

大規模データ処理システム

Big Data Processing System

Hadoop streaming

- 基本的にはHadoopのプログラムはJavaで書く必要がある。
- Hadoopを使うためのツールの1つ
 - <http://hadoop.apache.org/docs/r2.7.0/hadoop-streaming/HadoopStreaming.html>
- Javaのプログラム(hadoop-streaming.jar)がラッパーとして動作する。
- MapperとReducerはコマンドを指定する
 - MapperやReducerとしてどのような言語でも利用することができる。
- Basically Hadoop program must be written in Java
- A tool to use hadoop platform
 - <http://hadoop.apache.org/docs/r2.7.0/hadoop-streaming/HadoopStreaming.html>
- Java program (hadoop-streaming.jar) runs as a wrapper.
- Mapper and Reducer are provided as commands.
 - Any programming language can be used as Mapper or Reducer

Wikipediaの閲覧状況を解析する (1)

Analysis of Wikipedia Page view statistics (1)

- Wikipediaは様々な情報を公開しています。 <http://dumps.wikimedia.org>
 - 中には page view の情報もあります。
<https://dumps.wikimedia.org/other/analytics/>
 - 2019年4月の統計をつかって、どのページが多く見られているかを調査。
-
- Wikipadia discloses various information; <http://dumps.wikimedia.org>
 - There is Wikipedia pageviews dataset
<https://dumps.wikimedia.org/other/analytics/>
 - Try to analyze the page view statistics in April 2019

Format of the data

- https://meta.wikimedia.org/wiki/Research:Page_view
- Here are a few sample lines from one file:
 - fr.b Special:Recherche/Achille_Baraguey_d%5C%27Hilliers 1 624
 - fr.b Special:Recherche/Acteurs_et_actrices_N 1 739
 - fr.b Special:Recherche/Agrippa_d/%27Aubign%C3%A9 1 743
 - fr.b Special:Recherche/All_Mixed_Up 1 730
 - fr.b Special:Recherche/Andr%C3%A9_Gazut.html 1 737
- In the above, the first column "fr.b" is the project name. The following abbreviations are used:
 - wikibooks: ".b"
 - wiktory: ".d"
 - wikimedia: ".m"
 - wikipedia mobile: ".mw"
 - wikinews: ".n"
 - wikiquote: ".q"
 - wikisource: ".s"
 - wikiversity: ".v"
 - mediawiki: ".w"
- The second column is the title of the page retrieved, the third column is the number of requests, and the fourth column is the size of the content returned (dummy now?).

Wikipediaの閲覧状況を解析する (2)

Analysis of Wikipedia Page view counts (2)

- まずは、統計情報のダウンロード
- First of all, download the statistical information.

– ダウンロードのためのリストを作る。 / make a list to download

```
$ curl -4 https://dumps.wikimedia.org/other/pageviews/2019/2019-04/ | grep
```

Quiz

– ファイルをダウンロードする。 / download files

```
$ mkdir pv201904
```

Quiz

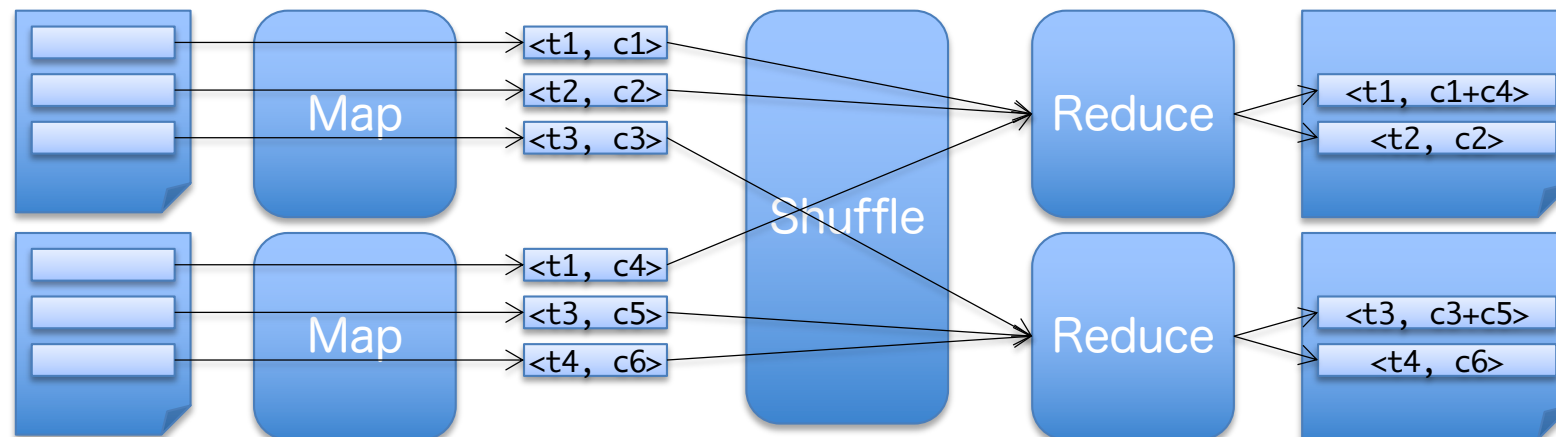
- ダウンロードしたファイルをHDFSに登録する。 / Put the files to HDFS

```
$ hdfs dfs -put pv201904
```

Wikipediaの閲覧状況を解析する (3)

Analysis of Wikipedia Page view counts (3)

- Map処理
 - 行毎に <タイトル, 閲覧数> というkey-valueの組を出力する。
- Reduce処理
 - Map処理が出力するkey-valueのkey（ここではタイトル）によってソートされてReduce処理に渡される。
 - 同じタイトルの場合は閲覧数を合計する。
- Map
 - make key-value pairs <title, count> per each line
- Reduce
 - Sorted key-value pairs which is made by Map is handed to Reduce. In this case, key is the title of pages.
 - Make total if title is same



Map, Shuffle and Reduce

Page view statistics

fr.b Acteurs_et_actrices_N 3 624

fr.b All_Mixed_Up 1 739

fr.b Acteurs_et_actrices_N 1 743

fr.b All_Mixed_Up 2 730

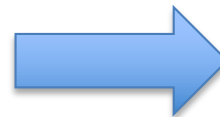
fr.b Acteurs_et_actrices_N 1 737



Map

Title	N
Acteurs_et_actrices_N	3
All_Mixed_Up	1
Acteurs_et_actrices_N	1
All_Mixed_Up	2
Acteurs_et_actrices_N	1

Shuffle



Title	N
Acteurs_et_actrices_N	5
All_Mixed_Up	3



Reduce

Title	N
Acteurs_et_actrices_N	3
	1
	1
All_Mixed_Up	1
	2

Wikipediaの閲覧状況を解析する (4)

Analysis of Wikipedia Page view counts (4)

- 行数を数えてみる。 / How many lines are there?

```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar -input pv201904 -output pv201904_out -mapper 'wc -l' -reducer cat
```

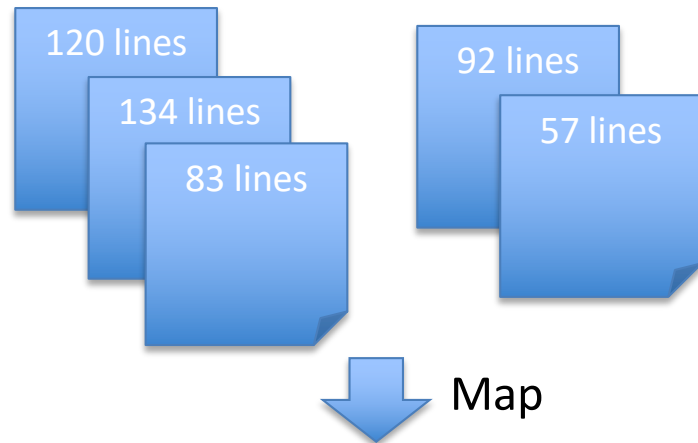
```
% hdfs dfs -cat 'pv201904_out/*' | awk '{sum = sum + $1} END{print sum}'  
4739497527
```

– もし、rubyの方が好みなら / if you prefer ruby

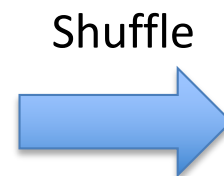
```
% hdfs dfs -cat 'pv201904_out/*' | ruby -ne 'BEGIN{$sum = 0}; $sum +=  
$_.to_i; END{puts $sum}'
```


Map, Shuffle and Reduce

```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar -input pv201904 -output pv201904_out -mapper 'wc -l' -reducer cat
```



Lines
120
134
83
92
57



Lines
57
83
92

Lines
120
134



Lines
57
83
92

Lines
120
134

Wikipediaの閲覧状況を解析する (4)

Analysis of Wikipedia Page view counts (4)

- 行数を数えてみる。 / How many lines are there?

```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar -input pv201904 -output pv201904_out -mapper 'wc -l' -reducer cat
```

```
% hdfs dfs -cat 'pv201904_out/*' | awk '{sum = sum + $1} END{print sum}'  
4739497527
```

- もし、rubyの方が好みなら / if you prefer ruby

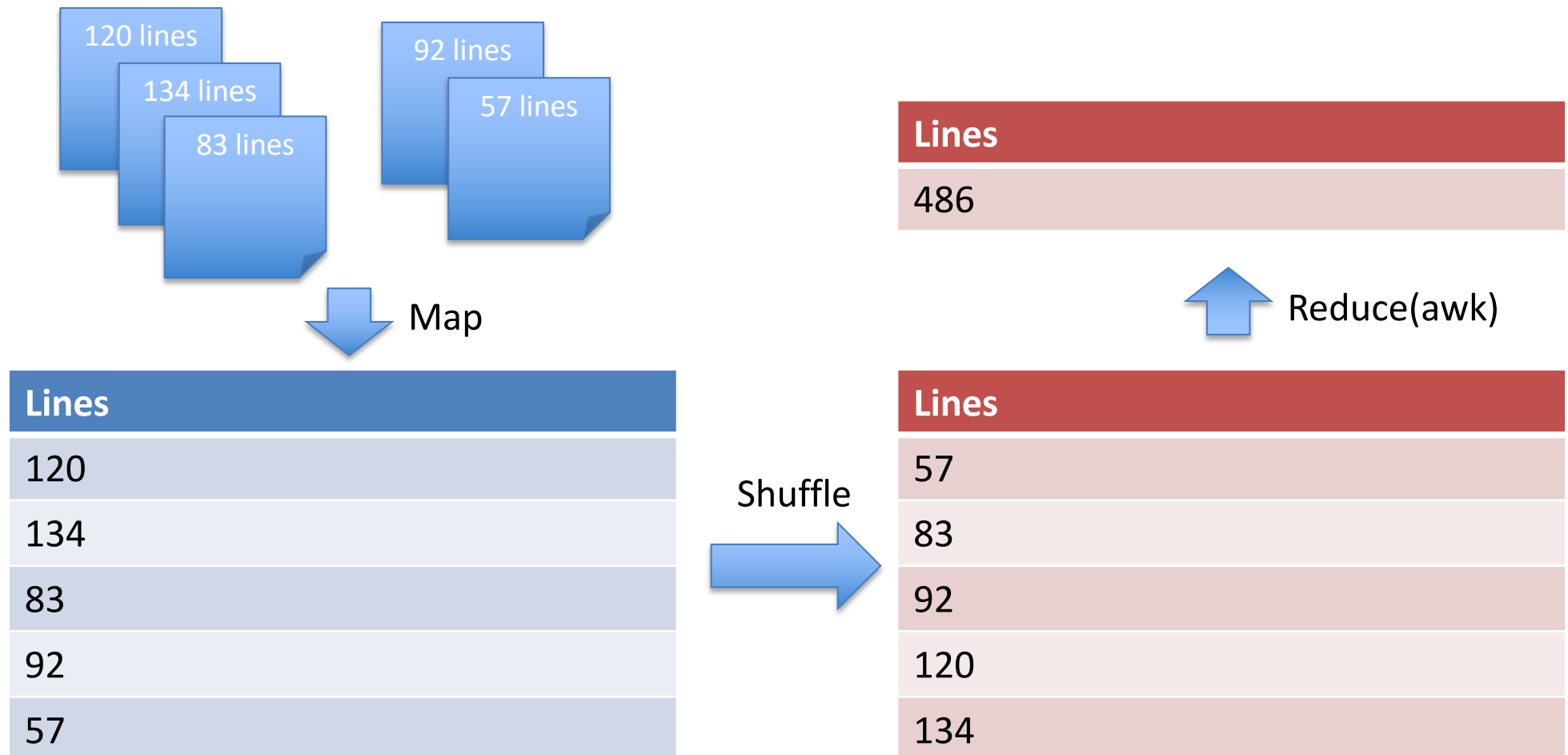
```
% hdfs dfs -cat 'pv201904_out/*' | ruby -ne 'BEGIN{$sum = 0}; $sum +=  
$_.to_i; END{puts $sum}'
```

- 全てをHadoopで行う。 / Everything is done by Hadoop.

```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar -input pv201904 -output pv201904_out -mapper 'wc -l' -reducer  
'awk "{sum = sum + $1} END{print sum}"' -numReduceTasks 1
```

Map, Shuffle and Reduce

```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar -input pv201904  
-output pv201904_out -mapper 'wc -l' -reducer 'awk "{sum = sum + $1} END{print sum}"'  
-numReduceTasks 1
```



Hadoop streaming

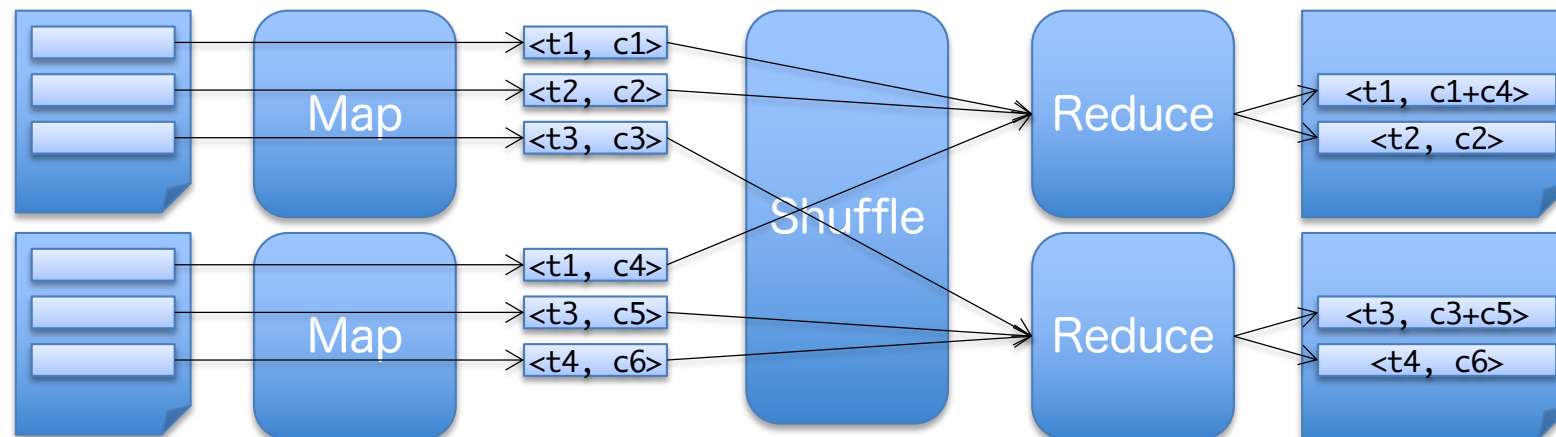
```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar [opt]
```

Parameter	Optional/ Required	Description
<code>-input directoryname or filename</code>	Required	Input location for mapper
<code>-output directoryname</code>	Required	Output location for reducer
<code>-mapper executable or JavaClassName</code>	Required	Mapper executable
<code>-reducer executable or JavaClassName</code>	Required	Reducer executable
<code>-files filenames</code>	Optional	Make the mapper, reducer, or combiner executable available locally on the compute nodes
<code>-inputformat JavaClassName</code>	Optional	Class you supply should return key/value pairs of Text class. If not specified, TextInputFormat is used as the default
<code>-outputformat JavaClassName</code>	Optional	Class you supply should take key/value pairs of Text class. If not specified, TextOutputFormat is used as the default
<code>-partitioner JavaClassName</code>	Optional	Class that determines which reduce a key is sent to
<code>-combiner streamingCommand or JavaClassName</code>	Optional	Combiner executable for map output
<code>-cmdenv name=value</code>	Optional	Pass environment variable to streaming commands
<code>-inputreader</code>	Optional	For backwards-compatibility: specifies a record reader class (instead of an input format class)
<code>-verbose</code>	Optional	Verbose output
<code>-lazyOutput</code>	Optional	Create output lazily. For example, if the output format is based on FileOutputFormat, the output file is created only on the first call to Context.write
<code>-numReduceTasks</code>	Optional	Specify the number of reducers
<code>-mapdebug</code>	Optional	Script to call when map task fails
<code>-reduceddebug</code>	Optional	Script to call when reduce task fails

Wikipediaの閲覧状況を解析する (3)

Analysis of Wikipedia Page view counts (3)

- Map処理
 - 行毎に <タイトル, 閲覧数> というkey-valueの組を出力する。
- Reduce処理
 - Map処理が出力するkey-valueのkey（ここではタイトル）によってソートされてReduce処理に渡される。
 - 同じタイトルの場合は閲覧数を合計する。
- Map
 - make key-value pairs <title, count> per each line
- Reduce
 - Sorted key-value pairs which is made by Map is handed to Reduce. In this case, key is the title of pages.
 - Make total if title is same



Wikipediaの閲覧状況を解析する (5) python

Analysis of Wikipedia Page view counts (5) python

- Mapper

```
#!/usr/bin/env python3.6
import sys

args = sys.argv

for line in sys.stdin:
    line = line.strip()
    if len(line.split()) != 4:
        continue
    projectName, title, count, size = line.split()
    if projectName != args[1]:
        continue
    print(title, "%t", count)
```

mapper.py

言語チェック

Wikipediaの閲覧状況を解析する (5) python

Analysis of Wikipedia Page view counts (5) python

- Reducer

```
#!/usr/bin/env python3.6
import sys
```

reducer.py

```
args = sys.argv
```

```
t, c = sys.stdin.readline().strip().split("¥t")
```

```
title, count = t, int(c)
```

```
for line in sys.stdin:
```

```
    t, c = line.strip().split("¥t")
```

```
    if title != t:
```

```
        if count > int(args[1]):
```

```
            print(title, "¥t", count);
```

```
            title, count = t, int(c)
```

```
    else:
```

```
        count += int(c)
```

```
if count > int(args[1]):
```

```
    print(title, "¥t", count);
```

get only popular pages

令和 1 58
令和 1 53
:
平成 17 18
平成 17 11
:

Wikipediaの閲覧状況を解析する (5) ruby

Analysis of Wikipedia Page view counts (5) ruby

- Mapper

```
#!/usr/bin/env ruby
STDIN.each do | l |
  projectName, title, count, size = l.chomp.split(" ")
  print "#{title}¥t#{count}¥n" if projectName == ARGV[0]
end
```

mapper.rb

- Reducer

```
#!/usr/bin/env ruby
t, c = STDIN.readline.chomp.split("¥t")
title, count = t, c.to_i
STDIN.each do | l |
  t, c = l.chomp.split("¥t")
  if title != t
    print "#{title}¥t#{count}¥n" if count > ARGV[0].to_i
    title, count = t, c.to_i
  else
    count += c.to_i
  end
end
print "#{title}¥t#{count}¥n" if count > ARGV[0].to_i
```

reducer.rb

Wikipediaの閲覧状況を解析する (6)

Analysis of Wikipedia Page view counts (6)

- 少数のデータで、hadoopを使わずに処理の確認をする。
Check the programs using small number of data without hadoop
 - 日本のWikipediaのみに着目し、1,000アクセス以上のものを表示する。
Print page title which is viewed more than 1000 times and written in Japanese

```
% gzip -dc pv201904/pageviews-2019040[1-2]-000000.gz | ./mapper.py  
ja | sort | ./reducer.py 1000 | nkf -w --url-input
```

- 全データでhadoopを使って処理する。 / process all data using hadoop

```
% hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-  
streaming.jar -files reducer.py,mapper.py -input pv201904 -output  
pv201904_out -mapper 'mapper.py ja' -reducer 'reducer.py 100000'
```

- よくアクセスされたもののトップ10を表示する。
Print top 10 accessed files pages

```
% hdfs dfs -cat 'pv201904_out/part-*' | sort -r -n -k2 | nkf -w --  
url-input | less
```

... sends files from master node from child node

Hiveを使う

Using Hive

- HiveはHadoopファミリのデータウェアハウス。
Hive is a data warehouse in Hadoop family.
- クエリを行うのにHiveQL (SQLのような言語)が用いられる。
HiveQL (SQL like language) is used to make a query.

- データをHiveに登録する。 / Register data to Hive

% hive

```
hive> CREATE TABLE pv201904(proj STRING, title STRING, count INT, size INT) row  
format delimited fields terminated by ' ' lines terminated by '¥n';  
hive> LOAD DATA INPATH '/user/kei/pv201904/*' OVERWRITE INTO TABLE pv201904;
```

- 行数は? / How many lines?

SELECT count(*) FROM pv201904

Quiz (SQL sentence)

- Hiveでトップ10を表示。 / Get top 10 page view counts

```
SELECT title, SUM(count), AS num FROM pv201904 WHERE proj="ja" GROUP BY title ORDER BY num DESC LIMIT  
10;
```

Quiz (SQL sentence)

HiveQL

- Hiveでトップ10を表示。 / Get top 10 page view counts

```
hive> SELECT title, SUM(count) AS num FROM pv201904 WHERE proj = "ja" GROUP BY title  
ORDER BY num DESC LIMIT 10;
```

- SELECT: データの抽出 / retrieving data
- SUM: 合計を算出する関数 / a function to make a summation
- AS: 別名 / alias name
- FROM: テーブル指定 / specify a table
- WHERE: 条件 / condition
- GROUP BY: グループニング / Grouping (used with COUNT, SUM, AVE…)
- ORDER BY … DESC: 並び替え(降順) / Sort (Descending order)
- LIMIT: 個数制限 / limit lines

Impalaを使う

Using Impala

- Impalaは別のデータウェアハウス / Impala is another data warehouse
- ネイティブで実装されている / It is implemented in native language.
- 爆速 / Very high speed

- Hiveで作成したテーブルをImpalaで使えるようにする。

Prepare the Impala's metadata used in Hive

```
% impala-shell -i dn01
```

```
> INVALIDATE METADATA;
```

- Impalaでトップ10を表示。 / Get top 10 PVC using Impala

```
> SELECT title, SUM(count) AS num FROM pv201904 WHERE proj = "ja"  
GROUP BY title ORDER BY num DESC LIMIT 10;
```