I wonder what BigData, Hadoop and MapReduce is all about ☹

How Big is BigBigBigData?

Facebook handles 180 PB of data in a year

Twitter handles 1.2 million tweets per second

2.3 Trillion GB everyday ; 40 Trillion GB by 2020

Volume} ☹

What's wrong with data?

Log files → INFO: 192.12.135.34 /products/raspberry

JSON , XML , TXT , Videos , MP3s and your guess is as good as mine.

The point is there is simply too ~~much beauty in this world~~.

Or

Variety} ☹

in

Data

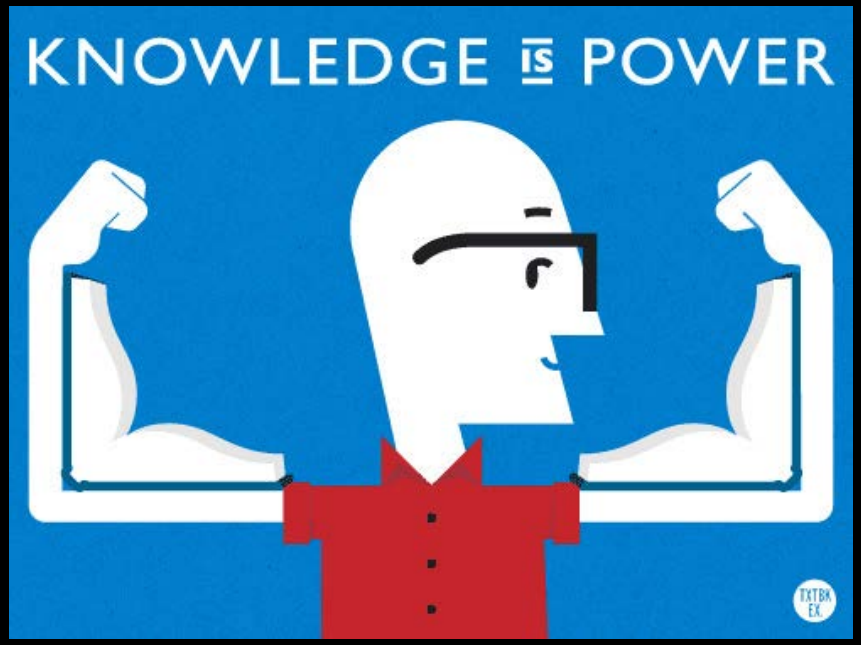
We will need Time machines!

- → Data is getting generated at a faster pace than what we can process.
- → Think of a SQL query with complex join operations across multiple tables each having millions of records.
- → Elegant Solution : build a time machine ; Go back in time and do data processing.

Velocity} ☹

Why can’t we just chuck this data away?

Francis Bacon said something wise.



DATA = ADVANTAGE

Think of Flipkart/Amazon/Snapdeal

→ Competitors

→ The company which understands the need of user better will eventually dominate.

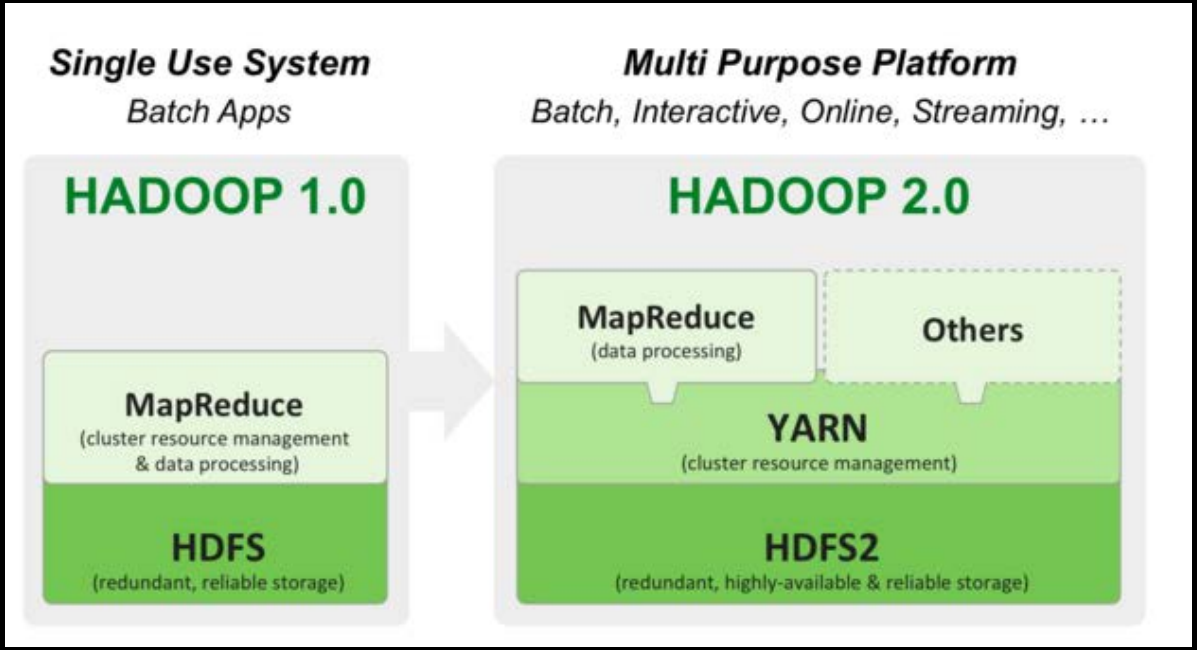
→ The more analysis they perform on their data logs , better their understanding on user.

→ ML, IoT and all new technologies of future would need BigData support.

Hadoop Hadoop Hadoop!

→ Top level project under Apache Software Foundation (Open Source)

→ Created based on Google research papers on how Google processed their large volume of data.



Two Questions

Q1) Where does all our Big Big data goes and gets stored ?

Ans . Hadoop Distributed File System (HDFS)

Q2) How do we process all this Big Big data ?

Ans. MapReduce

Q3) What is the key idea behind big data processing?

Ans. Next Slide

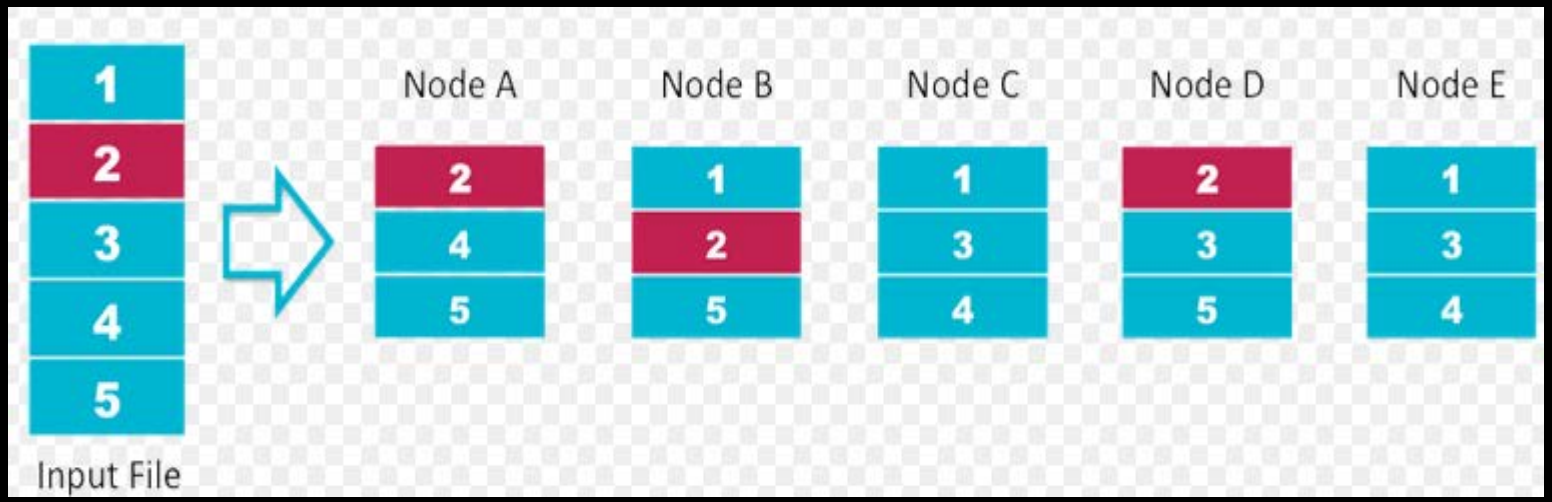
Key Ideas

- Step 1: HDFS → Our large data gets stored as small chunks of data in many machines (nodes) in a cluster.

- Step 2 : Data Local Computing → We process this small chunk of data simultaneously at the nodes in which data is present.

- Step 3 : Aggregates the results from Step2 across various nodes and save the money for building time machines.

HDFS



Replication Factor = 3
Cluster size = 5

Map Reduce

→ Is a programming paradigm (FP)

→ **Mapper** : The part of your program that
a) reads the file from HDFS ,
b) performs filtering , splitting and cleansing of data
c) Emits the data as key, value pair to reducer

→ **Reducer :** The part of your program that
a) performs aggregation or summary operation
b) the place where our program logic usually resides
c) Writes the output to HDFS

"Output from the Mapper is the input to Reducer"

A little more on MR

Mapping

- Mapper
- Each node applies the mapper function to the small chunk of data that it holds simultaneously and produces key, value pairs.

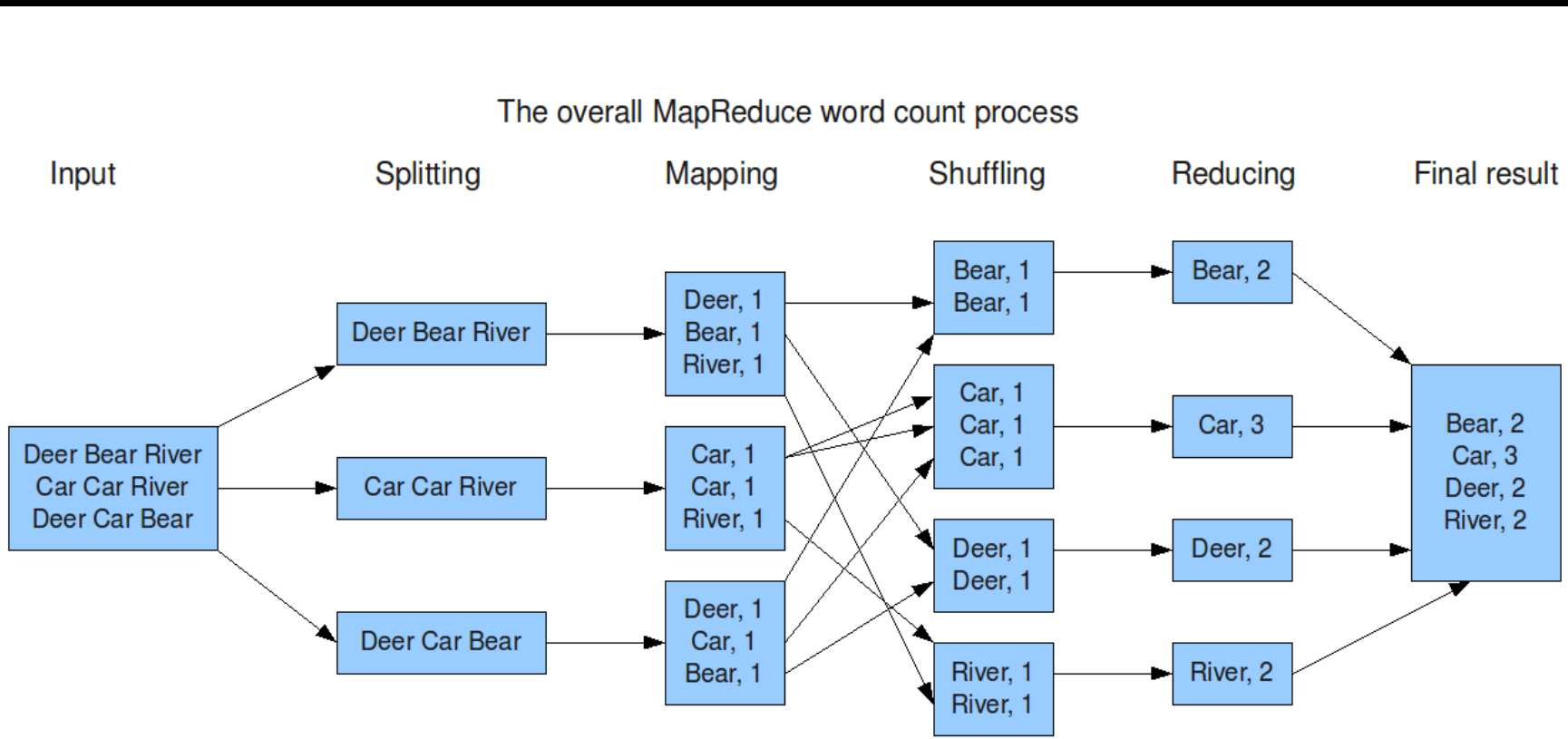
Shuffle and Sort

- Hadoop internal
- Nodes redistribute data based on the output from Mapper such that all data belonging to one key is located on the same node.

Reducing

- Reducer
- Nodes now process each group of output data, per key, in parallel

Word Count


The overall MapReduce word count process
Input
Splitting
Mapping
Shuffling
Reducing
Final result
Deer Bear River
Car Car River
Deer Car Bear
Deer Bear River
Car Car River
Deer Car Bear
Deer, 1
Bear, 1
River, 1
Car, 1
Car, 1
River, 1
Deer, 1
Car, 1
Bear, 1
Bear, 1
Bear, 1
Car, 1
Car, 1
Car, 1
Deer, 1
Deer, 1
River, 1
River, 1
Bear, 2
Car, 3
Deer, 2
River, 2
Bear, 2
Car, 3
Deer, 2
River, 2

Where does Python come in ?

→ Hadoop is written mainly in Java.

→ We can run mapreduce jobs via Hadoop Streaming (HS).

→ Using python for HS is quite easy.

→ We can do almost all the things with HS that normal Hadoop execution provides.

→ There are going to be 2 parts to our mapreduce program.

→ Any guesses ?

Mapper.py

```
1  #!/usr/bin/env python
2
3  import sys
4
5  # input comes from STDIN (standard input)
6  for line in sys.stdin:
7      # remove leading and trailing whitespace
8      line = line.strip()
9      # split the line into words
10     words = line.split()
11     # increase counters
12     for word in words:
13         # write the results to STDOUT (standard output);
14         # what we output here will be the input for the
15         # Reduce step, i.e. the input for reducer.py
16         #
17         # tab-delimited; the trivial word count is 1
18         print '%s\t%s' % (word, 1)
```

Reducer.py

```python
import sys

current_word = None
current_count = 0
word = None

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    word, count = line.split('\t', 1)


    # this IF-switch only works because Hadoop sorts map output
    # by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
            # write result to STDOUT
            print '%s\t%s' % (current_word, current_count)
        current_count = count
        current_word = word

# do not forget to output the last word if needed!
if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

How to get Started ?

→ Udacity : Intro to Hadoop and Map Reduce

→ MapR Free training lessons

→ Yahoo Hadoop Tutorials

→ BigData University

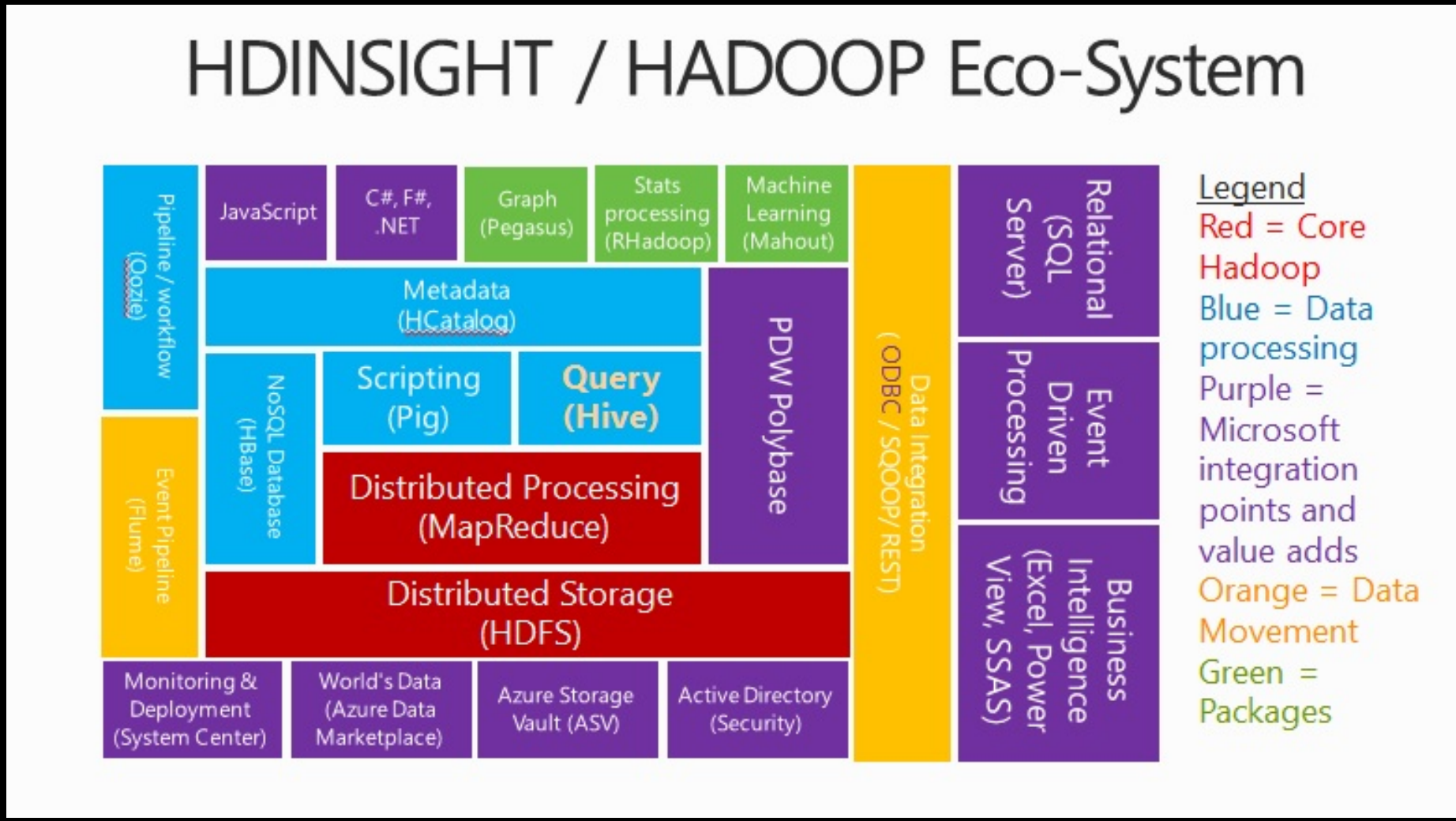
→ Hortonworks Webinars

Book : The Definitive Guide to Hadoop

Into to HDFS ; Youtube video

Tip : use the Cloudera VM provided by Udacity. It is light weight.

Hadoop EcoSystem



Thank You ☺