Swami Keshvanand Institute of Technology, Management &Gramothan,
Ramnagaria, Jagatpura, Jaipur-302017, INDIA
Approved by AICTE, Ministry of HRD, Government of India
Recognized by UGC under Section 2(f) of the UGC Act, 1956
Tel. : +91-0141- 5160400Fax: +91-0141-2759555
E-mail: info@skit.ac.in Web: www.skit.ac.in

UNIT-4 (PIG: Hadoop Programming Made Easier)

Overview of PIG: Pig is an open source Apache project for analyzing unstructured or semi-structured data available on Hadoop. It provides a level of abstraction over MapReduce thereby eliminating the need for writing low-level MapReduce code.

In 2006, Pig was started as a research project at Yahoo!. Yahoo developed a new language called Pig Latin for their Hadoop users that fit well between the declarative style of SQL and the procedural style of MapReduce.

Pig was open sourced through Apache and was first released in 2008. Later it became one of the sub projects of Apache Hadoop. By 2009 many companies started adopting Pig for their data processing.

Pig provides an engine for executing various data pipeline operations (load, filter, sort, store etc.) in parallel on Hadoop. It executes these operations with the help of MapReduce.

Pig is rich in terms of data structures and operators. Pig Latin is the language used for writing data flows with Pig.

Philosophy

- Eats anything : Pig can process or analyse any type of data such as relational, unstructured or nested in nature
- Lives anywhere : Pig is not tied to any particular parallel framework although it is first implemented for Hadoop
- we can customize or extend Pig features by adding user defined functions written in Java or other languages

- Can fly : Pig is capable of processing terabytes of data very quickly

Key features

- Pig provides various built in operators to perform all kinds of ETL Extract , Transform, Load) operations on the data
- Pig Latin scripts are internally converted to a series of MapReduce jobs
- Pig has an optimizer that optimizes the Pig Scripts to give consistent performance by reordering or rearranging the operations involved in the data flow.

Pig Execution modes

Pig scripts can be executed in two ways

- Local mode: This mode is suitable for small sized data. Here Pig scripts are executed locally on a single machine. Prototyping and debugging scenarios can be easily implemented with this mode.
- Distributed mode: Here the Pig scripts are executed on a distributed cluster and is suitable for huge sized data. This mode is the default one which translates Pig scripts into series of MapReduce jobs

Interactive mode using Grunt shell:

Interactive mode execution is done with the command shell named 'grunt'

It enables line-by-line execution of the Pig scripts

Grunt shell is invoked using the below commands

In case of distributed mode, pig or pig -x is used to invoke grunt shell

```
[pig_User001@pig_trngsrvr01 ~]$ pig
18/11/26 12:52:12 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
18/11/26 12:52:12 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
18/11/26 12:52:12 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
grunt>
```

- In case of local mode, **pig -x local** is used to invoke grunt shell

```
[pig_User001@pig_trngsrvr01 ~]$ pig -x local
18/11/26 10:44:45 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
18/11/26 10:44:45 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
grunt>
```

Batch mode:

- Pig scripts can be saved with a **.pig** extension
- File with **.pig** extension can be executed using **run** or **exec** commands in grunt shell
- Pig file can be executed along with the mode of execution as below

```
[pig_User001@pig_trngsrvr01 ~]$ pig -x local DemoScript.pig
```

Embedded mode:

- User can write their own User Defined Functions (UDF) in programming languages like Java
- UDFs can be embedded with Pig scripts for data manipulation

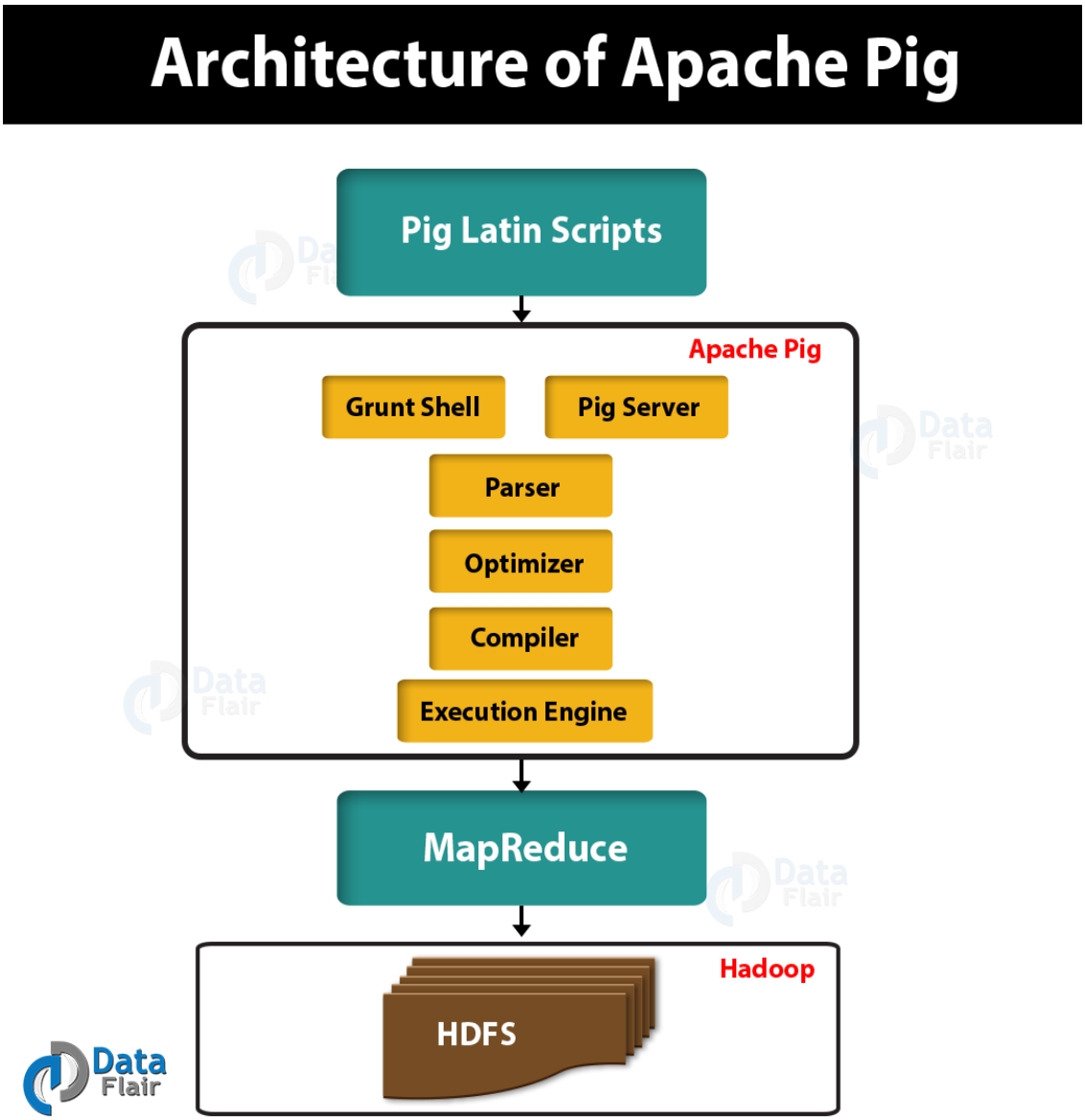
Swami Keshvanand Institute of Technology, Management &Gramothan,
Ramnagaria, Jagatpura, Jaipur-302017, INDIA
Approved by AICTE, Ministry of HRD, Government of India
Recognized by UGC under Section 2(f) of the UGC Act, 1956
Tel. : +91-0141- 5160400Fax: +91-0141-2759555
E-mail: info@skit.ac.in Web: www.skit.ac.in

Going with the Pig Latin Application Flow:

Pig Architecture: In Pig, there is a language we use to analyse data in **Hadoop**. That is what we call Pig Latin. In addition, a high-level data processing language offers a rich set of data types and operators to perform several operations on the data.

Moreover, in order to perform a particular task, programmers need to write a Pig script using the Pig Latin language and execute them using any of the execution mechanisms (Grunt Shell, UDFs, Embedded) using Pig.

To produce the desired output, these scripts will go through a series of transformations applied by the Pig Framework, after execution. Pig converts these scripts into a series of **MapReduce** jobs internally. Therefore, it makes the programmer's job easy. Here, is the architecture of Apache Pig.



i. Parser

At first, the Parser handles all the Pig Scripts. Parser checks the syntax of the script, does type checking, and other miscellaneous checks. Afterwards, Parser has output will be a

DAG (directed acyclic graph) that represents the Pig Latin statements as well as logical operators.

The logical operators of the script are represented as the nodes and the data flows are represented as edges in DAG (the logical plan)

ii. Optimizer

Afterwards, the logical plan (DAG) is passed to the logical optimizer. It carries out the logical optimizations further such as projection and push down.

iii. Compiler

Then compiler compiles the optimized logical plan into a series of MapReduce jobs.

iv. Execution engine

Eventually, all the MapReduce jobs are submitted to Hadoop in a sorted order. Ultimately, it produces the desired results while these MapReduce jobs are executed on Hadoop.

Pig Latin Data Model: Pig Latin data model is fully nested. Also, it allows complex non-atomic data types like map and tuple. Let's discuss this data model in detail:

i. Atom

Atom is defined as any single value in Pig Latin, irrespective of their data. We can use it as string and number and store it as the string. Atomic values of Pig are int, long, float, double, char array, and byte array. Moreover, a field is a piece of data or a simple atomic value in Pig.

For Example - 'Shubham' or '25'

ii. Tuple

Tuple is a record that is formed by an ordered set of fields. However, the fields can be of any type. In addition, a tuple is similar to a row in a table of RDBMS.

For Example - (Shubham, 25)

iii. Bag

An unordered set of tuples is what we call Bag. To be more specific, a Bag is a collection of tuples (non-unique). Moreover, each tuple can have any number of fields (flexible schema).

Generally, we represent a bag by '{}'.

For Example - {(Shubham, 25), (Pulkit, 35)}

In addition, when a bag is a field in a relation, in that way it is known as the inner bag.

Example - {Shubham, 25, {9826022258, Shubham@gmail.com,}}

iv. Map

A set of key-value pairs is what we call a map (or data map). Basically, the key needs to be of type char array and should be unique. Also, the value might be of any type. And, we represent it by '[]'

For Example - [name#Shubham, age#25]

v. Relation

A bag of tuples is what we call Relation. In Pig Latin, the relations are unordered. Also, there is no guarantee that tuples are processed in any particular order.

Swami Keshvanand Institute of Technology, Management &Gramothan,
Ramnagaria, Jagatpura, Jaipur-302017, INDIA
Approved by AICTE, Ministry of HRD, Government of India
Recognized by UGC under Section 2(f) of the UGC Act, 1956
Tel. : +91-0141- 5160400Fax: +91-0141-2759555
E-mail: info@skit.ac.in Web: www.skit.ac.in

Scripting With Pig Latin:

What is Apache Pig Running Scripts?

Basically, to place Pig Latin statements and Pig commands in a single file, we use Pig scripts. It is good practice to identify the file using the *pig extension.

Executing Pig Script in Batch mode

Further, follow these steps, while we execute Pig script in batch mode.

Step 1

At very first, write all the required Pig Latin statements and commands in a single file. Then save it as a .pig file.

Step 2

Afterwards, execute the Apache Pig script. To execute Pig script from the shell (Linux), see:

- **Local mode**

```
$ pig -x local Sample_script.pig
```

- **MapReduce mode**

```
$ pig -x MapReduce Sample_script.pig
```

It is possible to execute it from the Grunt shell as well using the exec command.

```
grunt> exec /sampleScript.pig
```

Executing a Pig Script from HDFS: Also, we can execute a Pig script that resides in the HDFS. Let's assume there is a Pig script with the name Sample_script.pig in the HDFS directory named /pig_data/. To execute it, see.

```
$ pig -x mapreduce hdfs://localhost:9000/pig_data/Sample_script.pig
```

Suppose we have a file Employee_details.txt in HDFS with the following content.

```
Employee_details.txt
001,mehul,chourey,21,9848022337,Hyderabad
002,Ankur,Dutta,22,9848022338,Kolkata
003,Shubham,Sengar,22,9848022339,Delhi
004,Prerna,Tripathi,21,9848022330,Pune
005,Sagar,Joshi,23,9848022336,Bhuwaneshwar
006,Monika,sharma,23,9848022335,Chennai
007,pulkit,pawar,24,9848022334,trivendram
008,Roshan,Shaikh,24,9848022333,Chennai
```

Now, also we have a sample script with the name sample_script.pig, in the same HDFS directory. It contains statements performing operations and transformations on the Employee relation.

```
Employee = LOAD 'hdfs://localhost:9000/pig_data/Employee_details.txt' USING
PigStorage(',') as (id:int, firstname:chararray, lastname:chararray, phone:chararray,
city:chararray);

Employee_order = ORDER Employee BY age DESC;

Employee_limit = LIMIT Employee_order 4;

Dump Employee_limit;
```

- The script will load the data in the file named Employee_details.txt as a relation named Employee, in the first statement.

- Moreover, the script will arrange the tuples of the relation in descending order, based on age, and store it as Employee_order, in the second statement.
- The script will store the first 4 tuples of Employee_order as Employee_limit, in the third statement.
- Ultimately, last and the 4rth statement will dump the content of the relation Employee_limit.

Further, let's execute the sample_script.pig.

$ pig -x mapreduce

hdfs://localhost:9000/pig_data/sample_script.pig

In this way, Pig gets executed and gives you the output like:

```
(7,Pulkit,Pawar,24,9848022334,trivendram)
(8,Roshan,Shaikh,24,9848022333,Chennai)
(5,Sagar,Joshi,23,9848022336,Bhuwaneshwar)
(6,Monika,Sharma,23,9848022335,Chennai)
2015-10-19 10:31:27,446 [main] INFO  org.apache.pig.Main - Pig script completed in 12
minutes, 32 seconds and 751 milliseconds (752751 ms)
```

Word Count Problem Solution with Pig Script:

1.Load the data from HDFS:

Use Load statement to load the data into a relation .

As keyword used to declare column names, as we dont have any columns, we declared only one column named line.

```
data_input= LOAD 'hdfs://localhost:9000/wc_count/myfile.txt'
as (line:Chararray);
```

2. Convert the Sentence into words:

The data we have is in sentences. So we have to convert that data into words using

TOKENIZE Function

TOKENIZE(line));

(or)

If we have any delimeter like space we can specify as

TOKENIZE(line,' '));

3. Convert Column into Rows: I mean we have to convert every line of data into multiple rows ,for this we have function called FLATTEN in pig.

Using FLATTEN function the bag is converted into tuple, means the array of strings converted into multiple rows.

Words = FOREACH data_input GENERATE FLATTEN(TOKENIZE(line,' ')) AS word;

3. Apply GROUP BY:

We have to count each word occurance, for that we have to group all the words.

```
Grouped = GROUP Words BY word;
```

4. Generate word count:

```
wordcount = FOREACH Grouped GENERATE group, COUNT(Words);
```

5. We can print the word count on console using Dump.

```
DUMP wordcount;
```

Output will be like below.

```
(a,2)
(is,2)
(This,1)
(class,1)
(hadoop,2)
(bigdata,1)
(technology,1)
```

Below is the complete given script. That can be saved with word_count.pig extension.

```
data_input= LOAD 'hdfs://localhost:9000/wc_count/myfile.txt' as (line:Chararray);
Words = FOREACH data_input GENERATE FLATTEN(TOKENIZE(line,' ')) AS word;
Grouped = GROUP Words BY word;
wordcount = FOREACH Grouped GENERATE group, COUNT(Words);
DUMP wordcount;
```

Script can be executed by using following command:

```
grunt >  pig -x mapreduce wordcount.pig
```