Introduction to Apache Pig

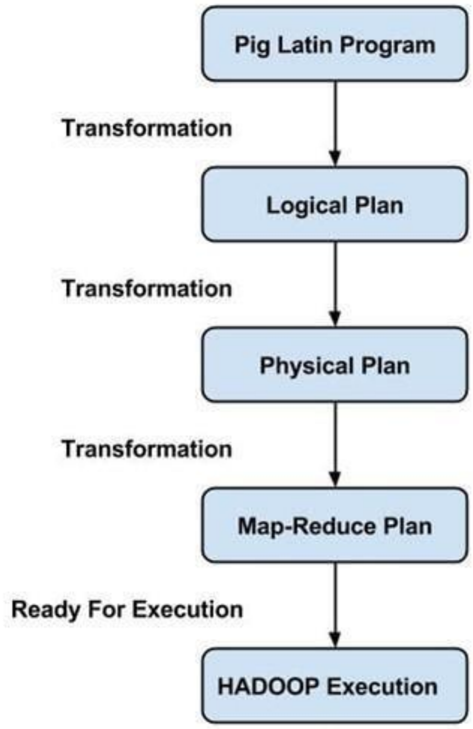
What is Apache Pig?

- Pig is a high-level programming language useful for analyzing large data sets. Pig was a result of development effort at Yahoo!.
- Apache Pig enables people to focus more on analyzing bulk data sets and to spend less time writing Map-Reduce programs. Similar to Pigs, who eat anything, the Apache Pig programming language is designed to work upon any kind of data. That's why the name, Pig!.

gith
akp0312000

- The Architecture of Pig consists of two components:
  - **Pig Latin**, which is a language.
  - **A runtime environment**, for running Pig Latin programs.

- A Pig Latin program consists of a series of operations or transformations which are applied to the input data to produce output. These operations describe a data flow which is translated into an executable representation, by Hadoop Pig execution environment. Underneath, results of these transformations are series of MapReduce jobs which a programmer is unaware of. So, in a way, Pig in Hadoop allows the programmer to focus on data rather than the nature of execution.



Execution modes:

Pig in Hadoop has two execution modes:

1. **Local mode**: In this mode, Hadoop Pig language runs in a single JVM and makes use of local file system. This mode is suitable only for analysis of small datasets using Pig in Hadoop.

2. **Map Reduce mode**: In this mode, queries written in Pig Latin are translated into MapReduce jobs and are run on a Hadoop cluster. MapReduce mode is useful of running Pig on large datasets.

Commands:

Command Name	**Syntax**	**Example**

Load	LOAD 'info' [USING FUNCTION] [AS SCHEMA];
	• LOAD is a relational operator.
	• 'info' is a file that is required to load. It contains any type of data.
	• USING is a keyword.
	• FUNCTION is a load function.
	• AS is a keyword.
	• SCHEMA is a schema of passing file, enclosed in parentheses.
		grunt> A = LOAD '/pigexample/pload.txt' USING PigStorage(',') AS (a1:int,a2:int,a3:int,a4:int) ;
		grunt> B = LOAD '/pigexample/pload.txt' USING PigStorage(',') AS (user:chararray,url:chararray,timestamp:chararray);
		Now, execute and verify the data.
		grunt> DUMP A;
		To know, structure or schema of table
		grunt> DESCRIBE A;

Store	STORE Relation_name INTO 'required_directory_path' [USING function];	STORE A INTO '/user/PigExamples/PigOutput/' USING PigStorage('|');

github.com/akp0312000

		Note: PigOutput directory will be created automatically.

Filter		grunt> B = LOAD '/pigexample/pfilter.txt' USING PigStorage(',') AS (a1:int,a2:int);
		grunt> DUMP B;
		1,2
		2,8
		4,5
		9,3
		7,8
		grunt> Result = FILTER A BY a2==8;
		grunt> DUMP B;
		2,8
		7,8

Foreach	The Apache Pig FOREACH operator generates data transformations based on columns of data.	grunt> A = LOAD '/pigexample/pforeach.txt' USING PigStorage(',') AS (a1:int,a2:int,a3:int) ; grunt> DUMP A; 1,2,3 4,5,6 7,8,9 grunt> fe = FOREACH A generate a1,a2; grunt> DUMP fe; 1,2 4,5 7,8

github.com/akp03112000

Distinct	The Apache Pig DISTINCT operator is used to remove duplicate tuples in a relation.	grunt> A = LOAD '/pigexample/pdistinct.txt' USING PigStorage(',') as (a1:int,a2:int,a3:int);
		grunt> DUMP A;
		1,3,5
		2,1,4
		1,3,5
		1,4,2
		2,1,4
		grunt> Result = DISTINCT A;
		grunt> DUMP Result;
		1,3,5
		1,4,2
		2,1,4

Group	It groups the tuples than contains similar kind of key.	grunt> A = LOAD '/pigexample/piginput2.txt' USING PigStorage(',') AS (fname:chararray,l_name:chararray,id:int); grunt> DUMP A; Jason,Roy,1 Chris,Roy,3 Nick,Holder,4 James,William,5 Chris,Holder,6 Mark,Holder,6 Anty,Thomson,5 grunt> groupbylname = group A by l_name ; DUMP groupbylname;

github.com/akp03112000

		(Roy,{(Chris,Roy,3),(Jason,Roy,1)}) (Holder,{(Mark,Holder,6),(Chris,Holder,6),(Nick,Holder,4)}) (Thomson,{(Anty,Thomson,5),(John,Thomson,2)}) (William,{(James,William,5)})
Limit	The Apache Pig LIMIT operator is used to limit the number of output tuples. However, if you specify the limit of output tuples equal to or more than the number of tuples exists, all the tuples in the relation are returned.	grunt> A = LOAD '/pigexample/plimit.txt' USING PigStorage(',') AS (a1:int,a2:int,a3:int) ; grunt> DUMP A; 5,2,1 3,2,7 8,2,3 4,3,2 9,2,1 grunt> Result = LIMIT A 2; grunt> DUMP Result; 5,2,1 3,2,7
Order By	The Apache Pig ORDER BY operator sorts a relation based on one or more fields. It maintains the order of tuples.	grunt> A = LOAD '/pigexample/porder.txt' USING PigStorage(',') AS (a1:int,a2:int,a3:int) ; grunt> DUMP A; 5,2,1 3,2,7 8,2,3 4,3,2 9,2,1

github.com/akp03112000

		grunt> Result = ORDER A BY a1 DESC; grunt> DUMP Result; 9,2,1 8,2,3 5,2,1 4,3,2 3,2,7
Split	The Apache Pig SPLIT operator breaks the relation into two or more relations according to the provided expression.	grunt> A = LOAD '/pigexample/psplit.txt' USING PigStorage(',') AS (a1:int,a2:int) ; grunt> DUMP A; 3,2 1,8 4,9 2,6 1,7 2,1 grunt> SPLIT A INTO X IF a1<=2, Y IF a1>2; grunt> DUMP X; 1,8 2,6 1,7 2,1 grunt> DUMP Y; 3,2

github.com/akp03112000

		4,9

Union	The Apache Pig UNION operator is used to compute the union of two or more relations. It doesn't maintain the order of tuples. It also doesn't eliminate the duplicate tuples.	grunt> A = load '/pigexample/punion1.txt' using PigStorage(',') as (a1:int,a2:int);
		grunt> DUMP A;
		1,2
		3,4
		grunt> B = LOAD '/pigexample/punion2.txt' USING PigStorage(',') AS (b1:int,b2:int,b3:int);
		grunt> DUMP B;
		5,6,7
		8,9,10
		grunt> Result = UNION A,B;
		grunt> DUMP Result;
		5,6,7
		8,9,10
		1,2
		3,4

github.com/akp03112000