Pig Basics
CHEAT SHEET

Apache Pig

It is a high level platform for creating programs that runs on Hadoop, the language is known as Pig Latin. Pig can execute its Hadoop jobs in MapReduce

Datatypes

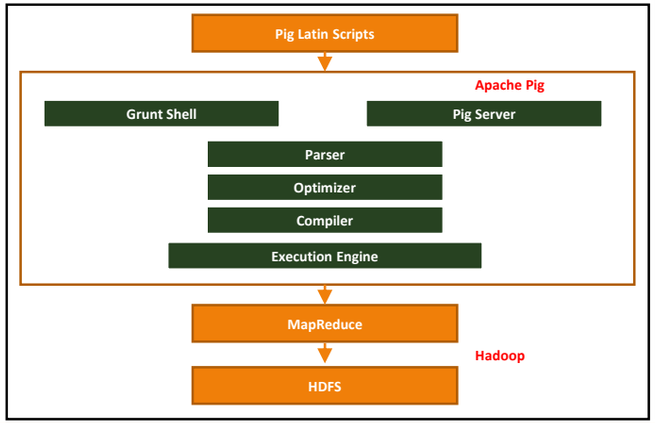
- Simple data types:
  - Int- It is a signed 32 bit integer
  - Long- It is a signed 64 bit integer
  - Float- 32 bit floating point
  - Double- 64 bit floating point
  - Chararray- Character array in UTF 8 format
  - Bytearray- byte array (blob)
  - Boolean: True or False
- Complex data types:
  - Tuple- It is an ordered set of fields
  - Bag- It is a collection of tuples
  - Map- A set of key value pairs

Components

- Parser: Parser is used to check the syntax of the scripts.
- Optimizer: It is used for the logical optimizations such as projection and push down
- Compiler: Compiler is used to compile the optimized logical plan into a series of MapReduce jobs
- Execution engine: The MapReduce jobs are executed on Hadoop, and the desired results are obtained

Simplification Item

- Grunt mode: Used for testing syntax & ad hoc data exploration
- Script mode: Used to run set of instructions from a file
- Embedded mode: Used to execute pig programs from java
- Local mode: Entire pig job runs as a single JVM process
- MapReduce Mode: Pig runs the jobs as a series of map reduce
- Tez: In this mode, pig jobs runs as a series of tez jobs



Pig Commands

Functions	Pig commands
SELECT	FOREACH alias GENERATE column_name,column_name;
SELECT*	FOREACH alias GENERATE *;
DISTINCT	DISTINCT(FOREACH aliasgenerate column_name, column_name);
WHERE	FOREACH (FILTER alias BY column_nameoperator value)GENERATE column_name, column_name;
AND/OR	FILTER alias BY (column_name operator value1AND column_name operator value2)OR column_name operator value3;
ORDER BY	ORDER alias BY column_name ASC|DESC,column_name ASC|DESC;
TOP/LIMIT	FOREACH (GROUP alias BY column_name)GENERATE LIMIT alias number;TOP(number, column_index, alias);
GROUP BY	FOREACH (GROUP alias BY column_name)GENERATE function(alias.column_name);
LIKE	FILTER alias BY REGEX_EXTRACT(column_name,pattern, 1) IS NOT NULL;
IN	FILTER alias BY column_name IN(value1, value2,...);
JOIN	FOREACH (JOIN alias1 BY column_name,alias2 BY column_name)GENERATE column_name(s);
LEFT/RIGHT/FULL OUTERJOIN	FOREACH(JOINalias1 BY column_name LEFT|RIGHT|FULL,alias2 BY column_name) GENERATE column_name(s);
UNION ALL	UNION alias1, alias2;
AVG	FOREACH (GROUP Alias ALL) GENERATEAVG(alias.column_name);
COUNT	FOREACH (GROUP alias ALL) GENERATE COUNT(alias);
COUNT DISTINCT	FOREACH alias{Unique _column=DISTINT Column_name);};
MAX	FOREACH(GROUP aliasALL) GENERATE MAX(alias.column_name);
MIN	FOREACH (GROUP aliasALL)GENERATE MIN(alias.column_name)
SUM	FOREACH (GROUP aliasALL)GEENRATE SUM(alias.column_name);
HAVING	FILTER alias BYAggregate_function(column_name)operatorValue;
UCASE/UPPER	FOREACH aliasGENERATEUPPER(column_name);
LCASE/LOWER	FOREACH aliasGENERATELOWER(column_name);
SUBSTRING	FOREACH aliasGENERATESUBSTRING(column_name,start,Star+length) as Some_name;
LEN	FOREACH aliasGENERATE SIZE(column_name)
ROUND	FOREACH aliasGENEARATE ROUND(column_name);

Pig Operators

Type	Command	Description
Loading and storing	LOAD DUMP STORE	It is used to load data, dump data into the console and stores in a location
Grouping data and joining	GROUP COGROUP CROSS JOIN	Groups based on the key will group the data from multiple relations Cross join is used to join two or more relations
Storing	LIMIT ORDER	It is used for limiting the results It is used for sorting by categories or fields
Data sets	UNION SPLIT	It is used for combining multiple relations It is used for splitting the relations

Relational Operators

Operators	Description
COGROUP/ GROUP	COGROUP operator groups together the tuples that has the same group key
CROSS	This operator is used to compute the cross product of two or more relations
DEFINE	This operator assigns an alias to an UDF
DISTINCT	This operator will remove the duplicate tuples
FILTER	Used to generate the transformation for each statement
FOREACH	Selects the tuples for a relation based
IMPORT	This operator imports macros defined in a separate file
JOIN	This operator performs inner join of two or more relations
LOAD	This operator loads the data from a file system
MAPREDUCE	This operator executes the native MapReduce jobs
ORDER BY	This will sort the relation based on two or more fields
SAMPLE	Divides the relation into two or more relations, and selects a random data sample based on a specified size
SPLIT	This will partition the relation based on some conditions or expressions as specified
STORE	This will store or save the result in a file system
STREAM	This operator sends the data to an external script
UNION	This operator is used to compute unions

Basic Operators

Operators	Description
Arithmetic operators	+, -, *, /, %, ?, :
Boolean operators	And, or, not
Casting operators	Casting from one datatype to another
Comparison Operators	==, !=, >, <, >=, <=, matches
Construction operators	Used to construct tuple(), bag{}, map[]
Dereference operators	Used to dereferencing as tuples(tuple.id or tuple.(id,...)), bags(bag.id or bag.(id,...))and Maps
Disambiguate operators	It used to identify field names after JOIN,COGROUP,CROSS, or FLATTEN Operators
Flatten operator	It is used to flatten un-nests tuples as well as bags
Null operator	Is null, is not null
Sign operators	+-> has no effect, -->It changes the sign of a positive/negative number

Diagnostic Operators

Operator	Description
Describe	Returns the schema of the relation
Dump	It will dump or display the result on screen
Explain	Displays execution plans
Illustrate	It displays the step by step execution for the sequence of statements