Map Reduce - UNI7

Fundamentals

Roberiogomes@gmail.com

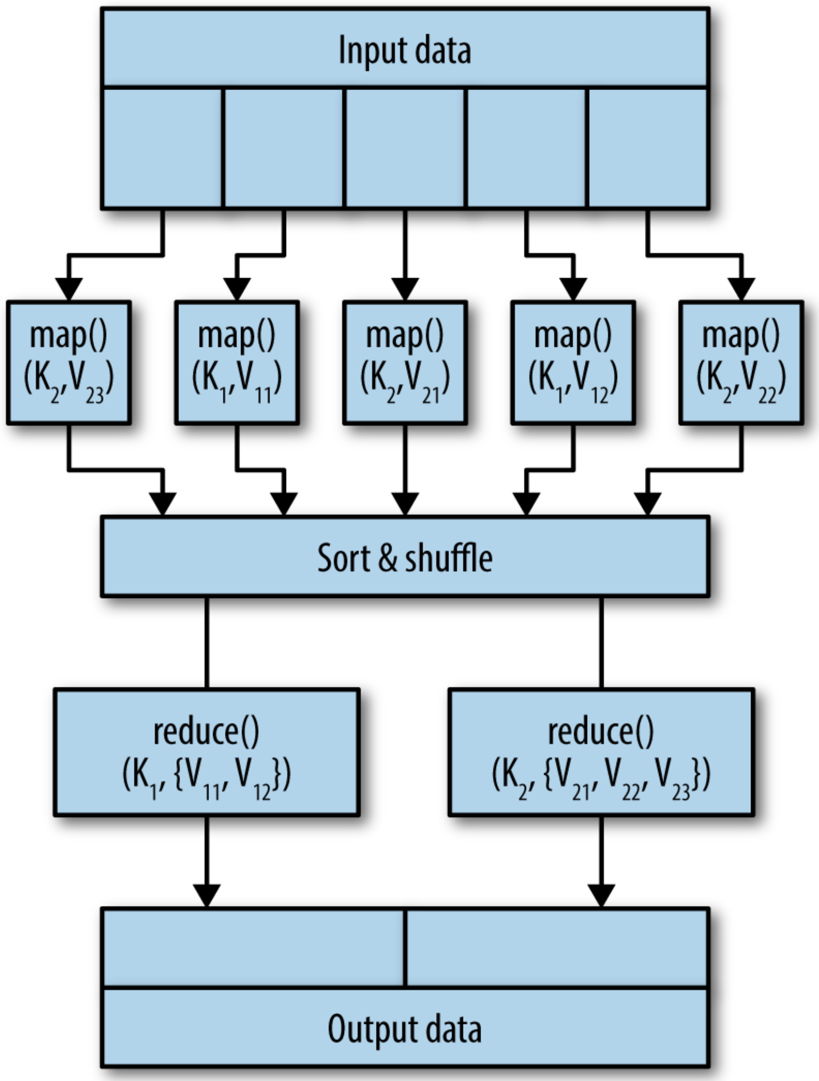
Agenda

- Definitions
- Big Picture
- Functions
- Examples
- Map Reduce with Python
- MRJob
- Common Problems - Part I
- Home work

What is MapReduce?

- MapReduce is a programming paradigm that allows for **massive scalability across hundreds or thousands of servers in a cluster environment.**
- The term MapReduce originated from functional programming and was introduced by Google in a paper called “MapReduce: Simplified Data Processing on Large Clusters.”

MapReduce process



What is MapReduce?

- Simply put, MapReduce is about scalability.
- Using the MapReduce paradigm, you focus on writing two functions:
  - map() ⟶ Filters and aggregates data
  - reduce() ⟶ Reduces, groups, and summarizes by keys generated by map()

Map() function

- The master node takes the input, partitions it into smaller data chunks, and distributes them to worker (slave) nodes.
- The worker nodes apply the same transformation function to each data chunk, then pass the results back to the master node.

$$\text{map}(): (\text{Key}_1, \text{Value}_1) \longrightarrow [(\text{Key}_2, \text{Value}_2)]$$

Reduce() function

- The master node shuffles and clusters the received results based on unique key-value pairs;
- Then, through another redistribution to the workers/slaves, these values are combined via another type of transformation function.

$$\text{reduce}(): (\text{Key}_2, [\text{Value}_2]) \longrightarrow [(\text{Key}_3, \text{Value}_3)]$$

Table P-1. Mappers' output

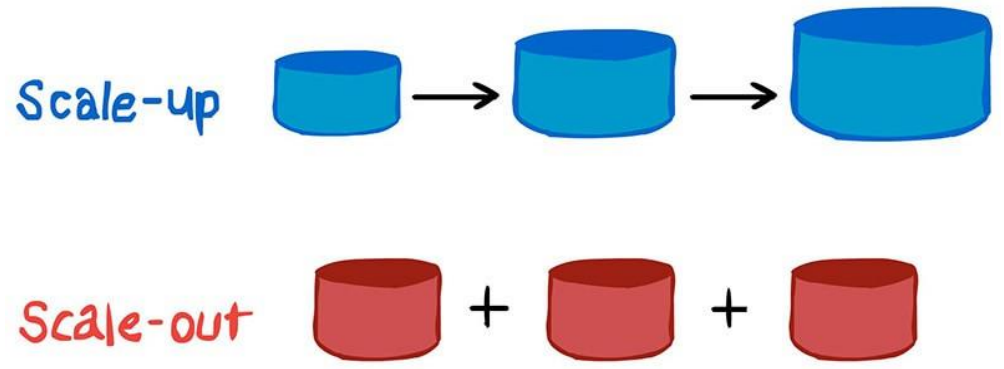
Key	Value
K_1	V_{11}
K_2	V_{21}
K_1	V_{12}
K_2	V_{22}
K_2	V_{23}

Table P-2. Reducers' input

Key	Value
K_1	$\{V_{11}, V_{12}\}$
K_2	$\{V_{21}, V_{22}, V_{23}\}$

MapReduce in Action

- When writing your map() and reduce() functions, make sure that your solution is scalable.
- Scalability is the heart of MapReduce.
- When we talk about scalability, we mean scaling out .



Simple Explanation

- Let’s say that we want to count the number of books in a library that has 1,000 shelves and report the final result to the librarian.
- Solution #1 (using map() and reduce()):
  - map(): Hire 1,000 workers; each worker counts one shelf.
  - reduce(): All workers get together and add up their individual counts

Simple Explanation

- Solution #2 (using map(), combine(), and reduce()):

- map(): Hire 1,110 workers (1,000 workers, 100 managers, 10 supervisors—each supervisor manages 10 managers, and each manager manages 10 workers); each worker counts one shelf, and reports its count to its manager.

- combine(): Every 10 managers add up their individual counts and report the total to a supervisor.

- reduce(): All supervisors get together and add up their individual counts (by reporting the results to the librarian)

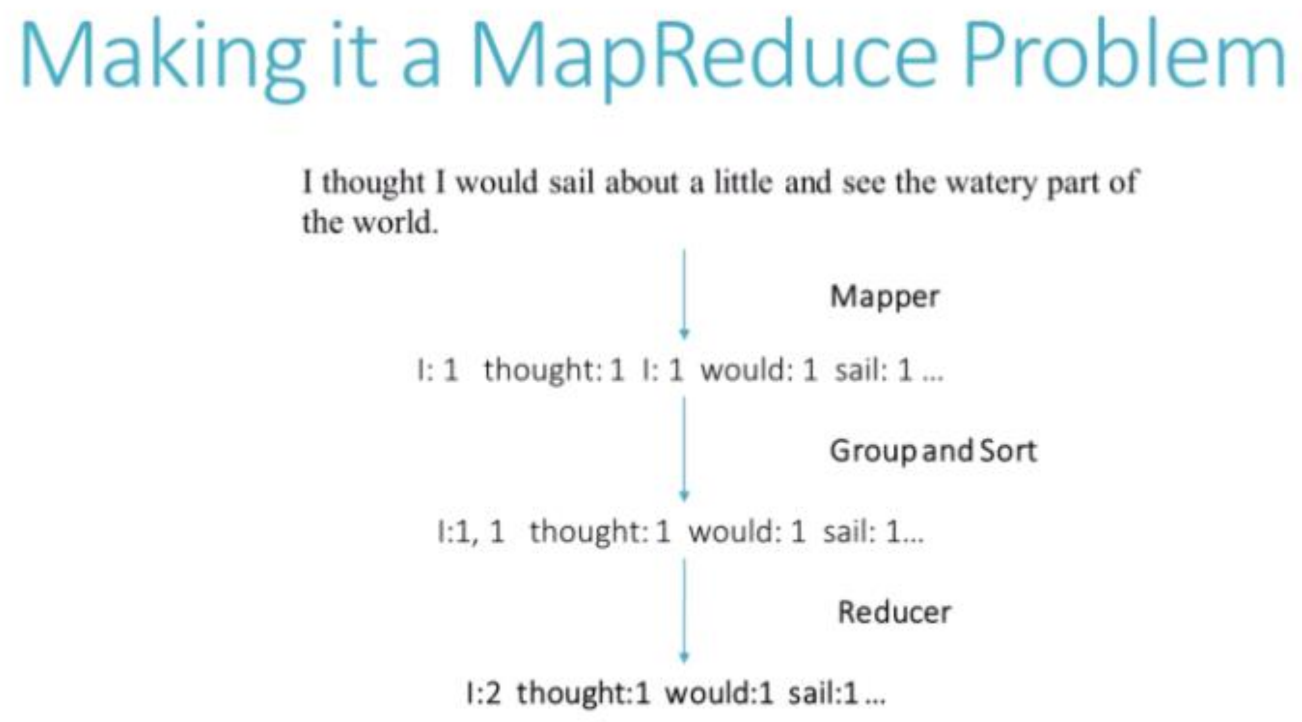
Hello World Map Reduce

Input Data

Text from a book:

"I thought I would sail about a little and see the watery part of the world."

Hello World Map Reduce



Making it a MapReduce Problem

I thought I would sail about a little and see the watery part of the world.

$\downarrow$ Mapper

I: 1 thought: 1 I: 1 would: 1 sail: 1 ...

$\downarrow$ Group and Sort

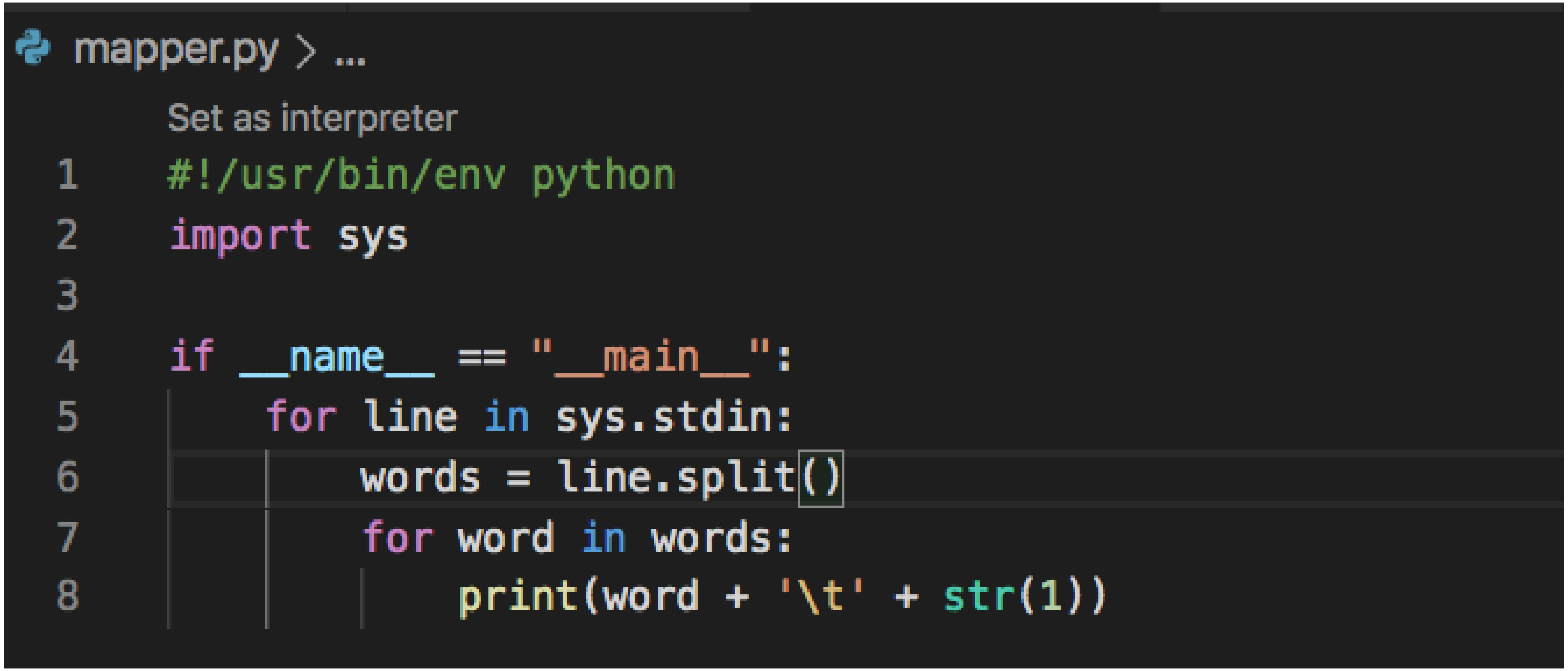
I:1, 1 thought: 1 would: 1 sail: 1...

$\downarrow$ Reducer

I:2 thought:1 would:1 sail:1 ...

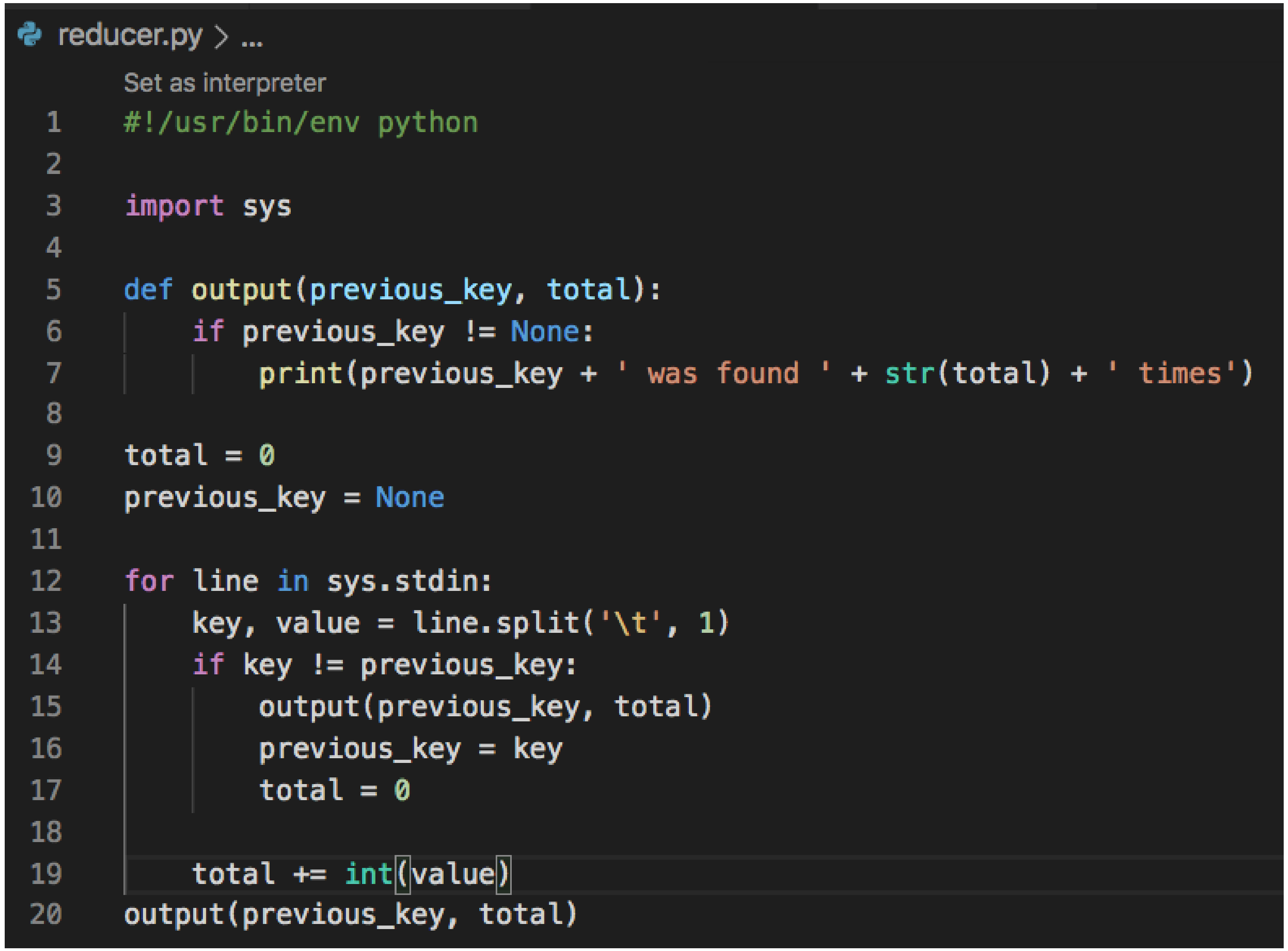
MapReduce with Python

MapReduce with Python


mapper.py > ...

Set as interpreter

```python
#!/usr/bin/env python
import sys

if __name__ == "__main__":
    for line in sys.stdin:
        words = line.split()
        for word in words:
            print(word + '\t' + str(1))
```


reducer.py > ...

Set as interpreter

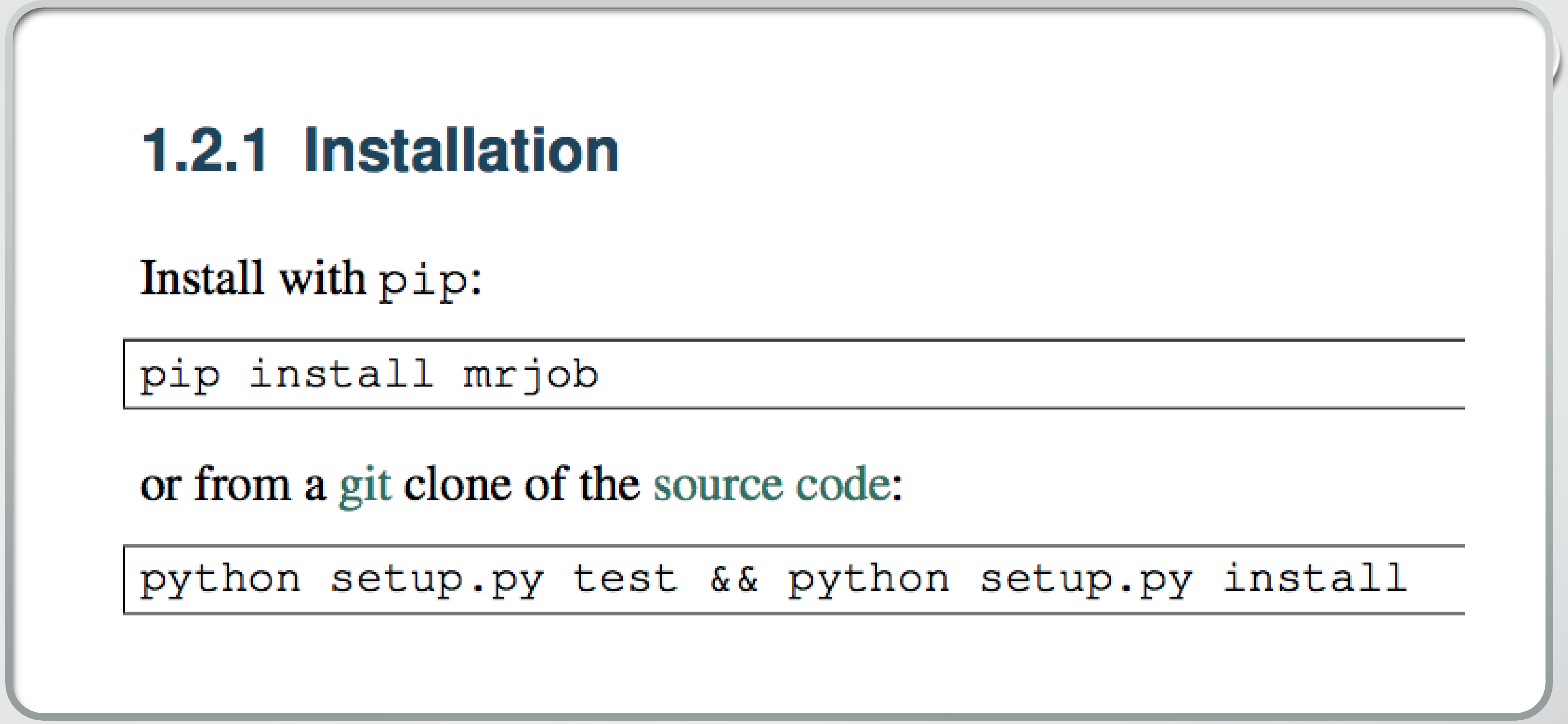
```python
#!/usr/bin/env python

import sys

def output(previous_key, total):
    if previous_key != None:
        print(previous_key + ' was found ' + str(total) + ' times')

total = 0
previous_key = None

for line in sys.stdin:
    key, value = line.split('\t', 1)
    if key != previous_key:
        output(previous_key, total)
        previous_key = key
        total = 0

    total += int(value)
output(previous_key, total)
```

Let's Practice!!!

MapReduce with Python - MRJob

Why mrjob?

- mrjob is the easiest route to writing Python programs that run on Hadoop. If you use mrjob, you'll be able to test your code locally without installing Hadoop or run it on a cluster of your choice.
- Additionally, mrjob has extensive integration with Amazon Elastic MapReduce. Once you're set up, it's as easy to run your job in the cloud as it is to run it on your laptop.
- If you don't want to be a Hadoop expert but need the computing power of MapReduce, mrjob might be just the thing for you.



1.2.1 Installation

Install with `pip`:

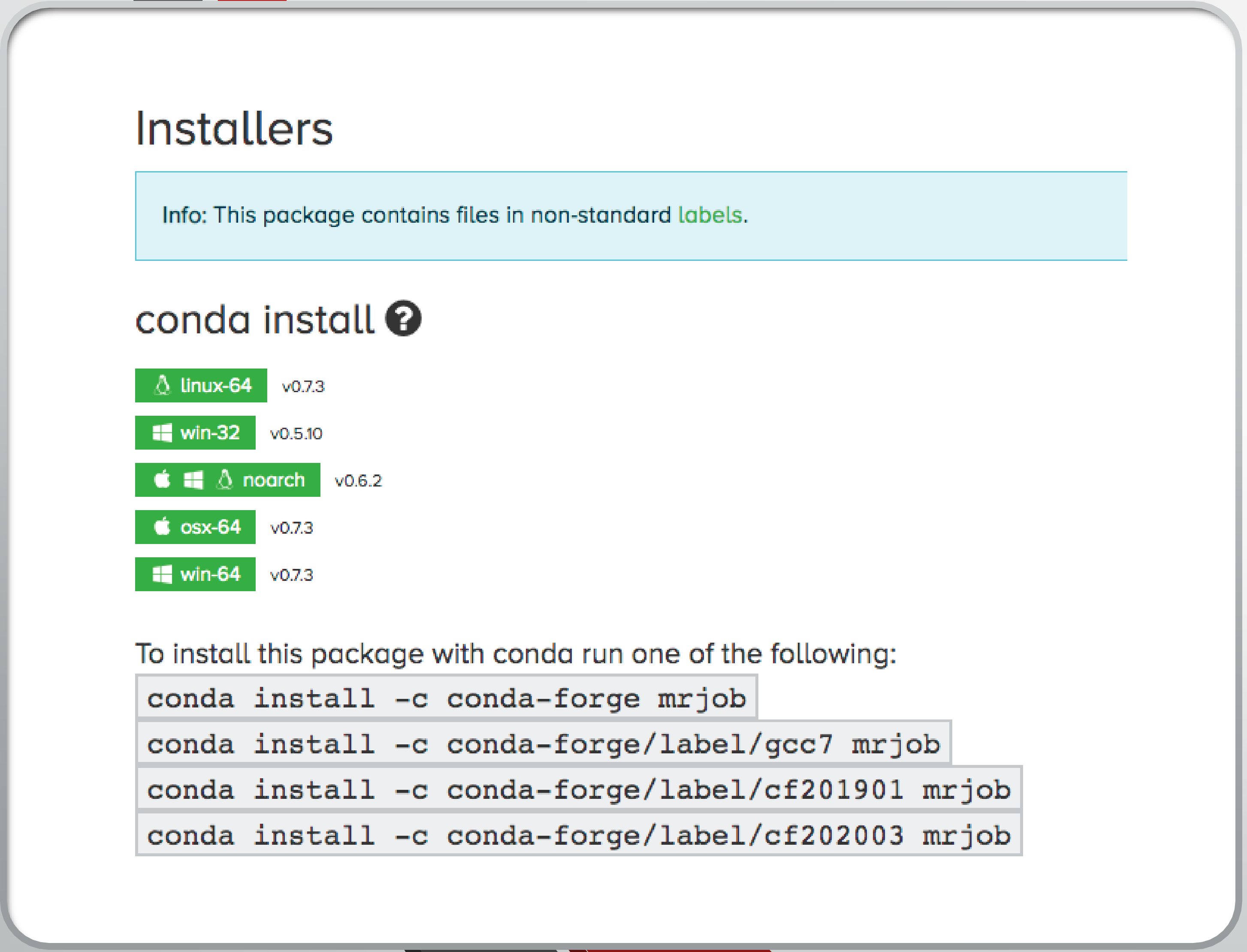
```
pip install mrjob
```

or from a git clone of the source code:

```
python setup.py test && python setup.py install
```

MapReduce with Python - MRJob

MapReduce with Python - MRJob



Installers

Info: This package contains files in non-standard labels.

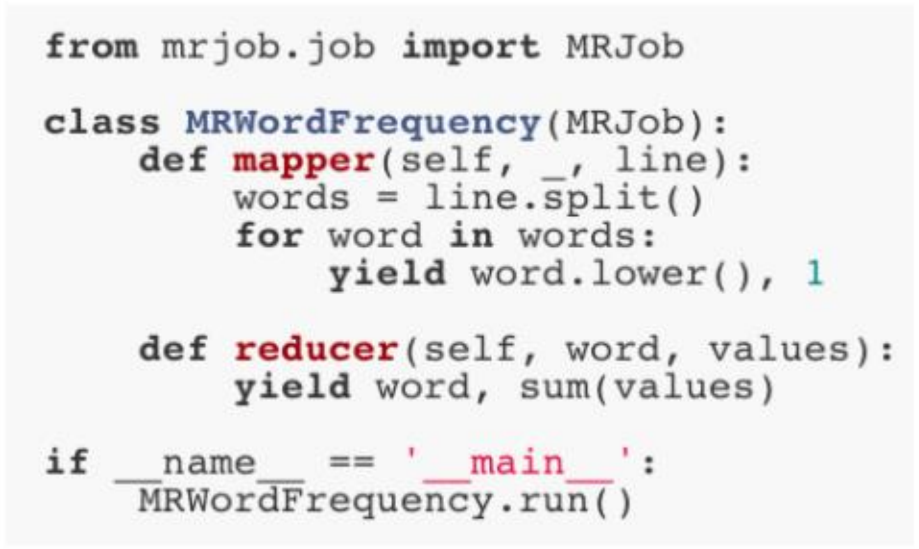
conda install

linux-64	v0.7.3
win-32	v0.5.10
noarch	v0.6.2
osx-64	v0.7.3
win-64	v0.7.3

To install this package with conda run one of the following:

```
conda install -c conda-forge mrjob
conda install -c conda-forge/label/gcc7 mrjob
conda install -c conda-forge/label/cf201901 mrjob
conda install -c conda-forge/label/cf202003 mrjob
```

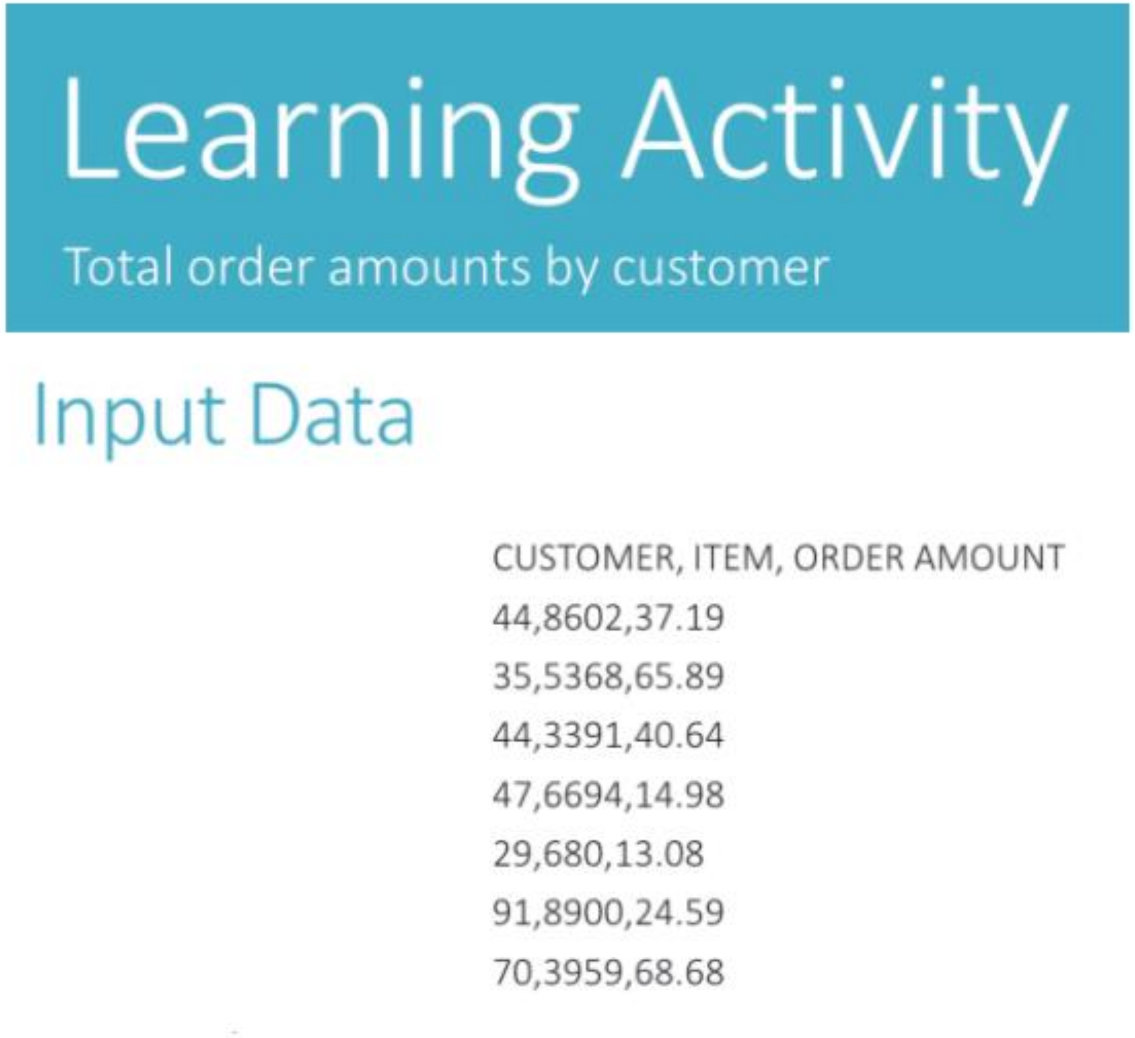
Hello World Map Reduce - Code



```python
from mrjob.job import MRJob

class MRWordFrequency(MRJob):
    def mapper(self, _, line):
        words = line.split()
        for word in words:
            yield word.lower(), 1

    def reducer(self, word, values):
        yield word, sum(values)

if __name__ == '__main__':
    MRWordFrequency.run()
```

Let's Practice!!!

PROBLEMS
SOLUTIONS



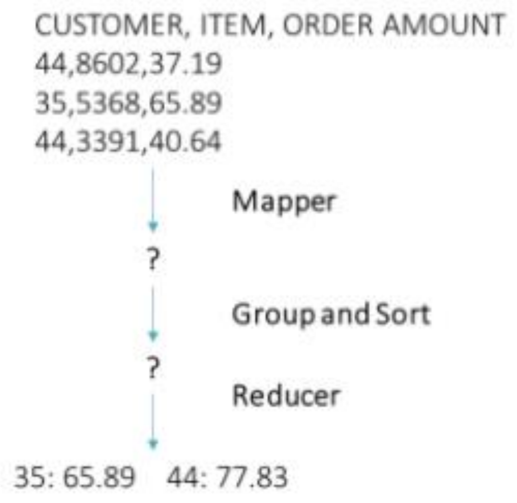
Learning Activity

Total order amounts by customer

Input Data

```
CUSTOMER, ITEM, ORDER AMOUNT
44,8602,37.19
35,5368,65.89
44,3391,40.64
47,6694,14.98
29,680,13.08
91,8900,24.59
70,3959,68.68
```

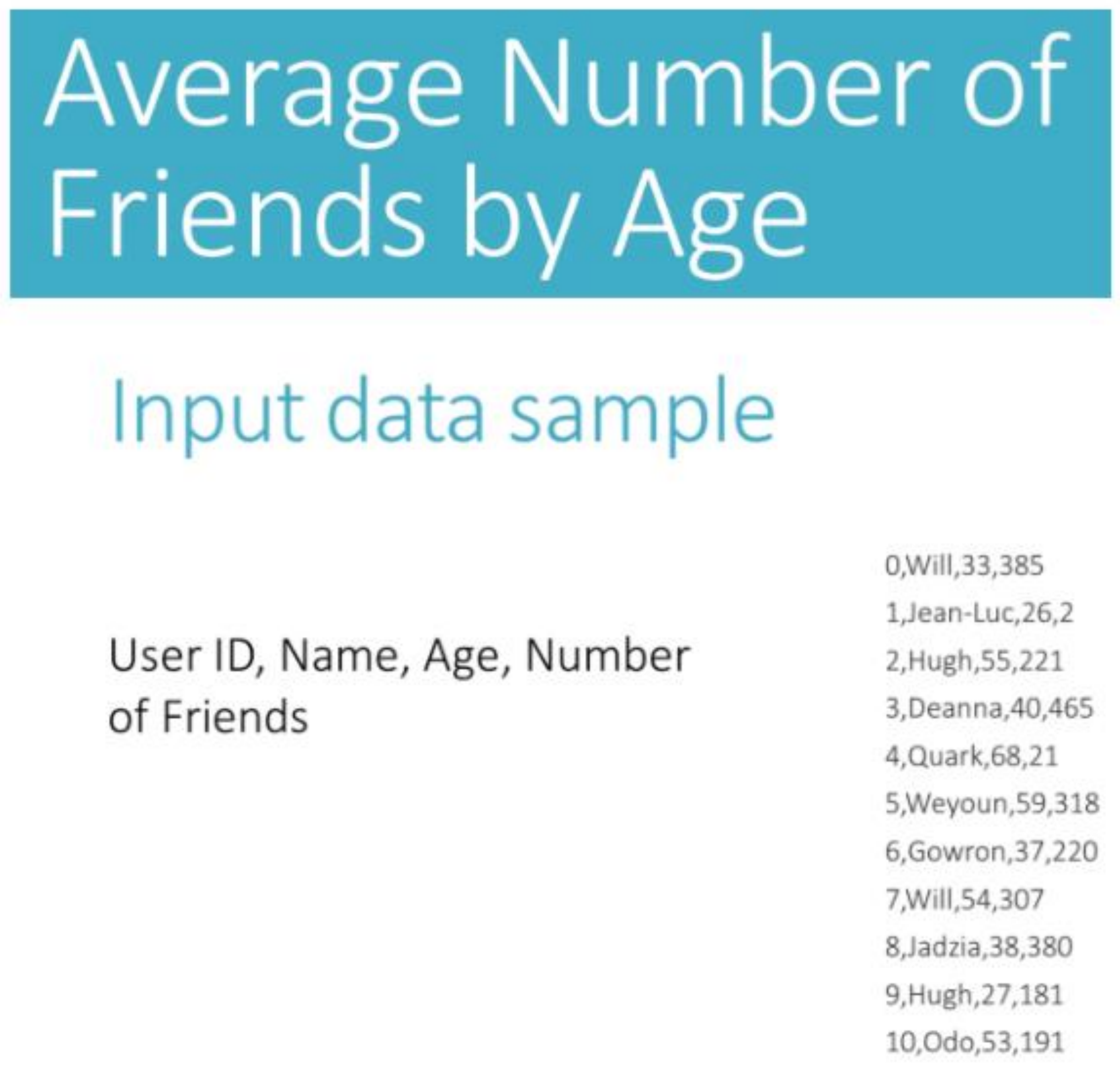
What will your Mapper and Reducer Do?



CUSTOMER, ITEM, ORDER AMOUNT
44,8602,37.19
35,5368,65.89
44,3391,40.64

35: 65.89   44: 77.83

Let's Practice!!!



Average Number of Friends by Age

Input data sample

User ID, Name, Age, Number of Friends

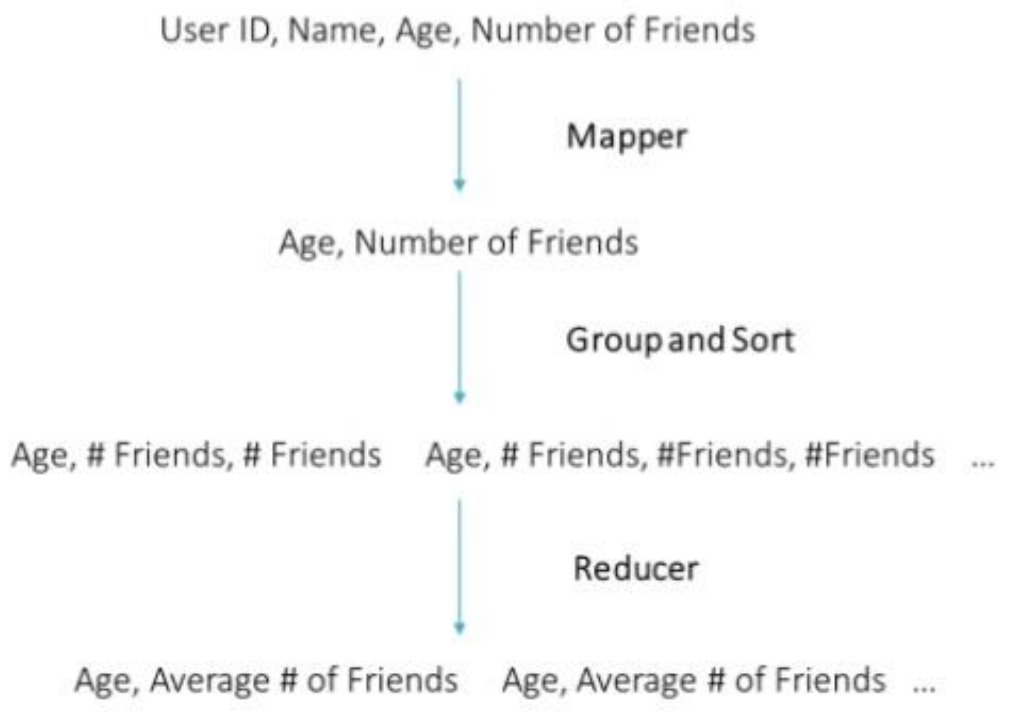
```
0,Will,33,385
1,Jean-Luc,26,2
2,Hugh,55,221
3,Deanna,40,465
4,Quark,68,21
5,Weyoun,59,318
6,Gowron,37,220
7,Will,54,307
8,Jadzia,38,380
9,Hugh,27,181
10,Odo,53,191
```

What fields do we care about?

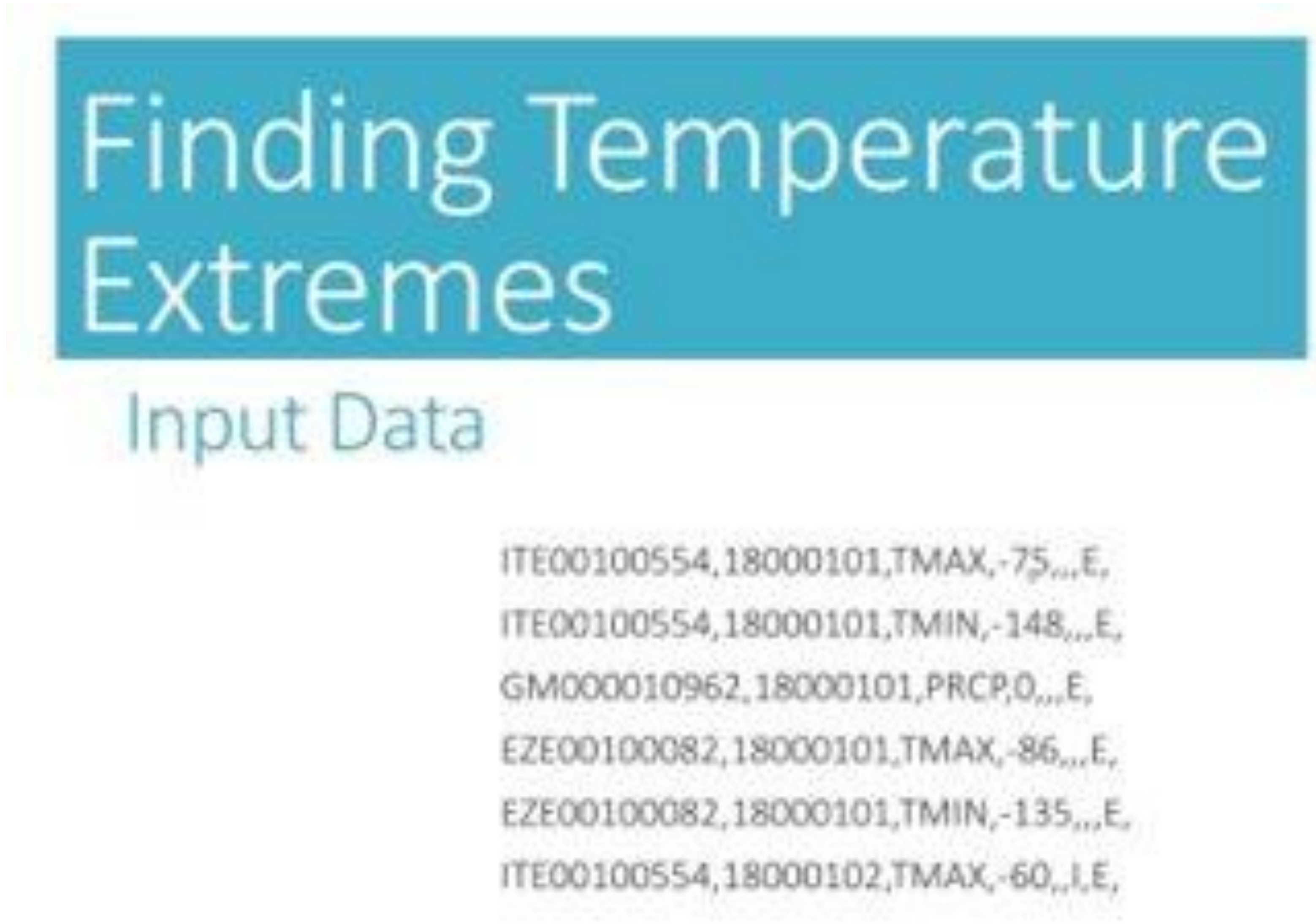
User ID, Name, Age, Number of Friends

```
0,Will,33,385
1,Jean-Luc,26,2
2,Hugh,55,221
3,Deanna,40,465
4,Quark,68,21
5,Weyoun,59,318
6,Gowron,37,220
7,Will,54,307
8,Jadzia,38,380
9,Hugh,27,181
10,Odo,53,191
```

Making it a MapReduce Problem


User ID, Name, Age, Number of Friends
Mapper
Age, Number of Friends
Group and Sort
Age, # Friends, # Friends
Age, # Friends, #Friends, #Friends
...
Reducer
Age, Average # of Friends
Age, Average # of Friends
...

Let's Practice!!!

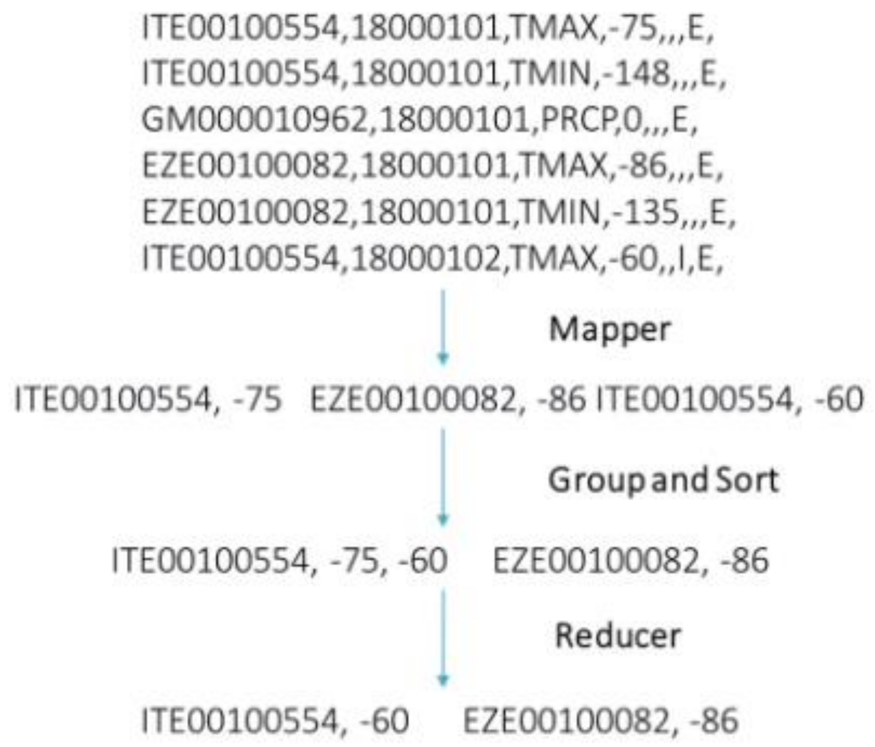


Finding Temperature Extremes

Input Data

```
ITE00100554,18000101,TMAX,-75,,,E,
ITE00100554,18000101,TMIN,-148,,,E,
GM000010962,18000101,PRCP,0,,,E,
EZE00100082,18000101,TMAX,-86,,,E,
EZE00100082,18000101,TMIN,-135,,,E,
ITE00100554,18000102,TMAX,-60,,I,E,
```

Making it a MapReduce Problem



```
ITE00100554,18000101,TMAX,-75,,,E,
ITE00100554,18000101,TMIN,-148,,,E,
GM000010962,18000101,PRCP,0,,,E,
EZE00100082,18000101,TMAX,-86,,,E,
EZE00100082,18000101,TMIN,-135,,,E,
ITE00100554,18000102,TMAX,-60,,I,E,
```

Mapper

```
ITE00100554, -75   EZE00100082, -86 ITE00100554, -60
```

Group and Sort

```
ITE00100554, -75, -60    EZE00100082, -86
```

Reducer

```
ITE00100554, -60    EZE00100082, -86
```

Let's Practice!!!