

# Bigdata Assignment 2.4

Entered into pig shell – **pig -x local**



```
acadgild@localhost:~  
File Edit View Search Terminal Help  
[acadgild@localhost ~]$ pig -x local  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]  
18/05/22 12:12:42 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL  
18/05/22 12:12:42 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType  
2018-05-22 12:12:42,996 [main] INFO org.apache.pig.Main - Apache Pig version 0.16.0 (r1746530) compiled Jun 01 2016, 23:10:49  
2018-05-22 12:12:42,997 [main] INFO org.apache.pig.Main - Logging error messages to: /home/acadgild/pig_1526971362994.log  
2018-05-22 12:12:43,026 [main] INFO org.apache.pig.impl.util.Utils - Default bootstrap file /home/acadgild/.pigbootstrap not found  
2018-05-22 12:12:43,213 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address  
2018-05-22 12:12:43,214 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS  
2018-05-22 12:12:43,217 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: file:///   
2018-05-22 12:12:43,317 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum  
2018-05-22 12:12:43,347 [main] INFO org.apache.pig.PigServer - Pig Script ID for the session: PIG-default-5d559615-1b65-44b4-ba3e-1bbfb80a239d  
2018-05-22 12:12:43,347 [main] WARN org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false  
grunt>
```

Loaded the text file into a relation – **emp\_details = LOAD**  
**'/home/acadgild/INPUT/PIG/employee\_details.txt' using PigStorage(',') as**  
**(id:int , name:chararray , salary:int , rating:int);**  
**emp\_expenses = LOAD**  
**'/home/acadgild/INPUT/PIG/employee\_expenses.txt' as (id:int ,**  
**expenses:int);**



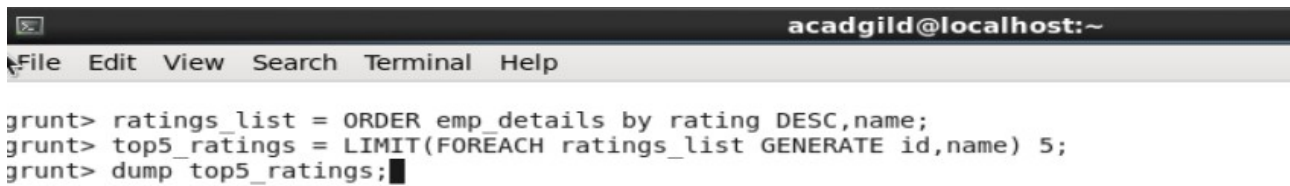
```
acadgild@localhost:~  
File Edit View Search Terminal Help  
grunt> emp_details = LOAD '/home/acadgild/INPUT/PIG/employee_details.txt' using PigStorage(',') as (id:int , name:chararray, salary:int , rating:int);  
2018-05-22 12:18:01,965 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum  
2018-05-22 12:18:01,966 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS  
grunt> emp_expenses = LOAD '/home/acadgild/INPUT/PIG/employee_expenses.txt' as (id:int , expenses:int);  
2018-05-22 12:19:31,370 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum  
2018-05-22 12:19:31,370 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS  
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)

Ans) We first ordered the by rating in descending order then by name in ascending order. Then selected top 5 employees based on the given sorting.

**ratings\_list = ORDER emp\_details by rating DESC , name;**

```
top5_ratings = LIMIT(FOREACH ratings_list GENERATE
                    id,name) 5;
dump top5_ratings;
```



```
acadgild@localhost:~
File Edit View Search Terminal Help

grunt> ratings_list = ORDER emp_details by rating DESC,name;
grunt> top5_ratings = LIMIT(FOREACH ratings_list GENERATE id,name) 5;
grunt> dump top5_ratings;
```

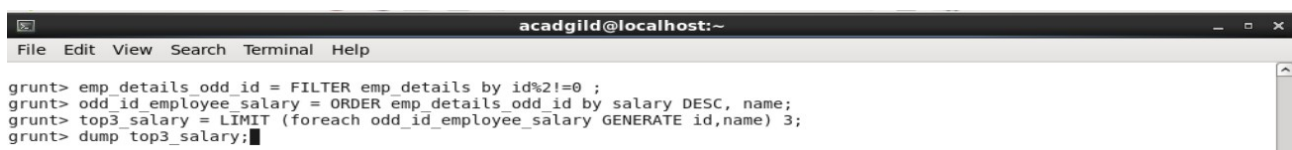
**OUTPUT-** As we can see in the below screenshot of the output we got the top 5 employee with topratings.

```
2018-05-22 19:43:08,130 [mai
cess : 1
(105,Pawan)
(110,Priyanka)
(104,Anubhav)
(109,Katrina)
(103,Akshay)
grunt>
```

(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)

Ans) Filtered out the odd id employees , then sorted them by salary in descending order and then by name in ascending order. Thwn selected top 3 employees of the list.

```
emp_details_odd_id = FILTER emp_details by id%2!=0 ;
odd_id_employee_salary = ORDER emp_details_odd_id by salary
                        DESC , name;
top3_salary = LIMIT (foreach odd_id_employee_salary GENERATE
                    id , name) 3;
dump top3_salary;
```



```
acadgild@localhost:~
File Edit View Search Terminal Help

grunt> emp_details_odd_id = FILTER emp_details by id%2!=0 ;
grunt> odd_id_employee_salary = ORDER emp_details_odd_id by salary DESC, name;
grunt> top3_salary = LIMIT (foreach odd_id_employee_salary GENERATE id,name) 3;
grunt> dump top3_salary;
```

## OUTPUT -

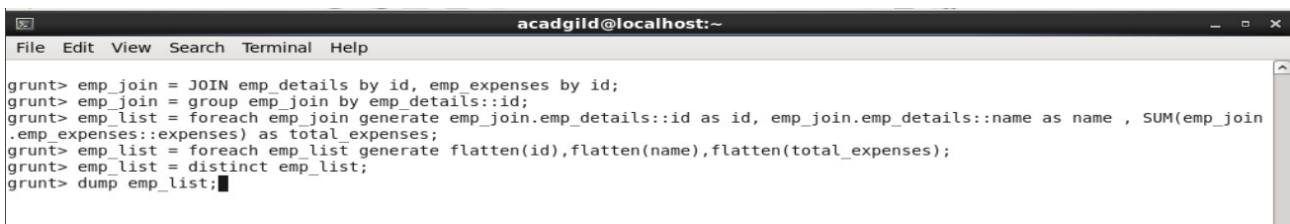
We got the employee id and its name with highest salary whose id is odd no.

```
2018-03-22 13:21:27,743 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapReduceUtil - total input paths to process : 1
(101,Amitabh)
(107,Salman)
(103,Akshay)
grunt> █
```

(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)

Ans - We made a inner join table of emp\_details and emp\_expenses. Then grouped them by id. Then made a list of each employee's id , name and its total expenses. Then we flattened the data and made a distinct list of it. Selected the top expense by sorting it in descending order.

```
emp_join = JOIN emp_details by id , emp_expenses by id;
emp_join = group emp_join by emp_details::id;
emp_list = foreach emp_join generate emp_join.emp_details::id as id ,
emp_join.emp_details::name as name ,
SUM(emp_join.emp_expenses::expenses) as total_expenses;
emp_list = foreach emp_list generate flatten(id) , flatten(name), flatten(total_expenses);
emp_list = distinct emp_list;
dump emp_list;
top_expense = LIMIT (ORDER emp_list by total_expenses DESC) 1;
dump top_expense;
```



```
acadgiid@localhost:~
File Edit View Search Terminal Help

grunt> emp_join = JOIN emp_details by id, emp_expenses by id;
grunt> emp_join = group emp_join by emp_details::id;
grunt> emp_list = foreach emp_join generate emp_join.emp_details::id as id, emp_join.emp_details::name as name, SUM(emp_join
.emp_expenses::expenses) as total_expenses;
grunt> emp_list = foreach emp_list generate flatten(id), flatten(name), flatten(total_expenses);
grunt> emp_list = distinct emp_list;
grunt> dump emp_list;█
```

```
grunt> top_expense = LIMIT (ORDER emp_list by total_expenses DESC) 1;
grunt> dump top_expense;█
```

OUTPUT – Got the highest expense employee id and name.

```
1000 : 1
(102,Shahrukh,500)
results █
```

(d) List of employees (employee id and employee name) having entries in employee\_expenses file.

Ans- Made an inner join by id of the details and expense tables. Then made an distinct list of it each employee name and it's id.

```
emp_join = join emp_details by id;  
emp_entry = distinct (foreach emp_join generate emp_details::id ,  
emp_details::name);  
dump emp_entry;
```



```
acadgild@localhost:~  
grunt> emp_join = join emp_details by id , emp_expenses by id;  
grunt> emp_entry = distinct(foreach emp_join generate emp_details::id , emp_details::name);  
grunt> dump emp_entry;
```

OUTPUT - The list of the employees having entry in expenses file.

```
2018-05-22 21:21:32,  
cess : 1  
(101,Amitabh)  
(102,Shahrukh)  
(104,Anubhav)  
(105,Pawan)  
(110,Priyanka)  
(114,Madhuri)  
grunt>
```

(e) List of employees (employee id and employee name) having no entry in employee\_expenses file.

Ans - Made a full outer join by id of tables emp\_details and emp\_expenses. Then filtered out the tuples which have id as NULL in emp\_expenses. Then made a distinct list of it having employee name and its id.

```
emp_join = JOIN emp_details by id FULL OUTER, emp_expenses by id;  
emp_expense_noentry = FILTER emp_join by (emp_expenses::id IS  
NULL) and (emp_details::id > 0);  
emp_expense_noentry_distinct = DISTINCT(foreach  
emp_expense_noentry GENERATE emp_details::id, emp_details::name);  
dump emp_expense_noentry_distinct;
```

```
acadgild@localhost:~/INPUT/PIG
File Edit View Search Terminal Help

grunt> emp_join = JOIN emp_details by id FULL OUTER , emp_expenses by id;
grunt> emp_expense_noentry = FILTER emp_join by (emp_expenses::id IS NULL) and (emp_details::id > 0);
grunt> emp_expense_noentry_distinct = DISTINCT(foreach emp_expense_noentry GENERATE emp_details::id , emp_details::name);
grunt> dump emp_expense_noentry_distinct;
```

OUTPUT - It shows the details of employee having details in emp-details.txt but not in emp\_expenses.txt.

```
cess : 1
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
grunt> █
```