Inceptez
Web: Inceptez.com Mail: info@inceptez.com Call: 7871299810, 7871299817

Apache PIG

Introduction

Pig provides an engine for executing data flows in parallel on Hadoop. It includes a high level data flow language, *Pig Latin*. Pig Latin includes operators for many of the traditional data operations (join, sort, filter, etc.), as well as the ability for users to develop their own functions for reading, processing, and writing data.

- ❑ High level data flow language for exploring very large datasets.
- ❑ Provides an engine for executing data flows in parallel on Hadoop.
- ❑ Compiler that produces sequences of Map Reduce programs
- ❑ Structure is amenable to substantial parallelization
- ❑ Operates on files in HDFS
- ❑ Metadata not required, but used when available

Key Properties of Pig:

Ease of programming: Trivial to achieve parallel execution of simple and parallel data analysis tasks

Optimization: Allows the user to focus on semantics rather than efficiency

Extensibility: Users can create their own functions to do special-purpose processing

Why PIG

- Makes writing hadoop jobs a lot simpler
  - 5% of the code, 5% of time
- You don't have to be a programmer to write Pig scripts
- Provides major functionality required for DW and Analytics
  - Load, Filter, Join, Group By, Order, Transform, UDFs, Store.
- User can write custom UDFs (User Defined Function)

History

Pig started out as a research project in **Yahoo** Research, where Yahoo! scientists designed it and produced an initial implementation. Hadoop "is too low-level and rigid, and leads to a great deal of custom user code that is hard to maintain and reuse." At the same time they observed that many MapReduce users were not comfortable with declarative languages such as SQL. Thus they set out to produce "a new language called Pig Latin that we have designed to fit in a sweet spot between the

declarative style of SQL, and the low-level, procedural style of MapReduce." Yahoo! Hadoop users started to adopt Pig. So, a team of development engineers was assembled to take the research prototype and build it into a production-quality product. In year 2007, Pig was open sourced via the Apache Incubator. The first Pig release came a year later 2008 and graduated from the Incubator and became a subproject of Apache Hadoop. Early in 2009 other companies started to use Pig for their data processing. Amazon also added Pig as part of its Elastic MapReduce service. By the end of 2009 about half of Hadoop jobs at Yahoo! were Pig jobs. In 2010, Pig adoption continued to grow, and Pig graduated from a Hadoop subproject, becoming its own top-level Apache project.

PIG Philosophy

Pigs eat anything
Pig can operate on data whether it has metadata or not. It can operate on data that is relational, nested, or unstructured. And it can easily be extended to operate on data beyond files, including key/value stores, databases, etc.
Pigs live anywhere
Pig is intended to be a language for parallel data processing. It is not tied to one particular parallel framework. It has been implemented first on Hadoop, but we do not intend that to be only on Hadoop.
Pigs are domestic animals
Pig is designed to be easily controlled and modified by its users. Pig allows integration of user code wherever possible, so it currently supports user defined field transformation functions, user defined aggregates, and user defined conditionals. These functions can be written in Java or in scripting languages that can compile down to Java (e.g., Jython).
Pigs fly
Pig processes data quickly. We want to consistently improve performance, and not implement features in ways that weigh Pig down so it can't fly.

Application of PIG

What Is Pig Useful For?
Pig Latin use cases tend to fall into three separate categories: traditional extract transform load (ETL) data pipelines, research on raw data, and iterative processing.

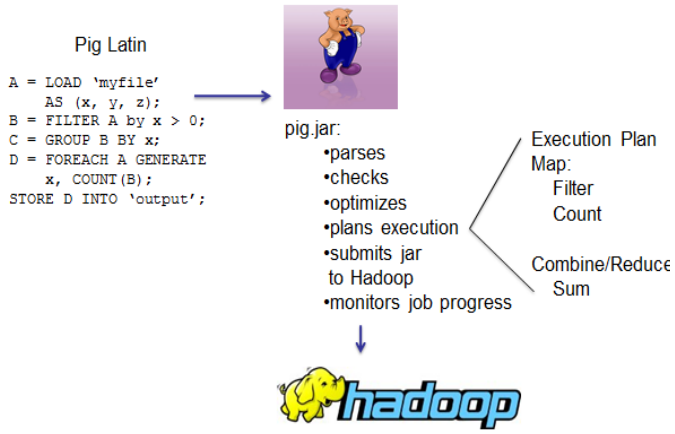
The largest use case is data pipelines. A common example is web companies bringing in logs from their web servers, cleansing the data, and pre computing common aggregates before loading it into their data warehouse. In this case, the data is loaded onto the grid, and then Pig is used to clean out records from bots and records with corrupt data.

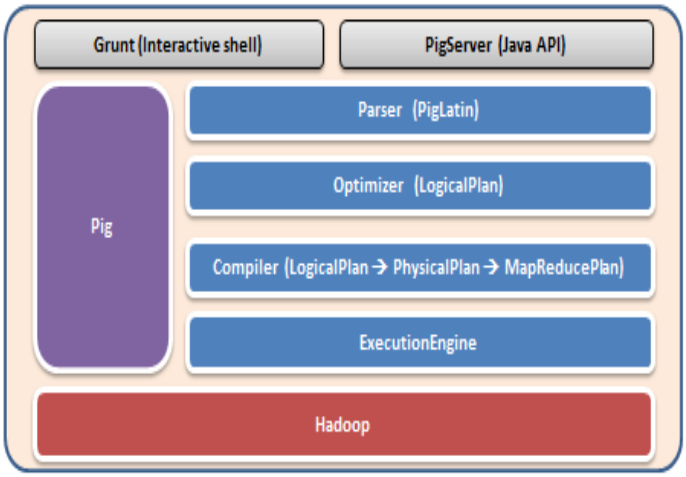
It is also used to join web event data against user databases so that user cookies can be connected with known user information. Another example of data pipelines is using Pig offline to build behavior prediction models. Pig is used to scan through all the user interactions with a website and split the users into various segments. Then, for each segment, a mathematical model is produced that predicts how members of that segment will respond to types of advertisements or news articles. In this way the website can show ads that are more likely to get clicked on, or offer news stories that are more likely to engage users and keep them coming back to the site.

Traditionally, ad-hoc queries are done in languages such as SQL that make it easy to quickly form a question for the data to answer. However, for research on raw data, some users prefer Pig Latin.

Because Pig can operate in situations where the schema is unknown, incomplete, or inconsistent, and because it can easily manage nested data, researchers who want to work on data before it has been cleaned and loaded into the warehouse often prefer Pig. Researchers who work with large data sets often use scripting languages such as Perl or Python to do their processing. Users with these backgrounds often prefer the dataflow paradigm of Pig over the declarative query paradigm of SQL.

PIG Architecture





Interfaces

Pig Latin	-	Submit a script directly
Grunt	-	Pig Shell
Java Interface	-	Java Class similar to JDBC interface

Execution Modes

Local Mode

PIG engine runs in a single machine will all installation and config files are run using your local host and file system. Pig can run in a non hadoop environment if runs using local mode. Is invoked by using the -x local flag
pig -x local

MapReduce Mode

Mapreduce mode is the default mode, Need access to a Hadoop cluster and HDFS installation. Can also be invoked by using the -x mapreduce flag or just pig
pig -x mapreduce

PIG Simple Datatypes

Simple Type	Description	Example
int	Signed 32-bit integer	10
long	Signed 64-bit integer	Data: 10L or 10l Display: 10L
float	32-bit floating point	Data: 10.5F or 10.5f or 10.5e2f or 10.5E2F Display: 10.5F or 1050.0F
double	64-bit floating point	Data: 10.5 or 10.5e2 or 10.5E2 Display: 10.5 or 1050.0
chararray	Character array (string) in Unicode UTF-8 format	hello world
bytearray	Byte array (blob)	
boolean	boolean	true/false (case insensitive)

PIG Complex Data types

Type	Description	Example
tuple	An ordered set of fields.	(19,2)
bag	An collection of tuples.	{(19,2), (18,1)}
map	A set of key value pairs.	[open#apache]

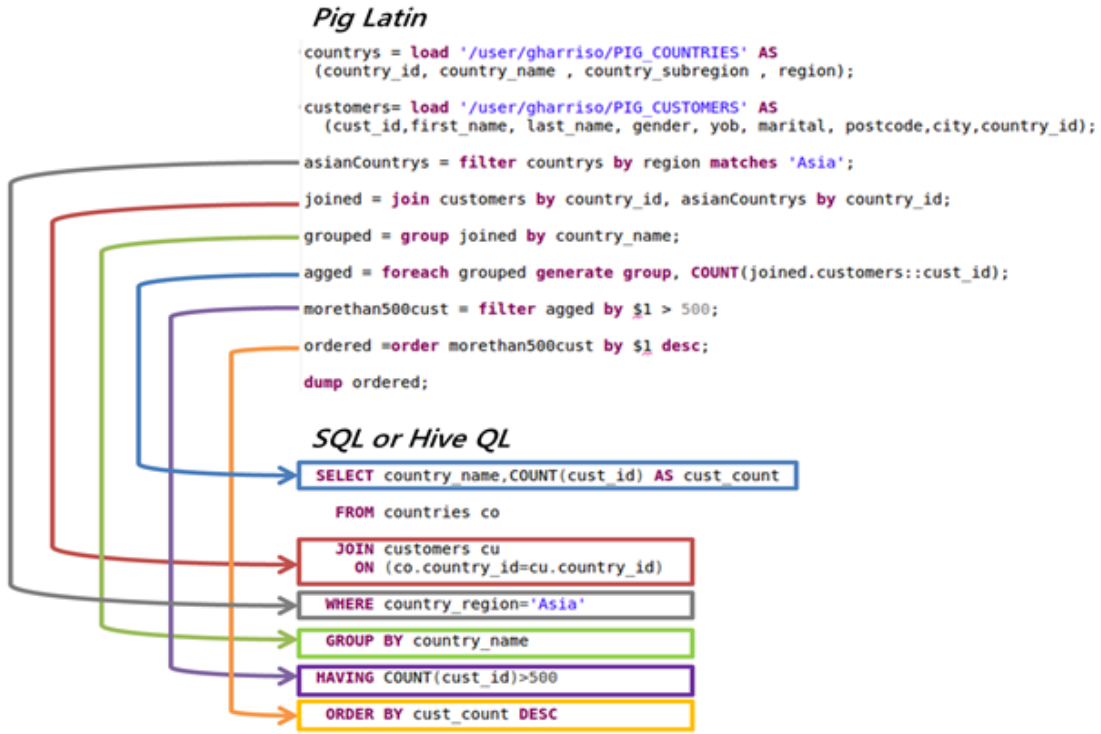
PIG Commands

Statement	Description
Load	Read data from the file system
Store	Write data to the file system
Dump	Write output to stdout
Foreach	Apply expression to each record and generate one or more records
Filter	Apply predicate to each record and remove records where false
Group / Cogroup	Collect records with the same key from one or more inputs
Join	Join two or more inputs based on a key
Order	Sort records based on a Key
Distinct	Remove duplicate records
Union	Merge two datasets
Limit	Limit the number of records
Split	Split data into 2 or more sets, based on filter conditions

PIG Diagnostic Statements

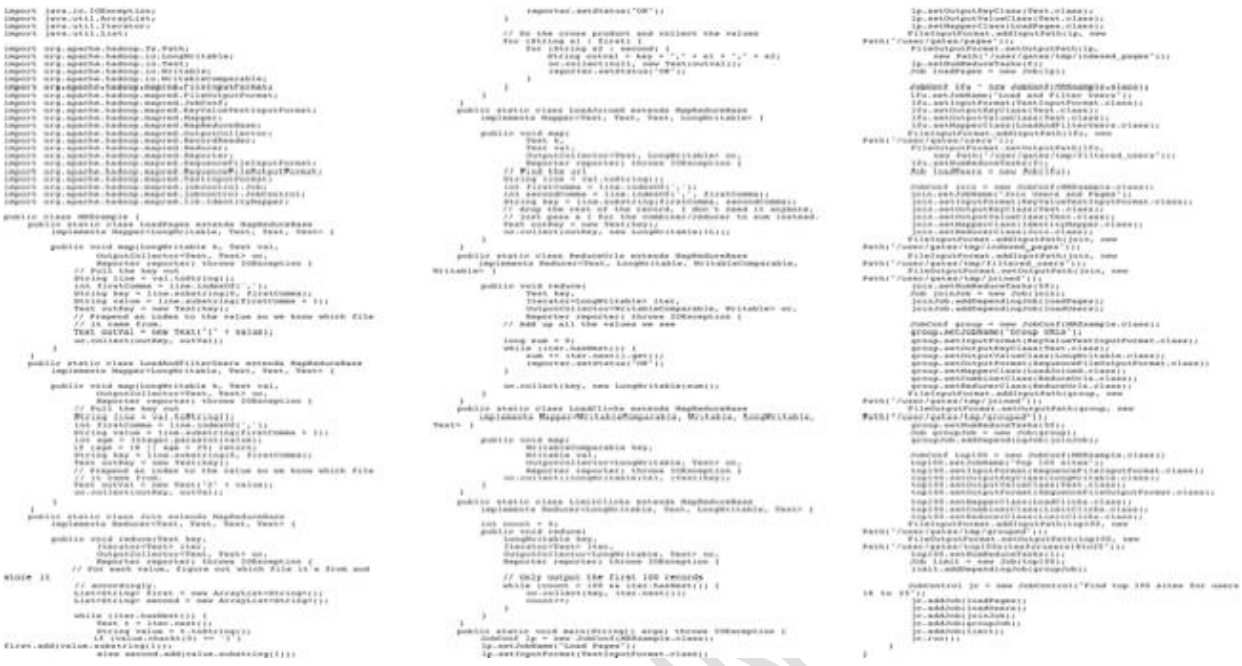
Statement	Description
Describe	Returns the schema of the relation
Dump	Dumps the results to the screen
Explain	Displays execution plans.
Illustrate	Displays a step-by-step execution of a sequence of statements

Pig Latin vs SQL



Pig	SQL
Dataflow	Declarative
Nested relational data model	Flat relational data model
Optional Schema	Schema is required
Scan-centric workloads	OLTP + OLAP workloads
Limited query optimization	Significant opportunity for query optimization

PIG equivalent java code



For comprehensive list of commands and scripting, refer the attached cheat sheet.

Mortar-Pig-Cheat-Sheet.pdf