Introduction to Pig programming

Yahoo Search Engineering

陳奕瑋 (**Yiwei Chen**)

任務！

殺很大、插很大(+瑤瑤寫真性感精選54P) @ osaki's Blog :: Xuite日誌

殺不用錢~~殺online瑤瑤性感變裝照+精選性感寫真童顏巨乳的娃娃音美少女瑤瑤 本名：郭書瑤 暱稱：瑤瑤身高：155cm 體重：42kg 三圍：33E/23/33 生日：1990/7/18 ...

blog.xuite.net/osaki99/blog/21865265 - 頁庫存檔 - 類似內容

電玩美少女瑤瑤精選影音(ヤオヤオ童顏Fカップ爆乳美少女映画videos ...

2008年8月31日 ... 18歲電玩少女瑤瑤半工半讀扛家計 (內有瑤瑤男友) http://blog.xuite.net /kaiger/daily/23136438 ... 20080913 我猜嗲嗲美少女第二段2號Kiki 3號瑤瑤 ...

blog.xuite.net/kaiger/daily/19128818 - 頁庫存檔 - 類似內容

顯示更多來自 blog.xuite.net 的結果

jays1943 分享正妹:NO.24 無名瑤瑤- 樂多日誌

瑤瑤也沒有哪裡得罪你們押你們為審麼這樣罵他說害女生生氣我看你們長的很醜吧不要自以為是喔死網友還罵人く死勒你要不要臉瑤瑤可是我的偶像你們最好是向一點 ...

blog.roodo.com/jays1943/archives/6850053.html - 頁庫存檔 - 類似內容

任務！

瑤瑤航空 - Powered by Discuz!

瑤瑤航空 - Discuz! Board ... 歡迎VIP旅客-魏如昀加入瑤瑤航空(2008-4-7) 歡迎VIP旅客-賴鋐偉加入瑤瑤航空(2008-3-13) ... 瑤瑤家族 瑤瑤在飛虎的第一家族 瑤瑤天空部落格 林佩瑤在天空的部落格 林佩瑤 無名網誌 瑤瑤的新照片都在無名啦! 無不癡齋 ...

www.yaoyaofly.com - 庫存頁面 - 更多此站結果

瑤瑤喵小屋~ - 無名小站

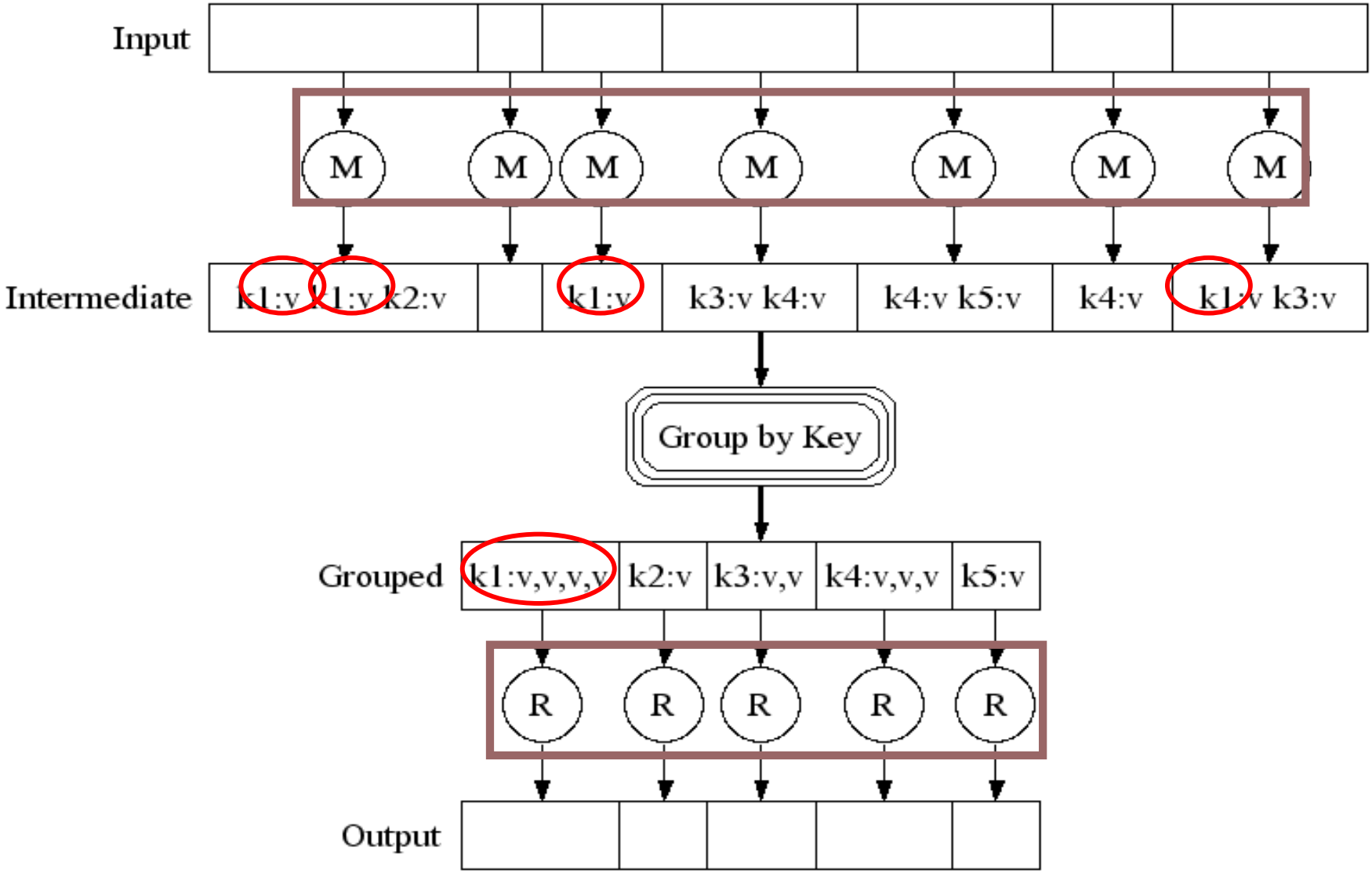
瑤瑤喵小屋~ - 無名小站 Blog Album... 最近好煩煩煩. 我覺得我的腦容量變小了... 好多事情消化不良 好多念頭讓我無法抉擇 (More.) goukigouki at 無名小站 at 02:39 PM post | Reply(27) | Trackback(0) | prosecute ...

www.wretch.cc/blog/goukigouki - 74k - 庫存頁面 - 更多此站結果

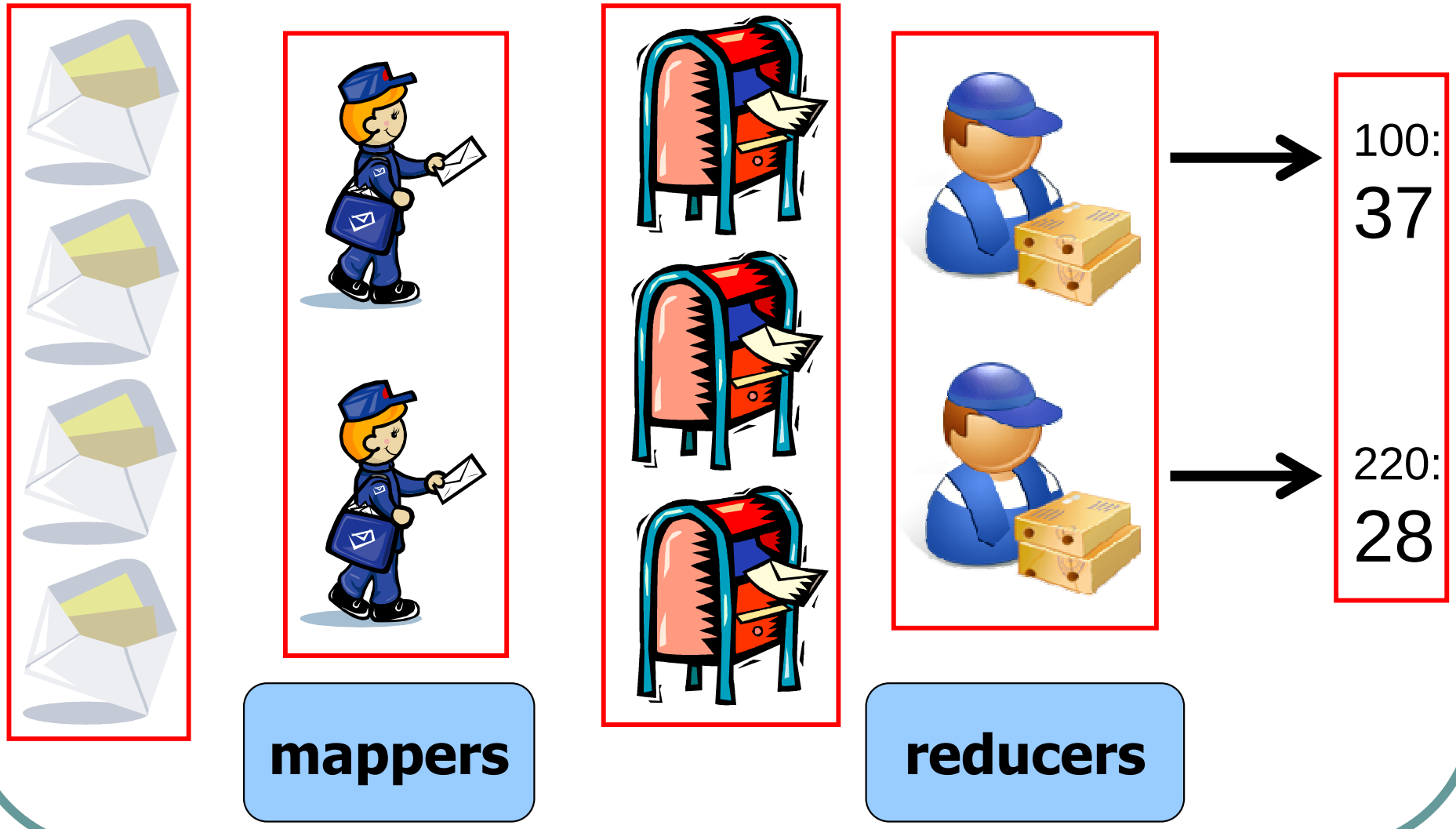
任務！

- 你怎麼知道我們放的網頁比較好？
- 你怎麼知道第一筆結果應該要多熱門？

Hadoop Programming – Map/Reduce



Map / Reduce



Map-Reduce

- 全新想法
- 須分別撰寫 mappers & reducers
- 會有超級無敵霹靂多的 mapper/reducer 要維護！

We usually do …

- 大部份時候:
  - filtering, projecting
  - grouping, aggregation, joining
- 今天有多少人搜尋「美國生」

Pig (Latin)

- Procedural dataflow language (Pig Latin) for Map-Reduce
  - 很像 SQL
    - group, join, filter, sort …
  - 人人都會 SQL

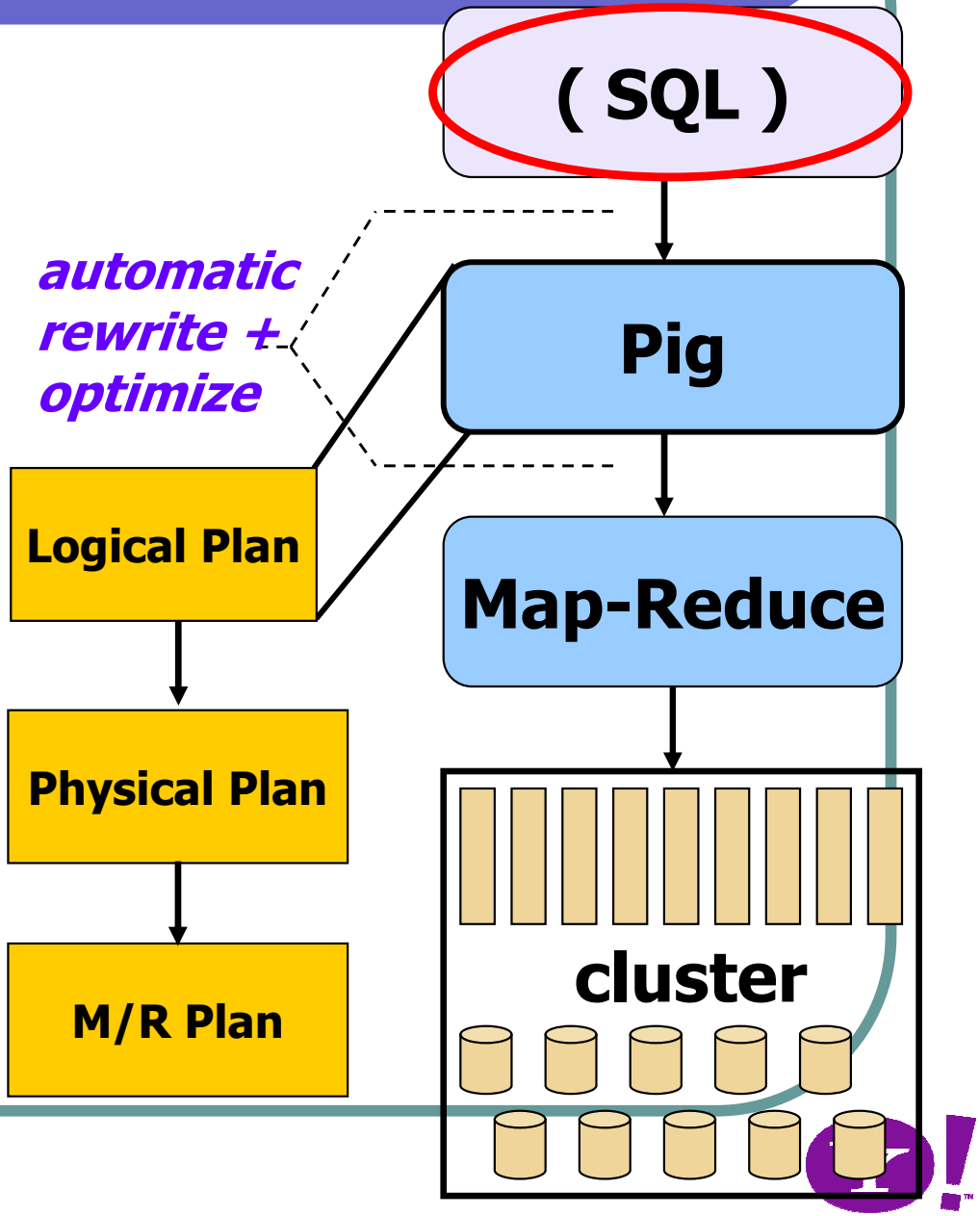
Pig Script Example

- Top sites visited by users aged 18 to 25

```
Users = LOAD ‘users.in’ AS (name, age);
Fltrd = FILTER Users by age >= 18 and age <= 25;

Pages = LOAD ‘pages.in’ AS (user, url);

Jnd   = JOIN Fltrd BY name, Pages BY user;
Grpd  = GROUP Jnd by url;
Smmd  = FOREACH Grpd GENERATE group, COUNT(Jnd) AS
        clicks;

Srtd  = ORDER Smmd BY clicks;
Top100 = LIMIT Srtd 100;

STORE Top100 INTO ‘top100sites.out’;
```

Pig script → Map/Reduce

- 不需懂底下 Map-Reduce 運作
- Pig 幫忙翻譯



Why Pig?

- 容易學
- 開發快
- 一目瞭然

Why Pig?

```
Users = LOAD ‘users’ AS (name, age);
Fltrd = FILTER Users by age >= 18 and age <= 25;
Pages = LOAD ‘pages’ AS (user, url);
Jnd   = JOIN Fltrd BY name, Pages BY user;
Grpd  = GROUP Jnd by url;
Smmd  = FOREACH Grpd GENERATE group, COUNT(Jnd) AS
        clicks;
Srtd  = ORDER Smmd BY clicks;
Top100 = LIMIT Srtd 100;

STORE Top100 INTO ‘top100sites’;
```

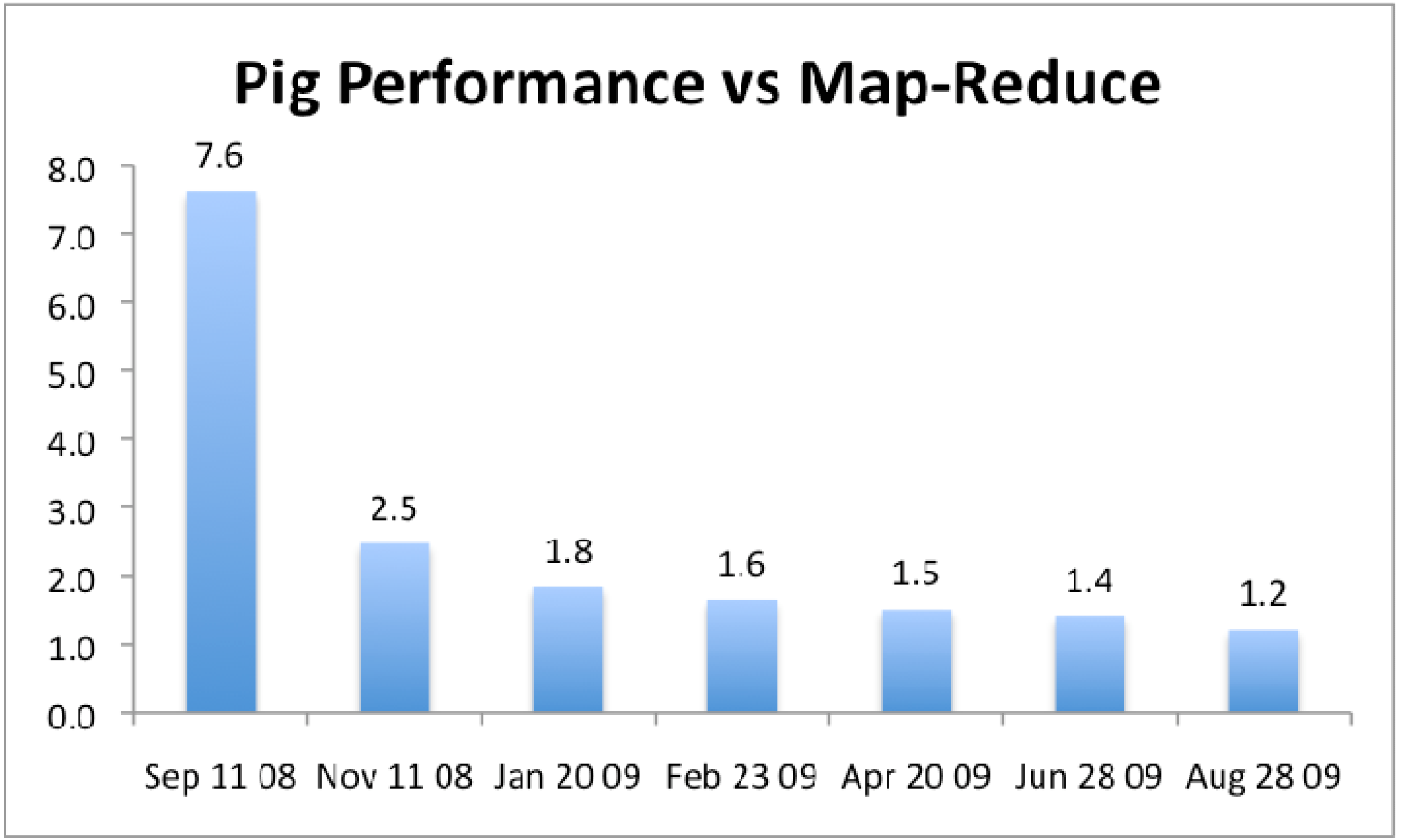
```
import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.KeyValueTextInputFormat;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.RecordReader;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.mapred.SequenceFileInputFormat;
import org.apache.hadoop.mapred.SequenceFileOutputFormat;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.jobcontrol.Job;
import org.apache.hadoop.mapred.jobcontrol.JobControl;
import org.apache.hadoop.mapred.lib.IdentityMapper;

public class MRExample {
    public static class LoadPages extends MapReduceBase
        implements Mapper<LongWritable, Text, Text, Text> {

        public void map(LongWritable k, Text val,
                OutputCollector<Text, Text> oc,
                Reporter reporter) throws IOException {
            // Pull the key out
            String line = val.toString();
            int firstComma = line.indexOf(',');
            String key = line.substring(0, firstComma);
            String value = line.substring(firstComma + 1);
            Text outKey = new Text(key);
            // Prepend an index to the value so we know which file
            // it came from.
            Text outVal = new Text("1" + value);
            oc.collect(outKey, outVal);
        }
    }
    public static class LoadAndFilterUsers extends MapReduceBase
        implements Mapper<LongWritable, Text, Text, Text> {

        public void map(LongWritable k, Text val,
                OutputCollector<Text, Text> oc,
                Reporter reporter) throws IOException {
            // Pull the key out
            String line = val.toString();
            int firstComma = line.indexOf(',');
            String value = line.substring(firstComma + 1);
            int age = Integer.parseInt(value);
            if (age < 18 || age > 25) return;
            String key = line.substring(0, firstComma);
            Text outKey = new Text(key);
            // Prepend an index to the value so we know which file
            // it came from.
            Text outVal = new Text("2" + value);
            oc.collect(outKey, outVal);
        }
    }
    public static class Join extends MapReduceBase
        implements Reducer<Text, Text, Text, Text> {

        public void reduce(Text key,
                Iterator<Text> iter,
                OutputCollector<Text, Text> oc,
                Reporter reporter) throws IOException {
            // For each value, figure out which file it's from and
store it
            // accordingly.
            List<String> first = new ArrayList<String>();
            List<String> second = new ArrayList<String>();

            while (iter.hasNext()) {
                Text t = iter.next();
                String value = t.toString();
                if (value.charAt(0) == '1')
first.add(value.substring(1));
                else second.add(value.substring(1));
                reporter.setStatus("OK");
            }

            // Do the cross product and collect the values
            for (String s1 : first) {
                for (String s2 : second) {
                    String outval = key + "," + s1 + "," + s2;
                    oc.collect(null, new Text(outval));
                    reporter.setStatus("OK");
                }
            }
        }
    }
    public static class LoadJoined extends MapReduceBase
        implements Mapper<Text, Text, Text, LongWritable> {

        public void map(
                Text key,
                Text val,
                OutputCollector<Text, LongWritable> oc,
                Reporter reporter) throws IOException {
            // Find the url
            String line = val.toString();
            int firstComma = line.indexOf(',');
            int secondComma = line.indexOf(',', firstComma);
            String key = line.substring(firstComma, secondComma);
            // drop the rest of the record, don't need it anymore,
            // just pass a 1 for the combiner/reducer to sum instead.
            Text outKey = new Text(key);
            oc.collect(outKey, new LongWritable(1L));
        }
    }
    public static class ReduceUrls extends MapReduceBase
        implements Reducer<Text, LongWritable, WritableComparable,
Writable> {

        public void reduce(
                Text key,
                Iterator<LongWritable> iter,
                OutputCollector<WritableComparable, Writable> oc,
                Reporter reporter) throws IOException {
            // Add up all the values we see

            long sum = 0;
            while (iter.hasNext()) {
                sum += iter.next().get();
                reporter.setStatus("OK");
            }

            oc.collect(key, new LongWritable(sum));
        }
    }
    public static class LoadClicks extends MapReduceBase
        implements Mapper<WritableComparable, Writable, LongWritable,
Text> {

        public void map(
                WritableComparable key,
                Writable val,
                OutputCollector<LongWritable, Text> oc,
                Reporter reporter) throws IOException {
            oc.collect((LongWritable)val, (Text)key);
        }
    }
    public static class LimitClicks extends MapReduceBase
        implements Reducer<LongWritable, Text, LongWritable, Text> {

        int count = 0;
        public void reduce(
            LongWritable key,
            Iterator<Text> iter,
            OutputCollector<LongWritable, Text> oc,
            Reporter reporter) throws IOException {

            // Only output the first 100 records
            while (count < 100 && iter.hasNext()) {
                oc.collect(key, iter.next());
                count++;
            }
        }
    }
    public static void main(String[] a
        JobConf lp = new JobConf(MRExa
        lp.setJobName("Load Pages");
        lp.setInputFormat(TextInputFor
        lp.setOutputKeyClass(Text.clas
        lp.setOutputValueClass(Text.cl
        lp.setMapperClass(LoadPages.cl
        FileInputFormat.addInputPath(l
Path("/user/gates/pages"));
        FileOutputFormat.setOutputPath
            new Path("/user/gates/tmp/
        lp.setNumReduceTasks(0);
        Job loadPages = new Job(lp);

        JobConf lfu = new JobConf(MRExa
        lfu.setJobName("Load and Filte
        lfu.setInputFormat(TextInputFo
        lfu.setOutputKeyClass(Text.cla
        lfu.setOutputValueClass(Text.c
        lfu.setMapperClass(LoadAndFilt
        FileInputFormat.addInputPath(l
Path("/user/gates/users"));
        FileOutputFormat.setOutputPath
            new Path("/user/gates/tmp/
        lfu.setNumReduceTasks(0);
        Job loadUsers = new Job(lfu);

        JobConf join = new JobConf(MRE
        join.setJobName("Join Users an
        join.setInputFormat(KeyValueTe
        join.setOutputKeyClass(Text.cl
        join.setOutputValueClass(Text.
        join.setMapperClass(IdentityMa
        join.setReducerClass(Join.clas
        FileInputFormat.addInputPath(j
Path("/user/gates/tmp/indexed_pages"))
        FileInputFormat.addInputPath(
Path("/user/gates/tmp/filtered_users"
        FileOutputFormat.setOutputPath
Path("/user/gates/tmp/joined"));
        join.setNumReduceTasks(50);
        Job joinJob = new Job(join);
        joinJob.addDependingJob(loadPa
        joinJob.addDependingJob(loadUs

        JobConf group = new JobConf(MR
        group.setJobName("Group URLs")
        group.setInputFormat(KeyValueT
        group.setOutputKeyClass(Text.c
        group.setOutputValueClass(Long
        group.setOutputFormat(Sequence
        group.setMapperClass(LoadJoine
        group.setCombinerClass(ReduceU
        group.setReducerClass(ReduceUr
        FileInputFormat.addInputPath(g
Path("/user/gates/tmp/joined"));
        FileOutputFormat.setOutputPath
Path("/user/gates/tmp/grouped"));
        group.setNumReduceTasks(50);
        Job groupJob = new Job(group);
        groupJob.addDependingJob(joinJ

        JobConf top100 = new JobConf(MR
        top100.setJobName("Top 100 sit
        top100.setInputFormat(Sequence
        top100.setOutputKeyClass(LongW
        top100.setOutputValueClass(Tex
        top100.setOutputFormat(Sequenc
        top100.setMapperClass(LoadClic
        top100.setCombinerClass(LimitC
        top100.setReducerClass(LimitCl
        FileInputFormat.addInputPath(t
Path("/user/gates/tmp/grouped"));
        FileOutputFormat.setOutputPath
Path("/user/gates/top100sitesforusers1
        top100.setNumReduceTasks(1);
        Job limit = new Job(top100);
        limit.addDependingJob(groupJob

        JobControl jc = new JobControl
18 to 25");
        jc.addJob(loadPages);
```

Why (NOT) Pig?

- 不是史上究極霹靂大無敵武器
  - Focus: aggregation, filter, join,…
- 另一種做分散運算工作的方式

Sweet spot between SQL – M/R

	SQL	Pig	Map-Reduce
Programming style	Large blocks of declarative constraints	➔	"Plug together pipes"
Built-in data manipulations	Group-by, Sort, Join, Filter, Aggregate, Top-k, etc...	➔	Group-by, Sort
Execution model	Fancy; trust the query optimizer	➔	Simple, transparent
Opportunities for automatic optimization	Many	➔	Few (logic buried in map() and reduce())
Data Schema	Must be known at table creation	➔	Not required, may be defined at runtime


Pig Performance vs Map-Reduce
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
7.6
2.5
1.8
1.6
1.5
1.4
1.2
Sep 11 08
Nov 11 08
Jan 20 09
Feb 23 09
Apr 20 09
Jun 28 09
Aug 28 09

Execution and Syntax

Pig Example

- Show users aged 18-25

```
Users = LOAD 'users.txt'
        USING PigStorage(',') AS (name, age);
Fltrd = FILTER Users
        BY age >= 18 AND age <= 25;
Names = FOREACH Fltrd GENERATE name;

STORE Names INTO 'names.out';
```

How to execute

- Local:
  - `pig -x local foo.pig`
- Hadoop (HDFS):
  - `pig foo.pig`
  - `pig -Dmapred.job.queue.name=xxx foo.pig`
    - `hadoop queue -showacls`

How to execute

- Interactive pig shell
  - `$ pig`
  - grunt> _

Load Data

```
Users = LOAD 'users.txt'
        USING PigStorage(',') AS (name, age);
```

- LOAD … AS …
- PigStorage(‘,’) to specify separator

```
John,18
Mary,20
Bob,30
```

name	age
John	18
Mary	20
Bob	30

Filter

```
Fltrd = FILTER Users
        BY age >= 18 AND age <= 25;
```

- FILTER … BY …
  - constraints can be composite

name	age
John	18
Mary	20
Bob	30

name	age
John	18
Mary	20

Generate / Project

```
Names = FOREACH Fltrd GENERATE name;
```

- FOREACH … GENERATE

name	age
John	18
Mary	20

name
John
Mary

Store Data

```
STORE Names INTO 'names.out';
```

- STORE … INTO …
  - PigStorage(‘,’) to specify separator if multiple fields

Command - JOIN

```
Users = LOAD 'users' AS (name, age);
Pages = LOAD 'pages' AS (user, url);
Jnd   = JOIN Users BY name, Pages BY user;
```

name	age
John	18
Mary	20
Bob	30

user	url
John	yaho
Mary	goog
Bob	bing

name	age	user	url
John	18	John	yaho
Mary	20	Mary	goog
Bob	30	Bob	bing

Command - GROUP

```
Grpd  = GROUP Jnd by url;
describe Grpd;
```

name	age	url
John	18	yhoo
Mary	20	goog
Dee	25	yhoo
Kim	40	bing
Bob	30	bing

yhoo	(John, 18, yhoo) (Dee, 25, yhoo)
goog	(Mary, 20, goog)
bing	(Kim, 40, bing) (Bob, 30, bing)

Other Commands

- `PARALLEL` – controls #reducer
- `ORDER` – sort by a field
- `COUNT` – eval: count #elements
- `COGROUP` – structured JOIN
- More at http://hadoop.apache.org/pig/docs/r0.5.0/piglatin_reference.html

Features

Parameter Substitution

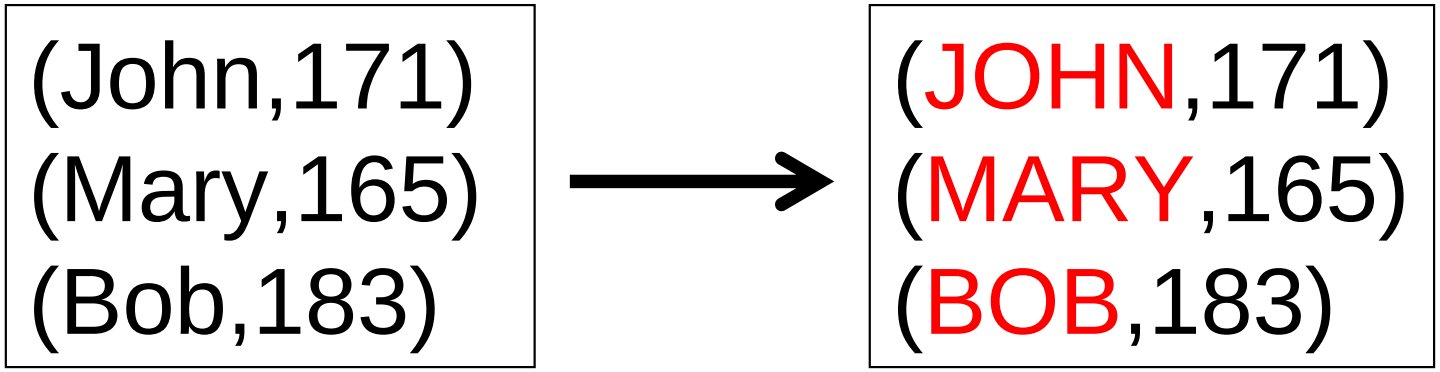
```
%default TYPE 'view'
%declare ID '18987'
A = load '/data/$DATE/$ID/$TYPE‘
```

- $ pig a.pig
- $ pig -param DATE=20091009 a.pig
- $ pig -param DATE=20091009 –param TYPE=click a.pig

RegEx Comparison

- `itsyou = FILTER urls by ($0 MATCHES 'http://.*\\.yahoo\\.com.*')`
- MATCHES matches ‘whole’ string
  - `‘aaaa’ MATCHES ‘aaa.*’` is true
  - `‘bbaaaa’ MATCHES ‘aaa.*’` is false
- pattern syntax: java.util.regex.Pattern

User-defined Function (UDF)



UDF – user function part

```
package myudf;
import java.io.IOException;
import org.apache.pig.EvalFunc;
import org.apache.pig.data.Tuple;

public class UPPER extends EvalFunc<String>
{
    public String exec(Tuple in) throws IOException {
        if (in == null || in.size() == 0) return null;
        String str = (String)in.get(0);
        return str.toUpperCase();
    }
}
```

UDF

- http://hadoop.apache.org/pig/docs/r0.3.0/udf.html
- http://hadoop.apache.org/pig/javadoc/docs/api/
- PiggyBank
  - Pig users UDF repo
  - http://wiki.apache.org/pig/PiggyBank

Embedded in Java

```
/* create a pig server in the main class*/
{
  PigServer pigserver = new PigServer(args[0]);
   runMyQuery(pigServer, "/user/viraj/mydata.txt")
}

/* submit in function runMyQuery */

runMyQuery(PigServer pigServer, String inputFile) throws
IOException {
        pigServer.registerQuery("A = load '" + inputFile +
"' as (f1,f2,f3);");
        pigServer.registerQuery("B = group A by f1;");
        pigServer.registerQuery("C = foreach B generate
flatten(group);");
        pigServer.store("C", "/user/viraj/myoutput");
}
```

References

- FAQ
  - http://wiki.apache.org/pig/FAQ
- Documentation
  - http://hadoop.apache.org/pig/docs/r0.5.0/
- Talks & papers
  - http://wiki.apache.org/pig/PigTalksPapers
  - http://www.cloudera.com/hadoop-training-pig-introduction

Questions?

Backup slides

Parameter Substitution

```
$ pig -param myparam=val foo.pig
```

```
B = filter A by ($0 eq '$myparam');
```

- `pig -dryrun` produces processed script

```
B = filter A by ($0 eq 'val');
```

Parameter Substitution

- Params in file instead of command line
- `$ pig -param_file myparams.txt a.pig`

```
#myparams.txt
DATE=20081009
TYPE=clicks
```

UDF – build user function

- javac
  -cp $PIG_HOME/lib/pig.jar
  -sourcepath src
  -d classes
  src/myudf/UPPER.java
- jar cf myudf.jar -C classes myudf/UPPER.class

UDF – pig latin part

```
register myudf.jar;
B =
foreach A generate
  myudf.UPPER(name),height;
```

SQL vs. Pig Latin

SQL	Pig	Example
From table	Load file(s)	**SQL**: from X; **Pig**: A = load 'mydata' using PigStorage('\t') as (col1, col2, col3);
Select	Foreach … generate	**SQL**: select col1 + col2, col3 … **Pig**: B = foreach A generate col1 + col2, col3;
Where	Filter	**SQL**: select col1 + col2, col3 from X where col2>2; **Pig**: C = filter B by col2 > '2';

(adapted from Viraj's slide)

SQL vs. Pig Latin

SQL	Pig	Example
Group by	Group + foreach … generate	**SQL:** select col1, col2, sum(col3) from X group by col1, col2; **Pig**: D = group A by (col1, col2); E = foreach D generate flatten(group), SUM(A.col3);
Having	Filter	**SQL**: select col1, sum(col2) from X group by col1 having sum(col2) > 5; **Pig**: F = filter E by $1 > '5';
Order By	Order … By	**SQL**: select col1, sum(col2) from X group by col1 order by col1; **Pig**: H = ORDER E by $0;

(adapted from Viraj's slide)

SQL vs. Pig Latin

SQL	Pig	Example
Distinct	Distinct	**SQL:** select distinct col1 from X; **Pig:** I = foreach A generate col1; J = distinct I;
Distinct Agg	Distinct in foreach	**SQL:** select col1, count (distinct col2) from X group by col1; **Pig:** K = foreach D { L = distinct A.col2; generate flatten(group), SUM(L); }

(adapted from Viraj's slide)

SQL vs. Pig Latin

SQL	Pig	Example
Join	**Cogroup + flatten** **(also shortcut: JOIN)**	**SQL:** select A.col1, B.col3 from A join B using (col1); **Pig:** A = load 'data1' using PigStorage('\t') as (col1, col2); B = load 'data2' using PigStorage('\t') as (col1, col3); C = cogroup A by col1 **inner**, B by col1 **inner**; D = foreach C generate flatten(A), flatten(B); E = foreach D generate A.col1, B.col3;

(adapted from Viraj's slide)

Debug Tips

- Use small data and `pig -x local`
- `LIMIT`
  - `A = LOAD 'data' AS (a1,a2,a3)`
  - `B = LIMIT A 3;`
- `DUMP , DESCRIBE`

FAQ

- http://wiki.apache.org/pig/FAQ
  - can assign #reducer
  - support regex
  - can use allocated HOD cluster

pig.vim

- http://www.vim.org/scripts/script.php?script_id=2186

```
A = load 'data.txt' as (f1,f2,f3);
dump A;
B = foreach A generate f1,f3;
dump B;
store B into 'output.txt' using PigStorage('-');
```