Pig Tutorial

Table of contents

1. Overview

The Pig tutorial shows you how to run two Pig scripts in local mode and mapreduce mode.

- **Local Mode**: To run the scripts in local mode, no Hadoop or HDFS installation is required. All files are installed and run from your local host and file system.
- **Mapreduce Mode**: To run the scripts in mapreduce mode, you need access to a Hadoop cluster and HDFS installation.

2. Check Your Setup

Check your run-time environment and do the following preliminary tasks:

1. Make sure the JAVA_HOME environment variable is set the root of your Java installation.
2. Make sure your PATH includes bin/pig (this enables you to run the tutorials using the "pig" command).

```
$ export PATH=/<my-path-to-pig>/pig-0.8.0/bin:$PATH
```

3. Set the PIG_HOME environment variable:

```
$ export PIG_HOME=/<my-path-to-pig>/pig-0.8.0
```

4. Create the pigtutorial.tar.gz file:
   - Move to the Pig tutorial directory (.../pig-0.8.0/tutorial).
   - Edit the build.xml file in the tutorial directory.

```
Change this:   <property name="pigjar" value="../pig.jar" />
To this:       <property name="pigjar" value="../pig-0.8.0-core.jar"
/>
```

   - Run the "ant" command from the tutorial directory. This will create the pigtutorial.tar.gz file.
5. Copy the pigtutorial.tar.gz file from the Pig tutorial directory to your local directory.
6. Unzip the pigtutorial.tar.gz file.

```
$ tar -xzf pigtutorial.tar.gz
```

7. A new directory named pigtmp is created. This directory contains the Pig tutorial files. These files work with Hadoop 0.20.2 and include everything you need to run the Pig scripts.

3. Running the Pig Scripts in Local Mode

To run the Pig scripts in local mode, do the following:

1. Move to the pigtmp directory.

2. Execute the following command using script1-local.pig (or script2-local.pig).

```
$ pig -x local script1-local.pig
```

   The output may contain a few Hadoop warnings which can be ignored:

```
2010-04-08 12:55:33,642 [main] INFO
org.apache.hadoop.metrics.jvm.JvmMetrics
- Cannot initialize JVM Metrics with processName=JobTracker, sessionId=
- already initialized
```

3. A directory named script1-local-results.txt (or script2-local-results.txt) is created. This directory contains the results file, part-r-0000.

4. Running the Pig Scripts in Mapreduce Mode

To run the Pig scripts in mapreduce mode, do the following:

1. Move to the pigtmp directory.
2. Copy the excite.log.bz2 file from the pigtmp directory to the HDFS directory.

```
$ hadoop fs –copyFromLocal excite.log.bz2 .
```

3. Set the PIG_CLASSPATH environment variable to the location of the cluster configuration directory (the directory that contains the core-site.xml, hdfs-site.xml and mapred-site.xml files):

```
export PIG_CLASSPATH=/mycluster/conf
```

4. Set the HADOOP_CONF_DIR environment variable to the location of the cluster configuration directory:

```
export HADOOP_CONF_DIR=/mycluster/conf
```

5. Execute the following command (using either script1-hadoop.pig or script2-hadoop.pig):

```
$ pig script1-hadoop.pig
```

6. Review the result files, located in the script1-hadoop-results or script2-hadoop-results HDFS directory:

```
$ hadoop fs -ls script1-hadoop-results
$ hadoop fs -cat 'script1-hadoop-results/*' | less
```

5. Pig Tutorial Files

The contents of the Pig tutorial file (pigtutorial.tar.gz) are described here.

File	Description
pig.jar	Pig JAR file

tutorial.jar	User-defined functions (UDFs) and Java classes
script1-local.pig	Pig Script 1, Query Phrase Popularity (local mode)
script1-hadoop.pig	Pig Script 1, Query Phrase Popularity (Hadoop cluster)
script2-local.pig	Pig Script 2, Temporal Query Phrase Popularity (local mode)
script2-hadoop.pig	Pig Script 2, Temporal Query Phrase Popularity (Hadoop cluster)
excite-small.log	Log file, Excite search engine (local mode)
excite.log.bz2	Log file, Excite search engine (Hadoop cluster)

The user-defined functions (UDFs) are described here.

UDF	**Description**
ExtractHour	Extracts the hour from the record.
NGramGenerator	Composes n-grams from the set of words.
NonURLDetector	Removes the record if the query field is empty or a URL.
ScoreGenerator	Calculates a "popularity" score for the n-gram.
ToLower	Changes the query field to lowercase.
TutorialUtil	Divides the query string into a set of words.

6. Pig Script 1: Query Phrase Popularity

The Query Phrase Popularity script (script1-local.pig or script1-hadoop.pig) processes a search query log file from the Excite search engine and finds search phrases that occur with particular high frequency during certain times of the day.

The script is shown here:

- Register the tutorial JAR file so that the included UDFs can be called in the script.

```
REGISTER ./tutorial.jar;
```

- Use the PigStorage function to load the excite log file (excite.log or excite-small.log) into the "raw" bag as an array of records with the fields **user**, **time**, and **query**.

```
raw = LOAD 'excite.log' USING PigStorage('\t') AS (user, time, query);
```

- Call the NonURLDetector UDF to remove records if the query field is empty or a URL.

```
clean1 = FILTER raw BY org.apache.pig.tutorial.NonURLDetector(query);
```

- Call the ToLower UDF to change the query field to lowercase.

```
clean2 = FOREACH clean1 GENERATE user, time,
org.apache.pig.tutorial.ToLower(query) as query;
```

- Because the log file only contains queries for a single day, we are only interested in the hour. The excite query log timestamp format is YYMMDDHHMMSS. Call the ExtractHour UDF to extract the hour (HH) from the time field.

```
houred = FOREACH clean2 GENERATE user,
org.apache.pig.tutorial.ExtractHour(time) as hour, query;
```

- Call the NGramGenerator UDF to compose the n-grams of the query.

```
ngramed1 = FOREACH houred GENERATE user, hour,
flatten(org.apache.pig.tutorial.NGramGenerator(query)) as ngram;
```

- Use the DISTINCT operator to get the unique n-grams for all records.

```
ngramed2 = DISTINCT ngramed1;
```

- Use the GROUP operator to group records by n-gram and hour.

```
hour_frequency1 = GROUP ngramed2 BY (ngram, hour);
```

- Use the COUNTfunction to get the count (occurrences) of each n-gram.

```
hour_frequency2 = FOREACH hour_frequency1 GENERATE flatten($0), COUNT($1)
as count;
```

- Use the GROUP operator to group records by n-gram only. Each group now corresponds to a distinct n-gram and has the count for each hour.

```

```

```
uniq_frequency1 = GROUP hour_frequency2 BY group::ngram;
```

- For each group, identify the hour in which this n-gram is used with a particularly high frequency. Call the ScoreGenerator UDF to calculate a "popularity" score for the n-gram.

```
uniq_frequency2 = FOREACH uniq_frequency1 GENERATE flatten($0),
flatten(org.apache.pig.tutorial.ScoreGenerator($1));
```

- Use the FOREACH-GENERATE operator to assign names to the fields.

```
uniq_frequency3 = FOREACH uniq_frequency2 GENERATE $1 as hour, $0 as ngram,
$2 as score, $3 as count, $4 as mean;
```

- Use the FILTER operator to move all records with a score less than or equal to 2.0.

```
filtered_uniq_frequency = FILTER uniq_frequency3 BY score > 2.0;
```

- Use the ORDER operator to sort the remaining records by hour and score.

```
ordered_uniq_frequency = ORDER filtered_uniq_frequency BY hour, score;
```

- Use the PigStorage function to store the results. The output file contains a list of n-grams with the following fields: **hour**, **ngram**, **score**, **count**, **mean**.

```
STORE ordered_uniq_frequency INTO '/tmp/tutorial-results' USING
PigStorage();
```

7. Pig Script 2: Temporal Query Phrase Popularity

The Temporal Query Phrase Popularity script (script2-local.pig or script2-hadoop.pig) processes a search query log file from the Excite search engine and compares the occurrence of frequency of search phrases across two time periods separated by twelve hours.

The script is shown here:

- Register the tutorial JAR file so that the user-defined functions (UDFs) can be called in the script.

```
REGISTER ./tutorial.jar;
```

- Use the PigStorage function to load the excite log file (excite.log or excite-small.log) into the “raw” bag as an array of records with the fields **user**, **time**, and **query**.

```
raw = LOAD 'excite.log' USING PigStorage('\t') AS (user, time, query);
```

- Call the NonURLDetector UDF to remove records if the query field is empty or a URL.

```
clean1 = FILTER raw BY org.apache.pig.tutorial.NonURLDetector(query);
```

- Call the ToLower UDF to change the query field to lowercase.

```
clean2 = FOREACH clean1 GENERATE user, time,
org.apache.pig.tutorial.ToLower(query) as query;
```

- Because the log file only contains queries for a single day, we are only interested in the hour. The excite query log timestamp format is YYMMDDHHMMSS. Call the ExtractHour UDF to extract the hour from the time field.

```
houred = FOREACH clean2 GENERATE user,
org.apache.pig.tutorial.ExtractHour(time) as hour, query;
```

- Call the NGramGenerator UDF to compose the n-grams of the query.

```
ngramed1 = FOREACH houred GENERATE user, hour,
flatten(org.apache.pig.tutorial.NGramGenerator(query)) as ngram;
```

- Use the DISTINCT operator to get the unique n-grams for all records.

```
ngramed2 = DISTINCT ngramed1;
```

- Use the GROUP operator to group the records by n-gram and hour.

```
hour_frequency1 = GROUP ngramed2 BY (ngram, hour);
```

- Use the COUNT function to get the count (occurrences) of each n-gram.

```
hour_frequency2 = FOREACH hour_frequency1 GENERATE flatten($0), COUNT($1)
as count;
```

- Use the FOREACH-GENERATE operator to assign names to the fields.

```
hour_frequency3 = FOREACH hour_frequency2 GENERATE $0 as ngram, $1 as hour,
$2 as count;
```

- Use the FILTERoperator to get the n-grams for hour ‘00’

```
hour00 = FILTER hour_frequency2 BY hour eq '00';
```

- Uses the FILTER operators to get the n-grams for hour ‘12’

```
hour12 = FILTER hour_frequency3 BY hour eq '12';
```

- Use the JOIN operator to get the n-grams that appear in both hours.

```
same = JOIN hour00 BY $0, hour12 BY $0;
```

- Use the FOREACH-GENERATE operator to record their frequency.

```
same1 = FOREACH same GENERATE hour_frequency2::hour00::group::ngram as
ngram, $2 as count00, $5 as count12;
```

- Use the PigStorage function to store the results. The output file contains a list of n-grams with the following fields: **hour**, **count00**, **count12**.

```
STORE same1 INTO '/tmp/tutorial-join-results' USING PigStorage();
```