UNIT-4

PIG

Hadoop Programming Made Easier

UNIT-IV

Pig – Hadoop Programming Made Easier

Syllabus

Admiring the Pig Architecture, Going with the Pig Latin Application Flow, Working through the ABC of Pig Latin, Uncovering Pig Latin Structures, Looking at Pig Data Type and syntax, Evaluating Local and Distributed Modes of Running Pig scripts, Checking Out Pig Script interfaces, Scripting with Pig Latin.

Introduction:

Apache Pig is an abstraction over MapReduce. It is a tool/platform which is used to analyze larger sets of data representing them as data flows. Pig is generally used with **Hadoop**: we can perform all the data manipulation operations in Hadoop using Pig.

To write data analysis programs, Pig provides a high-level language known as **Pig Latin**. This language provides various operators using which programmers can develop their own functions for reading, writing, and processing data.

To analyze data using **Apache Pig**, programmers need to write scripts using Pig Latin language. All these scripts are internally converted to Map and Reduce tasks. Apache Pig has a component known as **Pig Engine** that accepts the Pig Latin scripts as input and converts those scripts into MapReduce jobs.

Why Do We Need Apache Pig?

Programmers who are not so good at Java normally used to struggle working with Hadoop, especially while performing any MapReduce tasks. Apache Pig is a boon for all such programmers.

- Using **Pig Latin**, programmers can perform MapReduce tasks easily without having to type complex codes in Java.
- Apache Pig uses **multi-query approach**, thereby reducing the length of codes. For example, an operation that would require you to type 200 linesof code(LoC) in Java can be easily done by typing as less as just 10 LoC in

Apache Pig. Ultimately Apache Pig reduces the development time by almost 16 times.

- Pig Latin is **SQL-like language** and it is easy to learn Apache Pig when you arefamiliar with SQL.
- Apache Pig provides many built-in operators to support data operations like joins, filters, ordering, etc. In addition, it also provides nested data types like tuples, bags, and maps that are missing from MapReduce.

Features of Pig:

Apache Pig comes with the following features −

- **Rich set of operators** − It provides many operators to perform operationslike join, sort, filer, etc.
- **Ease of programming** − Pig Latin is similar to SQL and it is easy to write aPig script if you are good at SQL.
- **Optimization opportunities** − The tasks in Apache Pig optimize their execution automatically, so the programmers need to focus only on semantics of the language.
- **Extensibility** − Using the existing operators, users can develop their own functions to read, process, and write data.
- **UDF's** − Pig provides the facility to create **User-defined Functions** in other programming languages such as Java and invoke or embed them in Pig Scripts.
- **Handles all kinds of data** − Apache Pig analyzes all kinds of data, both structured as well as unstructured. It stores the results in HDFS.

Apache Pig Vs MapReduce

Listed below are the major differences between Apache Pig and MapReduce.

ApachePig	MapReduce
Apache Pig is a data flow language.	MapReduce is a data processing paradigm.
It is a high level language.	MapReduce is low level and rigid.
Performing a Join operation in Apache Pig is pretty simple.	It is quite difficult in MapReduce to perform a Join operation Between datasets.
Any novice programmer with a basic knowledge of SQL can work conveniently with Apache Pig.	Exposure to Java is must towork with MapReduce.
Apache Pig uses multi-query approach, thereby reducing the length of the codes to a great extent.	MapReduce will require almost 20 times more the number of lines to perform the same task.
There is no need for compilation. On execution, every Apache Pig operator is converted internally into a MapReduce job.	MapReduce jobs have a long compilation process.

Apache Pig Vs SQL

Listed below are the major differences between Apache Pig and SQL.

Pig	**SQL**
Pig Latin is a **procedural** language.	SQL is a **declarative** language.
In Apache Pig, **schema** is optional. We can store data without designing a schema (valuesare stored as $01, $02 etc.)	Schema is mandatory in SQL.
The data model in Apache Pig is **nestedrelational**.	The data model used in SQL **is flat relational**.
Apache Pig provides limited opportunityfor **Query optimization**.	There is more opportunity for query optimization in SQL.

In addition to above differences, Apache Pig Latin –

- Allows splits in the pipeline.
- Allows developers to store data anywhere in the pipeline.
- Declaresexecution plans.
- Provides operators to perform ETL (Extract, Transform, and Load) functions.

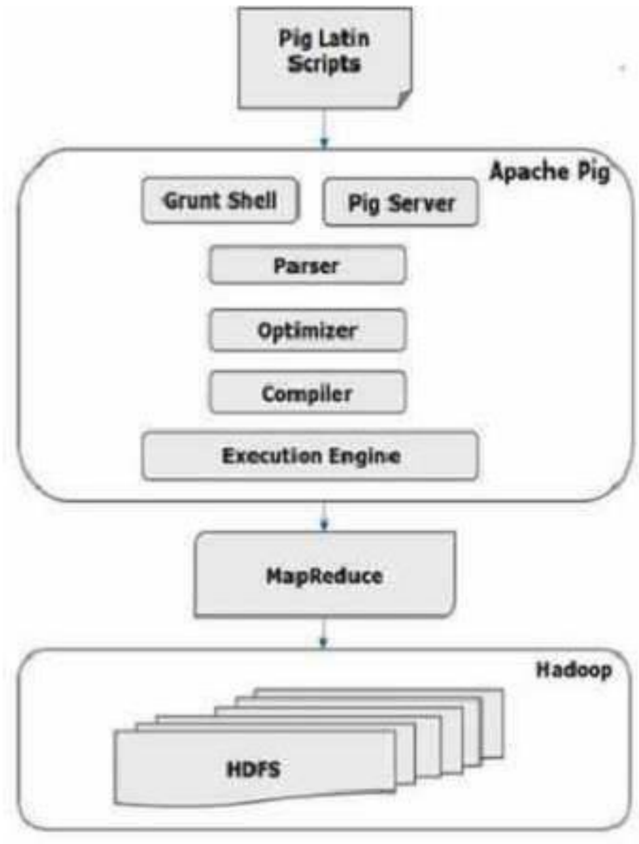
Apache Pig – Architecture

- The language used to analyze data in Hadoop using Pig is known as **Pig Latin**. It is a high level data processing language which provides a rich set of data types and operators to perform various operations on the data.
- To perform a particular task Programmers using Pig, programmers need to write a Pig script using the Pig Latin language, and execute

them using anyof the execution mechanisms

(Grunt Shell, UDFs, Embedded). After execution, these scripts will go through a series of transformations applied.by the Pig Framework, to produce the desired output.

- Internally, Apache Pig converts these scripts into a series of MapReduce jobs, and thus, it makes the programmer's job easy. The architecture of Apache Pig is shown below.



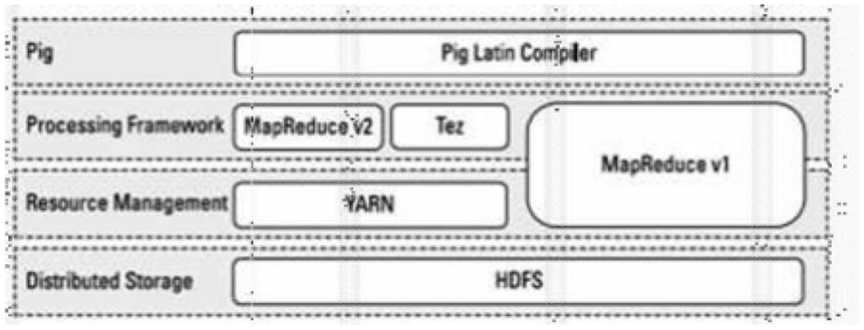
Admiring the pig architecture

Pig ismadeup of two components:

The language itself:_As proof that programmers have a sense of humor, the Programming language for Pig is known as Pig Latin, a high-level language that allows you to write dataprocessing and analysis programs.

The Pig Latin compiler: The Pig Latin compiler converts the Pig Latin code into executable code. The executable code is either in the form of MapReduce jobs or it can spawn a process where a virtual Hadoop instance is created to run the Pig node on a singlenode.

The sequence of MapReduce programs enables Pig programs to do data processing and analysis in parallel, leveraging Hadoop MapReduce and HDFS. Running the Pig job in the virtual Hadoop instance is a useful strategy for testing your Pig scripts.



Pig relates to the Hadoop ecosystem

Pig programs can run on MapReduce v1 or MapReduce v2 without any code changes, regardless of what mode your cluster is running. However, Pig scripts can also run using the Tez API instead. Apache Tez provides a more efficient execution framework than MapReduce. YARN enables application frameworks other than MapReduce (like Tez) to run on Hadoop. Hive can also run against the Tez framework.

Apache Pig Components

As shown in the figure, there are various components in the Apache Pig framework. Let us take a look at the major components.

Parser

Initially the Pig Scripts are handled by the Parser. It checks the syntax of the script, does type checking, and other miscellaneous checks. The output of the parser will be a DAG Latin statements and logical (directed acyclic graph), which represents the Pig operators.
In the DAG, the logical operators of the script are represented and the dataflows arerepresented as edges.

Optimizer

The logical plan (DAG) is passed to the logical optimizer, which carries out thelogical optimizations such as projection and pushdown.

Compiler

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

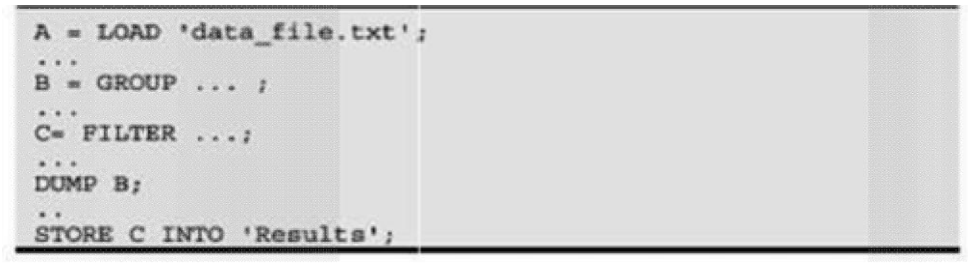
Execution engine

Finally the MapReduce jobs are submitted to Hadoop in a sorted order. Finally, these MapReducedesired results.

Going with the Pig Latin Application Flow

At its core, Pig Latin is a dataflow language, where we define a data stream and a series of transformationsthat areappliedto the dataas it flows throughyour application.

This is in contrast to a control flow language (like C or Java), where we write a series of instructions. In control flow languages, we use constructs like loops and conditional logic (like an if statement). You won’tfindloops and if statementsin Pig Latin.



```
A = LOAD 'data_file.txt';
...
B = GROUP ... ;
...
C= FILTER ...;
...
DUMP B;
..
STORE C INTO 'Results';
```

Sample pig code

Looking at each line in turn, you can see the basic flow of a Pig program

1. **Load:** we first load (LOAD) the data you want to manipulate. As in a typical MapReduce job, that data is stored in HDFS. For a Pig program to access the data, you first tell Pig what file or filesto use. For thattask, you use the LOAD'datafile'command.

Here, 'datafile' can specify either an HDFS file or a directory. If a directory is specified, all files in that directoryare loaded intothe program.

If the data is stored in a file format that isn’t natively accessible to Pig, you can optionally add the USING function to the LOAD statement to specify a user-defined function that can read in (and interpret) the data.

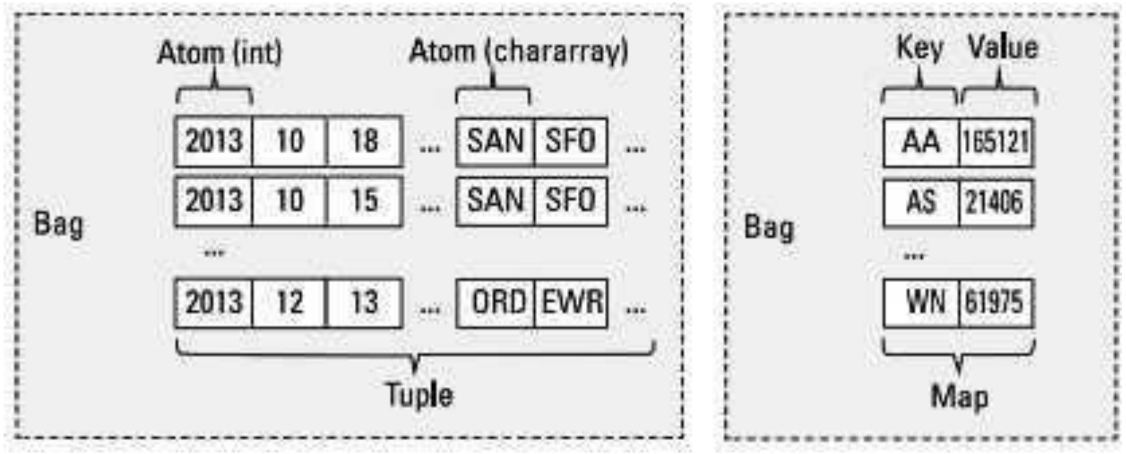
2. **Transform:** You run the data through a set of transformations that, way under the hoodand far removed from anything you have to concern yourself with, are translated into a set of Map and Reduce tasks.

The transformation logic is where all the data manipulation happens. Here, you can FILTER out rows that aren't of interest, JOIN two sets of data files, GROUP data to build aggregations, ORDER results, and do much, much more.

3. **Dump:** Finally, you dump (DUMP) the results to the screen or Store (STORE) the results in a file somewhere.

Pig Latin Data Model (pig data types)

The data model of Pig Latin is fully nested and it allows complex non-atomic data types such as **map** and **tuple**. Given below is the diagrammatical representation of Pig Latin's data model.



Atom: An atom is any single value, such as a string or a number 'Diego', **for example**. Pig's atomic values are scalar types that appear in most programming languages — int, long, float, double char array, and byte array.

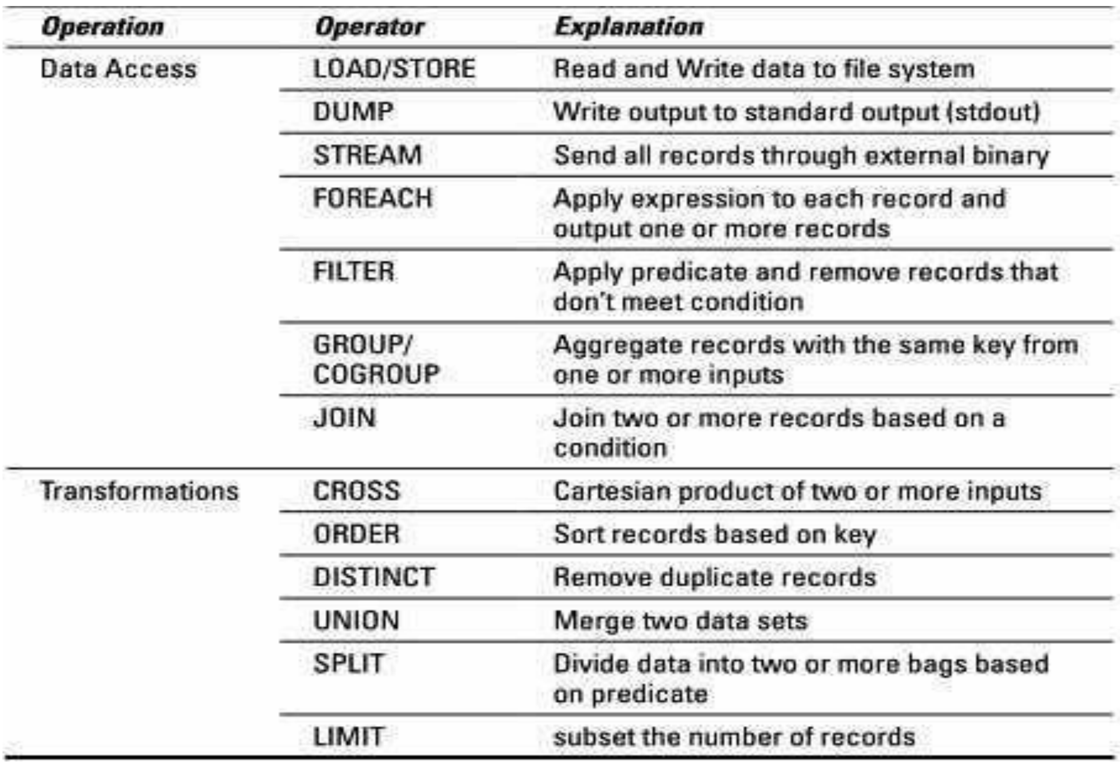
Tuple: A tuple is a record that consists of a sequence of fields. Each field can beof any type — 'Diego', 'Gomez', or 6, for example. Think of a tuple as a row ina table.

Bag: A bag is a collection of non-unique tuples. The schema of the bag is flexible — each tuple in the collection can contain an arbitrary number of fields, and each field can be of any type.

Map: A map is a collection of key value pairs. Any type can be stored in the value, and the key needs to be unique. The key of a map must be a char array and the value can be of any type.

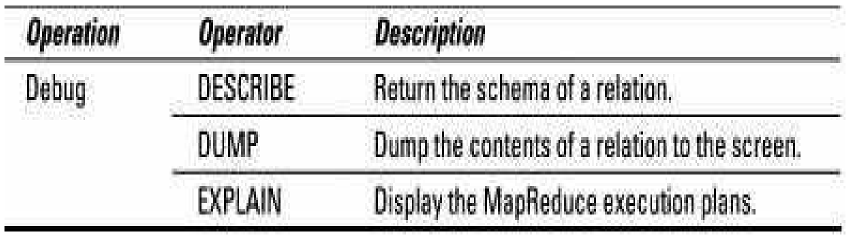
Pig latin opearations:

In a Hadoop context, accessing data means allowing developers to load, store, and stream data, whereas transforming data means taking advantage of Pig's ability to group, join, combine, split, filter, and sort data.



Operation	Operator	Explanation
Data Access	LOAD/STORE	Read and Write data to file system
	DUMP	Write output to standard output (stdout)
	STREAM	Send all records through external binary
	FOREACH	Apply expression to each record and output one or more records
	FILTER	Apply predicate and remove records that don't meet condition
	GROUP/ COGROUP	Aggregate records with the same key from one or more inputs
	JOIN	Join two or more records based on a condition
Transformations	CROSS	Cartesian product of two or more inputs
	ORDER	Sort records based on key
	DISTINCT	Remove duplicate records
	UNION	Merge two data sets
	SPLIT	Divide data into two or more bags based on predicate
	LIMIT	subset the number of records

Operators for Debugging and Trouble shooting



Operation	Operator	Description
Debug	DESCRIBE	Return the schema of a relation.
	DUMP	Dump the contents of a relation to the screen.
	EXPLAIN	Display the MapReduce execution plans.

Apache Pig Execution Modes

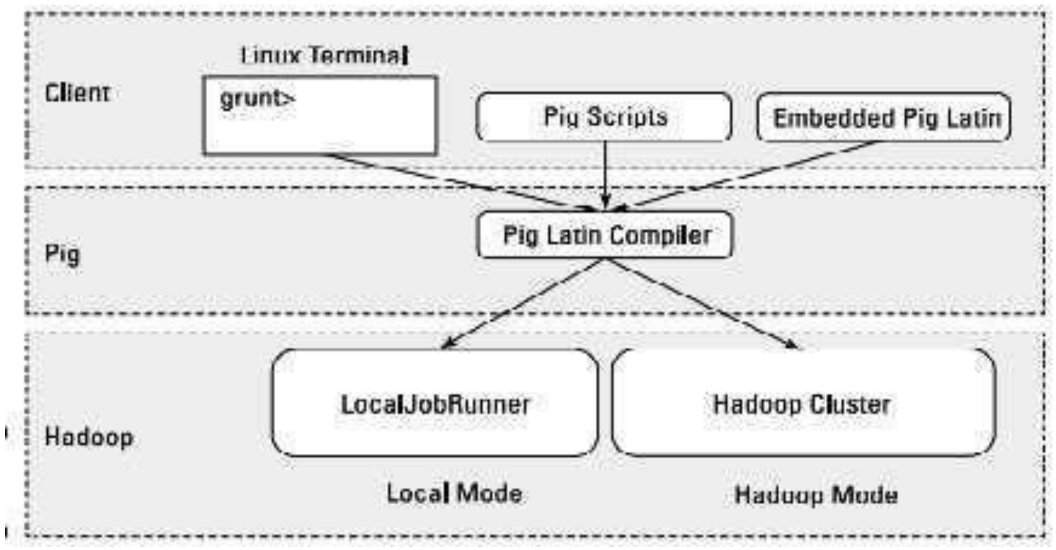
we can run Apache Pig in two modes, namely, **Local Mode** and **HDFS mode**.

Local Mode

In this mode, all the files are installed and run from your local host and local file system. There is no need of Hadoop or HDFS. This mode is generally used for testing purpose.

MapReduce Mode

MapReduce mode is where we load or process the data that exists in the Hadoop File System (HDFS) using Apache Pig. In this mode, whenever we execute the Pig Latin statements to process the data, a MapReduce job is invoked in the back-end to perform a particular operation on the data that exists in the HDFS.



Apache Pig Execution Mechanisms

Apache Pig scripts can be executed in three ways, namely, interactive mode, batch mode, and embedded mode.

Interactive Mode (Grunt shell) − You can run Apache Pig in interactive mode using the Grunt shell. In this shell, you can enter the Pig Latin statements and get the output (using Dump operator).

Batch Mode (Script) − You can run Apache Pig in Batch mode by writing the PigLatin script in a single file with **.pig** extension.

Embedded Mode (UDF) − Apache Pig provides the provision of defining our ownfunctions (**U**ser **D**efined **F**unctions) in programming languages such as Java, andusing them in our script.

Working through the ABCs of Pig Latin

Pig Latin is the language for Pig programs. Pig translates the Pig Latin script into MapReduce jobs that can be executed within Hadoop cluster. When coming up with Pig Latin, the development team followed three key design principles:

- ***Keep it simple.***
  Pig Latin provides a streamlined method for interacting with Java MapReduce. It‘s an abstraction, in other words, that simplifies the creation of parallel programs on the Hadoop cluster for data flows and analysis. Complex tasks may require a series of interrelated data transformations — such series are encoded as data flow sequences. Writing data transformation and flows as Pig Latin scripts instead of Java MapReduce programs makes these programs easier to write, understand, and maintain because a) you don‘t have to write the job in Java, b) you don‘t have to think in terms of MapReduce, and c) you don‘t need to come up with custom code to support rich data types.
  Pig Latin provides a simpler language to exploit your Hadoop cluster, thus making it easier for more people to leverage the power of Hadoop and become productive sooner.

- ***Make it smart.***
  You may recall that the Pig Latin Compiler does the work of transforming a Pig Latin program into a series of Java MapReduce jobs. The trick is to make sure that the compiler can optimize the execution of these Java MapReduce jobs automatically, allowing the user to focus on semantics rather than on how to optimize and access the data SQL is set up as a declarative query that you use to access structured data stored in an RDBMS. The RDBMS engine first translates the query to a data access method and then looks at the statistics and generates a series of data access approaches. The cost-based optimizer chooses the most efficient approach for execution.

- **Don’t limit development.**
  Make Pig extensible so that developers can add functions to address their particular business problems.

Uncovering Pig Latin structures

The problem we‘re trying to solve involves calculating the total number of flights flown by every carrier. Following listing is the Pig Latin script we‘ll use to answer this question.

```
records = LOAD '2013_subset.csv' USING PigStorage(',') AS
          (Year,Month,DayofMonth,DayOfWeek,DepTime,CRSDep
          Time,ArrTime,CRSArrTime,UniqueCarrier,FlightNum
          ,TailNum,ActualElapsedTime,CRSElapsedTime,AirTi
          me,ArrDelay,DepDelay,Origin,Dest,Distance:int,T
          axiIn,TaxiOut,Cancelled,CancellationCode,Divert
          ed,CarrierDelay,WeatherDelay,NASDelay,SecurityD
          elay,LateAircraftDelay);

milage_recs = GROUP records ALL;
tot_miles = FOREACH milage_recs GENERATE
          SUM(records.Distance);

DUMP tot_miles;
```

Listing : Pig script calculating the total miles flown

The Pig script is a lot smaller than the MapReduce application you‘d need to accomplish the same task — the Pig script only has 4 lines of code. And not only is the code shorter, but it‘s even semi-human readable.

Most Pig scripts start with the LOAD statement to read data from HDFS.

In this case, we‘re loading data from a .csv file. Pig has a data model it uses, so next we need to map the file‘s data model to the Pig data mode. This is accomplished with the help of the USING statement.

We then specify that it is a comma-delimited file with the Pig Storage(',') statement followed by the AS statement defining the name of each of the columns

Aggregations are commonly used in Pig to summarize data sets.

The GROUP statement is used to aggregate the records into a single record mileage recs. The ALL statement is used to aggregate all tuples into a single group. Note that some statements

— including the following SUM statement — requires a preceding GROUP ALL statementfor global sums.

FOREACH . . . GENERATE statements are used here to transform columns data In this case, we want to count the miles traveled in the records Distance column. The SUM statement computes the sum of the record Distance column into a single-column collection total miles.

The DUMP operator is used to execute the Pig Latin statement and display the results on the screen.

DUMP is used in interactive mode, which means that the statements are executable immediately and the results are not saved. Typically, you will either use the DUMP or STORE operators at the end of your Pig script.

Looking at Pig data types and syntax

Pig‘s data types make up the data model for how Pig thinks of the structure of the data it is processing. With Pig, the data model gets defined when the data is loaded. Any data you load into Pig from disk is going to have a particular schema and structure. In general terms, though, Pig data types can be broken into two categories: scalar types and complex types. Scalar types contain a single value, whereas complex types contain other types, such as the Tuple, Bag, and Map types. Pig Latin has these four types in its data model:

Atom: An atom is any single value, such as a string or a number — _Diego‘, for example. Pig‘s atomic values are scalar types that appear in most programming languages — int, long, float, double, chararray, and bytearray, for example. See Figure 2 to see sample atom types

Tuple: A tuple is a record that consists of a sequence of fields. Each field can be of any type — _Diego‘, _Gomez‘, or 6, for example. Think of a tuple as a row in a table.

Bag: A bag is a collection of non-unique tuples. The schema of the bag is flexible — each tuple in the collection can contain an arbitrary number of fields, and each field can be of any type.

Map: A map is a collection of key value pairs. Any type can be stored in the value, and the key needs to be unique. The key of a map must be a chararray and the value can be of any type. Figure -2 offers some fine examples of Tuple, Bag, and Map data types, as well.

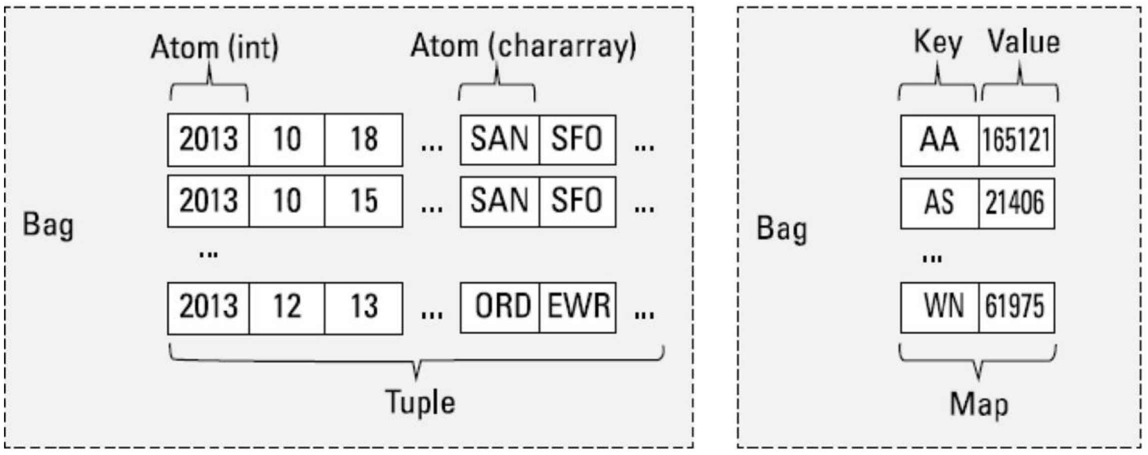


Figure 2:Sample Pig Data Types

In a Hadoop context, accessing data means allowing developers to load, store, and stream data, whereas transforming data means taking advantage of Pig‘s ability to group, join, combine, split, filter, and sort data. Table 1 gives an overview of the operators associated with each operation.

Operation	*Operator*	*Explanation*
Data Access	LOAD/STORE	Read and Write data to file system
	DUMP	Write output to standard output (stdout)
	STREAM	Send all records through external binary
	FOREACH	Apply expression to each record and output one or more records
	FILTER	Apply predicate and remove records that don't meet condition
	GROUP/ COGROUP	Aggregate records with the same key from one or more inputs
	JOIN	Join two or more records based on a condition
Transformations	CROSS	Cartesian product of two or more inputs
	ORDER	Sort records based on key
	DISTINCT	Remove duplicate records
	UNION	Merge two data sets
	SPLIT	Divide data into two or more bags based on predicate
	LIMIT	subset the number of records

Pig also provides a few operators that are helpful for debugging and troubleshooting, as shown in Table 2:

Operation	***Operator***	***Description***
Debug	DESCRIBE	Return the schema of a relation.
	DUMP	Dump the contents of a relation to the screen.
	EXPLAIN	Display the MapReduce execution plans.

The optional USING statement defines how to map the data structure within the file to the Pig data model —in this case, the PigStorage () data structure, which parses delimited text files. The optional AS clause defines a schema for the data that is being mapped. If youdon‘t use an AS clause, you‘re basically telling the default LOAD Func to expect a plain text file that is tab delimited.

Evaluating Local and Distributed Modes of Running Pig scripts

Before you can run your first Pig script, you need to have a handle on how Pig programs can be packaged with the Pig server.Pig has two modes for running scripts, as shown in

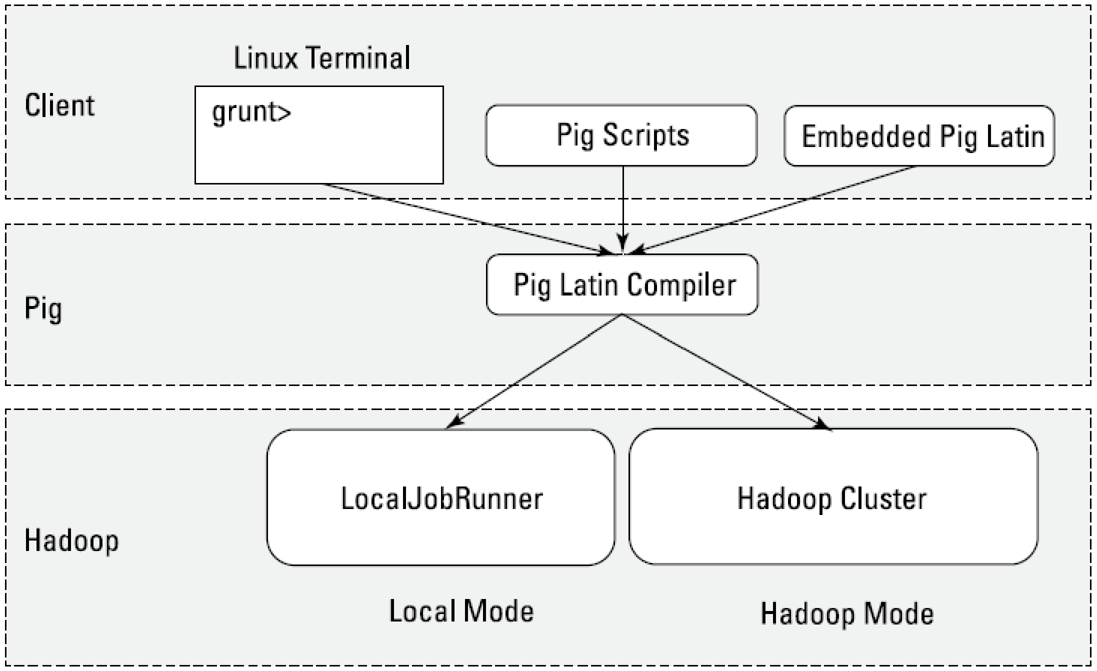


Figure 3.

Local mode

All scripts are run on a single machine without requiring Hadoop MapReduce and HDFS. This can be useful for developing and testing Pig logic. If you‘re using a small set of data to develope or test your code, then local mode could be faster than going through the

MapReduce infrastructure.

Local mode doesn‘t require Hadoop. When you run in Local mode, the Pig program runs in the context of a local Java Virtual Machine, and data access is via the local file system of a single machine. Local mode is actually a local simulation of MapReduce in Hadoop‘s Local Job Runner class

MapReduce mode (also known as Hadoop mode)

Pig is executed on the Hadoop cluster. In this case, the Pig script gets converted into a series of MapReduce jobs that are then run on the Hadoop cluster.If you have a terabyte of data that you want to perform operations on and you want to interactively develop a program, you may soon find things slowing down considerably, and you may start growing your storage.

Local mode allows you to work with a subset of your data in a more interactive manner so that you can figure out the logic (and work out the bugs) of your Pig program. After you have things set up as you want them and your operations are running smoothly, you can then run the script against the full data set using MapReduce mode.

Evaluating Local and Distributed Modes of Running Pig scripts

Before you can run your first Pig script, you need to have a handle on how Pig programs can be packaged with the Pig server.Pig has two modes for running scripts, as shown in Figure 3.

- **Script:** This method is nothing more than a file containing Pig Latin commands, identified by the .pig suffix (FlightData.pig, for example). Ending your Pig program with the .pig extension is a convention but not required. The commands are interpreted by the Pig Latin compiler and executed in the order determined by the Pig optimizer.
- Grunt: Grunt acts as a command interpreter where you can interactively enter Pig Latin at the Grunt command line and immediately see the response. This method is helpful for prototyping during initial development and with what-if scenarios.
- Embedded: Pig Latin statements can be executed within Java, Python, or JavaScript programs.

Pig scripts, Grunt shell Pig commands, and embedded Pig programs can run in either Local mode or MapReduce mode. The Grunt shell provides an interactive shell to submit Pig commands or run Pig scripts. To start the Grunt shell in Interactive mode, just submit the command pig at your shell. To specify whether a script or Grunt shell is executed locally or in Hadoop mode just specify it in the –x flag to the pig command. The following is an example of how you‘d specify running your Pig script in local mode

pig -x local milesPerCarrier.pig

Here‘s how you‘d run the Pig script in Hadoop mode, which is the default if you don‘t specify the flag: pig -x mapreduce milesPerCarrier.pig

By default, when you specify the pig command without any parameters, it starts the Grunt shell in Hadoop mode. If you want to start the Grunt shell in local mode just add the –x local flag to the command.

Here is an example:

pig -x local

Scripting with Pig Latin

Hadoop is a rich and quickly evolving ecosystem with a growing set of new applications. Rather than try to keep up with all the requirements for new capabilities, Pig is designed to be extensible via user-defined functions, also known as UDFs. UDFs can be written in a number of programming languages, including Java, Python, and JavaScript. Developers are also posting and sharing a growing collection of UDFs online. (Look for Piggy Bank and DataFu, to name just two examples of such online collections.) Some of the Pig UDFs that are part of these repositories are LOAD/STORE functions (XML, for example), date time functions, text, math, and stats functions.

Pig can also be embedded in host languages such as Java, Python, and JavaScript, which allows you to integrate Pig with your existing applications. It also helps overcome limitations in the Pig language. One of the most commonly referenced limitations is that Pig doesn‘t support control flow statements: if/else, while loop, for loop, and condition statements. Pig natively supports data flow, but needs to be embedded within another language to provide control flow. There are tradeoffs, however of embedding Pig in a control-flow language. For example if a Pig statement is embedded in a loop, every time the loop iterates and runs the Pig statement, this causes a separate MapReduce job to run.

Assignment -Cum -Tutorial Questions

SECTION –A

Objective Questions

Pig operates in mainly how many nodes?

a) Two b) Three c) Four d) Five

You can run Pig in batch mode using ___________

a) Pig Shell Command **b) Pig Script** c) Prig Option d) All

Pig Latin Statements are generally organized in one of the following ways?

a) A LOAD statement to read data from the file system.

b) A Series of "transformation" statements to process the data.

c) A DUMP statement to view results or a STORE statement to save the results

d) ALL

Which of the following function is used to read data in PIG?

a) WRITE b) READ **c) LOAD** d) None

You can run PIG in interactive mode using the ______shell

a) GRUNT b) FS D) HDFS d) None

Which of the following will run pig in local mode?

a) $pig – x – local

b) $pig –x tez_local

c) $pig

d) None

Which of the following is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs.

a) Pig Latin **b) Oozie** c) Pig d) Hive

Pig Latin Scripting language is not only a higher-level data flow language but also has operators similar to

a) JSON **b) SQL** c) XML d) None

Which of the following is data flow scripting language for analyzing unstructured data?

a) Mahoot b) Hive **c) Pig** d) None

which of the following command is used to show values to keys used in Pig?

a) set b) Declare c) Display d) None

PigUnit runs in Pig's__________________ mode by default.

a) Local b) Tex c) MapReduce d) None

Pig Operates in mainly how many nodes?

a) 2 b) 3 c) 4 d) 5

What language is used in Apache Pig?

a) Python b) Java c) Perl **d) Pig Latin**

What is the function of the Pig Latin statement "Group"

a) Groups data based on a specified key

b) Sorts data in ascending order

c) Joins tow datasets

d) Performs a cross-product of two datasets.

What is the function of the PIG Latin statement "FILTER"?

a) Groups data based on a specified key

b) Sorts data in ascending order

c) Filters data based on a specified condition

d) Performs a cross-product of two datasets

What is the function of Pig Latin statement "FOREACH"

a) Groups data based on a specified key

b) Sorts data in ascending order

c) Applies a transformation to each Record

d) Performs a cross-product of two datasets

What is the function of Pig Latin statement "JOIN"

a) Groups data based on a specified key

b) Sorts data in ascending order

c) Joins two datasets based on common key

d) Performs a cross-product of two datasets

What is the function of Pig Latin statement "ORDER"

a) Groups data based on a specified key

b) Sorts data in ascending order

c) Joins two datasets based on common key

d) Performs a cross-product of two datasets

What is the function of Pig Latin statement "LIMIT"

a) Groups data based on a specified key

b) Sorts data in ascending order

c) Joins two datasets based on common key

d) Limits the number of records returned

SECTION –B

SUBJECTIVE QUESTIONS

1. Illustrate Lazy Evaluation in pig with one example
2. Draw and explain Pig architecture.
3. Summarize Pig Data types with examples.
4. Explain how to run pig script in Distributed Mode.
5. Analyze Basic Pig Latin Statements
6. Examine the purpose of Joins in PIG
7. Illustrate various types of Operators in PIG
8. What is UDF? Explain types of PIG UDF's
9. Explain the architecture of Pig with a neat diagram.
10. Discuss about Pig Latin Application Flow.
11. Apache pig supports both primitive and complex data types. List and explain the above
12. data types with an example.
13. Describe the execution types of Apache Pig Scripts
14. Explain the following operators in Pig Latin.

    (i) flatten operator (ii) Relational operators

Queris

1. Consider the student data file (student.txt). Data in the following format Name, District, Age,

gender.

(i) Write a PIG Script to display Names of all male Students.

(ii) Write a PIG Script to find the number of students from Hyderabad district.

(iii)Write a PIG Script to display district wise count of all female students.

2. Consider the Olympics dataset. The data set consists of the following fields:

Athlete: Name of the athlete

Country: The name of the country participating in Olympics

Year: The year in which Olympics is conducted

Sport: Sports name

Total Medals: Total no. of medals

Find the total number of medals won by each country in swimming using Apache Pig

latin.

3. A text file contains following data (Id, Name, city) create a Pig Udf to convert Name in to

Upper case letters i.e. ROBIN and display it.

001, Robin, 22, newyork

4. Consider the Departmental Stores data file (stores.txt) in the following format

customerName, deptName, purchaseAmount.

(i) Write a Pig script to list total sales per departmental store.

(ii) Write a Pig script to list total sales per customer.