**Data Analytics Laboratory**

**Task 9**

**Pig Latin scripts: To sort, group, join for a given dataset**

## Aim

To performing Pig Latin Scripts to sort, group, join the given dataset in Hadoop distributed file system.

## Introduction to Pig Latin

* Pig is application that runs on top of MapReduce and abstracts Java MapReduce jobs away from developers.
* Pig Latin uses a lot fewer lines of code than the Java MapReduce script.
* The Pig Latin script was is easier to read for someone without a Java background.
* MapReduce jobs can be written in Pig Latin.
* Apache Pig has two main components – the Pig Latin language and the Pig Run-time Environment, in which Pig Latin programs are executed.
* Apache Pig follows ETL (Extract Transform Load) process. It can handle inconsistent schema (in case of unstructured data).
* Apache Pig automatically optimizes the tasks before execution, i.e., automatic optimization. Apache Pig handles all kinds of data.
* Pig allows programmers to write custom functions which is unavailable in Pig. User Defined Functions (UDF) can be written in different language like Java, Python, Ruby, etc. and embed them in Pig script.
* Pig Latin provides various built-in operators like join, sort, filter, etc. to read, write, and process large data sets.
* Java is a great and powerful language, but it has a higher learning curve than something like Pig Latin. Therefore, using a higher-level language, like Pig Latin, enables many more developers/analysts to write MapReduce jobs.

## Prerequisites

**1. What is Pig Latin?**

Pig Latin is Pig's language that allows developers to sort, join, parse, transform and calculate unstructured and semi-structured data in MapReduce all while using a language similar to SQL based language instead of writing MapReduce Java program. But, however the pig scripts are executed on HDFS by MapReduce framework.

**2. How Pig programming gets converted into MapReduce jobs?**

A program written in Pig Latin is a data flow language, which need an execution engine to execute the query. So, when a program is written in Pig Latin, Pig compiler converts the program into MapReduce jobs.

**3. What are the components of Pig Execution Environment?**

* Pig Scripts: Pig scripts are submitted to the Apache Pig execution environment which can be written in Pig Latin using built-in operators and UDFs can be embedded in it.
* Parser: The Parser does the type checking and checks the syntax of the script. The parser outputs a DAG (directed acyclic graph). DAG represents the Pig Latin statements and logical operators.
* Optimizer: The Optimizer performs the optimization activities like split, merge, transform, reorder operators, etc. The optimizer provides the automatic optimization feature to Apache Pig. The optimizer basically aims to reduce the amount of data in the pipeline.
* Compiler: The Apache Pig compiler converts the optimized code into MapReduce jobs automatically.
* Execution Engine: Finally, the MapReduce jobs are submitted to the execution engine. Then, the MapReduce jobs are executed, and the required result is produced.

**4. What are the different ways of executing Pig script?**

* Grunt Shell: This is Pig’s interactive shell provided to execute all Pig Scripts.
* Script File: Write all the Pig commands in a script file and execute the Pig script file. This is executed by the Pig Server.
* Embedded Script: If some functions are unavailable in built-in operators, we can programmatically create User Defined Functions (UDF) to bring that functionality using other languages like Java, Python, Ruby, etc. and embed it in the Pig Latin Script file. Then, execute that script file.

**5. What are the Built-In functions in Pig Latin?**

The extensive list of function and description can be found in the following link

<https://pig.apache.org/docs/r0.17.0/func.html>

## In-Lab Tasks

To complete this lab, you will need the following:

* Hadoop Environment installed preferably in Linux System.
* Configure Pig Latin environment using the text file provided along with this exercise.

1. Create a text file with comma separated values as given below (f denotes the value is float)

gopu,20,3.9f

arun,18,4.0f

alice,21,3.8f

bob,22,5.5f

kumar,19,3.8f

john,18,3.8f

jackson,18,3.8f

save the text file as pigexercise.txt

2. Copy the created text file from Linux filesystem to HDFS using the below command. You can also create a directory as per your needs.

hdfs dfs -mkdir /pig

hdfs dfs -put pigexercise.txt /pig/pigexercise.txt

3. To start the pig environment type pig in terminal the terminal will change to grunt shell

4. **Grouping**

grunt> A = LOAD '/pig/pigexercise.txt/ USING PigStorage (',') AS (name:chararray,age:int,gpa:float);

grunt>DUMP A

grunt>B = GROUP A BY age;

grunt>DUMP B

5. **Joining**

Create two text files as shown below

|  |  |
| --- | --- |
| Customers.txt | Orders.txt |
| 1,Ramesh,32,Ahmedabad,2000.00  2,Khilan,25,Delhi,1500.00  3,kaushik,23,Kota,2000.00  4,Chaitali,25,Mumbai,6500.00  5,Hardik,27,Bhopal,8500.00  6,Komal,22,MP,4500.00  7,Muffy,24,Indore,10000.00 | 102,2009-10-08 00:00:00,3,3000  100,2009-10-08 00:00:00,3,1500  101,2009-11-20 00:00:00,2,1560  103,2008-05-20 00:00:00,4,2060 |

copy both files to HDFS Directory.

grunt> customers = LOAD '/pig/customers.txt' USING PigStorage(',') as (id:int, name:chararray, age:int, address:chararray, salary:int);

grunt> orders = LOAD '/pig/orders.txt' USING PigStorage(',') as (oid:int, date:chararray, customer\_id:int, amount:int);

grunt> customer\_orders = JOIN customers BY id, orders BY customer\_id;

grunt> Dump customer\_orders

(2,Khilan,25,Delhi,1500,101,2009-11-20 00:00:00,2,1560)

(3,kaushik,23,Kota,2000,100,2009-10-08 00:00:00,3,1500)

(3,kaushik,23,Kota,2000,102,2009-10-08 00:00:00,3,3000)

(4,Chaitali,25,Mumbai,6500,103,2008-05-20 00:00:00,4,2060)

To store the output into a text file use the following command

STORE customer\_orders INTO '/pig/Output/' USING PigStorage (',');

6. **Sorting**

Create the text file as shown below

|  |
| --- |
| StudentDetails.txt |
| 001,Rajiv,Reddy,21,9848022337,Hyderabad  002,siddarth,Battacharya,22,9848022338,Kolkata  003,Rajesh,Khanna,22,9848022339,Delhi  004,Preethi,Agarwal,21,9848022330,Pune  005,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar  006,Archana,Mishra,23,9848022335,Chennai  007,Komal,Nayak,24,9848022334,trivendram  008,Bharathi,Nambiayar,24,9848022333,Chennai |

grunt> student\_details = LOAD /pig/StudentDetails.txt' USING PigStorage(',')as (id:int, firstname:chararray, lastname:chararray,age:int, phone:chararray, city:chararray);

grunt> order\_by\_data = ORDER student\_details BY age DESC;

grunt> Dump order\_by\_data;

(8,Bharathi,Nambiayar,24,9848022333,Chennai)

(7,Komal,Nayak,24,9848022334,trivendram)

(6,Archana,Mishra,23,9848022335,Chennai)

(5,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar)

(3,Rajesh,Khanna,22,9848022339,Delhi)

(2,siddarth,Battacharya,22,9848022338,Kolkata)

(4,Preethi,Agarwal,21,9848022330,Pune)

(1,Rajiv,Reddy,21,9848022337,Hyderabad)

Exercise Task

Write a pig script to count number of words in a given file

**Results**

The program is implemented in python and the output is observed.

**Faculty Signature**