Welcome to Pig

Pig - Introduction

An engine for executing data flows in parallel on Hadoop

- Language - Pig Latin
- Pig Engine
- Can be used with or without Hadoop

Pig - Why do we need it?

- Programmers struggle a lot in writing MapReduce tasks
- Pig scripts are easier to write and maintain
- Pig provides many relational operators which are hard to write in MapReduce

Pig - Use Cases

1. Analyzing Data
2. Iterative processing
3. Batch Processing
4. ETL - Extract, Transform and Load

Pig - Philosophy

A. Pigs eat anything
   - Data: Relational, nested, or unstructured.

B. Pigs live anywhere
   - Parallel processing Language. Not only Hadoop

C. Pigs are domestic animals
   - Controllable, provides custom functions in java/python

D. Pigs fly
   - Designed for performance

Pig Latin - A Data Flow Language

Allows describing:

1. How data flows
2. Should be read
3. Processed
4. Stored to multiple outputs in parallel

Complex Workflows

1. Multiple inputs are joined

Pig - Modes

1. MapReduce mode

- Access HDFS and Hadoop cluster
- Used in production

2. Local mode

- Access local files and local machine
- Used for testing locally
- Fastens the development

Pig - MapReduce Mode

- Login to CloudxLab Linux console.
- Type pig
- Invoke commands in grunt shell to access files in HDFS
- Control hadoop from grunt shell

Pig - Local Mode

- Login to CloudxLab Linux console.
- Type pig -x local
- Invoke commands in grunt shell to access files in local file system

Pig - Data Types

1. int - Signed 32-bit integer - Example - 8
2. long - Signed 64-bit integer - Example - 5L
3. float - 32-bit floating point - Example - 5.5F
4. double - 64-bit floating point - Example - 10.5
5. chararray - character array - Example - 'CloudxLab'
6. bytearray - blob - Example - Any binary data
7. datetime - Example - 1970-01-01T00:00:00.000+00:00

Pig - Complex Data Types

http://pig.apache.org/docs/r0.15.0/basic.html#Data+Types+and+More

Pig - Relational Operators - LOAD

- *divs = LOAD '/data/NYSE_dividends';*
- *divs = LOAD '/data/NYSE_dividends' USING PigStorage(',');*
- *divs = LOAD '/data/NYSE_dividends' AS (name: chararray, stock_symbol: chararray, date: datetime, dividend: float);*

Pig - Store / Dump

STORE

- Stores the data to HDFS and other storages

DUMP

- Prints the value on the screen - print()
- Useful for debugging

Pig - Lazy Evaluation

- Each processing step results in new relation
  - except for dump and store
- No value or operation is evaluated until the value or the transformed data is required
- This reduces the repeated calculation

Pig - Relational Operators - FOREACH

Takes expressions & applies to every record

```
1. divs = LOAD '/data/NYSE_dividends' AS (name:chararray,
   stock_symbol:chararray, date:chararray, dividends:float);
2. values = FOREACH divs GENERATE stock_symbol, dividends;
3. STORE values INTO 'values_1';
4. cat values_1
```

Pig - FOREACH - Question

Will below code generate any reducer code?

gain = FOREACH divs GENERATE ticker, val;
DUMP gain;

❌

Pig - Relational Operators - GROUP

Groups the data in relations based on the keys

A

```
(John, 18, 4.0F)
(Mary, 19, 3.8F)
(Bill, 20, 3.9F)
(Joe, 18, 3.8F)
```

B = GROUP A BY age;

```
(18, {(John, 18, 4.0F), (Joe, 18, 3.8F)})
(19, {(Mary, 19, 3.8F)})
(20, {(Bill, 20, 3.9F)})
```

```
DESCRIBE A;
A: {
        name: chararray,
        age: int,
        gpa: float
    }
```

```
DESCRIBE B;
B: {
        group: int,
        A: {
                name: chararray,
                age: int,
                gpa: float
            }
    }
```

Pig - Relational Operators - FILTER

- *divs = LOAD '/data/NYSE_dividends' AS (exchange: chararray, symbol: chararray, date: datetime, dividends: float);*
- *startswithcm = FILTER divs BY symbol matches 'CM.*';*

Pig - More Operators

http://pig.apache.org/docs/r0.15.0/basic.html

Pig - Hands-on - Average Dividend

hdfs: /data/NYSE_dividends

Exchange	**Symbol**	**Date**	**Dividends**
NYSE	CPO	2009-12-30	0.11
NYSE	CPO	2009-09-28	0.12
NYSE	CPO	2009-06-26	0.13
NYSE	CCS	2009-10-28	0.41
NYSE	CCS	2009-04-29	0.43
NYSE	CIF	2009-12-09	.029
NYSE	CIF	2009-12-09	.028

→

Average

CPO	0.14
CCS	0.41
CIF	.0214

Exchange - chararray
Stock symbol - chararray
Date - chararray
Dividends - float

Pig - Hands-on - Average Dividend

Exchange - chararray
Stock symbol - chararray
Date - chararray
Dividends - float

```
1. divs = LOAD '/data/NYSE_dividends' AS (exchange, stock_symbol, date,
   dividends);
2. grped = GROUP divs BY stock_symbol;
3. DUMP grped;
4. avged = FOREACH grped GENERATE  group,  AVG(divs.dividends);
5. STORE avged INTO 'avged';
6. cat avged/part-r-00000
```

Pig - Summary

- Basics
- Execution modes
- Data types
- Relational operators - FOREACH, GROUP, FILTER
- Hands-on demo