1. **Why Map-reduce program is needed in Pig Programming?**

* Apache Pig programs are written in a query language known as Pig Latin that is similar to the SQL query language. To execute the query, there is a need for an execution engine. The Pig engine converts the queries into MapReduce jobs and thus MapReduce acts as the execution engine and is needed to run the programs.
* The Pig Latin programs load the data from HDFS and run them through a series of business transformations (these transformations are internally converted to MapReduce task, so the developers don’t have to write the Java code for the business logic).
* Pigs’ operators are using Hadoops’ API depending upon the configurations the job is executed in local mode or Hadoop cluster. Pig is never passes any outputs to Hadoop instead set the inputs and data locations for map-reduce.

1. **What are advantages of pig over MapReduce?**

The following are the key differences between Apache Pig and MapReduce due to which Apache Pig came into picture:

* Apache Pig is a high-level data flow platform, whereas MapReduce is a low-level data processing paradigm.
* Without writing complex Java implementations in MapReduce, programmers can achieve the same implementations very easily using Pig Latin.
* Apache Pig provides nested data types like tuples, bags, and maps that are missing from MapReduce.
* Pig provides many built-in operators to support data operations like joins, filters, ordering, sorting etc. Whereas to perform the same function in MapReduce is a humongous task.
* Pig differs from MapReduce because, in MapReduce, the group by operation is performed at reducer side and filter, and also in the map phase the projection is implemented. Pig Latin provides the operations that are similar to MapReduce, such as groupby, orderby, and filters. We can analyze the Pig script and data flow to find the error checking. Pig Latin is lower in cost to write and maintain compared to MapReduce Java code.

|  |  |  |
| --- | --- | --- |
| **Characteristic** | **MapReduce** | **Pig** |
| Type of Language | Compiled Language | Scripting Language |
| Level of Abstraction | Low Level of Abstraction | Higher Level of Abstraction |
| Code | More lines of code is required. | Compatively less lines of code than Hadoop MapReduce. |
| Code Efficiency | Code efficiency is high. | Code efficiency is relatively less. |

1. **What is pig engine and what is its importance?**

Pig engine is an environment to execute the Pig Latin programs. It converts Pig Latin operators into a series of MapReduce jobs.

1. **What are the modes of Pig execution?**

The execution modes in Apache Pig are:

**MapReduce Mode**: This is the default mode, which requires access to a Hadoop cluster and HDFS installation. Since, this is a default mode, it is not necessary to specify -x flag (you can execute pig OR pig -x mapreduce). The input and output in this mode are present on HDFS.

**Local Mode:** With access to a single machine, all files are installed and run using a local host and file system. Here the local mode is specified using ‘-x flag’ (pig -x local). The input and output in this mode are present on local file system.

1. **What is grunt shell in Pig?**

Using Grunt i.e. Apache Pig’s interactive shell, users can interact with HDFS or the local file system.

grunt > run script.pig

To start Grunt, users should use *pig –x local*command . This command will prompt Grunt shell. To exit from grunt shell, press CTRL+D or just type exit.

1. **What are the features of Pig Latin language?**

* Apache Pig has two main components – the **Pig Latin** language and the **Pig Run-time Environment**, in which Pig Latin programs are executed.
* Pig Latin provides various built-in operators like join, sort, filter, etc. to read, write, and process large data sets.
* Its optional to specify schemas in PigLatin.

Pig Latin is procedural.

PigLatin has nested relational data model.

* **Case Sensitivity Nature of Pig**

Case Sensitive :

* Name of Relations
* Name of Fields
* Name of Pig Latin Functions (PigStorage, COUNT)

Case InSensitive :

* Name of Parameters
* Pig Latin Keywords (LOAD, USING, AS, GROUP, BY, FOREACH, GENERATE, DUMP)

Pig Latin can handle both atomic data types like int, float, long, double etc. and complex data types like tuple, bag and map.

Atomic or scalar data types are the basic data types which are used in all the languages like string, int, float, long, double, char[], byte[]. These are also called the primitive data types.

The complex data types supported by Pig Latin are:

* **Tuple**: Tuple is an ordered set of fields which may contain different data types for each field.
* **Bag**: A bag is a collection of a set of tuples and these tuples are a subset of rows or entire rows of a table.
* **Map**: A map is key-value pairs used to represent data elements. The key must be a chararray [] and should be unique like column name, so it can be indexed and value associated with it can be accessed on the basis of the keys. The value can be of any data type.

**7.Is Pig latin commands case sensitive?**

It is difficult to say whether Apache Pig is case sensitive or case insensitive. For instance, user defined functions, relations and field names in pig are case sensitive i.e. the function  COUNT is not the same as function count or X=load ‘foo’ is not same as x=load ‘foo’. On the other hand, keywords in Apache Pig are case insensitive i.e. LOAD is same as load.

Pig script is both case sensitive and case insensitive. For example, in user defined functions, the field name, and relations are case sensitive,i.e, INTELLIPAAT is not same as intellipaat or M=load ‘test’ is not same as m=load ‘test’. And Pig script keywords are case insensitive i.e, LOAD is same as aload

**8. What is a data flow language?**

In a dataflow language, you have a stream of data which is passed from instruction to instruction to be processed.

dataflow programming is a [programming paradigm](https://en.wikipedia.org/wiki/Programming_paradigm) that models a program as a [directed graph](https://en.wikipedia.org/wiki/Directed_graph) of the data flowing between operations, thus implementing [dataflow](https://en.wikipedia.org/wiki/Dataflow) principles and architecture.