense)

https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_expense_s.txt

- (a) Top 5 employees (employee id and employee name) with highest rating. (In case two employees have same rating, employee with name coming first in dictionary should get preference)

- (b) Top 3 employees (employee id and employee name) with highest salary, whose employee id is an odd number. (In case two employees have same salary, employee with name coming first in dictionary should get preference)

- (c) Employee (employee id and employee name) with maximum expense (In case two employees have same expense, employee with name coming first in dictionary should get preference)
- (d) List of employees (employee id and employee name) having entries in [employee_expenses](#) file.
- (e) List of employees (employee id and employee name) having no entry in [employee_expenses](#) file.

- Task 3

- Implement the use case present in below blog link and share the complete steps along with screenshot(s) from your end.

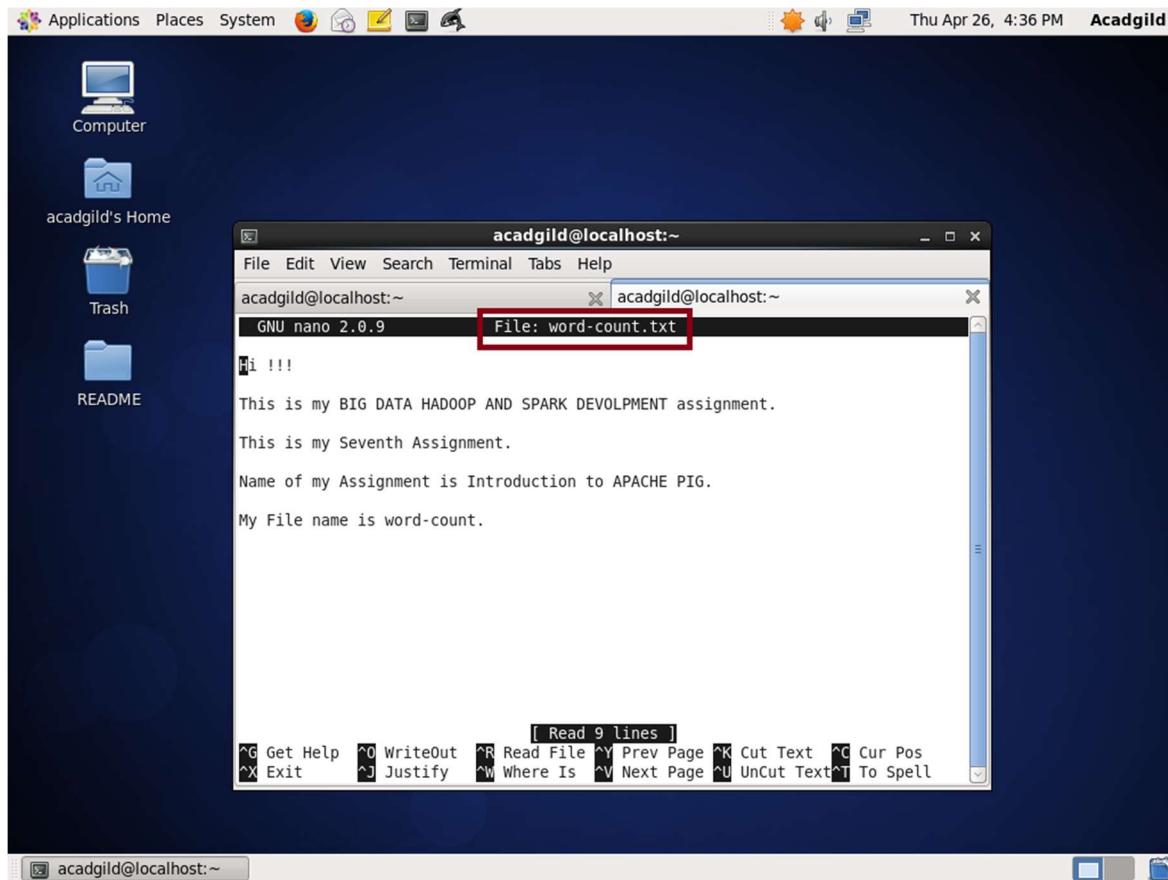
<https://acadgild.com/blog/aviation-data-analysis-using-apache-pig>

4. Expected Output

- Task 1

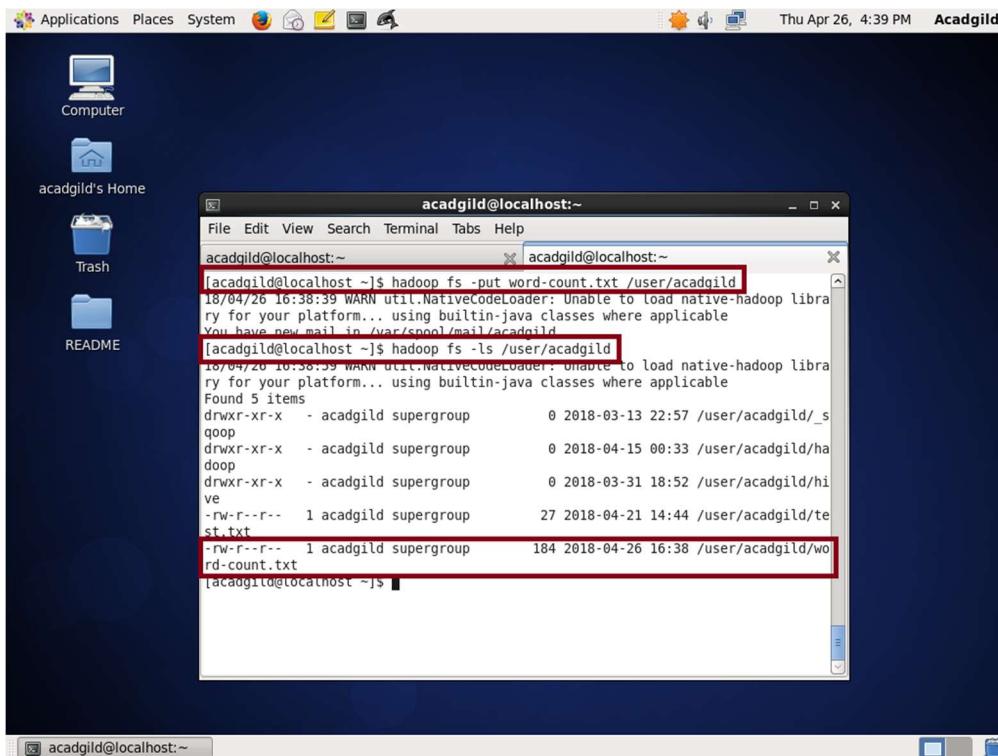
- i. Write a program to implement wordcount using Pig

Create the `word-count.txt` file using, `nano word-count.txt` command and save the text file with some text data.



Save the text file and copy the text file to Hadoop file system by using `-put` command. By using the following command, the `word-count.txt` is copied from local to Hadoop file system.

```
hadoop fs -put word-count.txt /user/acadgild
```



By using hadoop fs –ls /

We can list down the directories and files by using the **-ls /** command

We check the text file is copied or not by following command.

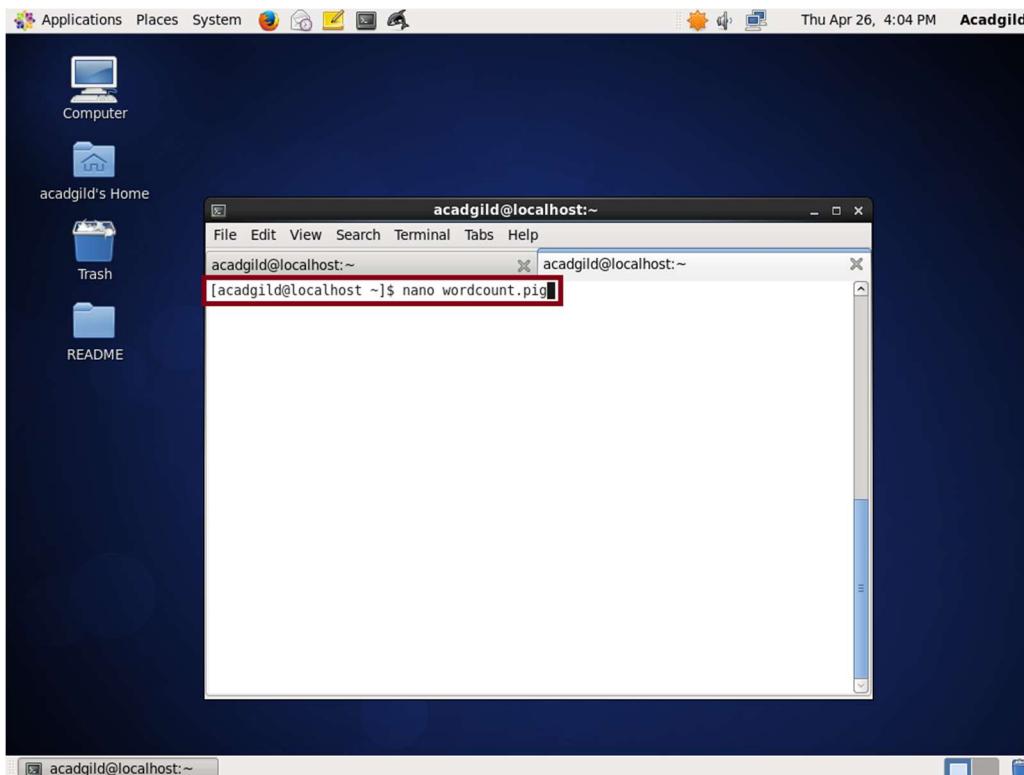
hadoop fs –ls /user/acadgild/word-count.txt

Creating a pig file:

Pig file is created with the extension of **.pig** , Pig file is created by using following command,

nano wordcount.pig

Create the **wordcount.pig** file using, **nano wordcount.pig** command and save the pig file with word count program codes to implement the wordcount using Pig.



Add the following codes to the pig file to implement the word count of the text input file

```
Lines = LOAD '/user/acadgild/word-count.txt' AS (line:chararray);

words = FOREACH Lines GENERATE FLATTEN(TOKENIZE(line)) as word;

grouped = GROUP words BY word;

wordcount = FOREACH grouped GENERATE group, COUNT(words);

DUMP wordcount;
```

- **TOKENIZE operator:**

The above pig script, first splits each line in the text file into words using the TOKENIZE operator. The tokenize function creates a bag of words.

- **FOREACH Operator:**

Use the FOREACH operator to work with columns of data.

- **GROUP Operator:**

Use the GROUP operator to group data in a single relation.

- **DUMP Operator:**

Use the DUMP operator to display results to a screen.

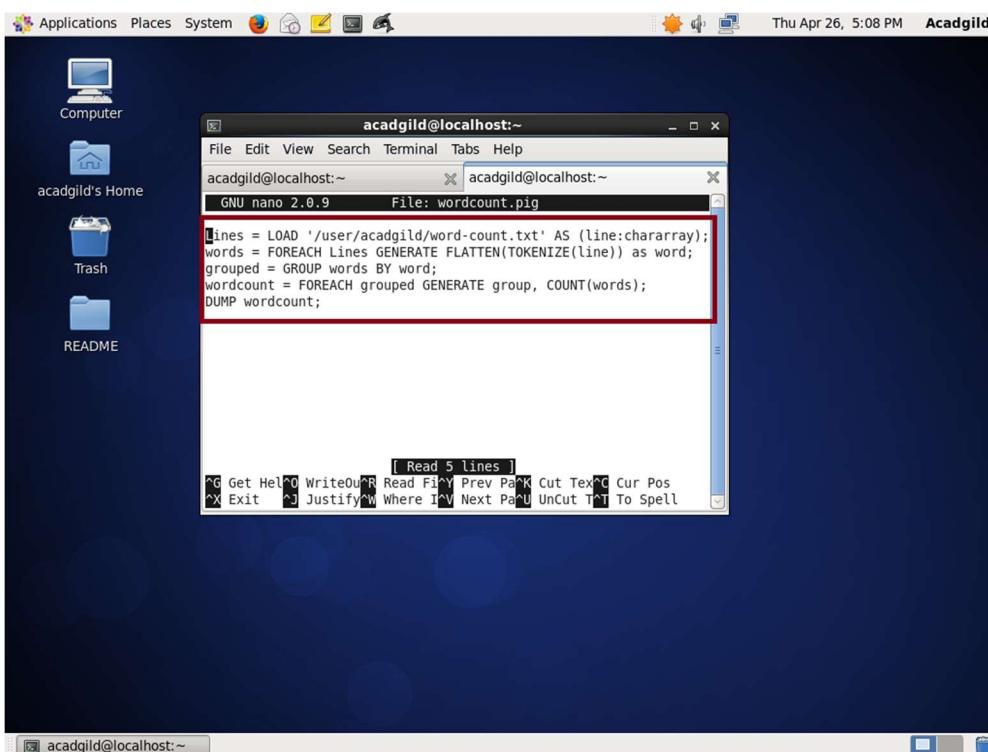
- **FLATTEN function:**

Using the **FLATTEN** function, the bag is converted into a tuple.

- **COUNT function:**

The **COUNT** function of Pig Latin is used to get the number of elements in a bag.

In the third statement, the words are grouped together so that the count can be computed in fourth statement.



In order to run or execute the pig script in local, the pig file has to be copied to the Hadoop distributed file system. By copying the pig file to Hadoop, the file can be executed from the grunt shell directly as pig local is situated top of the Hadoop.

By using,

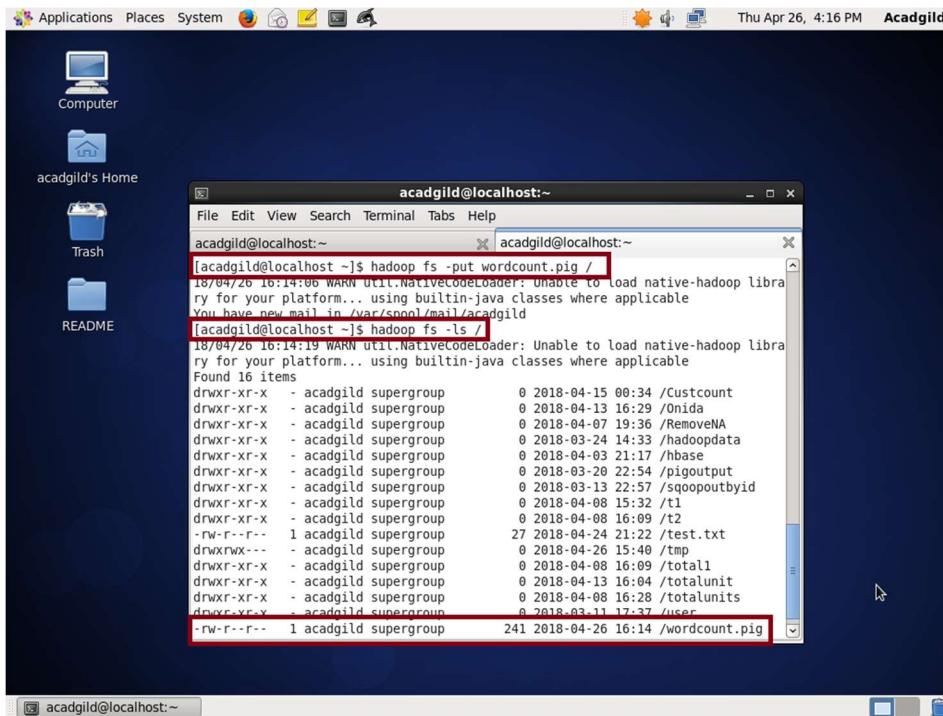
```
hadoop fs -put wordcount.pig /
```

The [wordcount.pig](#) is copied from local to Hadoop.

By using,

```
hadoop fs -ls /
```

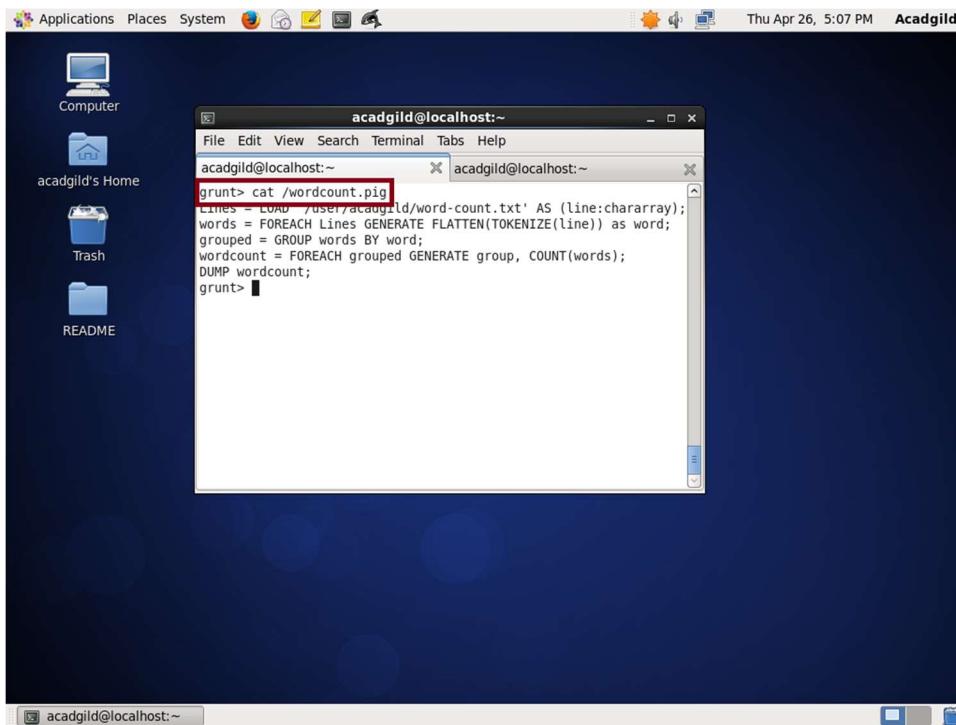
The [wordcount.pig](#) is listed in the Hadoop.



Grunt Mode or Interactive Mode:

The **grunt mode** can also be called as **interactive mode**. Grunt is pig's interactive shell. It is started when no file is specified for pig to run.

Using **cat** location path of **pig file** name, the file contents are shown in the **grunt console**. Below is the screenshot of the cat function shown in the grunt shell.



You can also run pig scripts from grunt using run and exec commands.

```
grunt> run scriptfile.pig  
grunt> exec scriptfile.pig
```

Utility Commands

exec - Run a Pig script.

Syntax

exec [-param param_name = param_value] [-param_file file_name] [script]

Terms

-param param_name = param_value

See Parameter Substitution.

-param_file file_name

See Parameter Substitution.

Script

The name of a Pig script.

Usage

Use the exec command to run a Pig script with no interaction between the script and the Grunt shell (batch mode).

Aliases defined in the script are not available to the shell. However, the files produced as the output of the script and stored on the system are visible after the script is run.

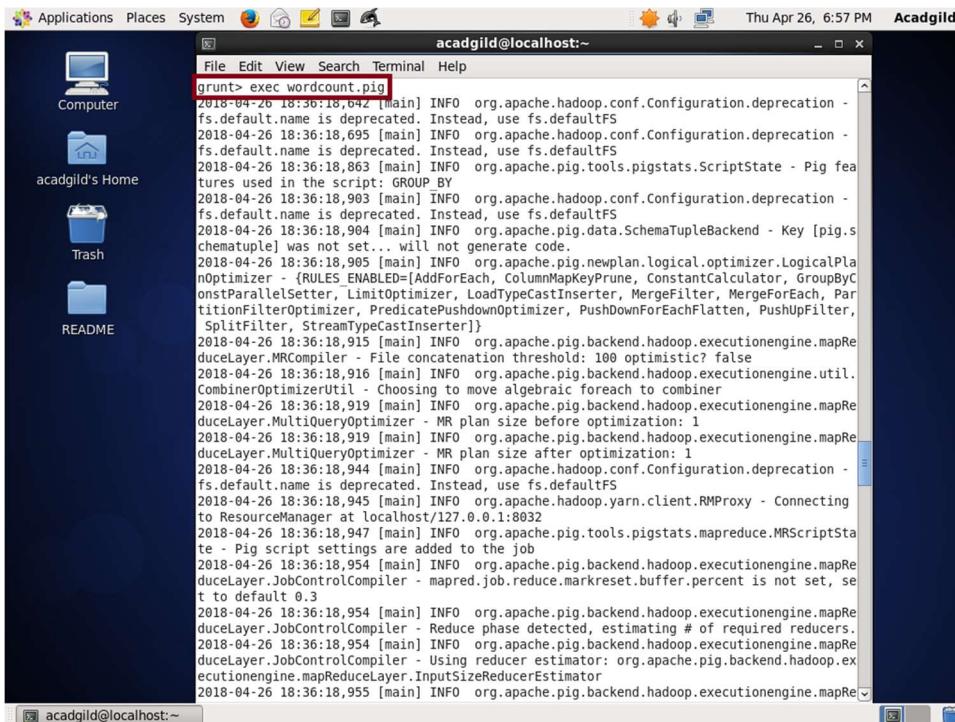
Aliases defined via the shell are not available to the script.

With the `exec` command, store statements will not trigger execution; rather, the entire script is parsed before execution starts.

Unlike the run command, exec does not change the command history or remembers the handles used inside the script.

Exec without any parameters can be used in scripts to force execution up to the point in the script where the exec occurs.

```
grunt> exec wordcount.pig
```



```

File Edit View Search Terminal Help
Thu Apr 26, 7:05 PM Acadgild
acadgild@localhost:~$ File Edit View Search Terminal Help
duceLayer.InputSizeReducerEstimator - BytesPerReducer=1000000000 maxReducers=999 totalInpu
putFileSize=192
2018-04-26 18:36:18,955 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - Setting Parallelism to 1
2018-04-26 18:36:18,956 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - This job cannot be converted run in-process
2018-04-26 18:36:19,073 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - Added jar file:/home/acadgild/install/pig/pig-0.16.0/pig-
0.16.0-core-h2.jar to DistributedCache through /tmp/temp-190214377/tmp-655610410/pig-0.1
6.0-core-h2.jar
2018-04-26 18:36:19,115 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - Added jar file:/home/acadgild/install/pig/pig-0.16.0/lib/
automaton-1.11-8.jar to DistributedCache through /tmp/temp-190214377/tmp-351359508/aut
omaton-1.11-8.jar
2018-04-26 18:36:19,157 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - Added jar file:/home/acadgild/install/pig/pig-0.16.0/lib/
antlr-runtime-3.4.jar to DistributedCache through /tmp/temp-190214377/tmp1447650838/antlr-
runtime-3.4.jar
2018-04-26 18:36:19,196 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - Added jar file:/home/acadgild/install/pig/pig-0.16.0/lib/
joda-time-2.9.3.jar to DistributedCache through /tmp/temp-190214377/tmp1762056144/joda-t
ime-2.9.3.jar
2018-04-26 18:36:19,201 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.JobControlCompiler - Setting up single store job
2018-04-26 18:36:19,204 [main] INFO org.apache.pig.SchemaTupleFrontend - Key [pig.schematuple] is false, will not generate code.
2018-04-26 18:36:19,204 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Starting process to move generated code to distributed cache
2018-04-26 18:36:19,204 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Setting k
ey [pig.schematuple.classes] with classes to deserialize []
2018-04-26 18:36:19,249 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - 1 map-reduce job(s) waiting for submission.
2018-04-26 18:36:19,258 [JobControl] INFO org.apache.hadoop.yarn.client.RMProxy - Conne
cting to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:19,291 [JobControl] WARN org.apache.hadoop.mapreduce.JobResourceUpload
er - No job jar file set. User classes may not be found. See Job or Job#setJar(String).
2018-04-26 18:36:19,311 [JobControl] INFO org.apache.pig.builtin.PigStorage - Using Pig
TextInputFormat
2018-04-26 18:36:19,317 [JobControl] INFO org.apache.hadoop.mapreduce.lib.input.FileInp
utFormat - Total input paths to process : 1
2018-04-26 18:36:19,317 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine
.util.MapRedUtil - Total input paths to process : 1
2018-04-26 18:36:19,319 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine
.util.MapRedUtil - Total input paths (combined) to process : 1
2018-04-26 18:36:19,370 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - nu
mber of splits:
2018-04-26 18:36:19,429 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - Su
bmitting tokens for job: job_1524729366758_0007
2018-04-26 18:36:19,438 [JobControl] INFO org.apache.hadoop.mapred.YARNRunner - Job jar
is not present. Not adding any jar to the list of resources.
2018-04-26 18:36:19,475 [JobControl] INFO org.apache.hadoop.yarn.client.api.impl.YarnCl
ientImpl - Submitted application application_1524729366758_0007
2018-04-26 18:36:19,479 [JobControl] INFO org.apache.hadoop.mapreduce.Job - The url to
track the job: http://localhost:8088/proxy/application_1524729366758_0007/
2018-04-26 18:36:19,745 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - HadoopJobId: job_1524729366758_0007
2018-04-26 18:36:19,746 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Processing aliases Lines,grouped,wordcount,words
2018-04-26 18:36:19,746 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - detailed locations: M: Lines[1..8],words[-1,-1],wordcount[4
..12],grouped[3..10] C: wordcount[4..12],grouped[3..10] R: wordcount[4..12]
2018-04-26 18:36:19,766 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - % complete
2018-04-26 18:36:19,766 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Running jobs are [job_1524729366758_0007]
2018-04-26 18:36:36,951 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - 50% complete
2018-04-26 18:36:36,951 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Running jobs are [job_1524729366758_0007]
2018-04-26 18:36:46,969 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Running jobs are [job_1524729366758_0007]
2018-04-26 18:36:49,979 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting
to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,005 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Ap
plication state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job histo

```

File input file one is processing. Mapreduce process is performed.

```

File Edit View Search Terminal Help
Thu Apr 26, 7:06 PM Acadgild
acadgild@localhost:~$ File Edit View Search Terminal Help
duceLayer.InputSizeReducerEstimator - BytesPerReducer=1000000000 maxReducers=999 totalInpu
putFileSize=192
2018-04-26 18:36:19,311 [JobControl] INFO org.apache.pig.builtin.PigStorage - Using Pig
TextInputFormat
2018-04-26 18:36:19,317 [JobControl] INFO org.apache.hadoop.mapreduce.lib.input.FileInp
utFormat - Total input paths to process : 1
2018-04-26 18:36:19,317 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine
.util.MapRedUtil - Total input paths to process : 1
2018-04-26 18:36:19,319 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine
.util.MapRedUtil - Total input paths (combined) to process : 1
2018-04-26 18:36:19,370 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - nu
mber of splits:
2018-04-26 18:36:19,429 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - Su
bmitting tokens for job: job_1524729366758_0007
2018-04-26 18:36:19,438 [JobControl] INFO org.apache.hadoop.mapred.YARNRunner - Job jar
is not present. Not adding any jar to the list of resources.
2018-04-26 18:36:19,475 [JobControl] INFO org.apache.hadoop.yarn.client.api.impl.YarnCl
ientImpl - Submitted application application_1524729366758_0007
2018-04-26 18:36:19,479 [JobControl] INFO org.apache.hadoop.mapreduce.Job - The url to
track the job: http://localhost:8088/proxy/application_1524729366758_0007/
2018-04-26 18:36:19,745 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - HadoopJobId: job_1524729366758_0007
2018-04-26 18:36:19,746 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Processing aliases Lines,grouped,wordcount,words
2018-04-26 18:36:19,746 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - detailed locations: M: Lines[1..8],words[-1,-1],wordcount[4
..12],grouped[3..10] C: wordcount[4..12],grouped[3..10] R: wordcount[4..12]
2018-04-26 18:36:19,766 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - % complete
2018-04-26 18:36:19,766 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Running jobs are [job_1524729366758_0007]
2018-04-26 18:36:36,951 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - 50% complete
2018-04-26 18:36:36,951 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Running jobs are [job_1524729366758_0007]
2018-04-26 18:36:46,969 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapRe
duceLayer.MapReduceLauncher - Running jobs are [job_1524729366758_0007]
2018-04-26 18:36:49,979 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting
to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,005 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Ap
plication state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job histo

```

Map reduce launcher shows 100% and Job status is shown with JobId, Maps, Reduces, Maxmaptime, MinMaptime, AvgMaptime, MedianMaptime, MaxReducetime, MinReducetime, AvgReducetime, MedianReducetime

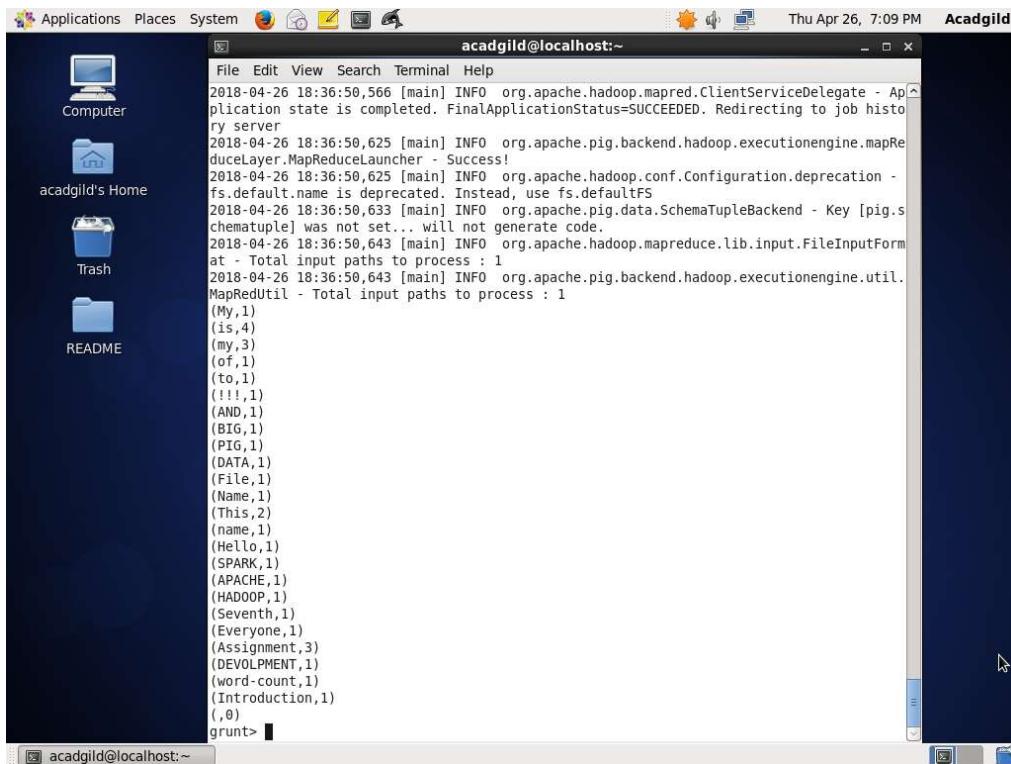
```
Applications Places System Terminal Thu Apr 26, 7:08 PM Acadgild
Computer acadgild's Home Trash README
File Edit View Search Terminal Help
acadgild@localhost:~ 
2018-04-26 18:36:50,005 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,257 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,266 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,341 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,349 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,393 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2018-04-26 18:36:50,397 [main] INFO org.apache.pig.tools.pigstats.mapreduce.SimplePigStats - Script Statistics:
HadoopVersion PigVersion UserId StartedAt FinishedAt Features
2.6.5 0.16.0 acadgild 2018-04-26 18:36:18 2018-04-26 18:36:50 GROUP_BY
Success!
Job Stats (time in seconds):
JobId Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime M axReduceTime MinReduceTime AvgReduceTime MedianReduceTime Alias FeatureO utputs
job_1524729366758_0007 1 1 5 5 5 5 6 6 6
6 Lines,grouped,wordcount,words GROUP_BY,COMBINER hdfs://localhost:8020/tmp/temp-190214377/tmp1077550077,
Input(s):
Successfully read 9 records (569 bytes) from: "/user/acadgild/word-count.txt"
Output(s):
Successfully stored 25 records (323 bytes) in: "hdfs://localhost:8020/tmp/temp-190214377/tmp1077550077"
Counters:
```

```
Applications Places System Terminal Thu Apr 26, 7:09 PM Acadgild
Computer acadgild's Home Trash README
File Edit View Search Terminal Help
acadgild@localhost:~ 
Counters:
Total records written : 25
Total bytes written : 323
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0
Job DAG:
job_1524729366758_0007

2018-04-26 18:36:50,398 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,417 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,487 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,495 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,551 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0.0.1:8032
2018-04-26 18:36:50,566 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,625 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-26 18:36:50,625 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-26 18:36:50,633 [main] INFO org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not generate code.
2018-04-26 18:36:50,643 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-26 18:36:50,643 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(My,1)
(is,4)
(my,3)
```

Result is shown below for wordcount.pig

The word count for the input file which is processed as shown below



A screenshot of a Linux desktop environment. On the left, there's a 'Places' panel with icons for Computer, acadgild's Home, Trash, and README. The main area shows a terminal window titled 'acadgild@localhost:~'. The terminal displays the following output:

```
File Edit View Search Terminal Help
2018-04-26 18:36:50,566 [main] INFO org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2018-04-26 18:36:50,625 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-26 18:36:50,625 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-26 18:36:50,633 [main] INFO org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not generate code.
2018-04-26 18:36:50,643 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-26 18:36:50,643 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(Hello,1)
(SPARK,1)
(APACHE,1)
(HADOOP,1)
(Seventh,1)
(Everyone,1)
(Assignment,3)
(DEVOLPMENT,1)
(word-count,1)
(Introduction,1)
(,0)
grunt>
```

- **Task 2:**

- We have employee_details and employee_expenses files. Use local mode while running Pig and write Pig Latin script to get below results:

- employee_details (EmpID, Name, Salary, DepartmentID)
https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_details.txt
- employee_expenses (EmpID, Expense)
https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_expenses.txt

Load

To load the data either from local filesystem or Hadoop filesystem.

Syntax:

LOAD 'path_of_data' [USING function] [AS schema];

Where;

path_of_data : file/directory name in single quotes.

USING : is the keyword.

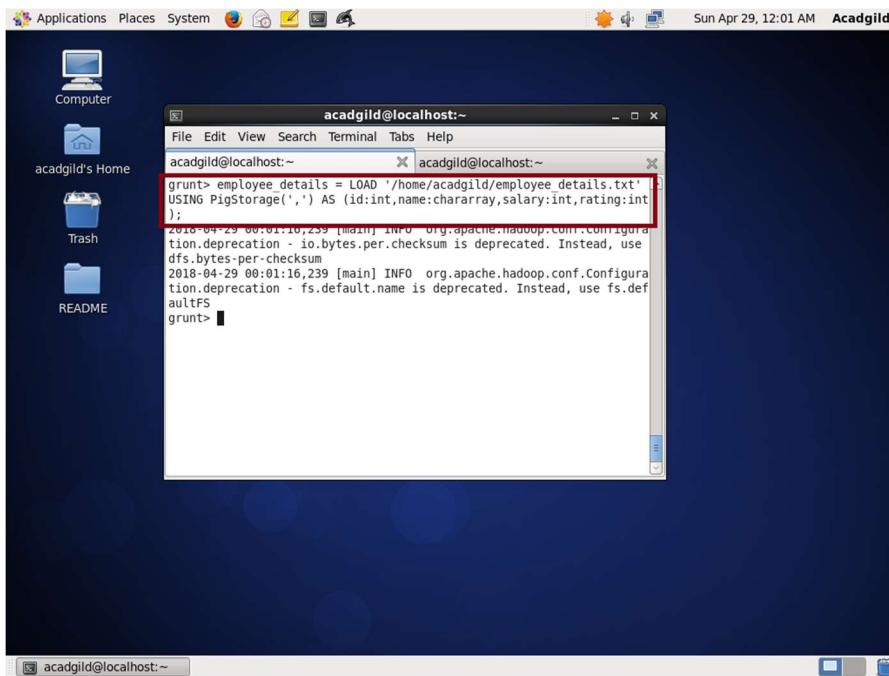
function : If you choose to omit this, default load function PigStorage() is used.

AS : is the keyword

schema : schema of your data along with data type.

The file named employee_details.txt is comma separated file and we are going to load it from local file system.

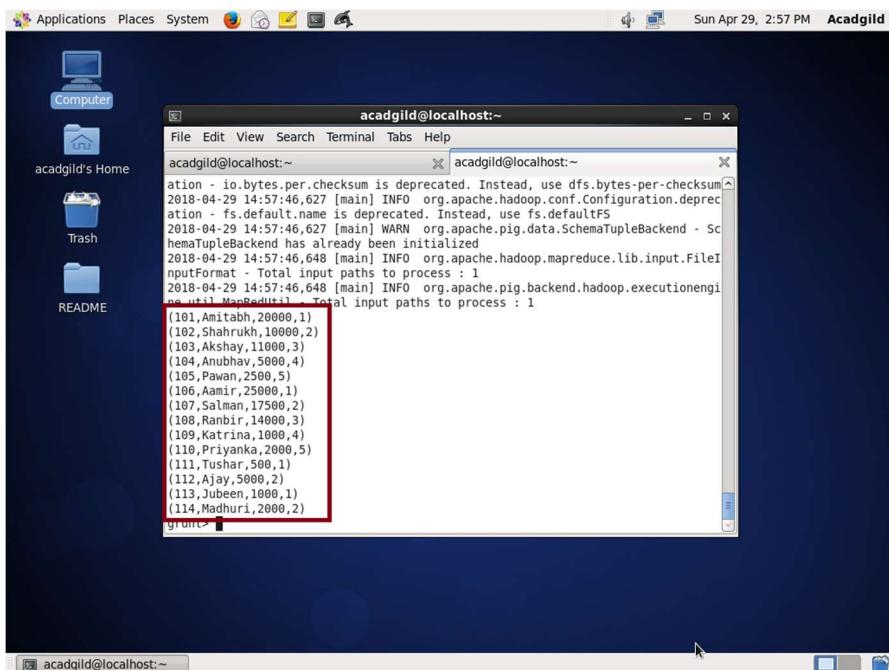
```
employee_details = LOAD '/home/acadgild/employee_details.txt' USING PigStorage(',') AS  
(id:int,name:chararray,salary:int,rating:int);
```



```
grunt> employee_details = LOAD '/home/acadgild/employee_details.txt'
USING PigStorage(',') AS (id:int,name:chararray,salary:int,rating:int);
;
```

2018-04-29 00:01:16,239 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-29 00:01:16,239 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS

To check the result, you can use DUMP command.

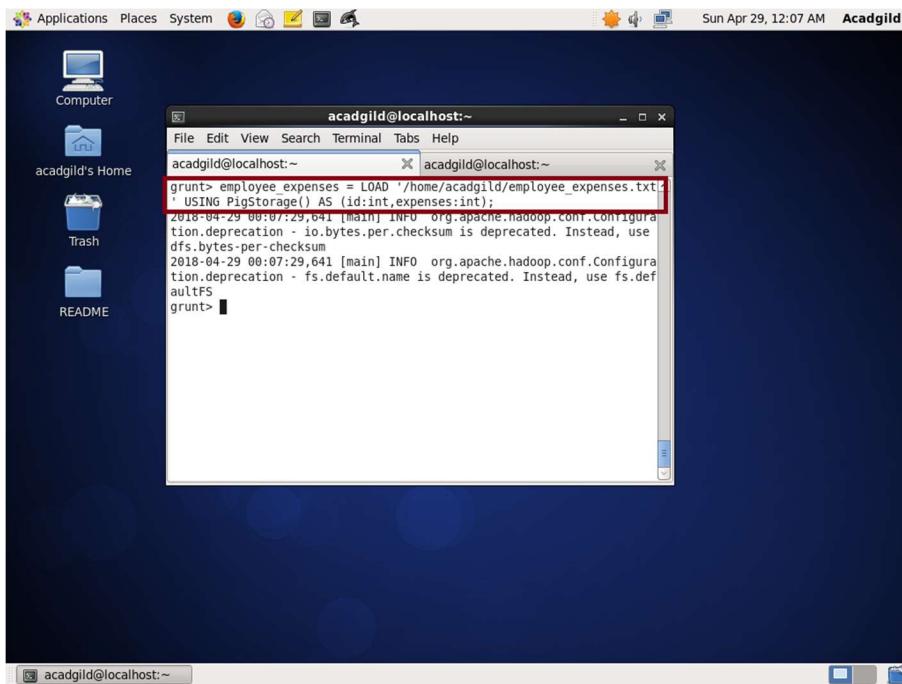


```
action - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-29 14:57:46,627 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-29 14:57:46,627 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-29 14:57:46,648 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-29 14:57:46,648 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceUtil - Total input paths to process : 1
(101,Amitabh,20000,1)
(102,Shahrukh,10000,2)
(103,Akshay,11000,3)
(104,Anubhav,5000,4)
(105,Pawan,25000,5)
(106,Aamir,25000,1)
(107,Salman,17500,2)
(108,Ranbir,14000,3)
(109,Katrina,1000,4)
(110,Priyanka,2000,5)
(111,Tushar,500,1)
(112,Ajay,5000,2)
(113,Jubeen,1000,1)
(114,Madhuri,2000,2)
```

Similarly, you can load another data into another relation, say employee_expenses.txt

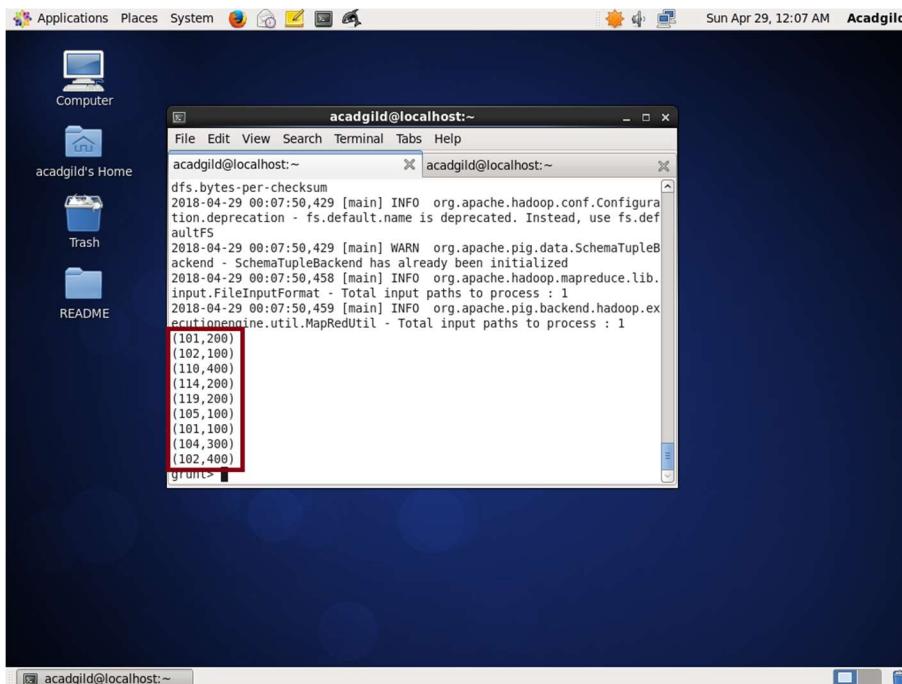
The file named employee_expenses.txt is tab separated file and we are going to load it from local file system.

```
employee_expenses = LOAD '/home/acadgild/employee_expenses.txt' USING PigStorage() AS  
(id:int,expenses:int);
```



```
acadgild@localhost:~  
acadgild@localhost:~  
grunt> employee_expenses = LOAD '/home/acadgild/employee_expenses.txt'  
' USING PigStorage() AS (id:int,expenses:int);  
2018-04-29 00:07:29,641 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use  
dfs.bytes-per-checksum  
2018-04-29 00:07:29,641 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.def  
aultFS  
grunt>
```

To check the result, you can use DUMP command.



```
acadgild@localhost:~  
acadgild@localhost:~  
dfs.bytes-per-checksum  
2018-04-29 00:07:50,429 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.def  
aultFS  
2018-04-29 00:07:50,429 [main] WARN org.apache.pig.data.SchemaTupleB  
ackend - SchemaTupleBackend has already been initialized  
2018-04-29 00:07:50,458 [main] INFO org.apache.hadoop.mapreduce.lib.  
input.FileInputFormat - Total input paths to process : 1  
2018-04-29 00:07:50,459 [main] INFO org.apache.pig.backend.hadoop.ex  
ecutionengine.util.MapRedUtil - Total input paths to process : 1  
(101,200)  
(102,100)  
(110,400)  
(114,200)  
(119,200)  
(105,100)  
(101,100)  
(104,300)  
(102,400)  
grunt>
```

- (a) Top 5 employees (employee id and employee name) with highest rating. (In case two employees have same rating, employee with name coming first in dictionary should get preference)

Order

Sorts a relation based on single or multiple fields.

Syntax:

alias = ORDER alias BY {field_name [ASC | DESC]}

Where;

alias : is the relation.

ORDER : is the keyword.

BY : is the keyword.

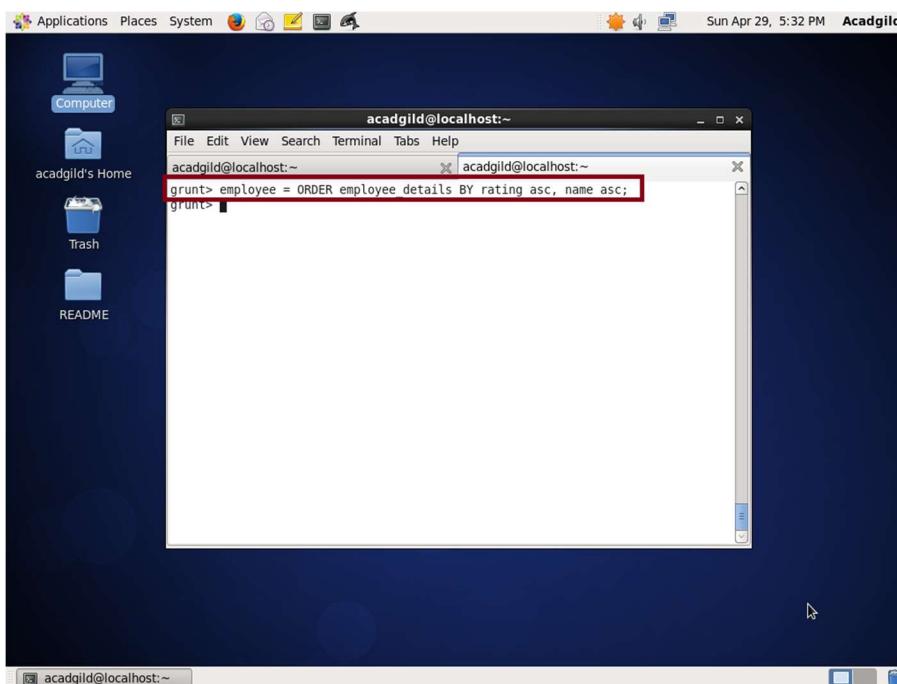
field_name : column on which you want to sort the relation.

ASC : sort in ascending order.

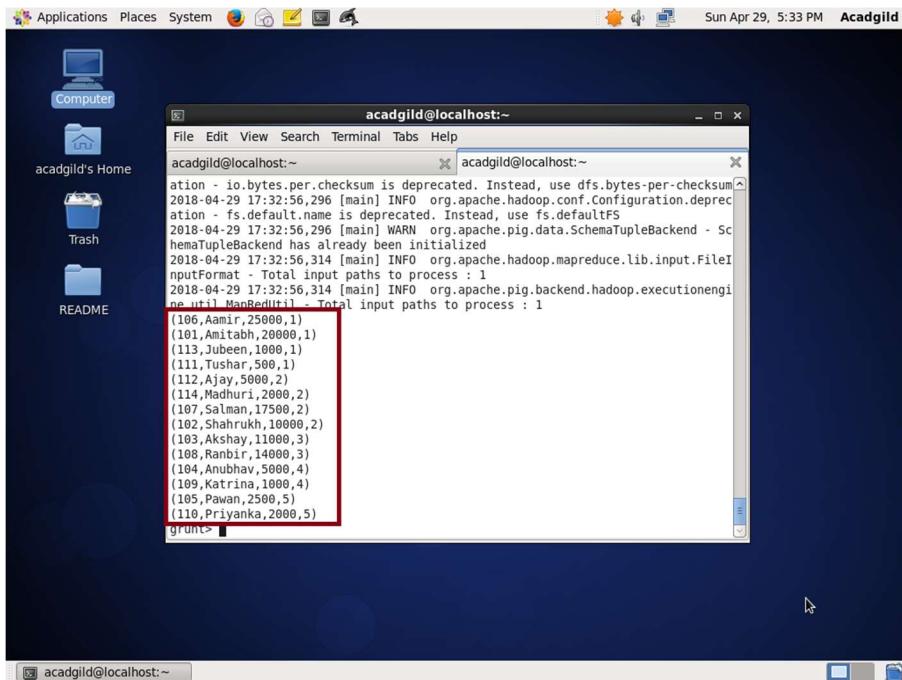
DESC : sort in descending order.

By using the following command the employee table is sorted.

employee = ORDER employee_details BY rating asc, name asc



To check the result, you can use DUMP command.



```
acadgild@localhost:~
```

```
action - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-29 17:32:56,296 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-29 17:32:56,296 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-29 17:32:56,314 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-29 17:32:56,314 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapReduceUtil - Total input paths to process : 1
(106,Aamir,25000,1)
(101,Amitabh,20000,1)
(113,Jubeen,1000,1)
(111,Tushar,500,1)
(112,Ajay,5000,2)
(114,Madhuri,2000,2)
(107,Salman,17500,2)
(102,Shahrukh,10000,2)
(103,Akshay,11000,3)
(108,Ranbir,14000,3)
(104,Anubhav,5000,4)
(109,Katrina,1000,4)
(105,Pawan,2500,5)
(110,Priyanka,2000,5)
grunt>
```

Limit

Used to limit the number of outputs to the desired number.

Syntax:

Alias = LIMIT alias n;

Where;

alias : name of the relation.

n : number of tuples to be displayed.

By using the following the command we can Limit the top 5 employee

Limited= Limit employee 5;

A screenshot of a Linux desktop environment. On the left is a vertical file manager sidebar with icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the command "grunt> Limited = Limit employee 5;" with the last part highlighted by a red box. The desktop background is dark blue.

To check the result, you can use DUMP command.

A screenshot of a Linux desktop environment. On the left is a vertical file manager sidebar with icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the command "grunt> Limited = Limit employee 5;" followed by a large amount of Hadoop Metrics log output. At the bottom of the terminal window, the results of the "DUMP" command are shown, with the entire list of employees highlighted by a red box. The desktop background is dark blue.

| ID | Name | Salary | Count |
|----------------|-----------|--------|-------|
| (106, Aamir, | 25000, 1) | | |
| (101, Amitabh, | 20000, 1) | | |
| (113, Jubeen, | 1000, 1) | | |
| (111, Tushar, | 500, 1) | | |
| (112, Ajay, | 5000, 2) | | |

Foreach

It generates data transformations based on desired columns of data.

Syntax:

alias = FOREACH alias GENERATE {expression | field};

Where;

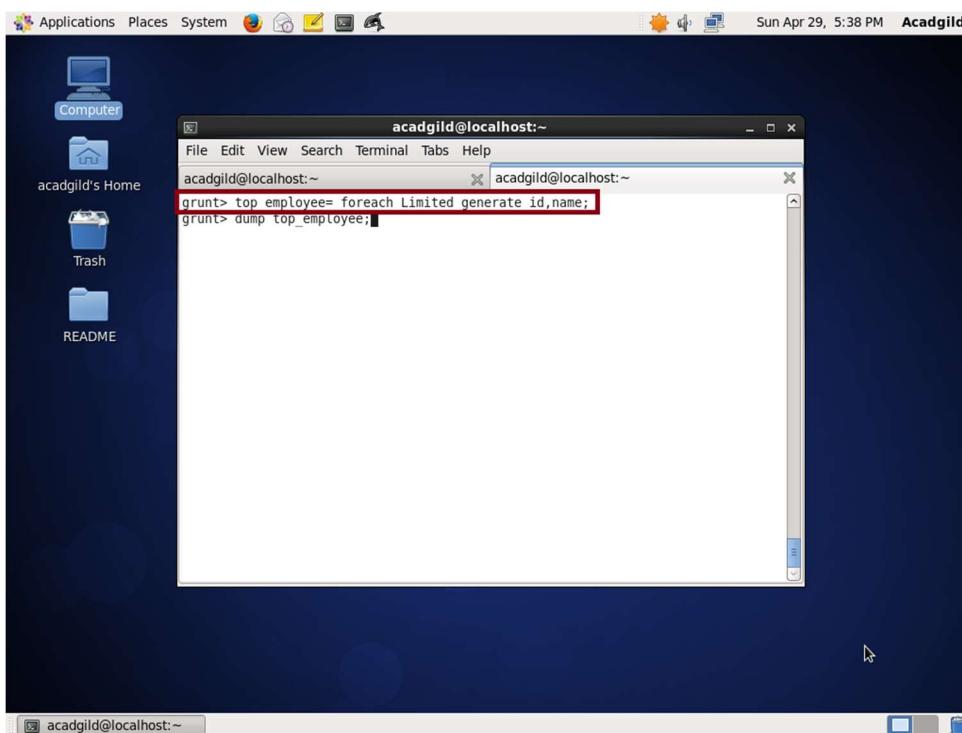
alias : is the relation

FOREACH : is the keyword

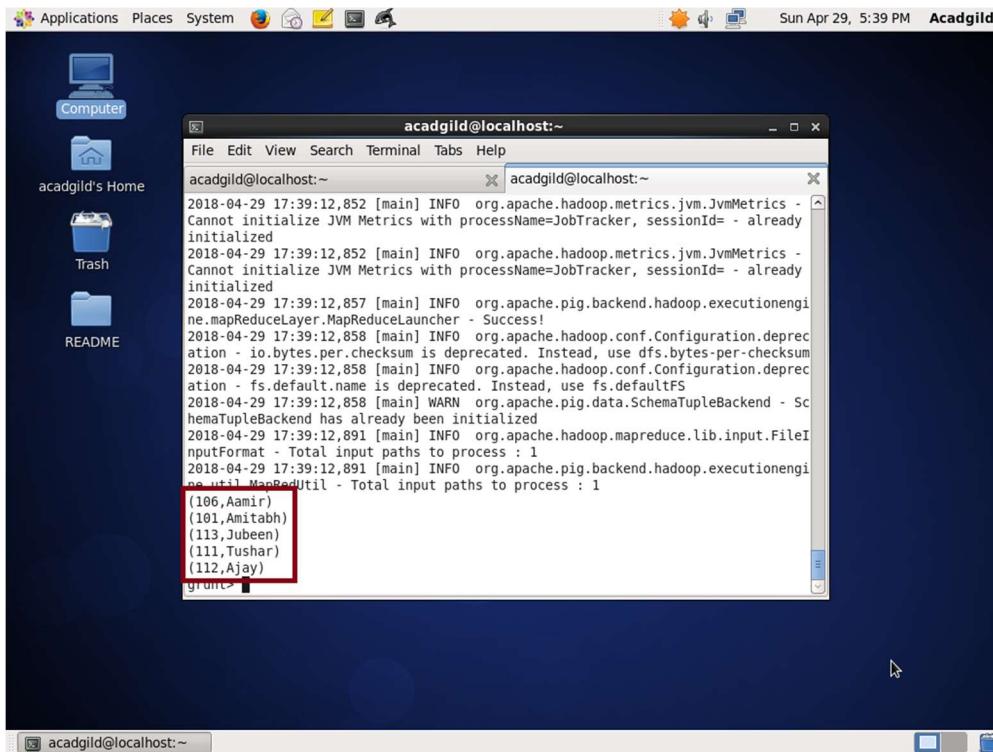
GENERATE : is the keyword

By using the following command value of top employees are stored in **top_employee**

top_employee= foreach Limited generate id,name;



To check the result, you can use DUMP command

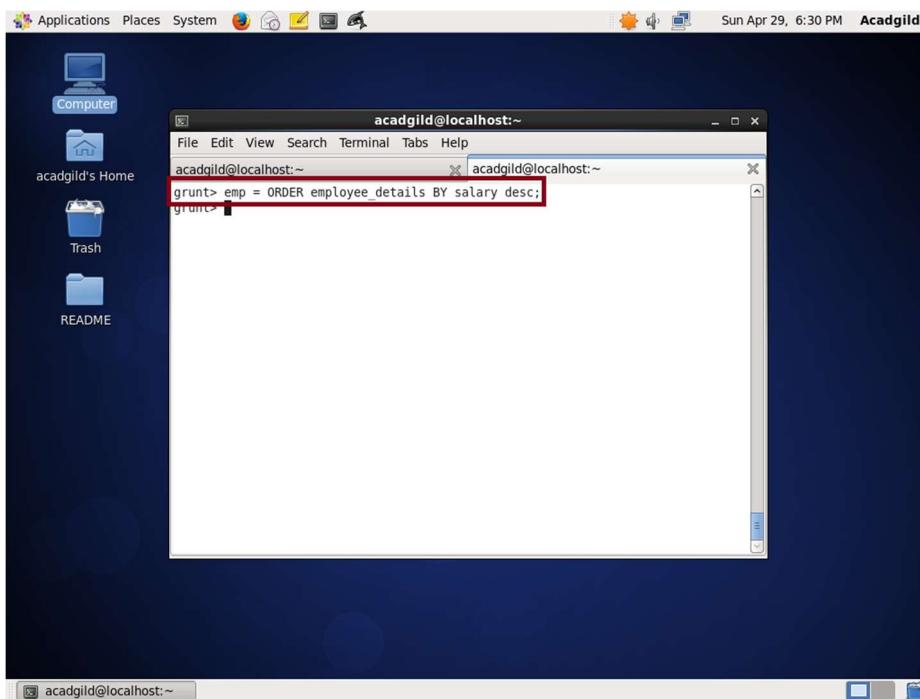


- **Top 3 employees (employee id and employee name) with highest salary, whose employee id is an odd number. (In case two employees have same salary, employee with name coming first in dictionary should get preference)**

The table of employee details are populated by using ORDER BY relation to get the highest salary details of employees.

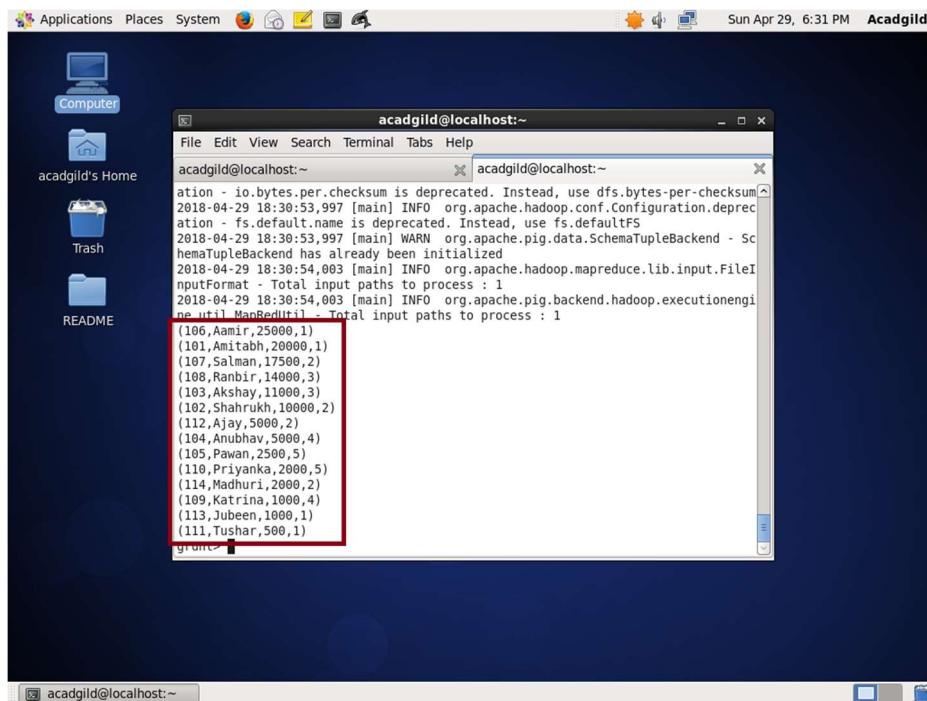
The following command explains the above,

emp = ORDER employee_details BY salary desc;



To check the result, you can use DUMP command

Here we get the list of employee details with highest salary.

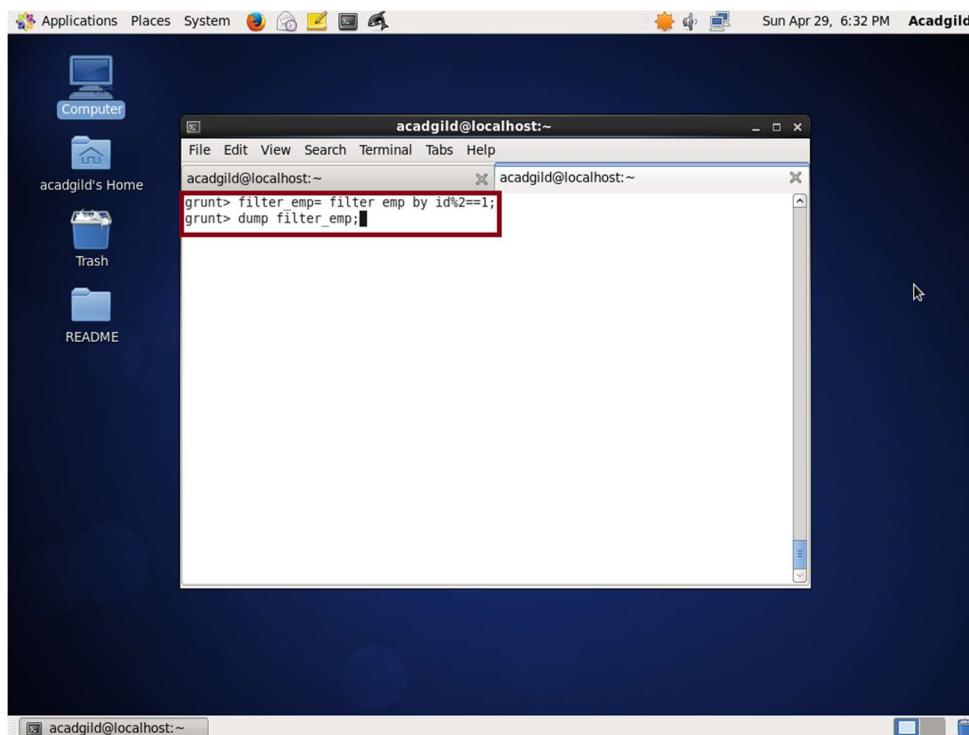


A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the following output:

```
action - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-29 18:30:53,997 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultF
2018-04-29 18:30:53,997 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-29 18:30:54,003 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2018-04-29 18:30:54,003 [main] INFO org.apache.pig.backend.hadoop.executionengi
ne_util.MapReduceUtil - Total input paths to process : 1
(106,Aamir,25000,1)
(101,Amitabh,20000,1)
(107,Salman,17500,2)
(108,Ranbir,14000,3)
(103,Akshay,11000,3)
(102,Shahrukh,10000,2)
(112,Ajay,5000,2)
(104,Anubhav,5000,4)
(105,Pawan,2500,5)
(110,Priyanka,2000,5)
(114,Madhuri,2000,2)
(109,Katrina,1000,4)
(113,Jubeen,1000,1)
(111,Tushar,500,1)
grunt>
```

By using the below filter relation, we can get the employee details with highest salary, whose id are odd

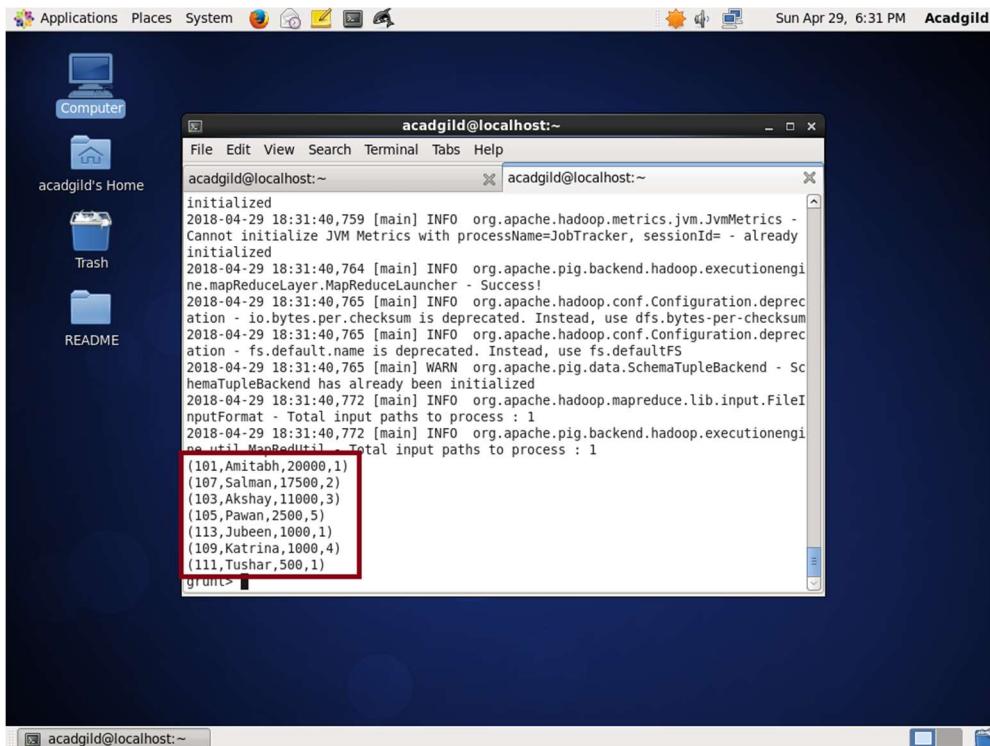
`filter_emp=filter emp by id%2==1;`



A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the following output:

```
acadgild@localhost:~ acadgild@localhost:~ acadgild@localhost:~
grun
filter_emp= filter emp by id%2==1;
grun> dump filter_emp;
```

To check the result, you can use DUMP command

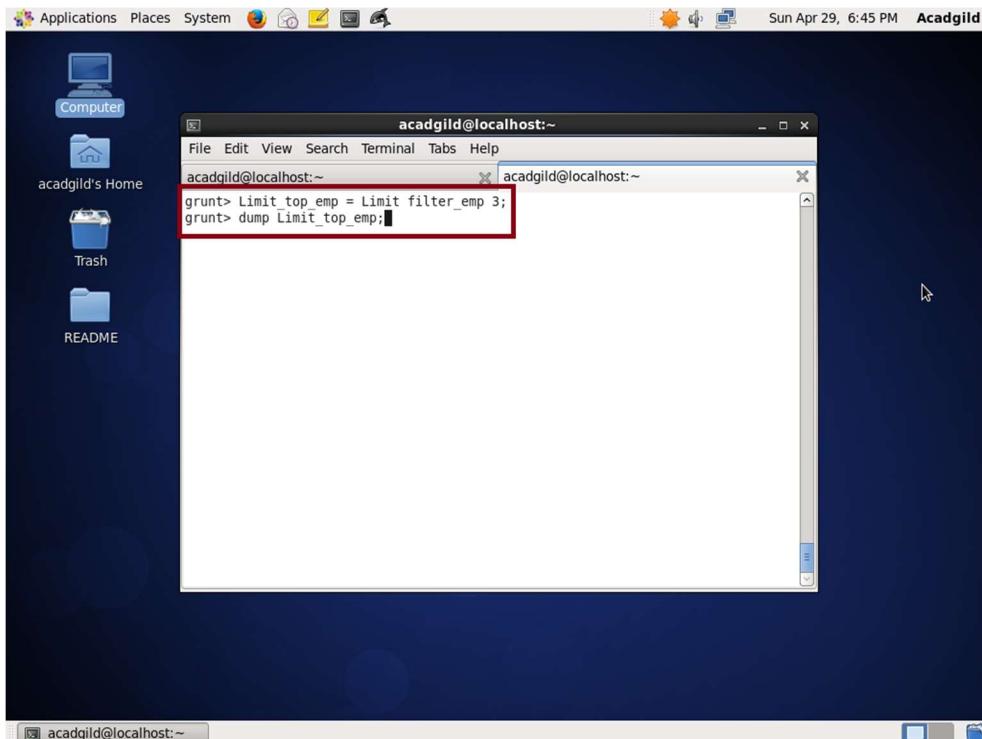


A screenshot of a Linux desktop environment. On the left, there's a file manager window titled "Computer" showing icons for "Computer", "acadgild's Home", "Trash", and "README". In the center, there's a terminal window titled "acadgild@localhost:~". The terminal shows the following text:

```
initialized
2018-04-29 18:31:40,759 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-29 18:31:40,764 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-29 18:31:40,765 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-29 18:31:40,765 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-29 18:31:40,765 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-29 18:31:40,772 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-29 18:31:40,772 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapReduceUtil - Total input paths to process : 1
(101,Anitabh,20000,1)
(107,Salman,17500,2)
(103,Akshay,11000,3)
(105,Pawan,2500,5)
(113,Jubeen,1000,1)
(109,Katrina,1000,4)
(111,Tushar,500,1)
grunt>
```

By using **Limit top_emp = Limit filter_emp 3;**

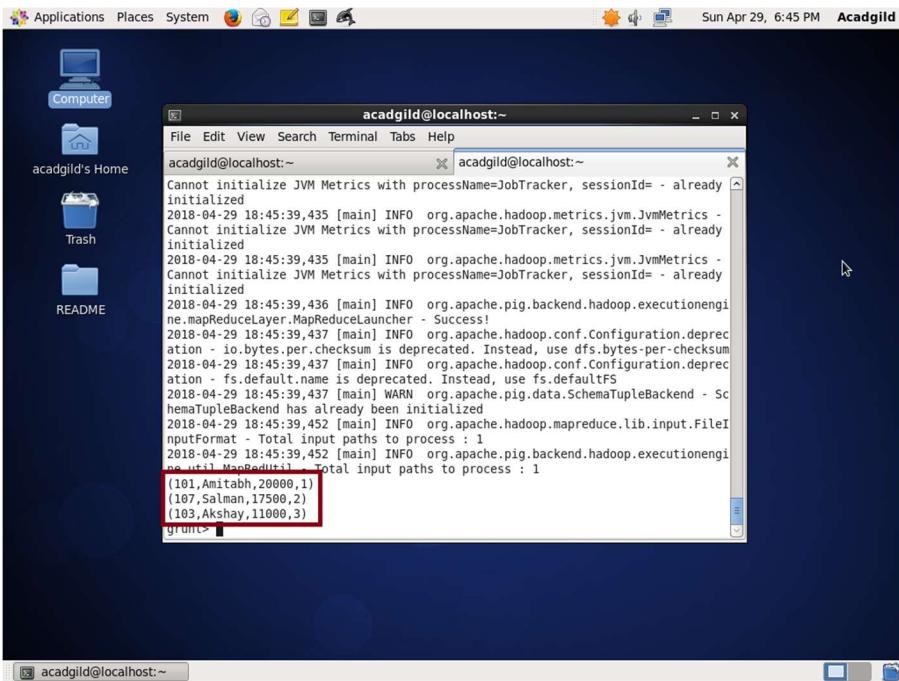
We can limit the top 3 employees with high salary.



A screenshot of a Linux desktop environment. On the left, there's a file manager window titled "Computer" showing icons for "Computer", "acadgild's Home", "Trash", and "README". In the center, there's a terminal window titled "acadgild@localhost:~". The terminal shows the following text:

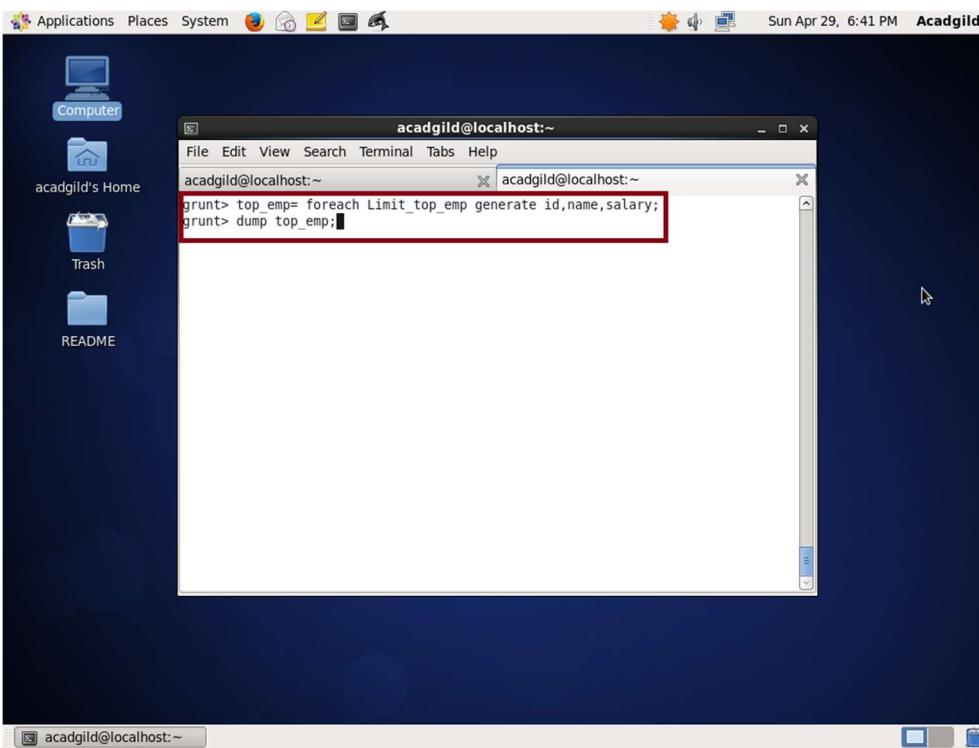
```
acadgild@localhost:~ acadgild@localhost:~
grunt> Limit top_emp = Limit filter_emp 3;
grunt> dump Limit_top_emp;
```

To check the result, you can use DUMP command



```
acadgild@localhost:~ acadgild@localhost:~  
Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized  
2018-04-29 18:45:39,435 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized  
2018-04-29 18:45:39,435 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized  
2018-04-29 18:45:39,436 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!  
2018-04-29 18:45:39,437 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum  
2018-04-29 18:45:39,437 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS  
2018-04-29 18:45:39,437 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized  
2018-04-29 18:45:39,452 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1  
2018-04-29 18:45:39,452 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapReduceUtil - Total input paths to process : 1  
(101,Amitabh,20000,1)  
(107,Salman,17500,2)  
(103,Akshay,11000,3)  
grunt>
```

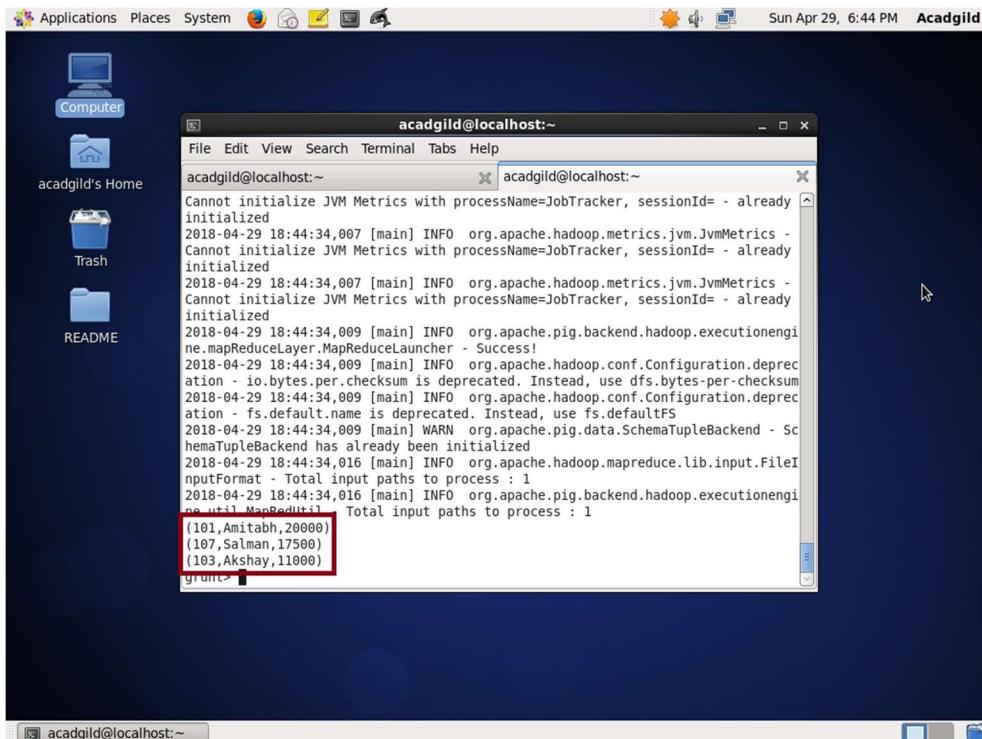
To specifically print the employee id and employee name with high salary, FOREACH is used to GENERATE the result



```
acadgild@localhost:~ acadgild@localhost:~  
grunt> top_emp= foreach Limit_top_emp generate id,name,salary;  
grunt> dump top_emp;
```

top_emp = foreach Limit_top_emp generate id, name , salary;

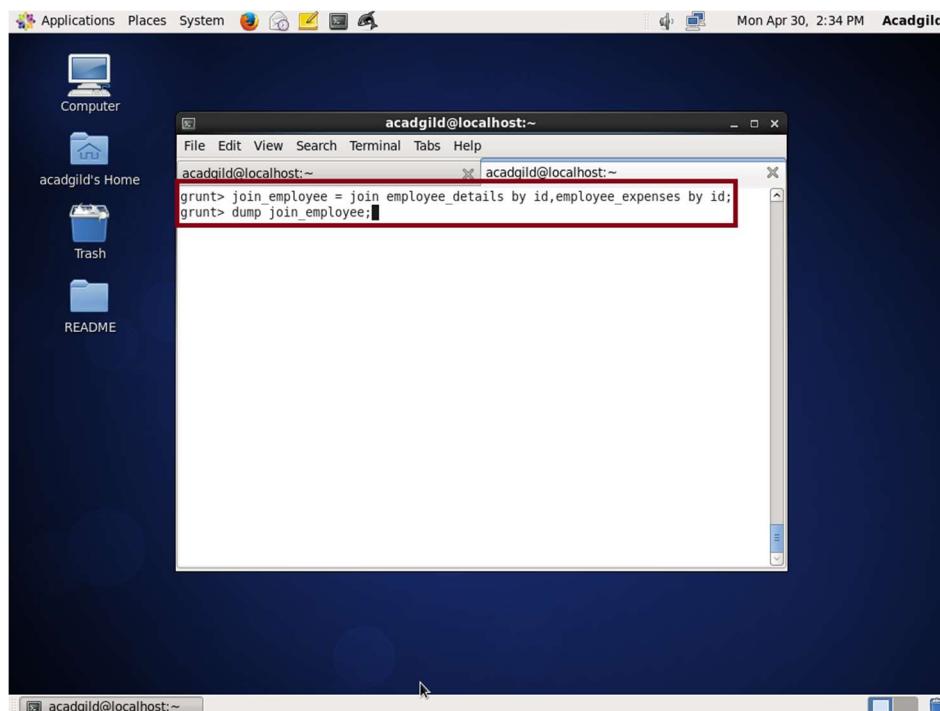
To check the result, you can use DUMP command



- (c) Employee (employee id and employee name) with maximum expense (In case two employees have same expense, employee with name coming first in dictionary should get preference)

Join relation is used to join two table, when both the table contents match.

It creates a new relation by combining column values of two relations (say A and B) based upon the join-predicate. The query compares each row of A with each row of B to find all pairs of rows which satisfy the join-predicate. When the join-predicate is satisfied, the column values for each matched pair of rows of A and B are combined into a result row.



A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the following command history:

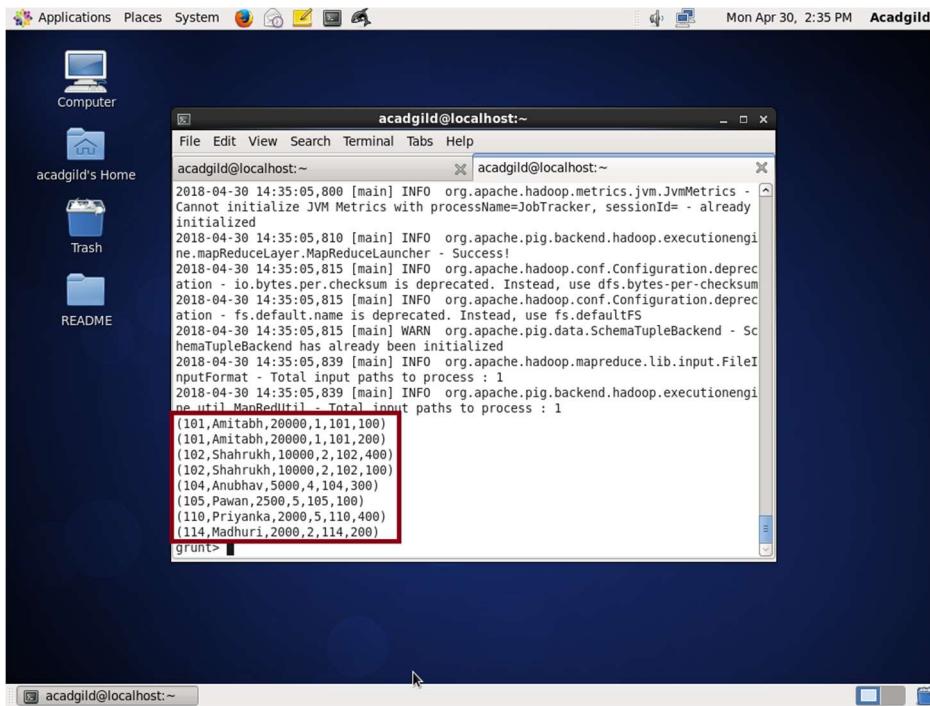
```
acadgild@localhost:~$ File Edit View Search Terminal Tabs Help
acadgild@localhost:~$ grunt> join_employee = join employee_details by id,employee_expenses by id;
acadgild@localhost:~$ grunt> dump join employee;
```

The last two lines of the command history are highlighted with a red rectangle.

By using the following command, the two tables are joined by matching pair in the rows of employee details and employee expenses are combined into result.

[join_employee = join employee_details by id, employee_expenses by id;](#)

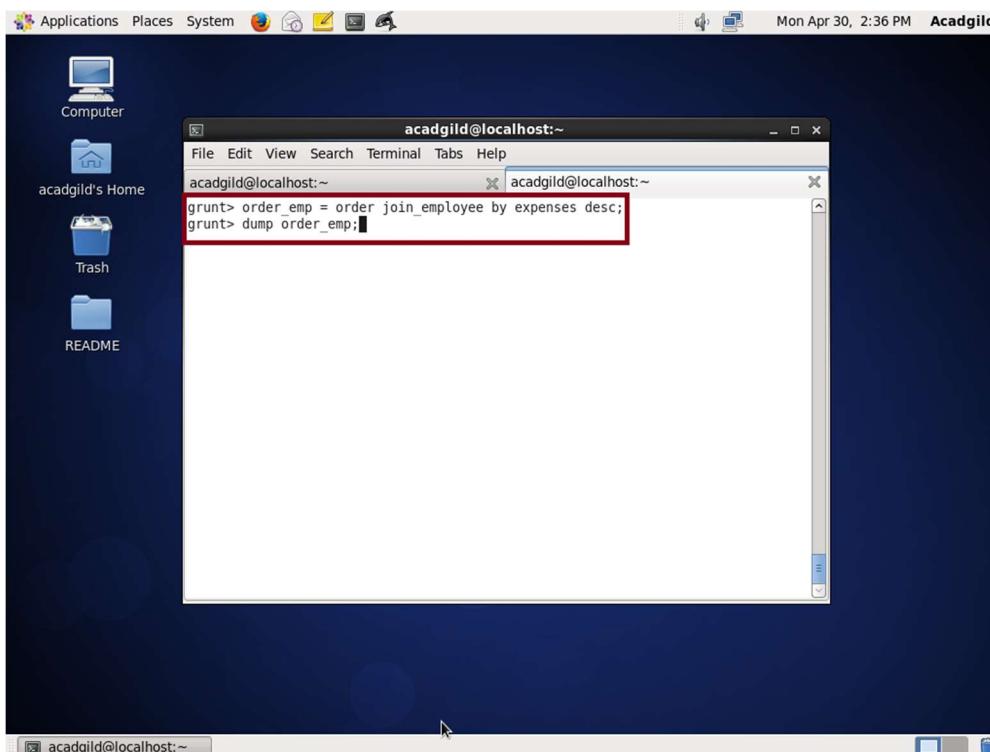
To check the result, you can use DUMP command



A screenshot of a Linux desktop environment. On the left is a file manager window showing icons for Computer, acadgild's Home, Trash, and README. To its right is a terminal window titled "acadgild@localhost:~". The terminal displays the following text:

```
2018-04-30 14:35:05,800 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:35:05,810 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 14:35:05,815 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 14:35:05,815 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 14:35:05,815 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 14:35:05,839 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 14:35:05,839 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(101,Anitabh,20000,1,101,100)
(101,Anitabh,20000,1,101,200)
(102,Shahrukh,10000,2,102,400)
(102,Shahrukh,10000,2,102,100)
(104,Anubhav,5000,4,104,300)
(105,Pawan,2500,5,105,100)
(110,Priyanka,2000,5,110,400)
(114,Madhuri,2000,2,114,200)
grunt>
```

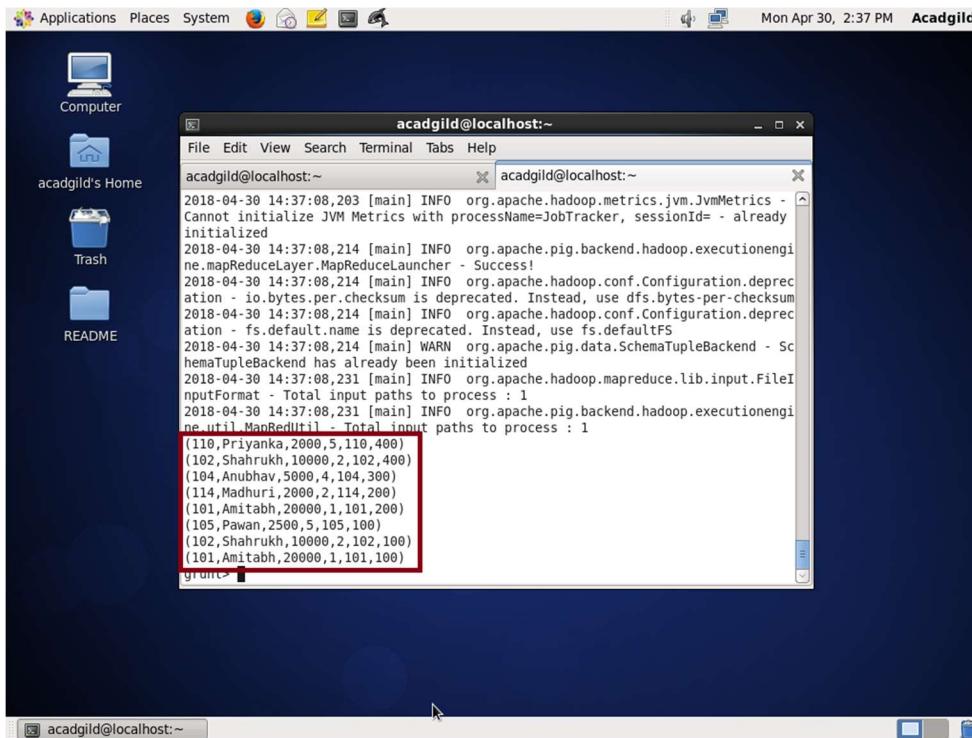
By using the following command, the result which is obtained by the
[order_emp = order join_employee by expenses desc;](#)



A screenshot of a Linux desktop environment. On the left is a file manager window showing icons for Computer, acadgild's Home, Trash, and README. To its right is a terminal window titled "acadgild@localhost:~". The terminal displays the following text:

```
2018-04-30 14:35:05,800 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:35:05,810 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 14:35:05,815 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 14:35:05,815 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 14:35:05,815 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 14:35:05,839 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 14:35:05,839 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
grunt> order_emp = order join_employee by expenses desc;
grunt> dump order_emp;
```

To check the result, you can use DUMP command

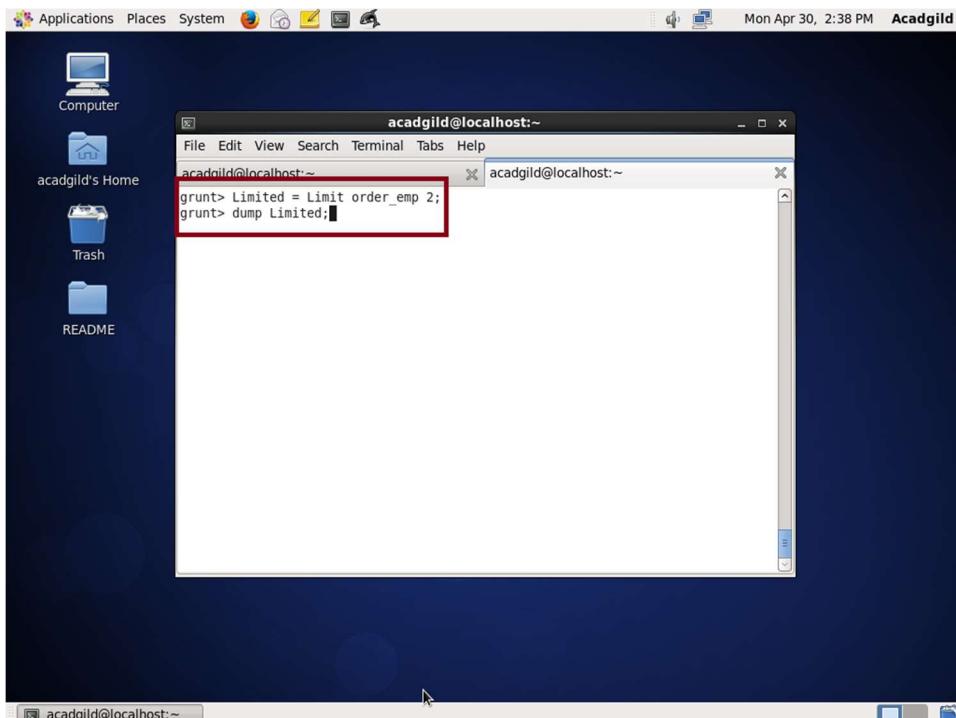


A screenshot of a Linux desktop environment. The desktop background is dark blue. On the left, there is a vertical dock with icons for Computer, acadgild's Home, Trash, and README. In the center, there is a terminal window titled "acadgild@localhost:~". The terminal shows the following text:

```
File Edit View Search Terminal Tabs Help
acadgild@localhost:~          acadgild@localhost:~
2018-04-30 14:37:08,203 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:37:08,214 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 14:37:08,214 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 14:37:08,214 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 14:37:08,214 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 14:37:08,231 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 14:37:08,231 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(110,Priyanka,2000,5,110,400)
(102,Shahrukh,10000,2,102,400)
(104,Anubhav,5000,4,104,300)
(114,Madhuri,2000,2,114,200)
(101,Amitabh,20000,1,101,200)
(105,Pawan,2500,5,105,100)
(102,Shahrukh,10000,2,102,100)
(101,Amitabh,20000,1,101,100)
grunt>
```

By using Limit operation, the data are limited to 2.

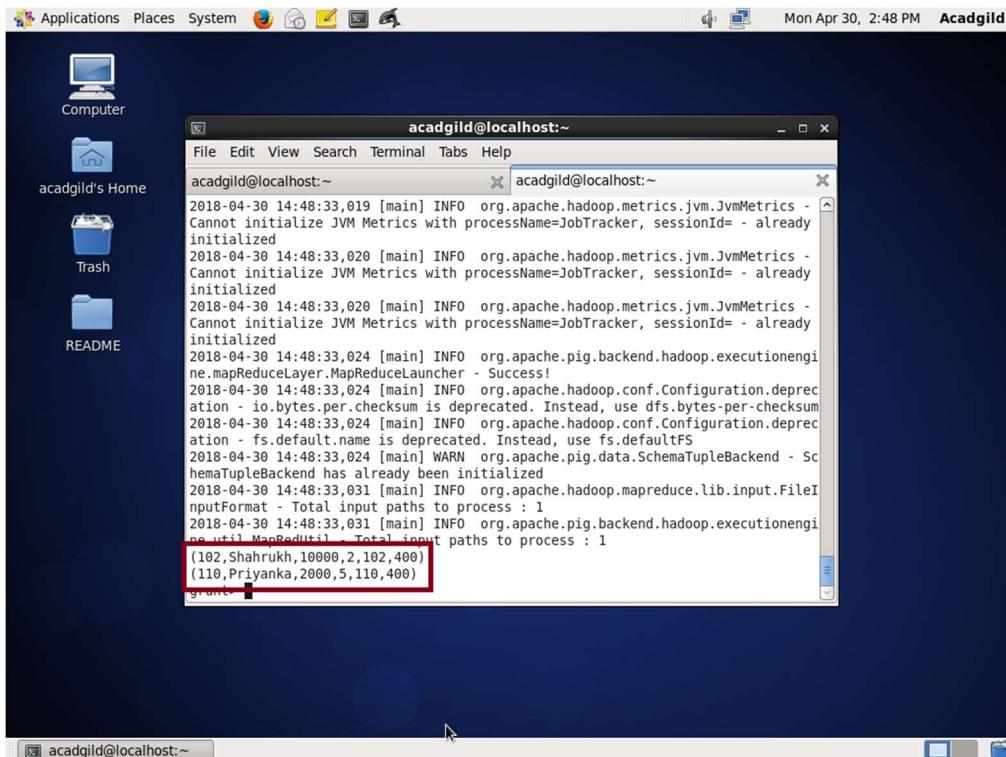
Limited = Limit order_emp 2;



A screenshot of a Linux desktop environment. The desktop background is dark blue. On the left, there is a vertical dock with icons for Computer, acadgild's Home, Trash, and README. In the center, there is a terminal window titled "acadgild@localhost:~". The terminal shows the following text:

```
File Edit View Search Terminal Tabs Help
acadgild@localhost:~          acadgild@localhost:~
grunt> Limited = Limit order_emp 2;
grunt> dump Limited;
```

To check the result, you can use DUMP command

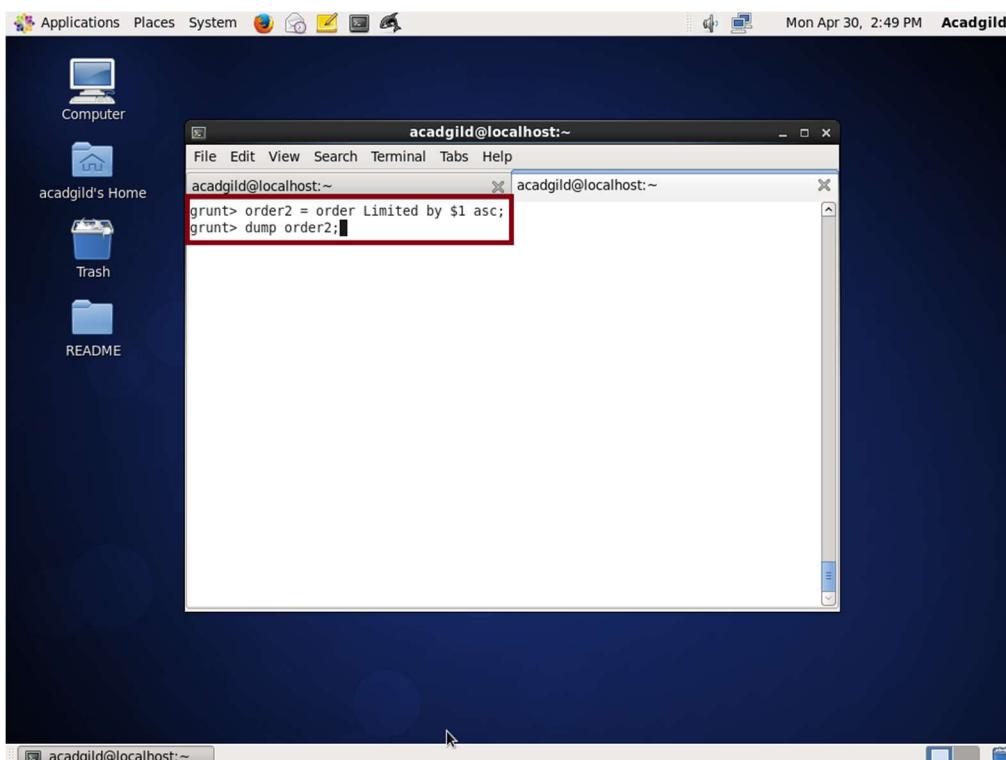


The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "acadgild@localhost:~". The terminal displays the following log output:

```
2018-04-30 14:48:33,019 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:48:33,020 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:48:33,020 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:48:33,024 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 14:48:33,024 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 14:48:33,024 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 14:48:33,024 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 14:48:33,031 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 14:48:33,031 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(102,Shahrukh,10000,2,102,400)
(110,Priyanka,2000,5,110,400)
```

The last two lines of the log, which show the data tuples, are highlighted with a red box.

Here the two employees have same expense, employee with name coming first in dictionary should get preference. Hence we use ORDER operation to get the Ascending order of the data obtained.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "acadgild@localhost:~". The terminal displays the following Pig Latin commands:

```
grunt> order2 = order Limited by $1 asc;
grunt> dump order2;
```

The first command, "order2 = order Limited by \$1 asc;", is highlighted with a red box.

To check the result, you can use DUMP command

A screenshot of a Linux desktop environment. On the left is a file manager window showing 'Computer', 'acadgild's Home', 'Trash', and 'README'. In the center is a terminal window titled 'acadgild@localhost:~'. The terminal shows Hadoop startup logs and then displays the output of a 'dump' command:

```
2018-04-30 14:51:56,051 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:51:56,054 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:51:56,054 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:51:56,055 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 14:51:56,055 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 14:51:56,055 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 14:51:56,055 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 14:51:56,065 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 14:51:56,065 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapReduceUtil - Total input paths to process : 1
(110,Priyanka,2000,5,110,400)
(102,Shahrukh,10000,2,102,400)
```

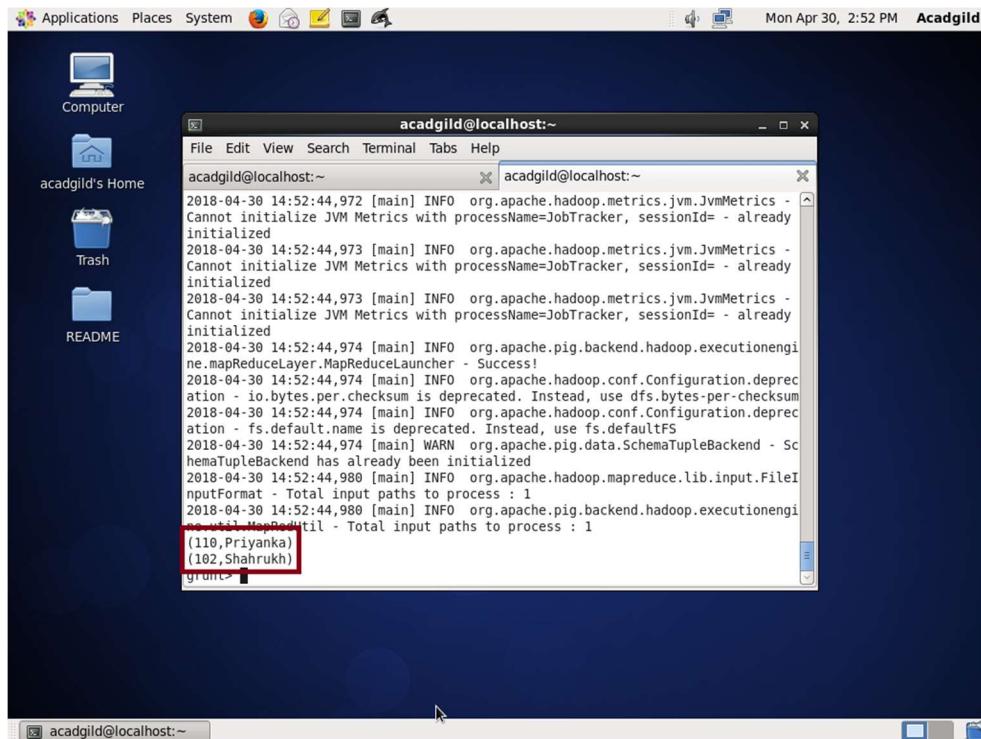
The Maximum Expense made by the employee are displayed with employee id and employee name.

max_emp = foreach order2 generate \$0,\$1;

A screenshot of a Linux desktop environment. On the left is a file manager window showing 'Computer', 'acadgild's Home', 'Trash', and 'README'. In the center is a terminal window titled 'acadgild@localhost:~'. The terminal shows a Pig Latin script being run:

```
grunt> max_emp = foreach order2 generate $0,$1;
grunt> dump max_emp;
```

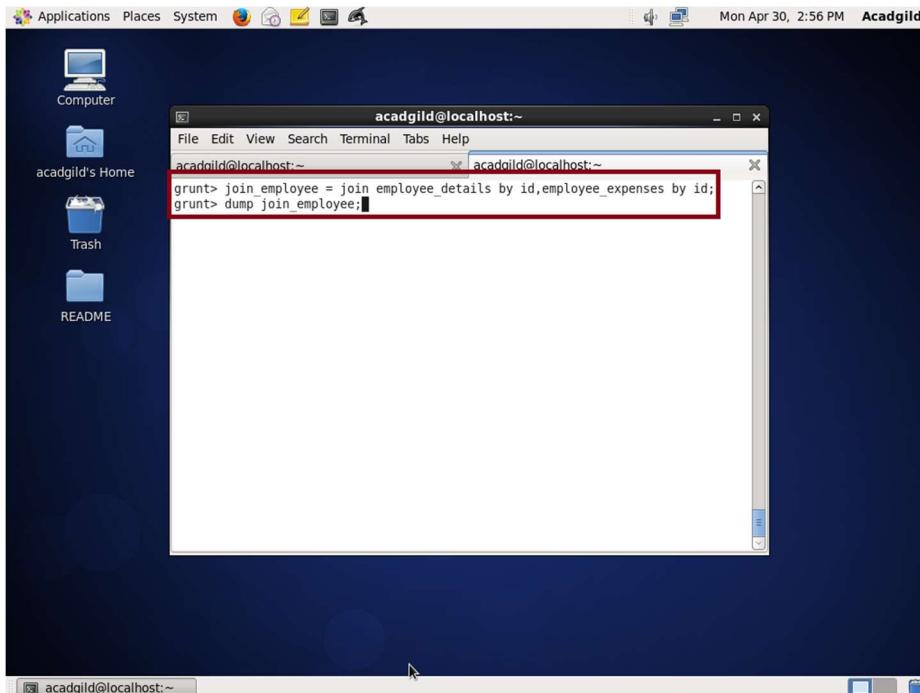
To check the result, you can use DUMP command



- (d) List of employees (employee id and employee name) having entries in employee_expenses file.

By using the join operator, the tables are joined and the data are obtained.

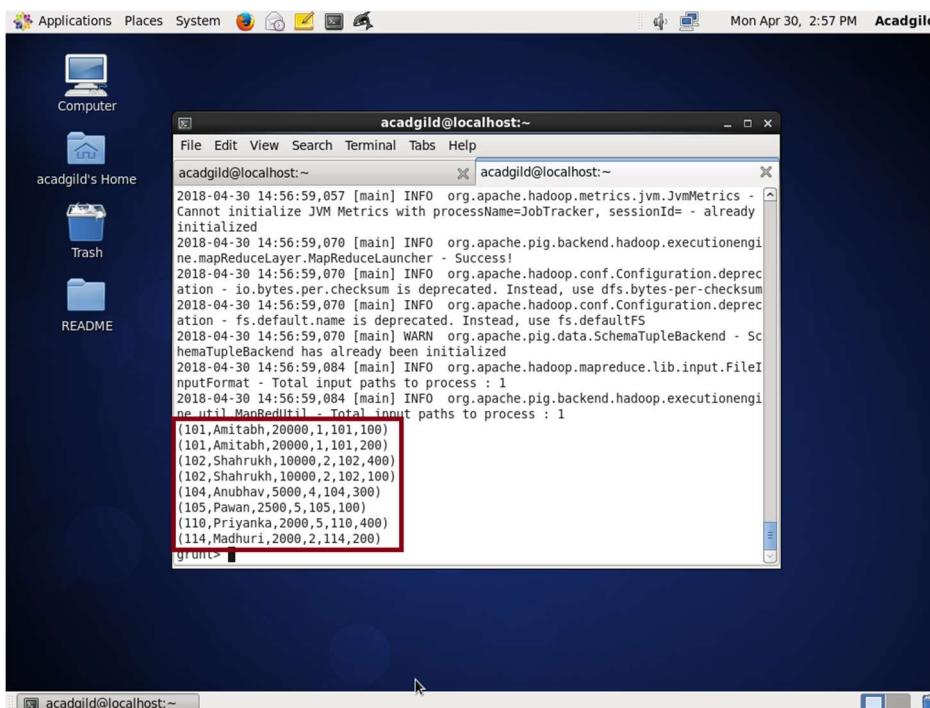
`join_employee = join employee_details by id, employee_expenses by id;`



```
acadgild@localhost:~
```

```
join_employee = join employee_details by id, employee_expenses by id;
grunt> dump join_employee;
```

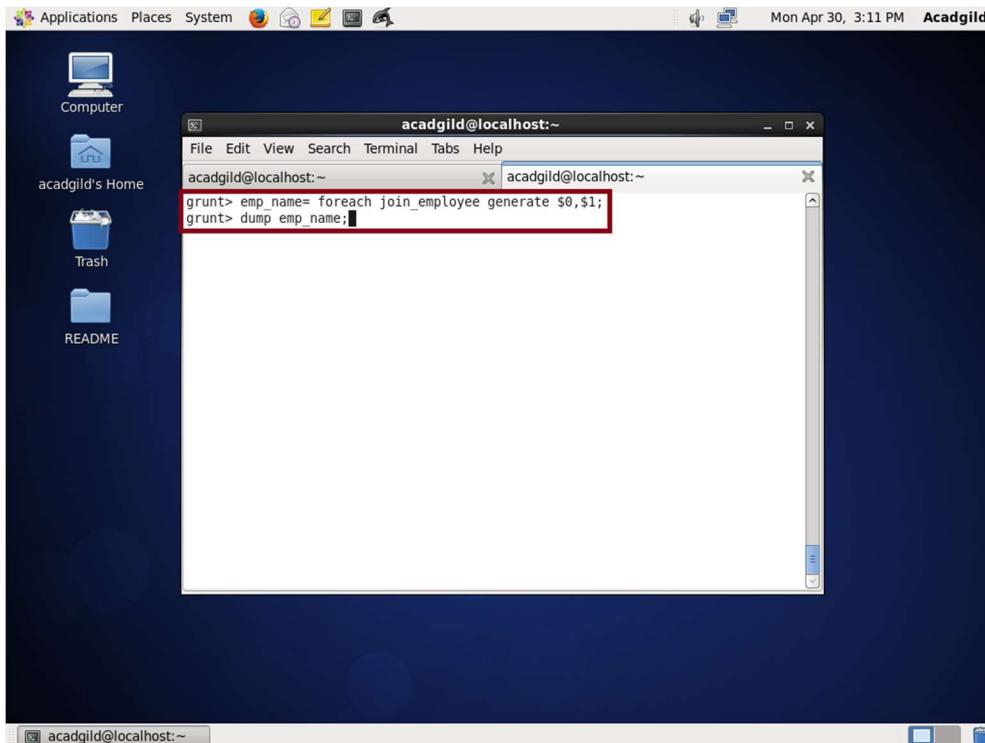
To check the result, you can use DUMP command



```
2018-04-30 14:56:59,057 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 14:56:59,070 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 14:56:59,070 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 14:56:59,070 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 14:56:59,070 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 14:56:59,084 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 14:56:59,084 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(101,Amitabh,20000,1,101,100)
(101,Amitabh,20000,1,101,200)
(102,Shahrukh,10000,2,102,400)
(102,Shahrukh,10000,2,102,100)
(104,Anubhav,5000,4,104,300)
(105,Pawan,2500,5,105,100)
(110,Priyanka,2000,5,110,400)
(114,Madhuri,2000,2,114,200)
```

By using the below command, we can generate the employee id and employee name who have the entries in the employee expenses table.

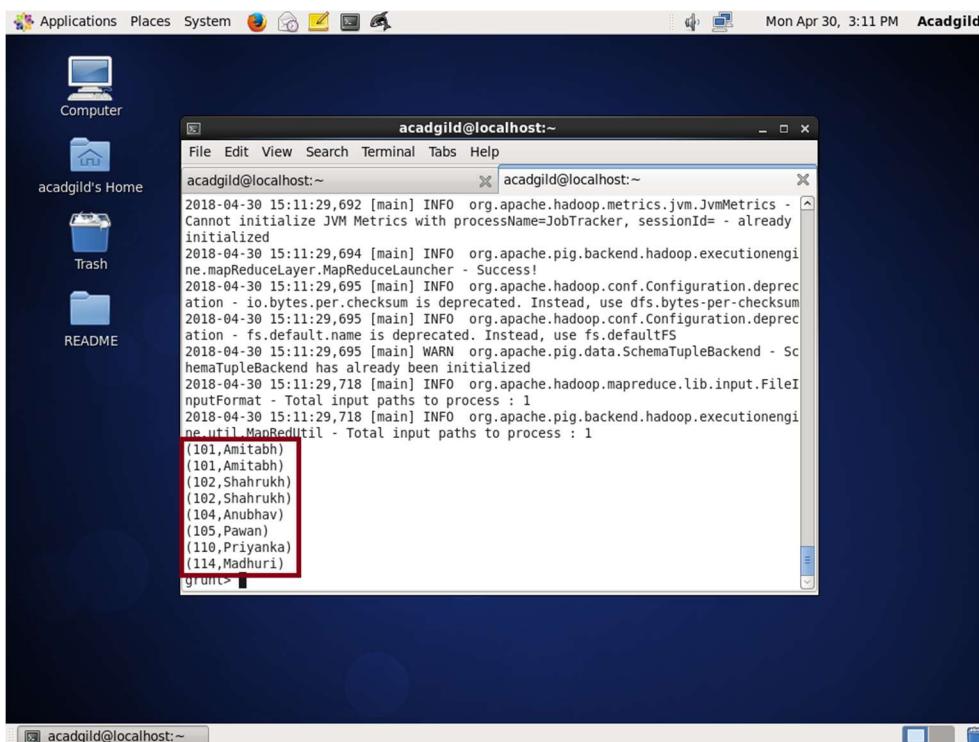
emp_name = foreach join_employee generate \$0, \$1;



A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing 'acadgild's Home' with icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled 'acadgild@localhost:~'. The terminal shows the following command being run:

```
acadgild@localhost:~$ grunt> emp_name= foreach join_employee generate $0,$1;
acadgild@localhost:~$ grunt> dump emp_name;
```

To check the result, you can use DUMP command



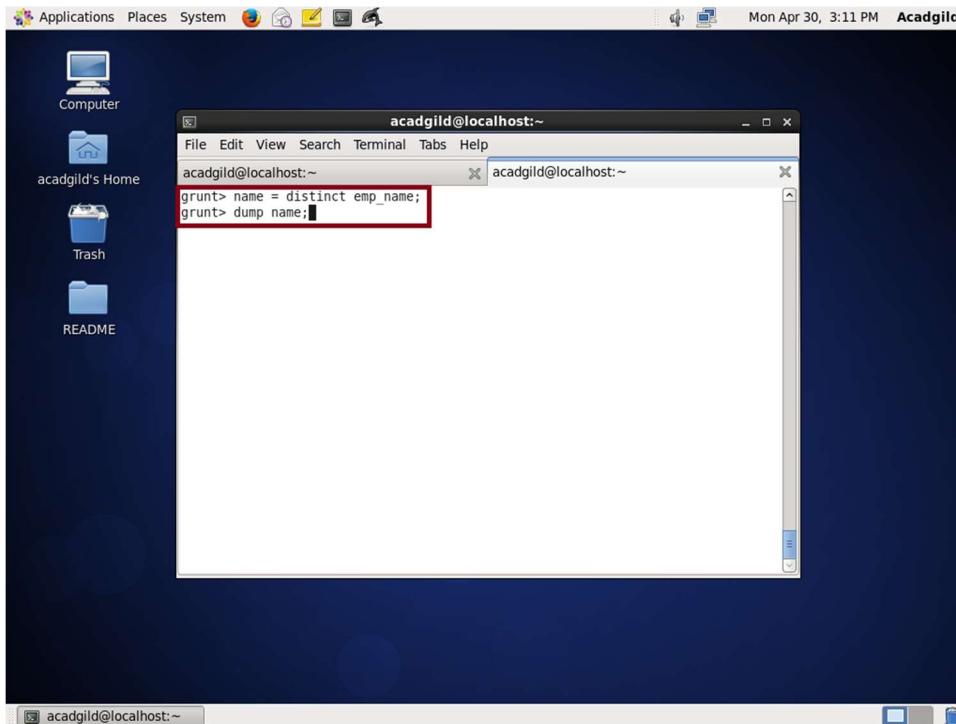
A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing 'acadgild's Home' with icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled 'acadgild@localhost:~'. The terminal shows the output of the 'dump' command:

```
2018-04-30 15:11:29,692 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 15:11:29,694 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 15:11:29,695 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 15:11:29,695 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 15:11:29,695 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 15:11:29,718 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 15:11:29,718 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(101,Amitabh)
(101,Amitabh)
(102,Shahrukh)
(102,Shahrukh)
(104,Anubhav)
(105,Pawan)
(110,Priyanka)
(114,Madhuri)
```

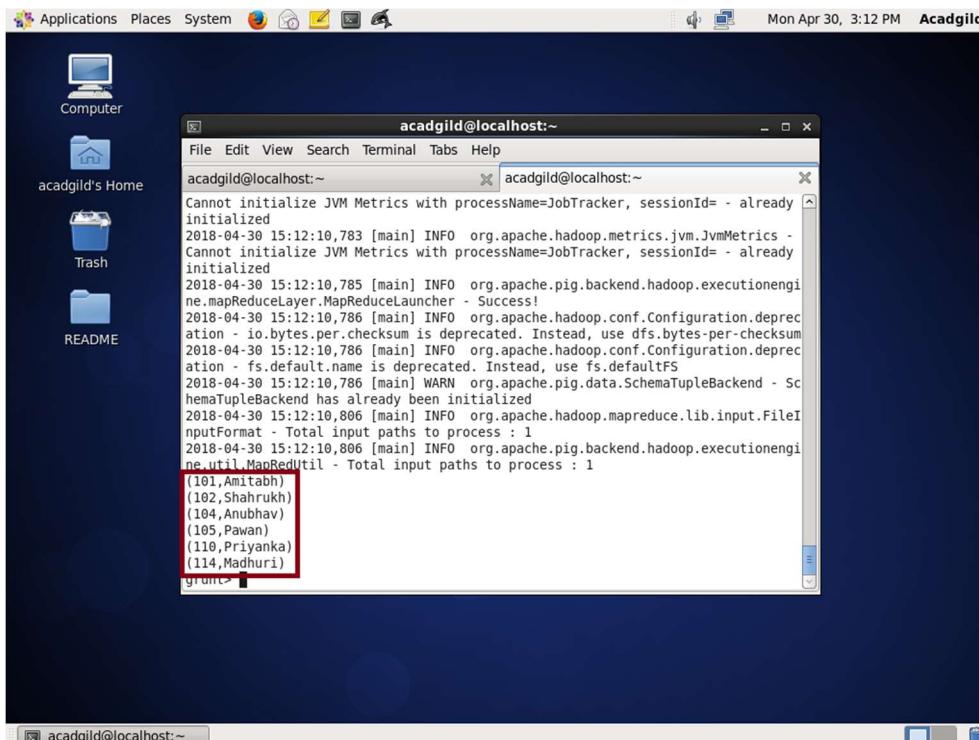
By using distinct operator, we can remove the duplicate entries in the table.

By using the following command, the duplicate entries are removed.

name = distinct emp_name;



To check the result, you can use DUMP command

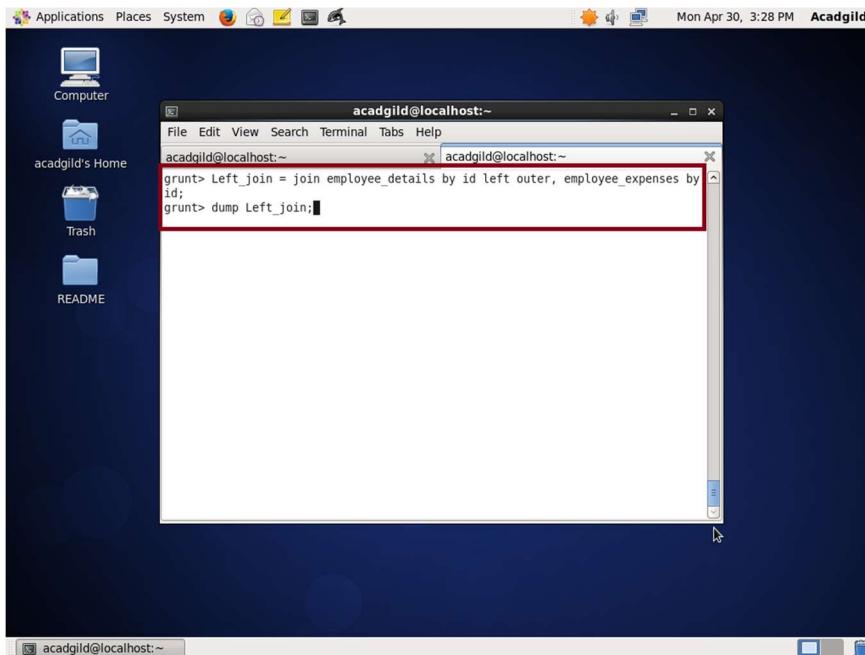


- o (e) List of employees (employee id and employee name) having no entry in employee_expenses file.

Left outer join is used to get all the data from the employee_details table to get the list of employees who have no entry in the employee_expenses file.

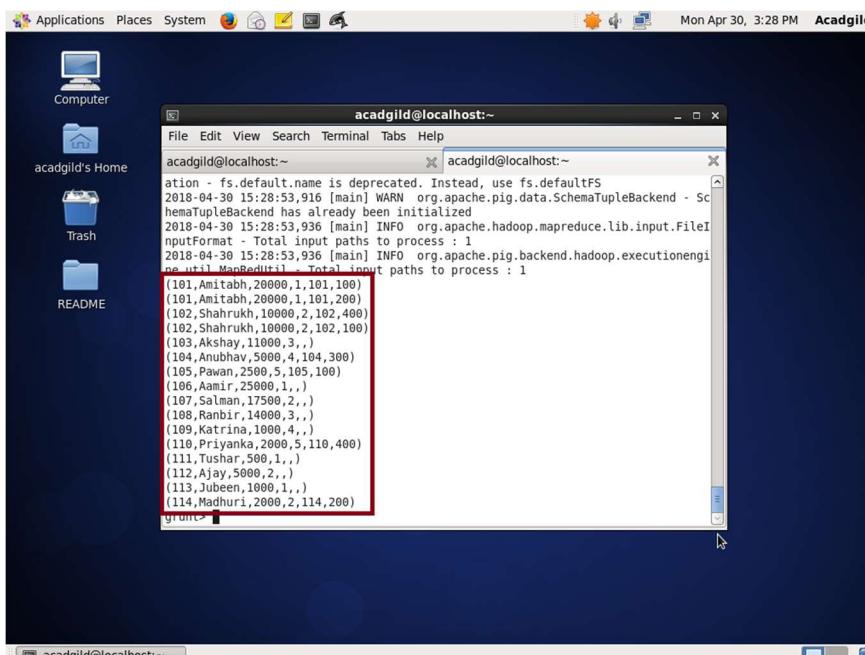
By using the following command, the Left outer join is performed.

Left_join =join employee_details by id left outer, employee_expenses by id;



```
acadgild@localhost:~
```

To check the result, you can use DUMP command



```
action - fs.default.name is deprecated. Instead, use fs.defaultFS
```

```
2018-04-30 15:28:53,916 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
```

```
2018-04-30 15:28:53,936 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
```

```
2018-04-30 15:28:53,936 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
```

```
(101,Amitabh,20000,1,101,100)
(101,Amitabh,20000,1,101,200)
(102,Shahrukh,10000,2,102,400)
(102,Shahrukh,10000,2,102,100)
(103,Akshay,11000,3,,)
(104,Anubhav,5000,4,104,300)
(105,Pawan,2500,5,105,100)
(106,Aamir,25000,1,,)
(107,Salman,17500,2,,)
(108,Ranbir,14000,3,,)
(109,Katrina,1000,4,,)
(110,Priyanka,2000,5,110,400)
(111,Tushar,500,1,,)
(112,Ajay,5000,2,,)
(113,Jubeen,1000,1,,)
(114,Madhuri,2000,2,114,200)
```

```
grunt>
```

By using the following filter operation to filter out the data entries of employee details which have null data entries in the employee expense table.

filter_emp = filter Left_join by \$4 is NULL and \$5 is NULL;

A screenshot of a Linux desktop environment. The desktop has a dark blue background. On the left side, there is a vertical dock with several icons: Computer (monitor), acadgild's Home (home folder), Trash (trash can), and README (text file). At the top, there is a horizontal dock with icons for Applications, Places, System, and a few others. The system tray at the bottom right shows the date and time as "Mon Apr 30, 3:22 PM Acadgild". In the center of the screen, there is a terminal window titled "acadgild@localhost:~". The window has a menu bar with File, Edit, View, Search, Terminal, Tabs, and Help. Below the menu bar, there are two tabs: "acadgild@localhost:~" and another tab which is partially visible. The main area of the terminal contains the following text:

```
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~
grunt> filter emp = filter Left_join by $4 is NULL and $5 is null;
grunt> dump filter_emp;
```

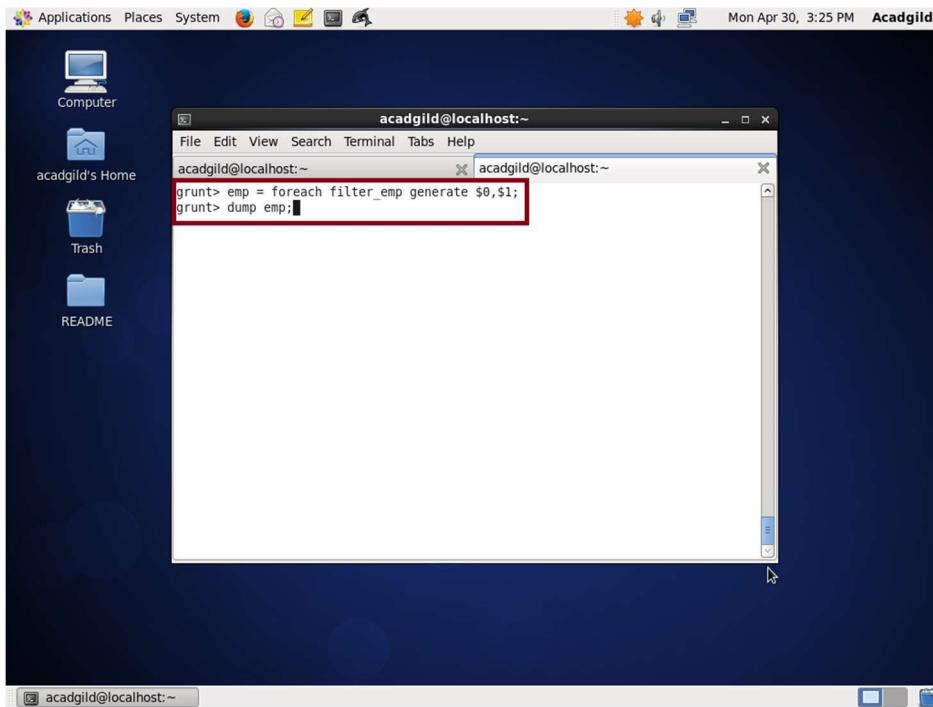
The last two lines of text are highlighted with a red rectangular box. The terminal window has standard window controls (minimize, maximize, close) at the top right.

To check the result, you can use DUMP command

The screenshot shows a Linux desktop interface with a dark blue theme. At the top, there's a dock with icons for Applications, Places, System, and a few system status indicators like battery and signal strength. The date and time are shown as Mon Apr 30, 3:22 PM Acadgild. On the left, there's a sidebar with icons for Computer, acadgild's Home, Trash, and README. The main area has a terminal window titled "acadgild@localhost:~". The terminal displays several lines of log output from a Hadoop application, including INFO messages about JVM Metrics, MapReduceLauncher success, configuration deprecation warnings, and SchemaTupleBackend initialization. A list of names and IDs is displayed at the bottom of the log, with the entire list highlighted and enclosed in a red rectangular box. The list includes: (103, Akshay, 11000, 3,), (106, Aamir, 25000, 1,), (107, Salman, 17500, 2,), (108, Ranbir, 14000, 3,), (109, Katrina, 1000, 4,), (111, Tushar, 500, 1,), (112, Ajay, 5000, 2,), and (113, Jubeen, 1000, 1,). Below the terminal window, the command "grunt>" is visible.

By using the following command, the employee id and employee name are generated.

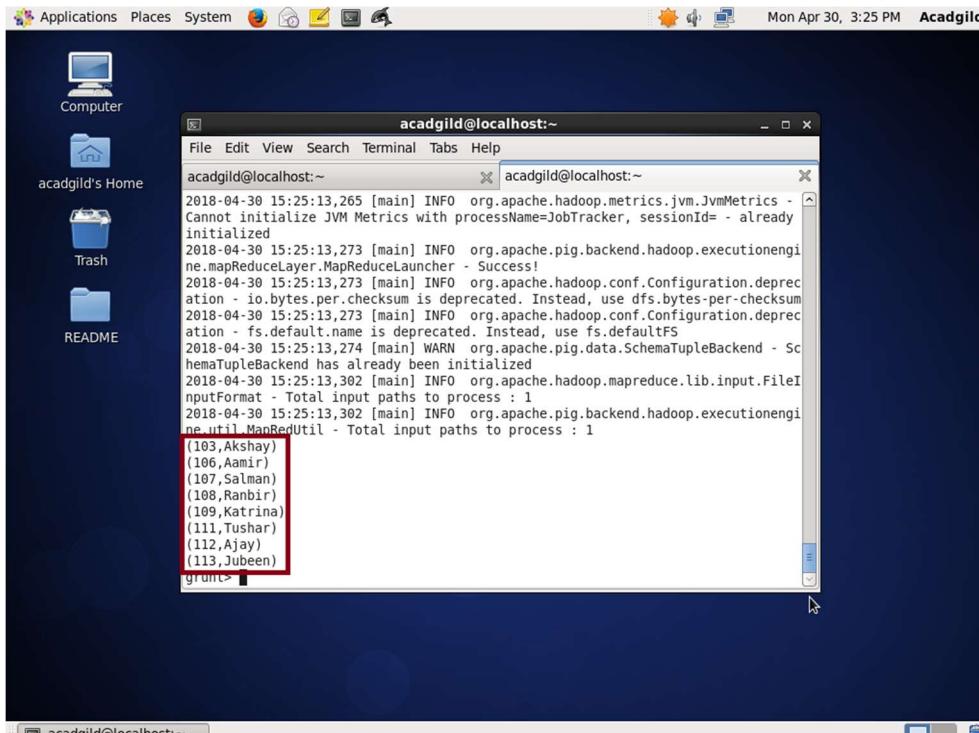
emp = foreach filter_emp generate \$0, \$1;



A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the following Grunt commands:

```
acadgild@localhost:~$ grunt> emp = foreach filter_emp generate $0,$1;
acadgild@localhost:~$ grunt> dump emp;
```

To check the result, you can use DUMP command



A screenshot of a Linux desktop environment. On the left is a Nautilus file manager window showing icons for Computer, acadgild's Home, Trash, and README. In the center is a terminal window titled "acadgild@localhost:~". The terminal shows the Grunt command "dump emp;" followed by the resulting data:

```
2018-04-30 15:25:13,265 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-04-30 15:25:13,273 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-04-30 15:25:13,273 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-04-30 15:25:13,273 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-04-30 15:25:13,274 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-04-30 15:25:13,302 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-04-30 15:25:13,302 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
```

- Task 3

- Implement the use case present in below blog link and share the complete steps along with screenshot(s) from your end.

<https://acadgild.com/blog/aviation-data-analysis-using-apache-pig>

The U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics (BTS) tracks the on-time performance of domestic flights operated by large air carriers. Summary information on the number of on-time, delayed, canceled, and diverted flights appears in DOT's monthly Air Travel Consumer Report, published about 30 days after the month's end, as well as in summary tables posted on this website. Summary statistics and raw data are made available to the public at the time the Air Travel Consumer Report is released.

You can download the datasets from the following links:

Delayed_Flights.csv

Airports.csv

These are 2 different datasets, i.e., Delayed_Flights.csv and Airports.csv. Let us understand one at a time.

Delayed_Flights.csv Datasets

There are 29 columns in this dataset. Some of them have been mentioned below:

Year: 1987 – 2008

Month: 1 – 12

FlightNum: Flight number

Canceled: Was the flight canceled?

CancelleationCode: The reason for cancellation.

For complete details, refer to [this link](#).

Airports.csv Datasets

- iata: the international airport abbreviation code
- name of the airport
- city and country in which airport is located.
- lat and long: the latitude and longitude of the airport

Now, using Apache pig, we will try to gain more insights from these datasets.

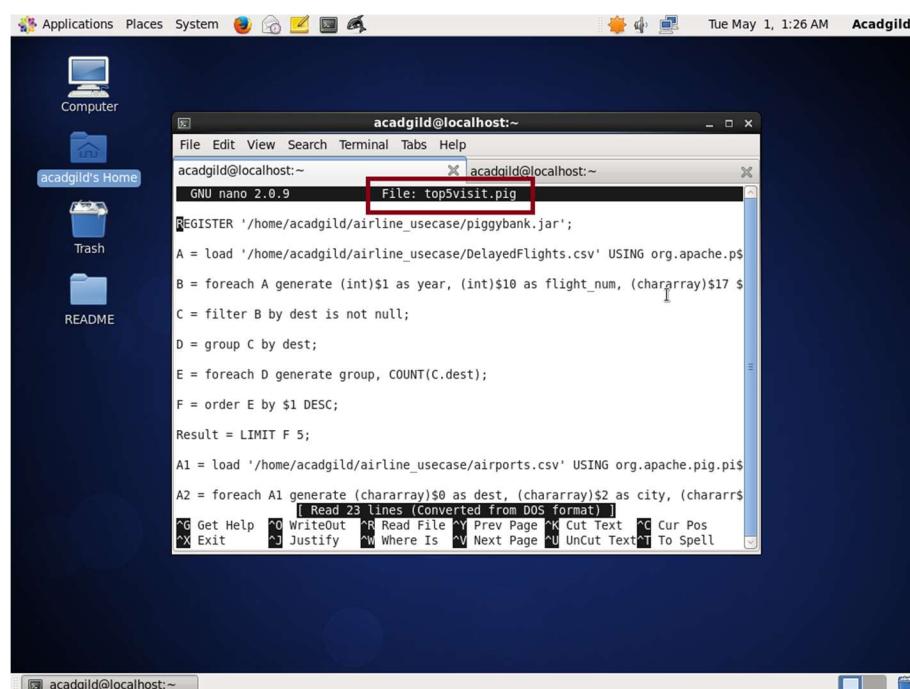
Problem Statement 1

Find out the top 5 most visited destinations.

nano top5visit.pig

Create the **top5visit.pig** file using, **nano top5visit.pig** command and save the file.

Here I create a pig file with the name **top5visit.pig** for the below code and run them to find the top 5 most visited destinations.



The screenshot shows a Linux desktop environment with a dark blue theme. On the left is a Nautilus file manager window showing files like 'Computer', 'acadgild's Home', 'Trash', and 'README'. In the center is a terminal window titled 'acadgild@localhost:~'. The terminal shows the following Pig Latin script:

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';
A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING org.apache.pig.piggybank.PiggyBank;
B = foreach A generate (int)$1 as year, (int)$10 as flight_num, (chararray)$17 $18 as dest;
C = filter B by dest is not null;
D = group C by dest;
E = foreach D generate group, COUNT(C.dest);
F = order E by $1 DESC;
Result = LIMIT F 5;
A1 = load '/home/acadgild/airline_usecase/airports.csv' USING org.apache.pig.piggybank.PiggyBank;
A2 = foreach A1 generate (chararray)$0 as dest, (chararray)$2 as city, (chararray)$3 as country;
G = JOIN Result BY dest, A2 BY dest;
H = foreach G generate dest, city, country, COUNT(Result);
I = ORDER H BY COUNT(Result) DESC;
I = LIMIT I 5;
I = foreach I generate dest, city, country;
```

The terminal window has a red box highlighting the file path 'File: top5visit.pig' in the title bar. The bottom of the terminal shows a menu bar with options like 'Get Help', 'WriteOut', 'Read File', 'Cur Pos', 'Exit', etc.

REGISTER

Registers a JAR file so that the UDFs in the file can be used.

Usage

Use the REGISTER statement inside a Pig script to specify the path of a Java JAR file containing UDFs.

You can register additional files (to use with your Pig script) via the command line using the -Dpig.additional.jars option.

For more information about UDFs, see the User Defined Function Guide. Note that Pig currently only supports functions written in Java.

Add the following codes to the pig file to implement the word count of the text input file

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage('','NO_MULTILINE','UNIX'
,'SKIP_INPUT_HEADER');

B = foreach A generate (int)$1 as year, (int)$10 as flight_num,
(chararray)$17 as origin,(chararray) $18 as dest;

C = filter B by dest is not null;

D = group C by dest;

E = foreach D generate group, COUNT(C.dest);

F = order E by $1 DESC;

Result = LIMIT F 5;

A1 = load '/home/acadgild/airline_usecase/airports.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage('','NO_MULTILINE','UNIX'
,'SKIP_INPUT_HEADER');

A2 = foreach A1 generate (chararray)$0 as dest, (chararray)$2 as city,
(chararray)$4 as country;

joined_table = join Result by $0, A2 by dest;

dump joined_table;
```

In Line 1: We are registering the piggybank jar in order to use the CSVExcelStorage class.

In relation A, we are loading the dataset using CSVExcelStorage because of its effective technique to handle double quotes and headers.

In relation B, we are generating the columns that are required for processing and explicitly typecasting each of them.

In relation C, we are filtering the null values from the “dest” column.

In relation D, we are grouping relation C by “dest.”

In relation E, we are generating the grouped column and the count of each.

Relation F and Result is used to order and limit the result to top 5.

These are the steps to find the top 5 most visited destinations. However, adding few more steps in this process, we will be using another table to find the city name and country as well.

In relation **A1**, we are loading another table to which we will look-up and find the city as well as the country.

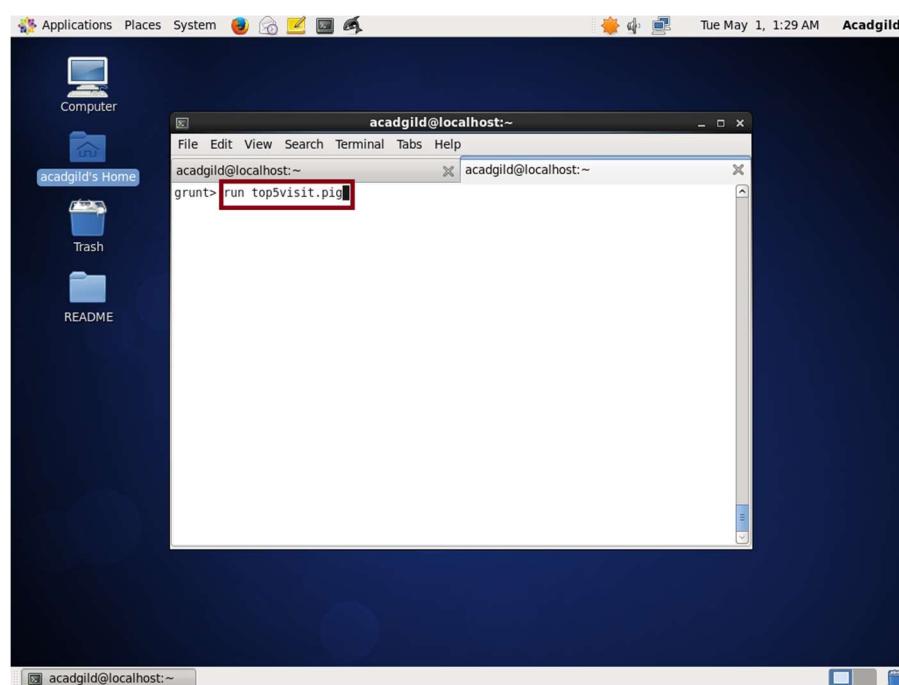
In relation **A2**, we are generating dest, city, and country from the previous relation.

In relation **joined_table**, we are joining Result and A2 based on a common column, i.e., "dest"

Finally, using dump, we are printing the result.

By using the following command, the top5visit.pig is made to run.

run top5visit.pig



acadgild@localhost:~

```
2018-05-01 02:00:36,907 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-05-01 02:00:36,907 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt>
grunt> B = foreach A generate (int)$1 as year, (int)$10 as flight_num, (chararray)$17 as origin,(chararray) $18 as dest;
grunt>
grunt> C = filter B by dest is not null;
grunt>
grunt> D = group C by dest;
grunt>
grunt> E = foreach D generate group, COUNT(C.dest);
grunt>
grunt> F = order E by $1 DESC;
grunt>
grunt> Result = LIMIT F 5;
grunt>
grunt> A1 = load '/home/acadgild/airline_usecase/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage('','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-05-01 02:00:38,254 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-05-01 02:00:38,257 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt>
grunt> A2 = foreach A1 generate (chararray)$0 as dest, (chararray)$2 as city, (chararray)$4 as country;
grunt>
grunt> joined_table = join Result by $0, A2 by dest;
grunt>
grunt> dump joined_table;
```

acadgild@localhost:~

```
Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-05-01 01:22:10,670 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2018-05-01 01:22:10,671 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2018-05-01 01:22:10,671 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-05-01 01:22:10,672 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-05-01 01:22:10,672 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-05-01 01:22:10,678 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-05-01 01:22:10,678 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(ATL,106898,ATL,Atlanta,USA)
(DEN,63003,DEN,Denver,USA)
(DFW,78657,DFW,Dallas-Fort Worth,USA)
(LAX,59969,LAX,Los Angeles,USA)
(ORD,108984,ORD,Chicago,USA)
grunt>
grunt>
```

Problem Statement 2

Which month has seen the most number of cancellations due to bad weather?

[nano Mostcancel.pig](#)

Create the [Mostcancel.pig](#) file using, [nano Mostcancel.pig](#) command and save the file.

Here I create a pig file with the name Mostcancel.pig for the below code and run them to find the most number of cancellations due to bad weather.

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage('','NO_MULTILINE','UNIX'
,'SKIP_INPUT_HEADER');

B = foreach A generate (int)$2 as month,(int)$10 as flight_num,(int)$22 as
cancelled,(chararray)$23 as cancel_code;

C = filter B by cancelled == 1 AND cancel_code =='B';

D = group C by month;

E = foreach D generate group, COUNT(C.cancelled);

F= order E by $1 DESC;

Result = limit F 1;

dump Result;
```

In Line 1: We are registering piggybank jar in order to use the CSVExcelStorage class.

In relation A, we are loading the dataset using CSVExcelStorage because of its effective technique to handle double quotes and header.

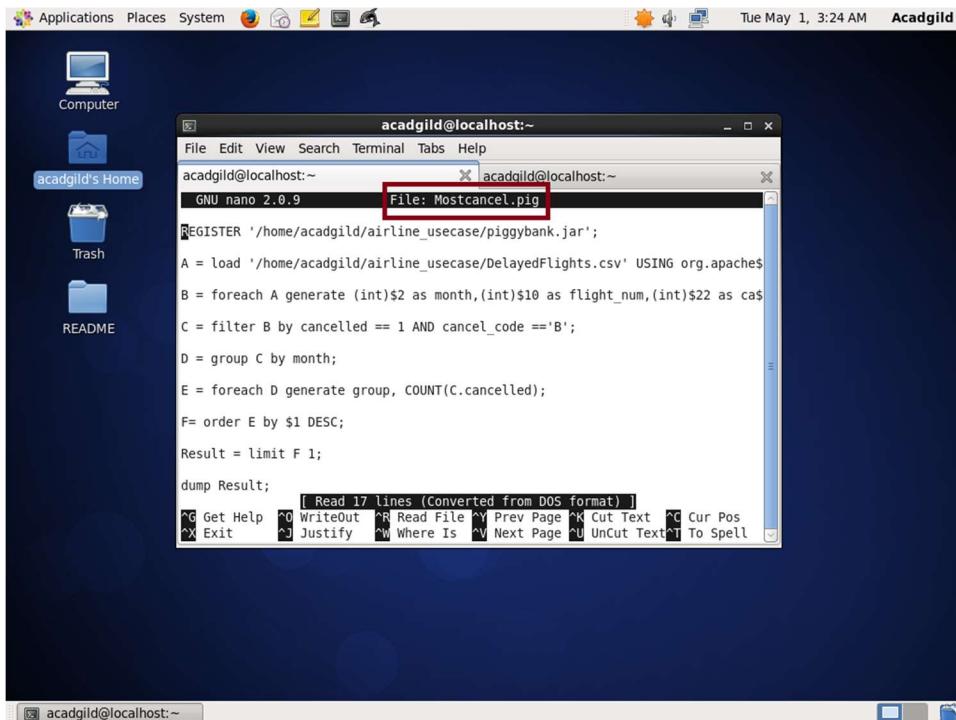
In relation B, we are generating the columns which are required for processing and explicitly typecasting each of them.

In relation C, we are filtering the data based on cancellation and cancellation code, i.e., canceled = 1 means flight have been canceled and cancel_code = 'B' means the reason for cancellation is "weather." So relation C will point to the data which consists of canceled flights due to bad weather.

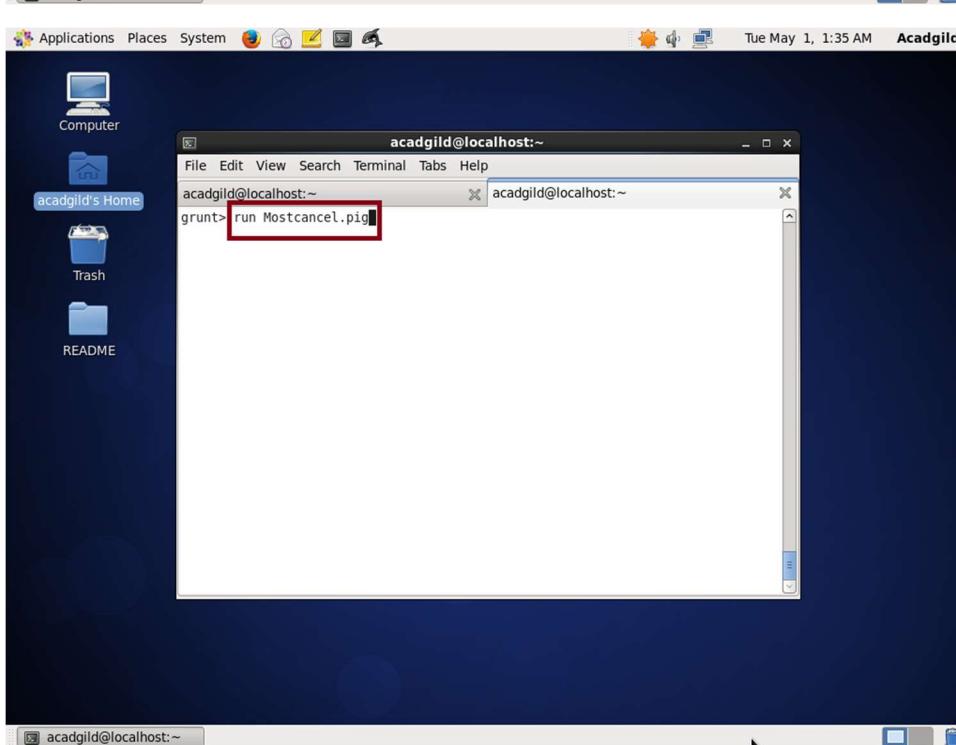
In relation D, we are grouping the relation C based on every month.

In relation E, we are finding the count of canceled flights every month.

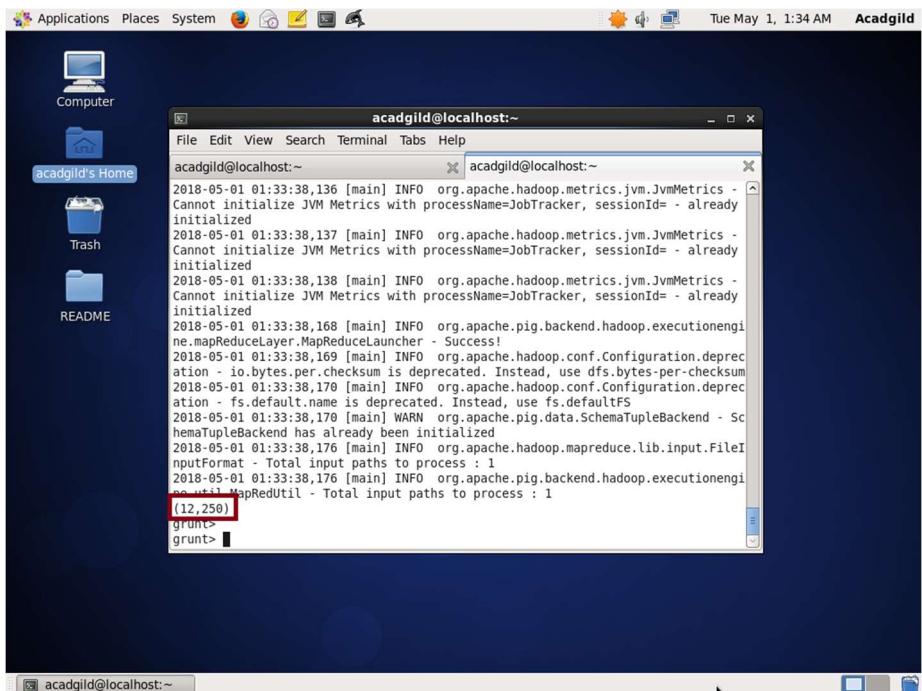
Relation F and Result is for ordering and finding the top month based on cancellation.



```
acadgild@localhost:~$ nano Mostcancel.pig
File Edit View Search Terminal Tabs Help
acadgild@localhost:~$ File: Mostcancel.pig
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';
A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CsvLoader AS (month:int, flight_num:int, cancel_code:chararray);
B = foreach A generate (int)$2 as month,(int)$10 as flight_num,(int)$22 as cancelled;
C = filter B by cancelled == 1 AND cancel_code =='B';
D = group C by month;
E = foreach D generate group, COUNT(C.cancelled);
F= order E by $1 DESC;
Result = limit F 1;
dump Result;
[ Read 17 lines (Converted from DOS format) ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U Uncut Text ^T To Spell
```



```
acadgild@localhost:~$ grunt> run Mostcancel.pig
```



Problem Statement 3

Top ten origins with the highest AVG departure delay

To find the top ten origins with high AVG departure delay, the following code is build and saved as pig file.

By using nano command the code is saved in to pig file.

nano topten.pig

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'NO_MULTILINE', 'UNIX'
,'SKIP_INPUT_HEADER');

B1 = foreach A generate (int)$16 as dep_delay, (chararray)$17 as origin;

C1 = filter B1 by (dep_delay is not null) AND (origin is not null);

D1 = group C1 by origin;

E1 = foreach D1 generate group, AVG(C1.dep_delay);

Result = order E1 by $1 DESC;

Top_ten = limit Result 10;

Lookup = load '/home/acadgild/airline_usecase/airports.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'NO_MULTILINE', 'UNIX'
,'SKIP_INPUT_HEADER');

Lookup1 = foreach Lookup generate (chararray)$0 as origin, (chararray)$2
as city, (chararray)$4 as country;

Joined = join Lookup1 by origin, Top_ten by $0;

Final = foreach Joined generate $0,$1,$2,$4;

Final_Result = ORDER Final by $3 DESC;

dump Final_Result;
```

Explanation of first 3 lines are the same as explained in the previous 2 problem statements.

In relation C1, we are removing the null values fields present if any.

In relation D1, we are grouping the data based on column "origin."

In relation E1, we are finding average delay from each unique origin.

Relations named Result and Top_ten are ordering the results in descending order and printing the top ten values.

These steps are good enough to find the top ten origins with the highest average departure delay.

However, rather than generating just the code of origin, we will be following a few more steps to find some more details like country and city.

In the relation Lookup, we are loading another table to which we will look up and find the city as well as the country.

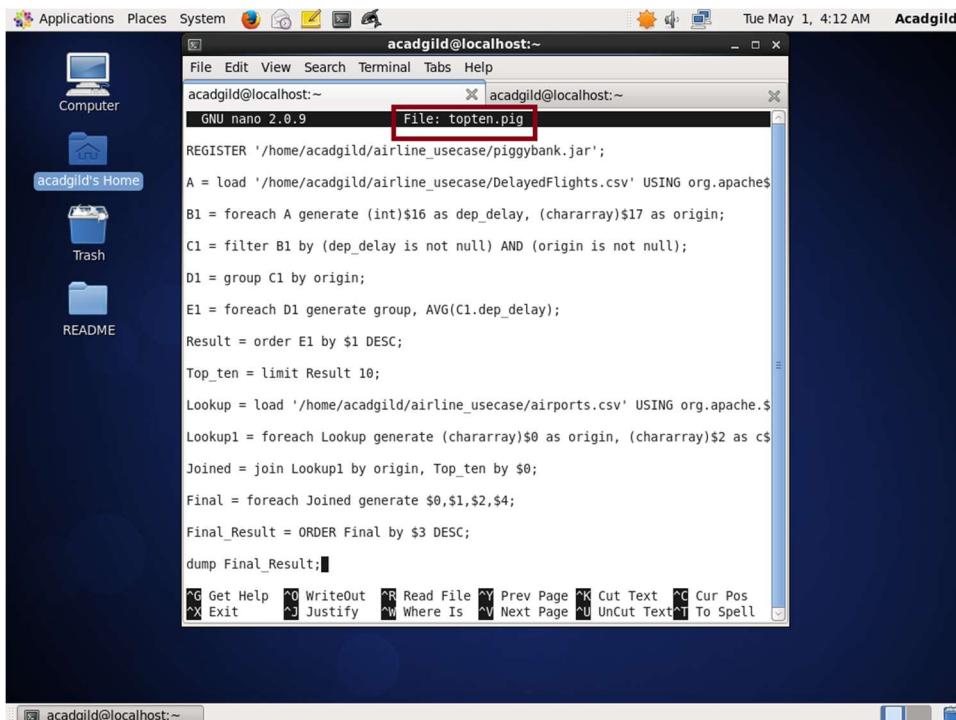
In the relation Lookup1, we are generating the destination, city, and country from the previous relation.

In the relation Joined, we are joining relation Top_ten and Lookup1 based on common a column, i.e., "origin."

In the relation Final, we are generating required columns from the Joined table.

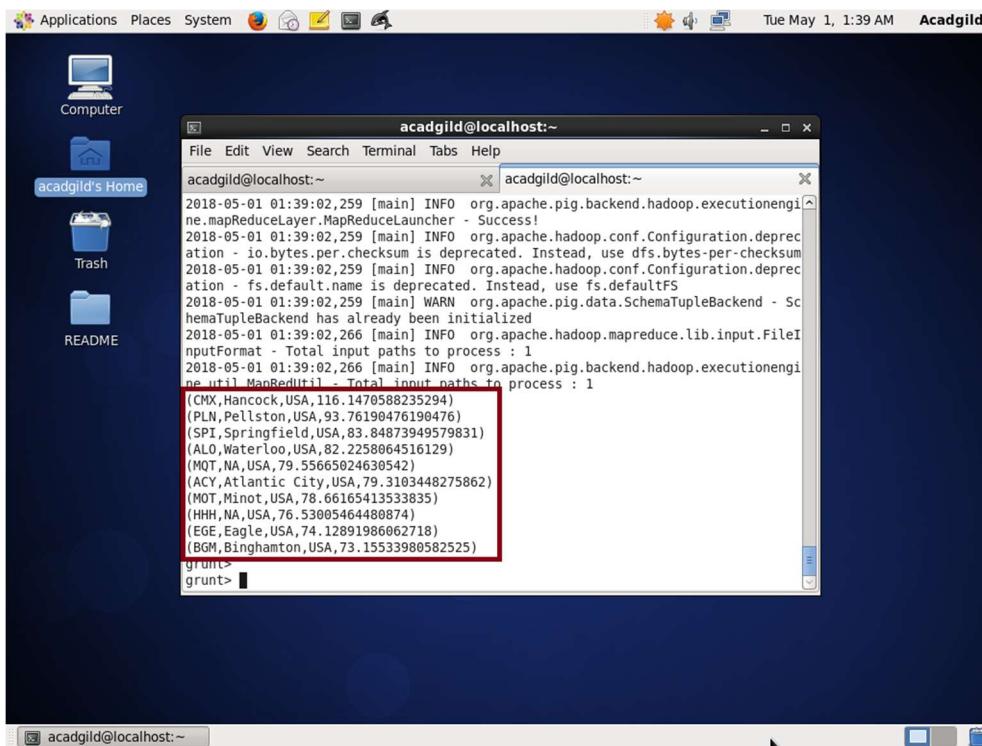
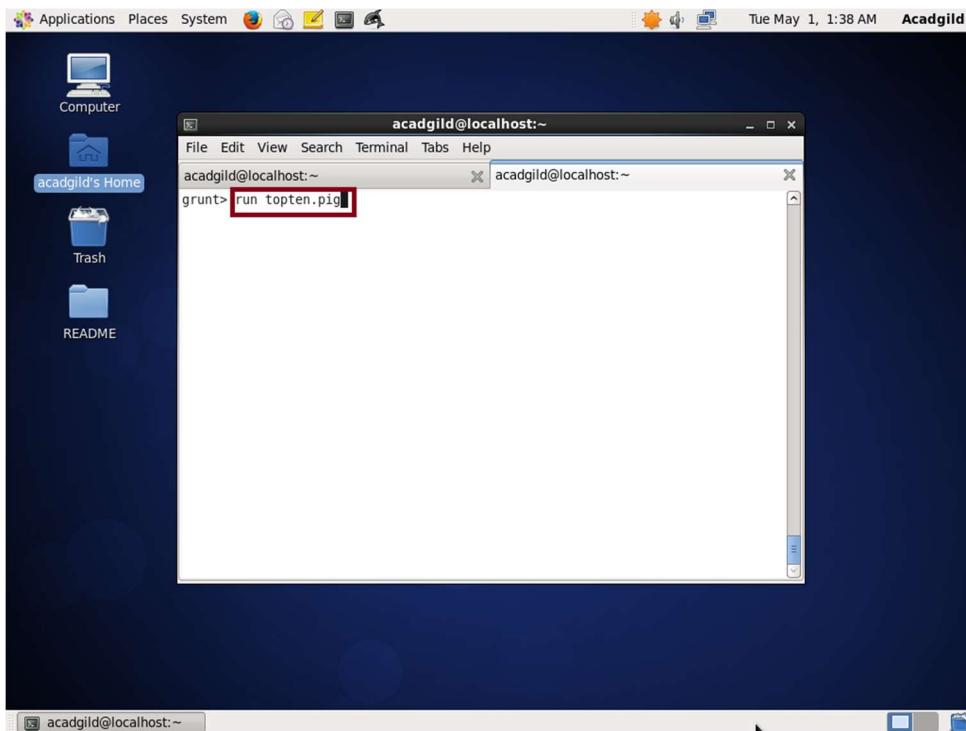
Finally, we are ordering and printing the results. In relation E, we are finding the count of canceled flights every month.

Relation F and Result is for ordering and finding the top month based on cancellation.



```
acadgild@localhost:~$ nano topen.pig
acadgild@localhost:~$ File: topen.pig
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING org.apache.pig.piggybank.PiggyBank;
B1 = foreach A generate (int)$16 as dep_delay, (chararray)$17 as origin;
C1 = filter B1 by (dep_delay is not null) AND (origin is not null);
D1 = group C1 by origin;
E1 = foreach D1 generate group, AVG(C1.dep_delay);
Result = order E1 by $1 DESC;
Top_ten = limit Result 10;
Lookup = load '/home/acadgild/airline_usecase/airports.csv' USING org.apache.pig.piggybank.PiggyBank;
Lookup1 = foreach Lookup generate (chararray)$0 as origin, (chararray)$2 as country;
Joined = join Lookup1 by origin, Top_ten by $0;
Final = foreach Joined generate $0,$1,$2,$4;
Final_Result = ORDER Final by $3 DESC;
dump Final_Result;
```



Problem Statement 4

Which route (origin & destination) has seen the maximum diversion?

To find the maximum diversion between the origin and destination, the following code is build and saved in the pig file using nano.

[nano maxdiversion.pig](#)

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage('','NO_MULTILINE','UNIX'
,'SKIP_INPUT_HEADER');

B = FOREACH A GENERATE (chararray)$17 as origin, (chararray)$18 as dest,
(int)$24 as diversion;

C = FILTER B BY (origin is not null) AND (dest is not null) AND (diversion
== 1);

D = GROUP C by (origin,dest);

E = FOREACH D generate group, COUNT(C.diversion);

F = ORDER E BY $1 DESC;

Result = limit F 10;

dump Result;
```

In Line 1: We are registering *piggybank* jar in order to use CSVExcelStorage class.

In relation A, we are loading the dataset using CSVExcelStorage because of its effective technique to handle double quotes and headers.

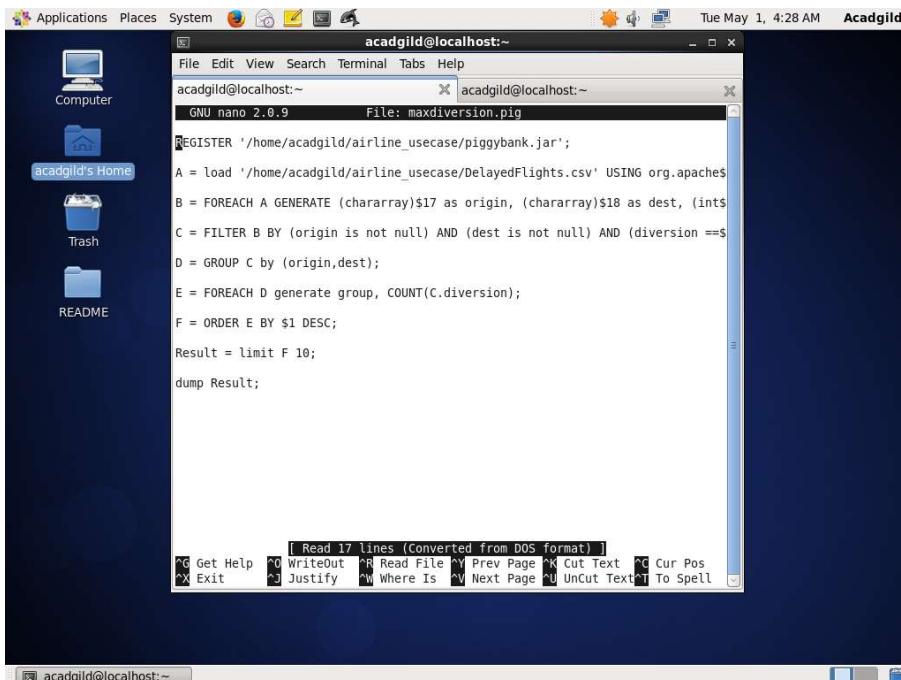
In relation B, we are generating the columns which are required for processing and explicitly type-casting each of them.

In relation C, we are filtering the data based on “not null” and diversion =1. This will remove the null records, if any, and give the data corresponding to the diversion taken.

In relation D, we are grouping the data based on origin and destination.

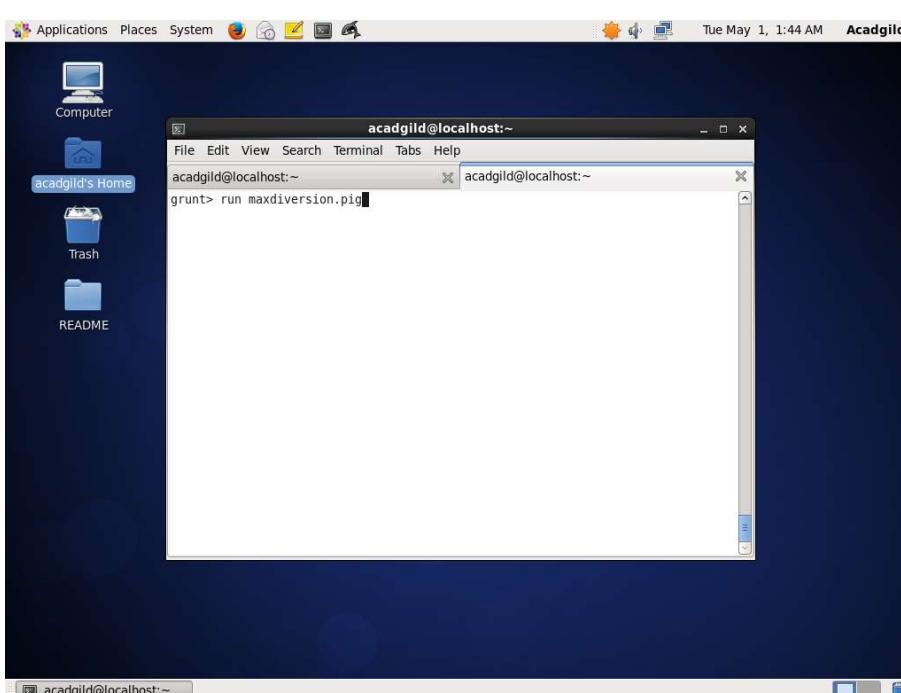
Relation **D** finds the count of diversion taken per unique origin and destination.

Relations **F** and **Result** orders the result and produces top 10 results.



The screenshot shows a Linux desktop environment with a dark blue theme. A terminal window titled "acadgild@localhost:~" is open, displaying a Pig Latin script named "maxdiversion.pig". The script reads a CSV file "DelayedFlights.csv" and performs several operations: it registers a JAR file, loads the CSV, generates pairs of origin and destination, filters non-null values, groups by origin and destination, counts diversions, orders the results by count in descending order, and limits the output to the top 10 entries. The terminal window also shows the status bar indicating "Tue May 1, 4:28 AM" and the user "Acadgild".

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';
A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVLoader;
B = FOREACH A GENERATE (chararray)$17 as origin, (chararray)$18 as dest, (int)$19 as diversion;
C = FILTER B BY (origin is not null) AND (dest is not null) AND (diversion == 1);
D = GROUP C by (origin,dest);
E = FOREACH D generate group, COUNT(C.diversion);
F = ORDER E BY $1 DESC;
Result = limit F 10;
dump Result;
```



The screenshot shows the same Linux desktop environment. The terminal window now displays the command "grunt> run maxdiversion.pig", indicating that the script has been executed. The terminal window title is "acadgild@localhost:~". The status bar shows "Tue May 1, 1:44 AM" and the user "Acadgild".

```
grunt> run maxdiversion.pig
```

