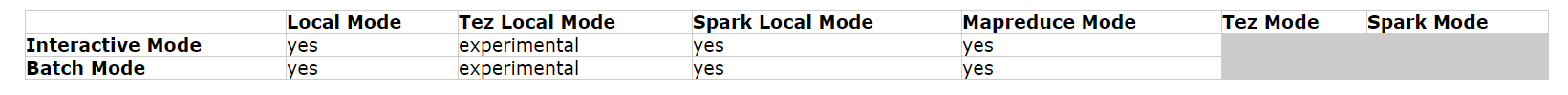
**Running Pig**

You can run Pig (execute Pig Latin statements and Pig commands) using various modes.



**Execution Modes**

Pig has six execution modes or exectypes:

* **Local Mode** - To run Pig in local mode, you need access to a single machine; all files are installed and run using your local host and file system. Specify local mode using the -x flag (pig -x local).
* **Tez Local Mode** - To run Pig in tez local mode. It is similar to local mode, except internally Pig will invoke tez runtime engine. Specify Tez local mode using the -x flag (pig -x tez\_local).

**Note:** Tez local mode is experimental. There are some queries which just error out on bigger data in local mode.

* **Spark Local Mode** - To run Pig in spark local mode. It is similar to local mode, except internally Pig will invoke spark runtime engine. Specify Spark local mode using the -x flag (pig -x spark\_local).

**Note:** Spark local mode is experimental. There are some queries which just error out on bigger data in local mode.

* **Mapreduce Mode** - To run Pig in mapreduce mode, you need access to a Hadoop cluster and HDFS installation. Mapreduce mode is the default mode; you can, *but don't need to*, specify it using the -x flag (pig OR pig -x mapreduce).
* **Tez Mode** - To run Pig in Tez mode, you need access to a Hadoop cluster and HDFS installation. Specify Tez mode using the -x flag (-x tez).
* **Spark Mode** - To run Pig in Spark mode, you need access to a Spark, Yarn or Mesos cluster and HDFS installation. Specify Spark mode using the -x flag (-x spark). In Spark execution mode, it is necessary to set env::SPARK\_MASTER to an appropriate value (local - local mode, yarn-client - yarn-client mode, mesos://host:port - spark on mesos or spark://host:port - spark cluster. For more information refer to spark documentation on Master URLs, *yarn-cluster mode is currently not supported*). Pig scripts run on Spark can take advantage of the [dynamic allocation](http://spark.apache.org/docs/latest/job-scheduling.html#dynamic-resource-allocation) feature. The feature can be enabled by simply enabling *spark.dynamicAllocation.enabled*. Refer to spark [configuration](http://spark.apache.org/docs/latest/configuration.html#dynamic-allocation) for additional configuration details. In general all properties in the pig script prefixed with *spark.* are copied to the Spark Application Configuration. Please note that Yarn auxillary service need to be enabled on Spark for this to work. See Spark documentation for additional details.

You can run Pig in either mode using the "pig" command (the bin/pig Perl script) or the "java" command (java -cp pig.jar ...).

***Examples***

This example shows how to run Pig in local and mapreduce mode using the pig command.

/\* local mode \*/

$ pig -x local ...

/\* Tez local mode \*/

$ pig -x tez\_local ...

/\* Spark local mode \*/

$ pig -x spark\_local ...

/\* mapreduce mode \*/

$ pig ...

or

$ pig -x mapreduce ...

/\* Tez mode \*/

$ pig -x tez ...

/\* Spark mode \*/

$ pig -x spark ...

# **Apache Pig Tutorial**

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.

# **Audience**

This tutorial is meant for all those professionals working on Hadoop who would like to perform MapReduce operations without having to type complex codes in Java.

# **Prerequisites**

To make the most of this tutorial, you should have a good understanding of the basics of Hadoop and HDFS commands.

## 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.

## Features of Pig

Apache Pig comes with the following features −

* **Rich set of operators** − It provides many operators to perform operations like join, sort, filer, etc.
* **Ease of programming** − Pig Latin is similar to SQL and it is easy to write a Pig 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.

|  |  |
| --- | --- |
| **Apache Pig** | **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 to work 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 (values are stored as $01, $02 etc.) | Schema is mandatory in SQL. |
| The data model in Apache Pig is **nested relational**. | The data model used in SQL **is flat relational**. |
| Apache Pig provides limited opportunity for **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.
* Declares execution plans.
* Provides operators to perform ETL (Extract, Transform, and Load) functions.

## Apache Pig Vs Hive

Both Apache Pig and Hive are used to create MapReduce jobs. And in some cases, Hive operates on HDFS in a similar way Apache Pig does. In the following table, we have listed a few significant points that set Apache Pig apart from Hive.

|  |  |
| --- | --- |
| **Apache Pig** | **Hive** |
| Apache Pig uses a language called **Pig Latin**. It was originally created at **Yahoo**. | Hive uses a language called **HiveQL**. It was originally created at **Facebook**. |
| Pig Latin is a data flow language. | HiveQL is a query processing language. |
| Pig Latin is a procedural language and it fits in pipeline paradigm. | HiveQL is a declarative language. |
| Apache Pig can handle structured, unstructured, and semi-structured data. | Hive is mostly for structured data. |

Applications of Apache Pig

Apache Pig is generally used by data scientists for performing tasks involving ad-hoc processing and quick prototyping. Apache Pig is used −

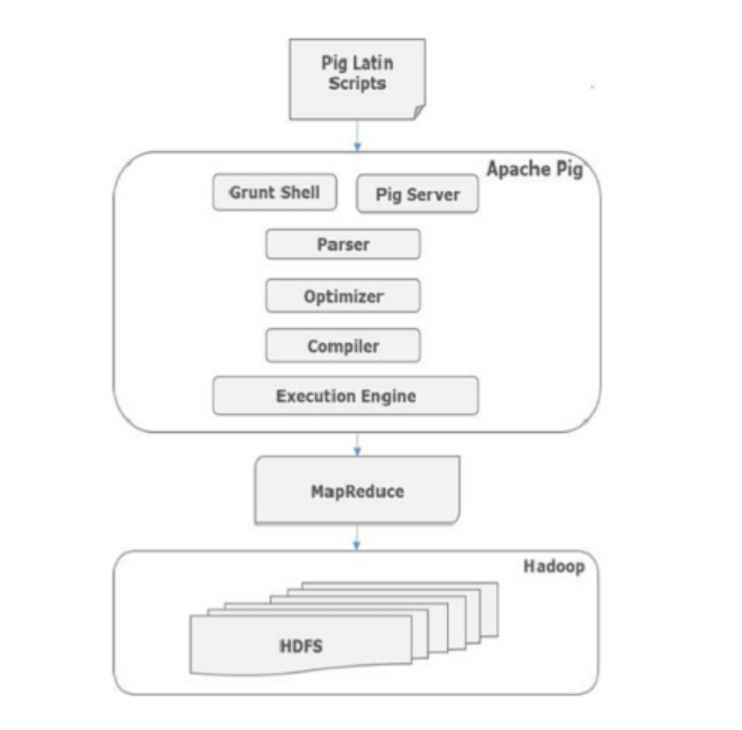
* To process huge data sources such as web logs.
* To perform data processing for search platforms.
* To process time sensitive data loads.

# **Apache Pig - Architecture**

The language used to analyze data in Hadoop using Pig is known as **Pig Latin**.

To perform a particular task Programmers using Pig, 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).

Internally, Apache Pig converts these scripts into a series of MapReduce jobs, and thus, it makes the programmer’s job easy



### Atom

Any single value in Pig Latin, irrespective of their data, type is known as an **Atom**. It is stored as string and can be used as string and number. int, long, float, double, chararray, and bytearray are the atomic values of Pig. A piece of data or a simple atomic value is known as a **field**.

**Example** − ‘raja’ or ‘30’

### Tuple

A record that is formed by an ordered set of fields is known as a tuple, the fields can be of any type. A tuple is similar to a row in a table of RDBMS.

**Example** − (Raja, 30)

### Bag

A bag is an unordered set of tuples. In other words, a collection of tuples (non-unique) is known as a bag. Each tuple can have any number of fields (flexible schema). A bag is represented by ‘{}’. It is similar to a table in RDBMS, but unlike a table in RDBMS, it is not necessary that every tuple contain the same number of fields or that the fields in the same position (column) have the same type.

**Example** − {(Raja, 30), (Mohammad, 45)}

A bag can be a field in a relation; in that context, it is known as **inner bag**.

**Example** − {Raja, 30, **{9848022338, raja@gmail.com,}**}

### Map

A map (or data map) is a set of key-value pairs. The **key** needs to be of type chararray and should be unique. The **value** might be of any type. It is represented by ‘[]’

**Example** − [name#Raja, age#30]

### Relation

A relation is a bag of tuples. The relations in Pig Latin are unordered (there is no guarantee that tuples are processed in any particular order).

## Prerequisites

It is essential that you have Hadoop and Java installed on your system before you go for Apache Pig. Therefore, prior to installing Apache Pig, install Hadoop and Java

Version:

## News

### Apache Pig 0.17.0 is released!

The highlights of this release is the introduction of Pig on Spark. See details on the [release page.](https://pig.apache.org/releases.html)

## Installation:

## Apache Pig Execution Modes

You 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 Pig Latin script in a single file with **.pig** extension.
* **Embedded Mode** (UDF) − Apache Pig provides the provision of defining our own functions (**U**ser **D**efined **F**unctions) in programming languages such as Java, and using them in our script.

## Invoking the Grunt Shell

You can invoke the Grunt shell in a desired mode (local/MapReduce) using the **−x** option as shown below.

|  |  |
| --- | --- |
| **Local mode** | **MapReduce mode** |
| **Command −**  $ ./pig –x local | **Command −**  $ ./pig -x mapreduce |