Apache Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

Pig's language layer currently consists of a textual language called Pig Latin, which has the following key properties:

- Ease of programming. It is trivial to achieve parallel execution of simple, "embarrassingly parallel" data analysis tasks. Complex tasks comprised of multiple interrelated data transformations are explicitly encoded as data flow sequences, making them easy to write, understand, and maintain.
- Optimization opportunities. The way in which tasks are encoded permits the system to optimize their execution automatically, allowing the user to focus on semantics rather than efficiency.
- Extensibility. Users can create their own functions to do special-purpose processing.

This pig tutorial use pig to do the same thing as spark tutorial. The default mode is mapreduce, you can also use other modes like local/tez_local/tez. For mapreduce mode, you need to have hadoop installed and export `HADOOP_CONF_DIR` in `zeppelin-env.sh`

The tutorial consists of 3 steps.

- Use shell interpreter to download bank.csv and upload it to hdfs
- use `%pig` to process the data
- use `%pig.query` to query the data

```
%sh

wget https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv
hadoop fs -put bank.csv .
```

```
--2017-01-22 12:51:48--  https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv
Resolving s3.amazonaws.com... 52.216.80.227
Connecting to s3.amazonaws.com|52.216.80.227|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 461474 (451K) [application/octet-stream]
Saving to: 'bank.csv.3'

     0K .......... .......... .......... .......... .......... 11%  141K 3s
    50K .......... .......... .......... .......... .......... 22%  243K 2s
   100K .......... .......... .......... .......... .......... 33%  449K 1s
   150K .......... .......... .......... .......... .......... 44%  413K 1s
   200K .......... .......... .......... .......... .......... 55%  746K 1s
   250K .......... .......... .......... .......... .......... 66%  588K 0s
   300K .......... .......... .......... .......... .......... 77%  840K 0s
   350K .......... .......... .......... .......... .......... 88%  795K 0s
   400K .......... .......... .......... .......... .......... 99% 1.35M 0s
   450K                                                       100% 13.2K=1.1s

2017-01-22 12:51:50 (409 KB/s) - 'bank.csv.3' saved [461474/461474]

17/01/22 12:51:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

```
%pig

bankText = load 'bank.csv' using PigStorage(';');
bank = foreach bankText generate $0 as age, $1 as job, $2 as marital, $3 as education, $5 as balance;
bank = filter bank by age != '"age"';
bank = foreach bank generate (int)age, REPLACE(job,'"','') as job, REPLACE(marital, '"', '') as marital, (int)(REPLACE(balance, '"', '')) as balance;

-- The following statement is optional, it depends on whether your needs.
-- store bank into 'clean_bank.csv' using PigStorage(';');
```

```
%pig.query

bank_data = filter bank by age < 30;
b = group bank_data by age;
foreach b generate group, COUNT($1);
```

group	col_1
19	4
20	3
21	7
22	9
23	20
24	24
25	44
26	77
27	94
28	103
29	97

```
%pig.query

bank_data = filter bank by age < ${maxAge=40};
b = group bank_data by age;
foreach b generate group, COUNT($1) as count;
```

group	count
19	4
20	3
21	7
22	9
23	20
24	24
25	44
26	77
27	94
28	103

```
29      97
30      150
31      199
32      224
33      186
34      231
35      180

%pig.query

bank_data = filter bank by marital=='${marital=single,single|divorced|married}';
b = group bank_data by age;
foreach b generate group, COUNT($1) as count;

group   count
23      3
24      11
25      11
26      18
27      26
28      23
29      37
30      56
31      104
32      105
33      103
34      142
35      109
36      117
37      100
38      99
39      88
40      105
41      97
42      91
43      79
44      68
45      76
46      82
47      78
48      91
49      87
50      74
51      63
52      66
53      75
54      56
55      68
56      50
57      78
58      67
59      56
60      36
61      15
62      5
63      7
64      6
65      4
66      7
67      5
68      1
69      5
70      5
71      5
72      4
73      6
74      2
75      3
76      1
77      5
78      2
79      3
80      6
81      1
83      2
86      1
87      1

%pig
```