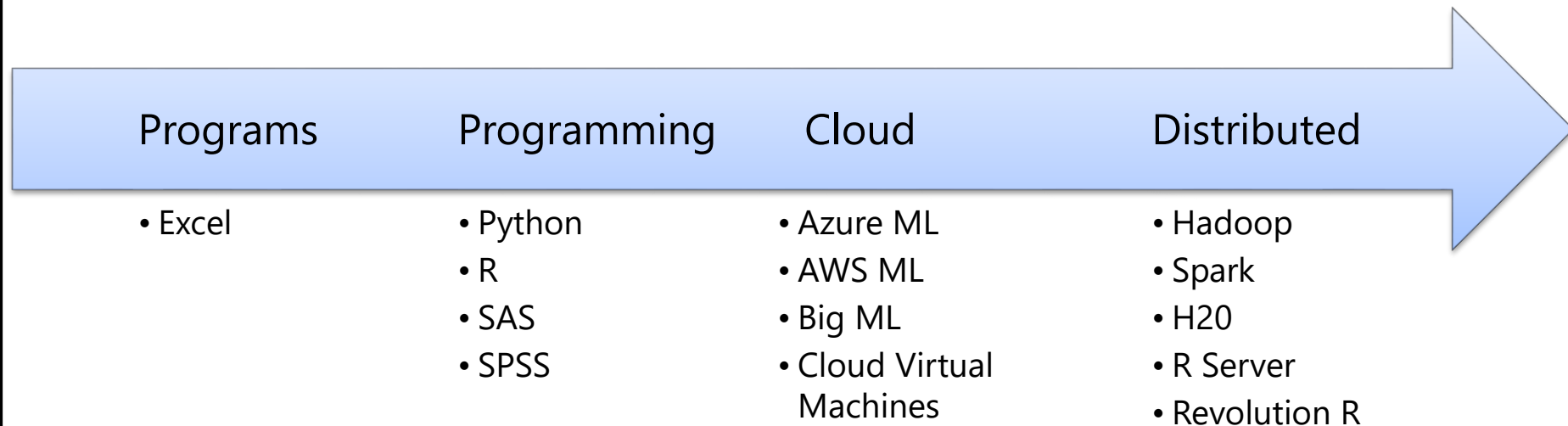


Scaling R to Big Data

Data Science Dojo

Machine Learning Scaling

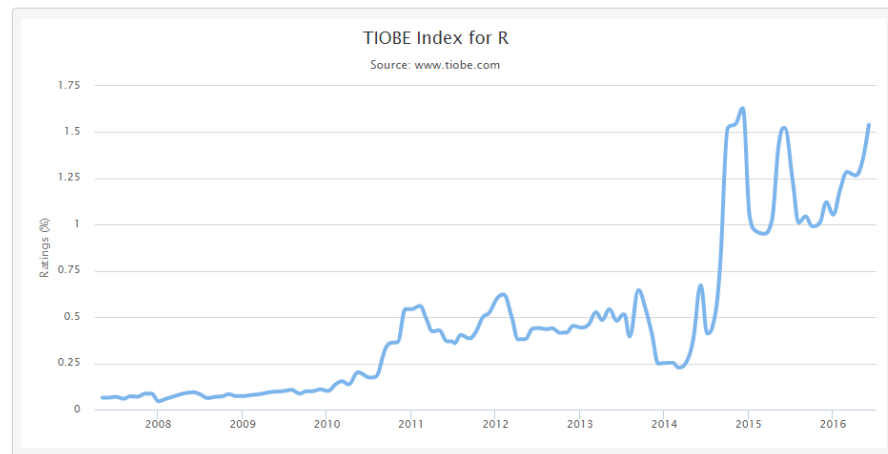


Distributed R Solutions:

<https://cran.r-project.org/web/views/HighPerformanceComputing.html>

Popularity Overall

Jun 2016	Jun 2015	Change	Programming Language	Ratings	Change
1	1		Java	20.794%	+2.97%
2	2		C	12.376%	-4.41%
3	3		C++	6.199%	-1.56%
4	6	▲	Python	3.900%	-0.10%
5	4	▼	C#	3.786%	-1.27%
6	8	▲	PHP	3.227%	+0.36%
7	9	▲	JavaScript	2.583%	+0.29%
8	12	▲	Perl	2.395%	+0.64%
9	7	▼	Visual Basic .NET	2.353%	-0.82%
10	16	▲	Ruby	2.336%	+0.98%
11	11		Visual Basic	2.254%	+0.41%
12	23	▲	Assembly language	2.119%	+1.36%
13	10	▼	Delphi/Object Pascal	1.939%	+0.07%
14	14		Swift	1.831%	+0.39%
15	5	▼	Objective-C	1.704%	-2.64%
16	13	▼	R	1.540%	+0.02%



Source: http://www.tiobe.com/tiobe_index

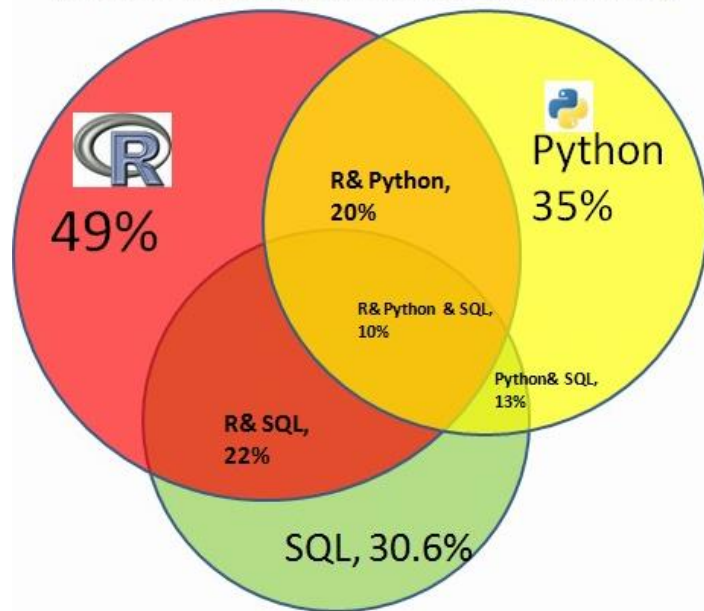
Popularity for Data Science

kdnuggets poll on data mining and data science

What programming/statistics languages you used for an analytics / data mining / data science work in 2014?

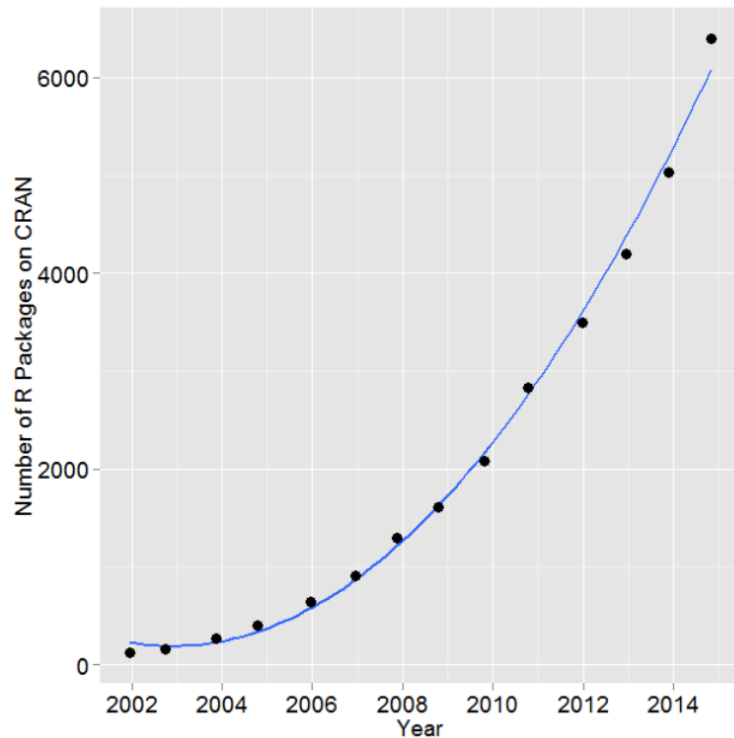
Language used	% voters in 2014 (719 total)	% voters in 2013 (713 total)	% voters in 2012 (579 total)
R (352 voters in 2014)	49.0%	60.9%	52.5%
SAS (262)	36.4%	20.8%	19.7%
Python (252)	35.0%	38.8%	36.1%
SQL (220)	30.6%	36.6%	32.1%
Java (89)	12.4%	16.5%	21.2%

KDnuggets 2014 Poll: Languages used for Analytics/Data Mining



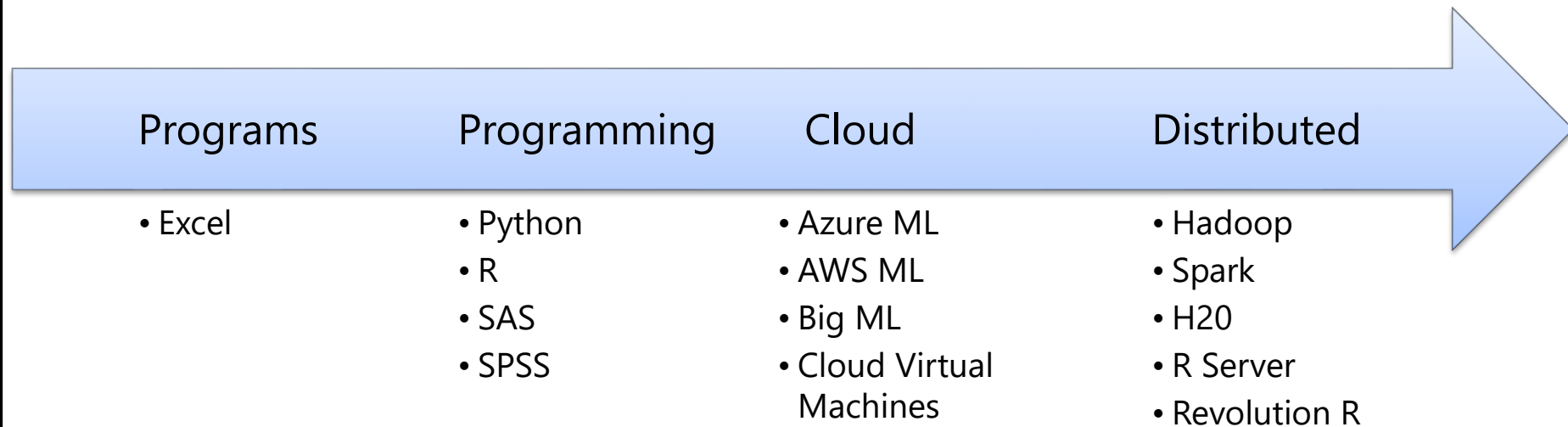
Source: <http://www.kdnuggets.com/2014/08/four-main-languages-analytics-data-mining-data-science.html>

R as a Movement



- Open-source
- 8000+ public packages
- Designed for statistical analysis, data analysis

Machine Learning Scaling



Distributed R Solutions:

<https://cran.r-project.org/web/views/HighPerformanceComputing.html>

R Limits

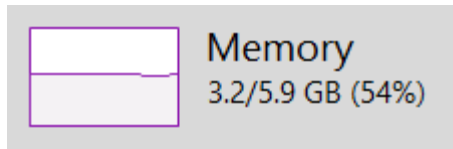
- Single core
- Single threaded
 - Sequential processing, not parallel
- All in memory (RAM)
- Cannot be distributed across computers

R Limits: RAM

- All in memory (RAM)

$$\text{Max Data Limit} = (\text{Total RAM Access} - \text{Normal RAM Usage}) \times 80\%$$

Phuc's Laptop Example:



$$\text{Max Data Limit} = (5.9\text{gb} - 3.2\text{gb}) \times 80\%$$

$$\text{Max Data Limit} = \sim 2.16\text{gb}$$

R Limits: RAM

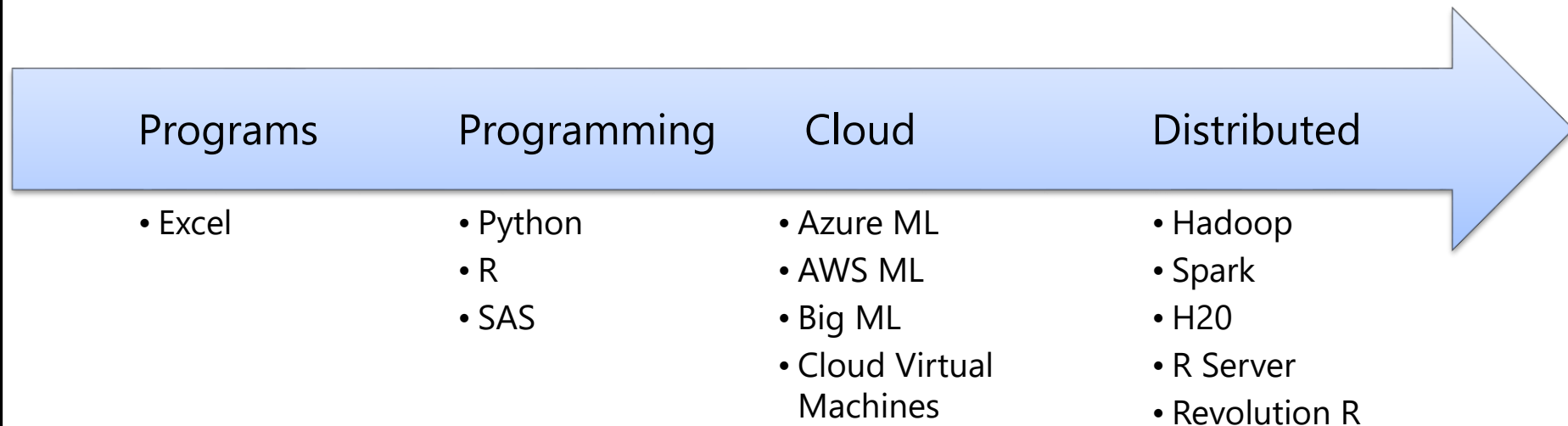
INSTANCE	CORES	RAM	DISK SIZES	PRICE
G1	2	28 GB	384 GB	\$0.67/hr (~\$498/mo)
G2	4	56 GB	768 GB	\$1.34/hr (~\$997/mo)
G3	8	112 GB	1,536 GB	\$2.68/hr (~\$1,994/mo)
G4	16	224 GB	3,072 GB	\$5.36/hr (~\$3,988/mo)
G5	32	448 GB	6,144 GB	\$9.65/hr (~\$7,180/mo)

Azure's Biggest Virtual Machine

Max Data Limit = (448gb – 1gb) x 80%

Max Data Limit = ~357.6gb

Machine Learning Scaling



Distributed R Solutions:

<https://cran.r-project.org/web/views/HighPerformanceComputing.html>

Unlocking the Potential of R

- Single core (scaleR)
- Single threaded (scaleR)
 - Sequential processing, not parallel
- All in memory (RAM) (scaleR)
- Cannot be distributed across computers (distributedR)

Why are we here?



Data (hidden value)

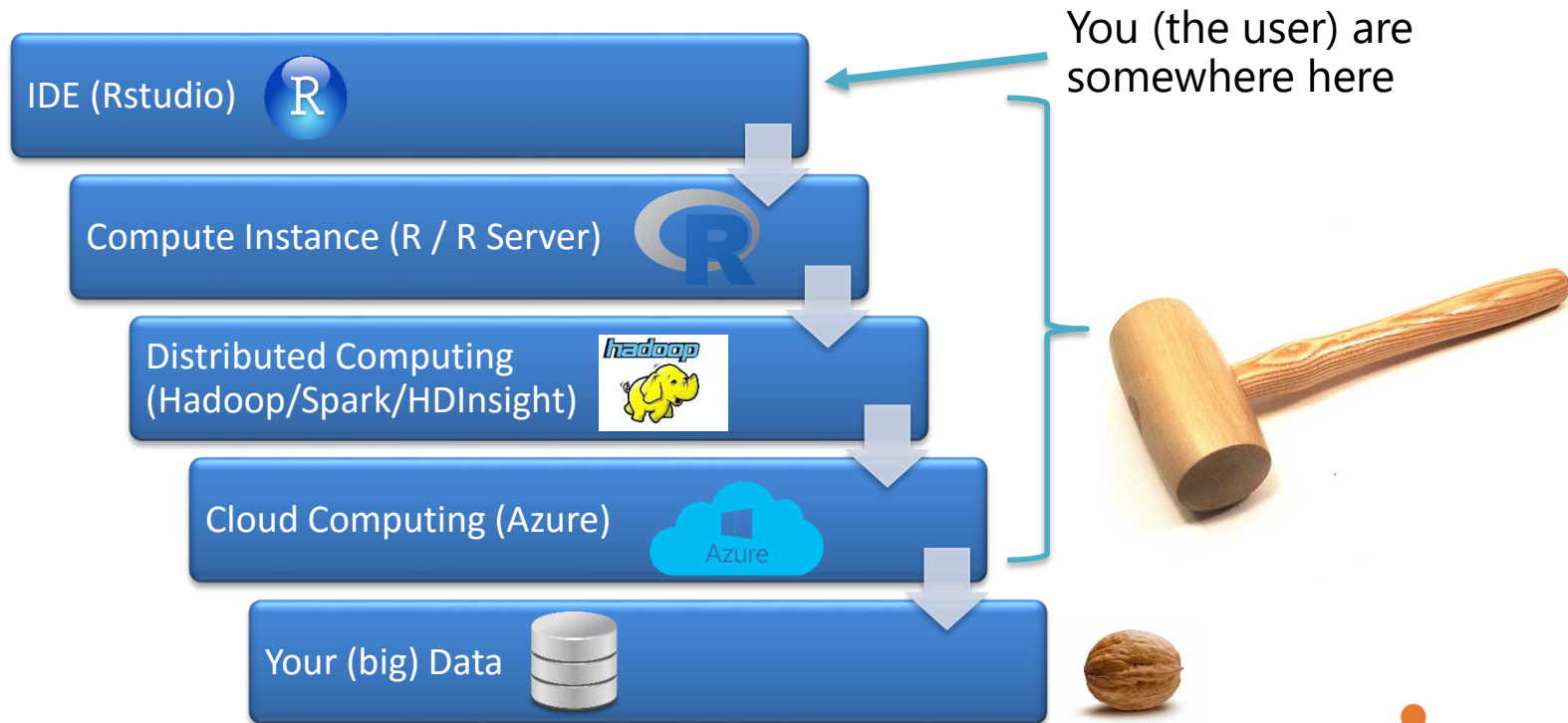


Big Data Tools

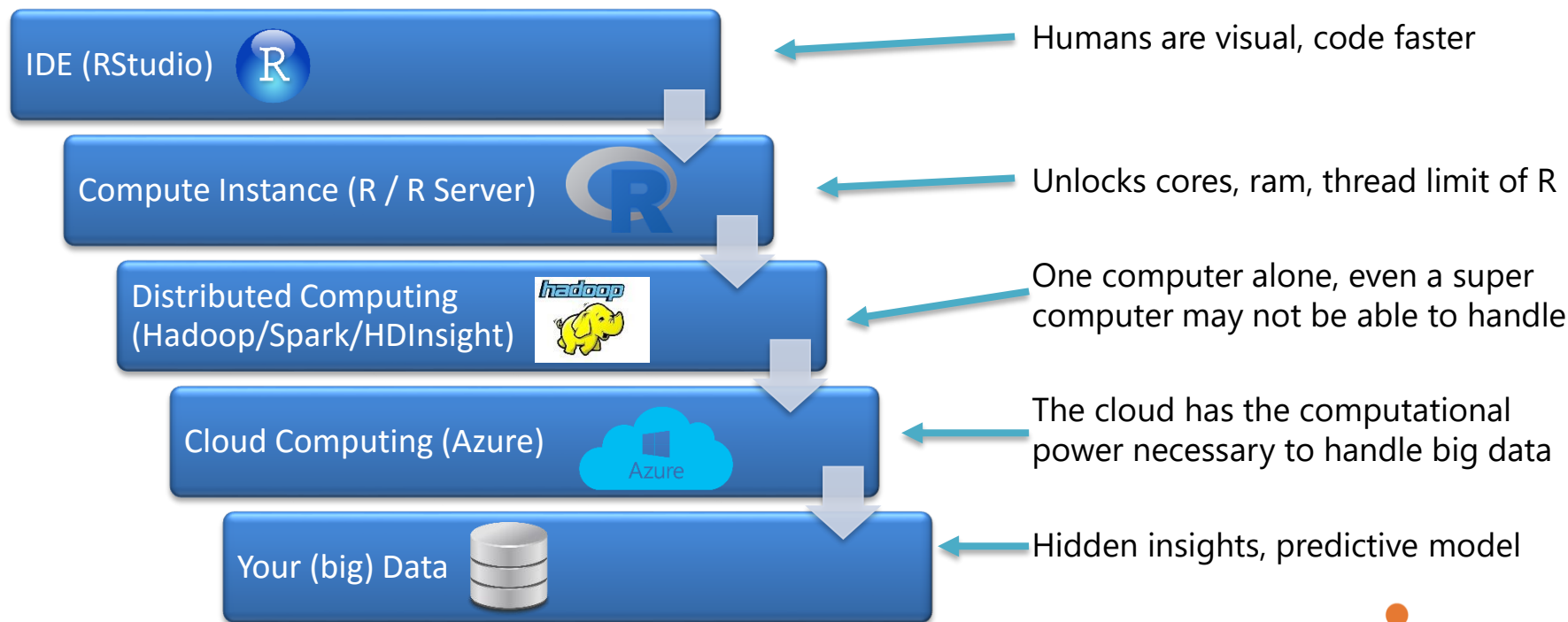


Insights

Turtles all the way down...

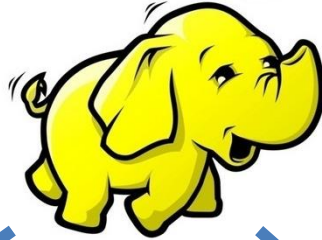


Why are these parts necessary?



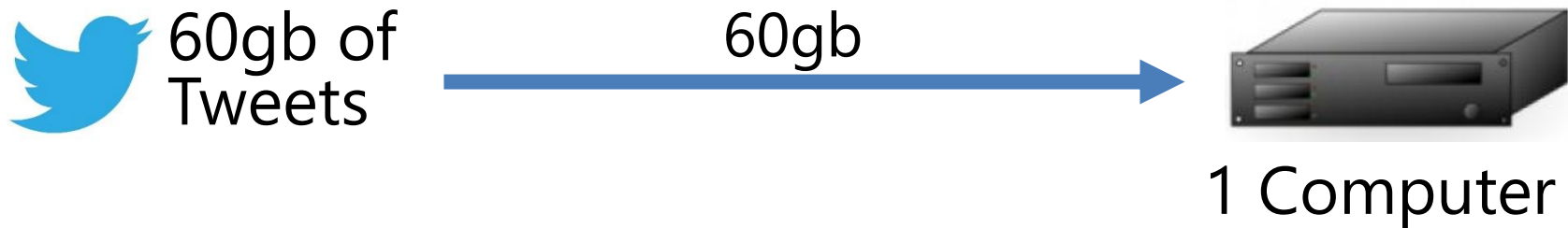
(Vanilla/Base) Hadoop

hadoop



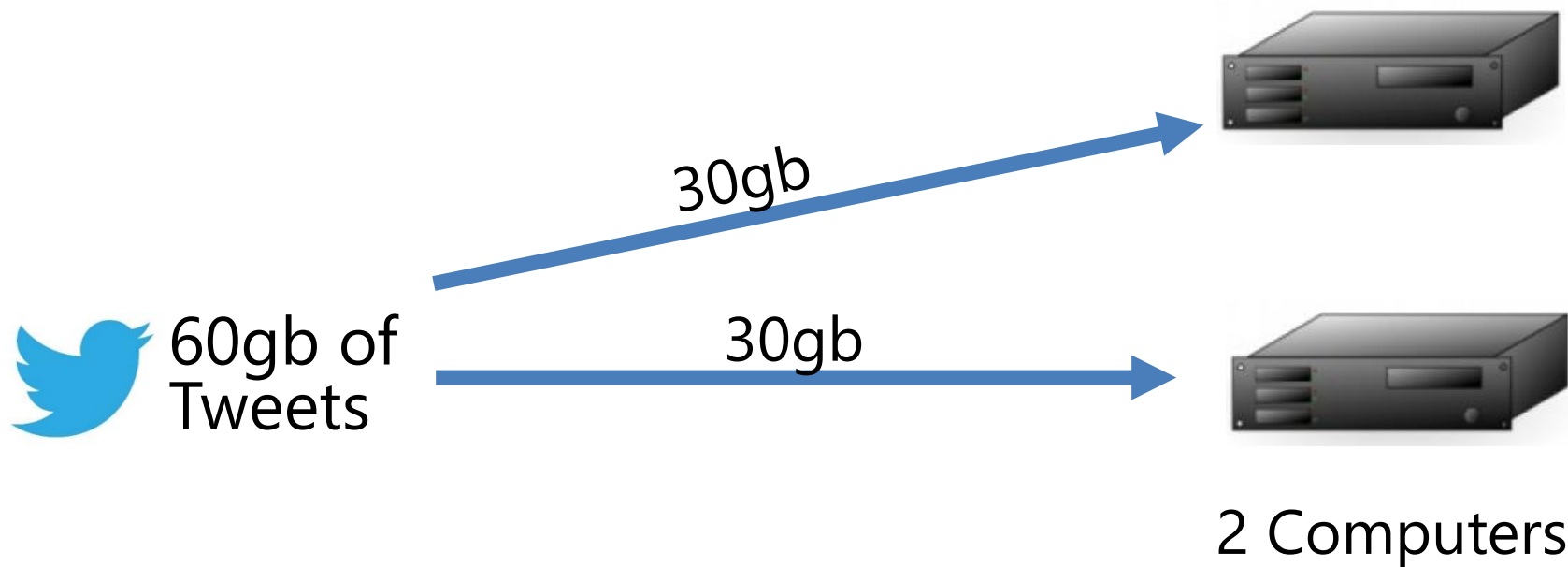
Processing engine for distributed batch processing.

HDFS & MapReduce



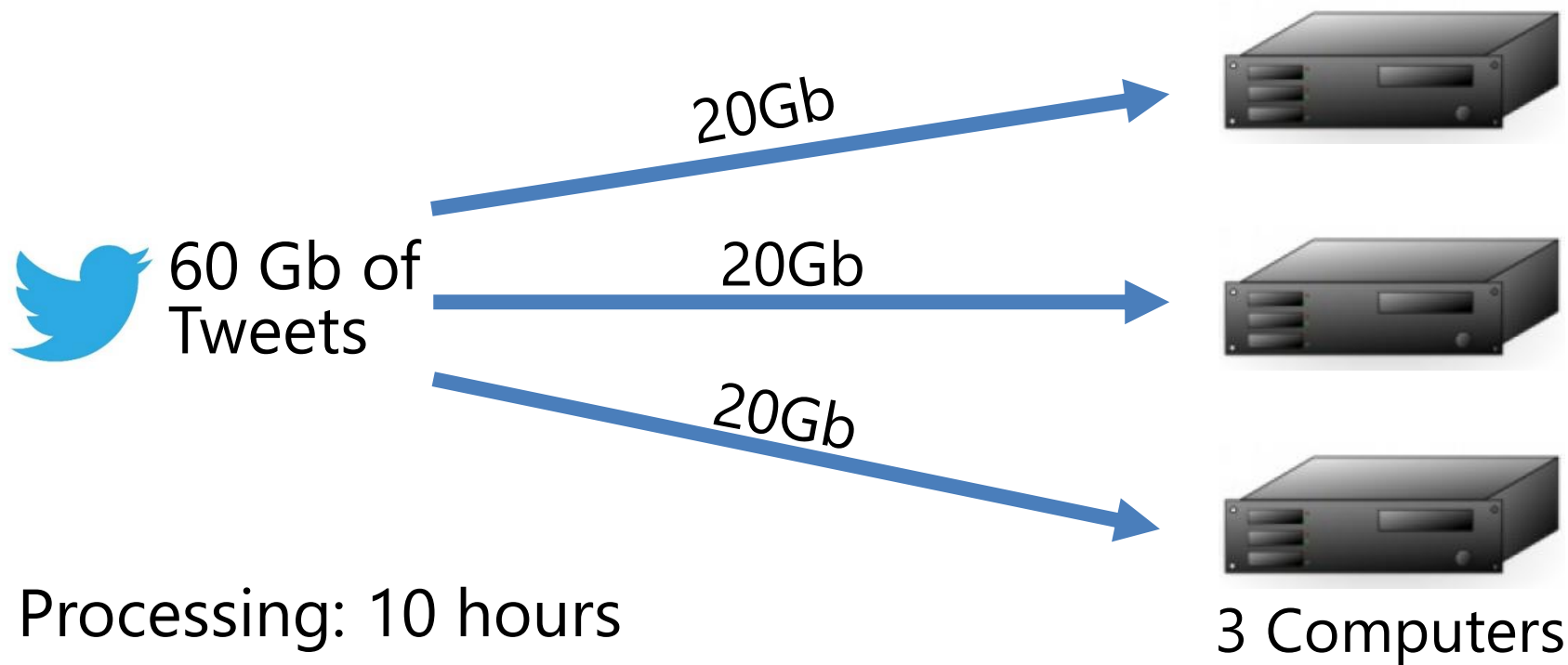
Processing: 30 hours

HDFS & MapReduce

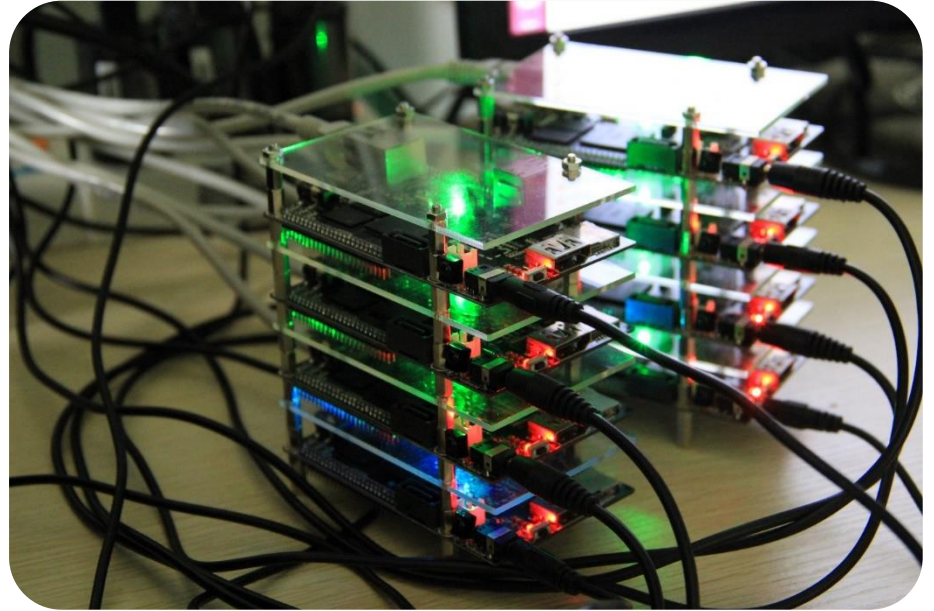
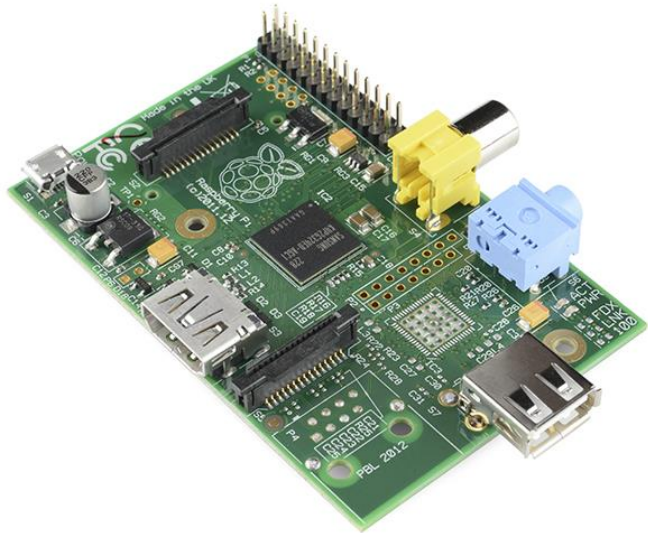


Processing: 15 hours

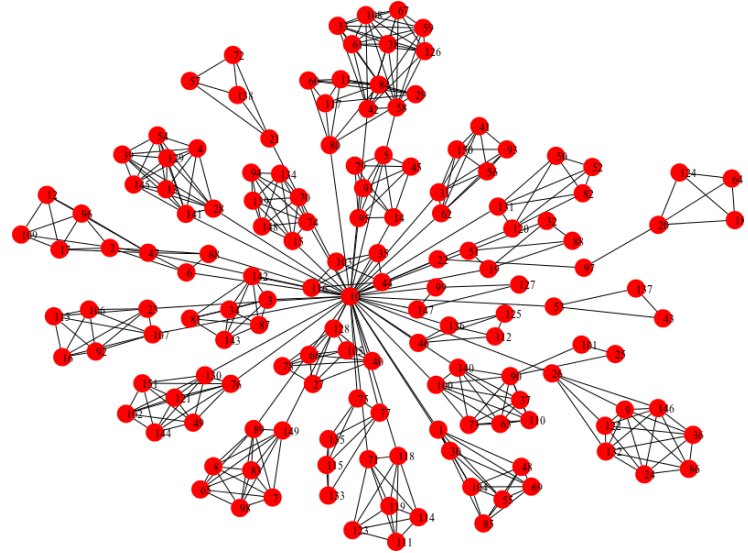
HDFS & MapReduce



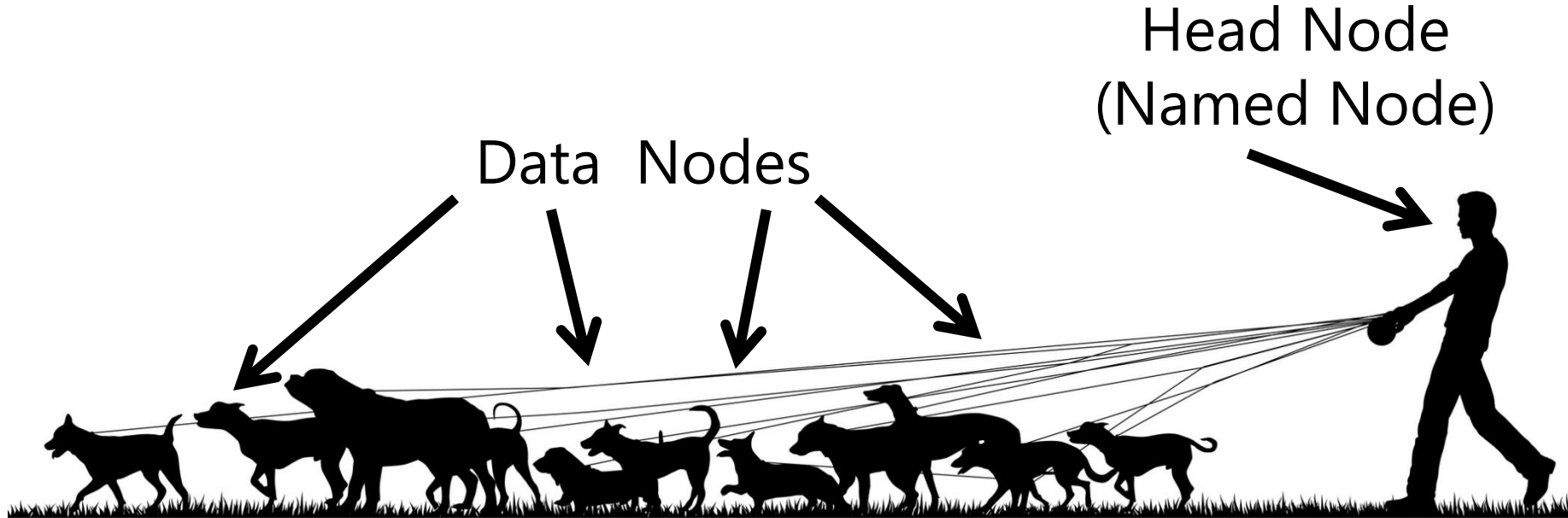
Distributed Computing



Cloud Computing



If dogs were servers...

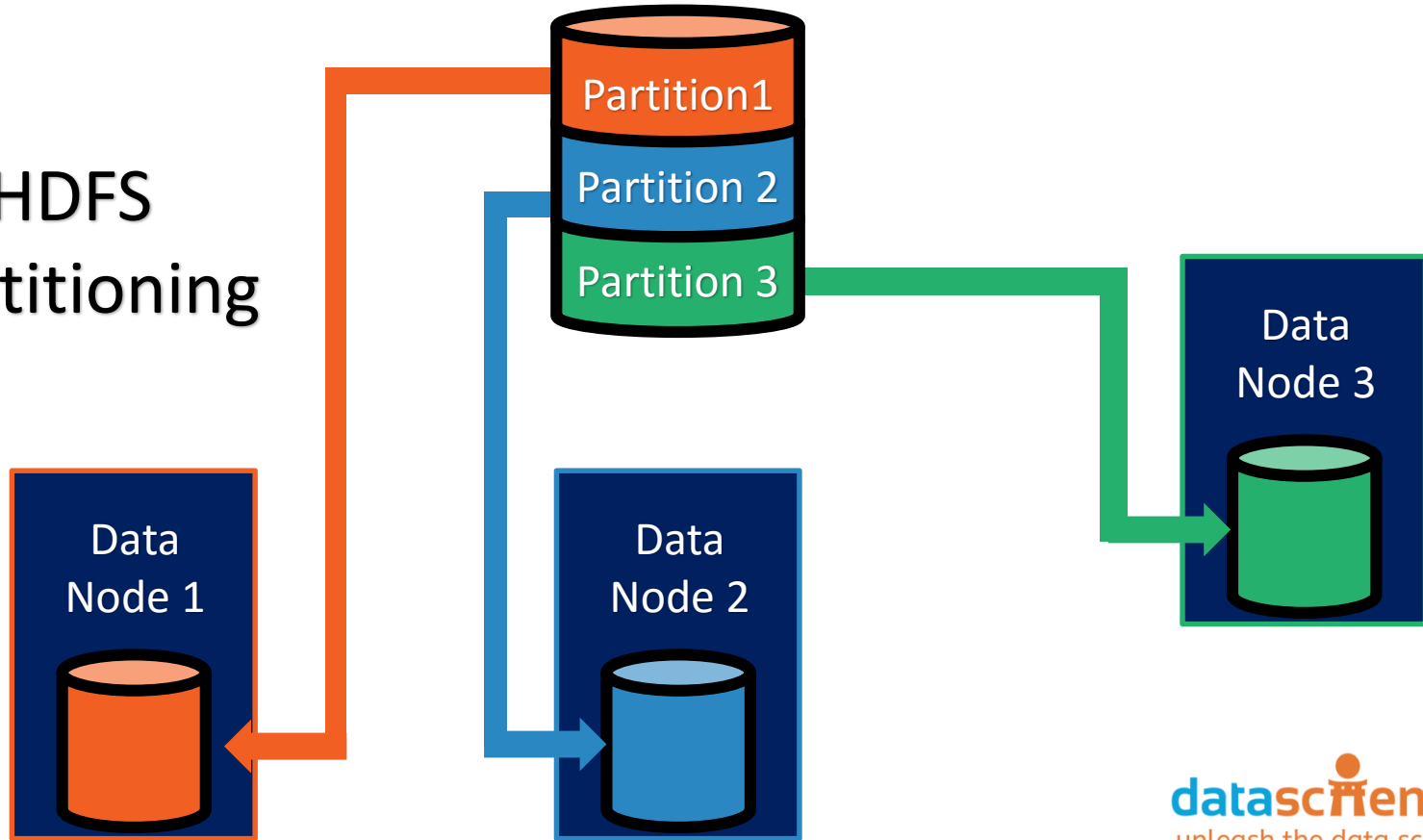


Most Cases, Linear Scaling Of Processing Power

Number of Computers	Processing Time (hours)
1	30
2	15
3	10
4	7.5
5	6
6	5
7	4.26
8	3.75
9	3.33

HDFS

HDFS Partitioning



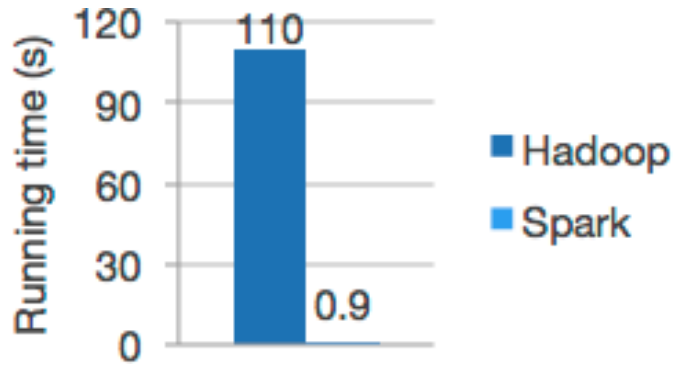
First: Distributed Compute

Hadoop

Lots of Computers

Cloud (Azure)





Microseconds
(ms)

Vs.



Nanoseconds
(ns)

- Ram + Solid State vs disk
- In-Memory: 100x times faster than Hadoop
- ~22x faster if Yarn is utilized in MR

Second: In Memory Compute

Spark

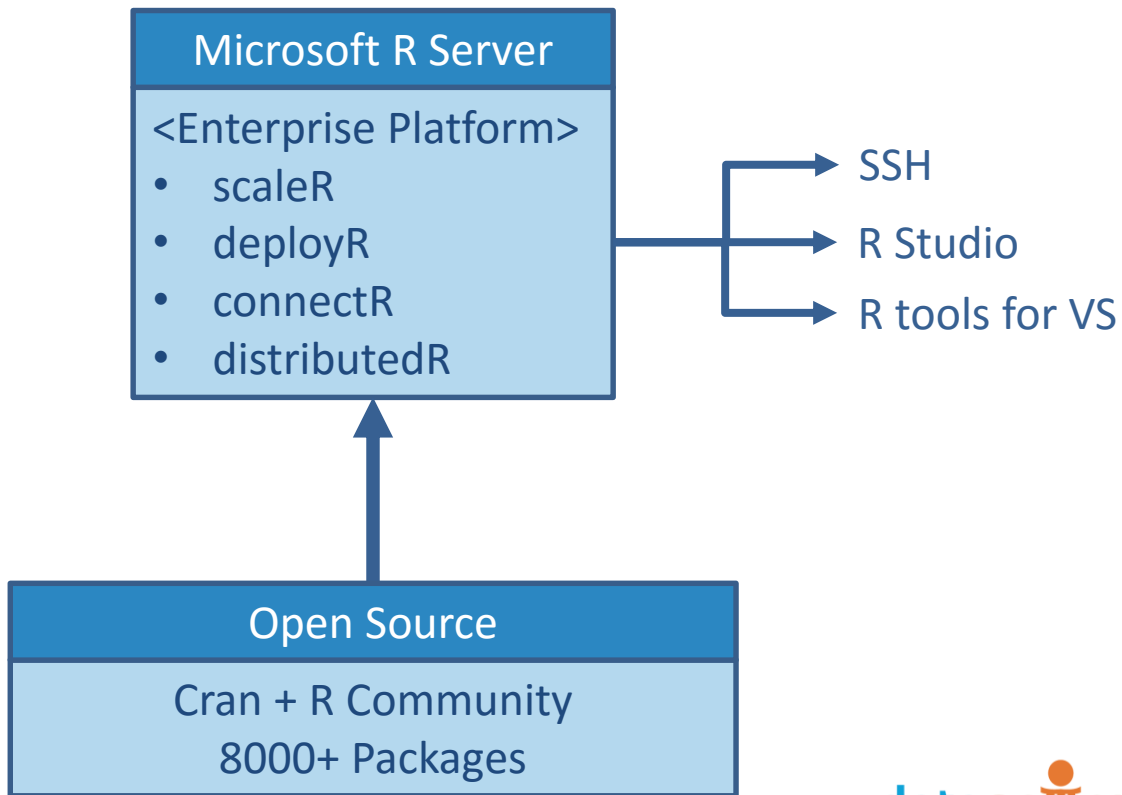
Hadoop

Lots of Computers (~2tb RAM each)

Cloud (Azure)

R Server

Can install from
`install.packages()`



R Server

Connect R

- Ingress

Scale R

- Parallelization in a single computer
- Multi-core
- Mult-threading
- Data stream (not all in memory compute)

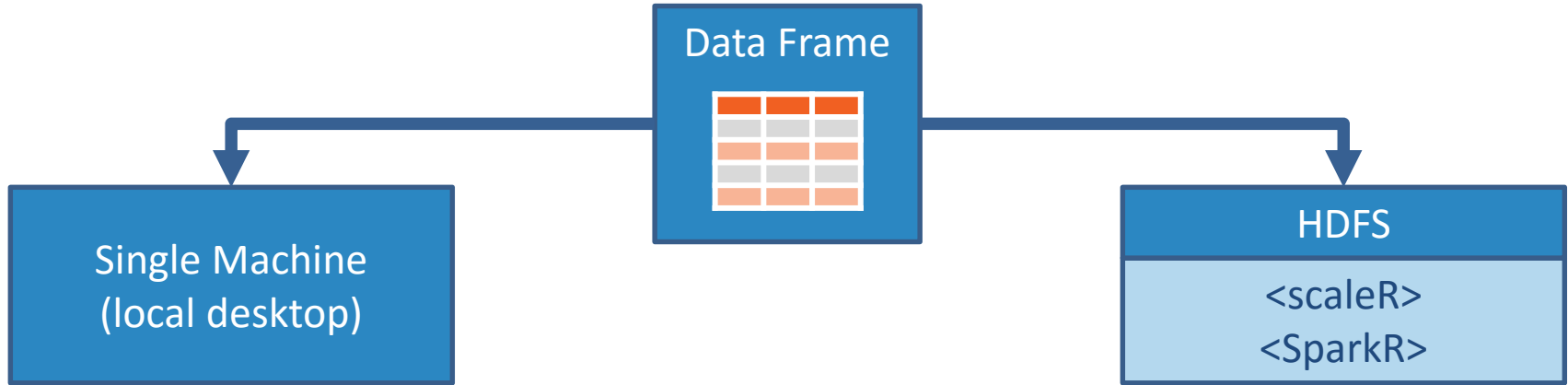
Distributed R

- Parallelization over multiple computers (HDFS)

Deploy R

- Talk to the outside world

R Server



ScaleR ← **RevolutionR** ← **Microsoft**

SparkR ← **Apache**

Third: R Context Compute

R Server

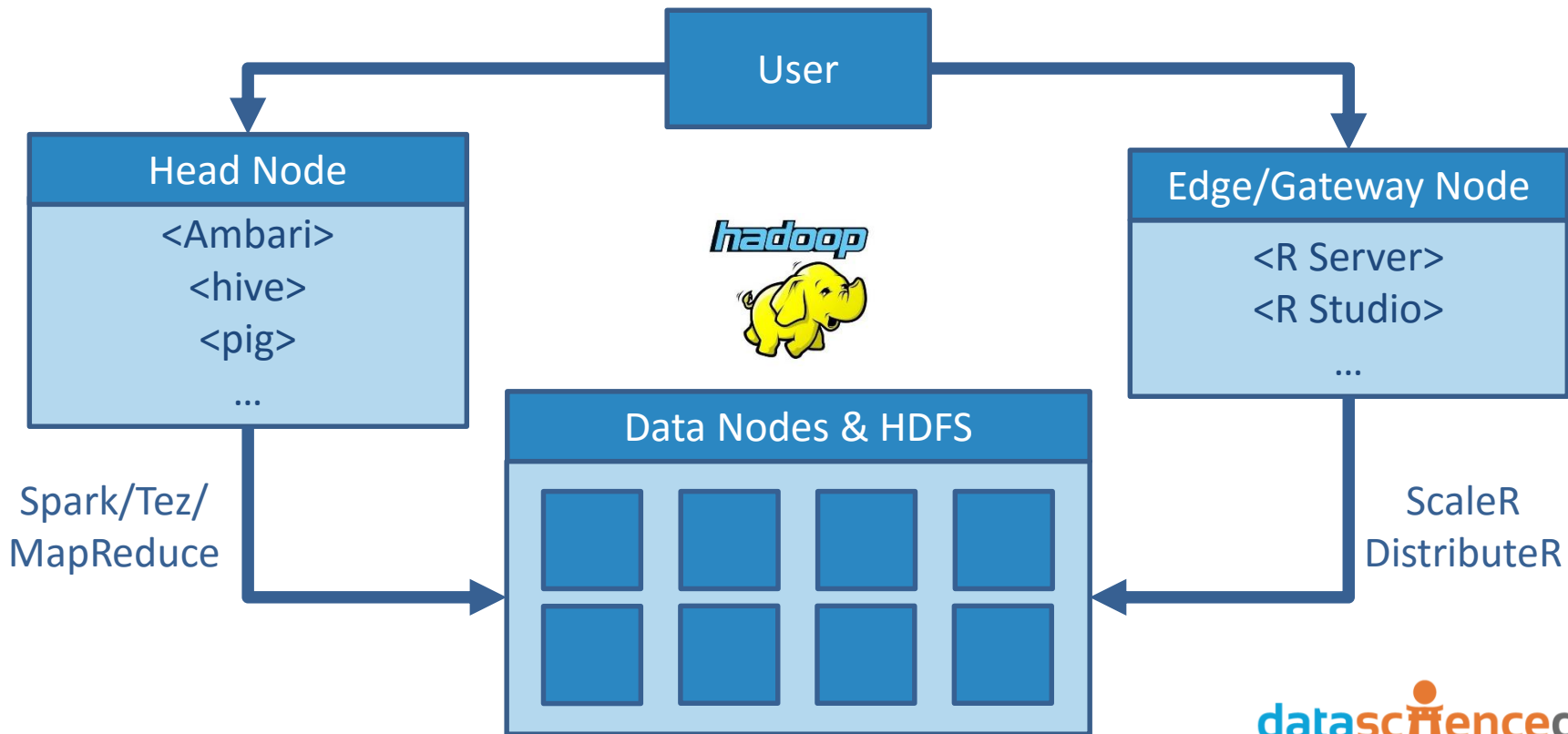
Spark

Hadoop

Lots of Computers (~2tb RAM each)

Cloud (Azure)

R Server on HDInsight



Four: IDE (for quality of life)

RStudio Client

R Server

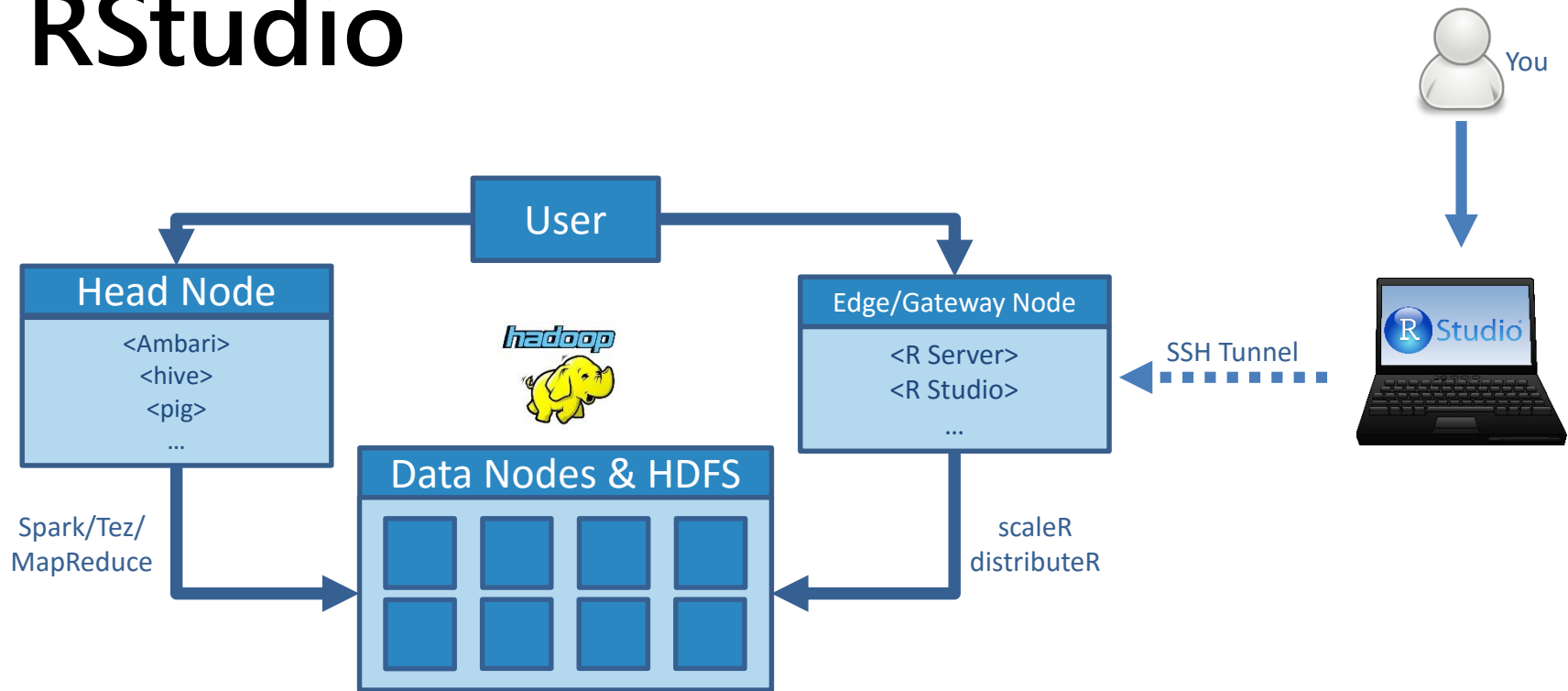
Spark

Hadoop

Lots of Computers (~2tb RAM each)

Cloud (Azure)

RStudio



Compute Context

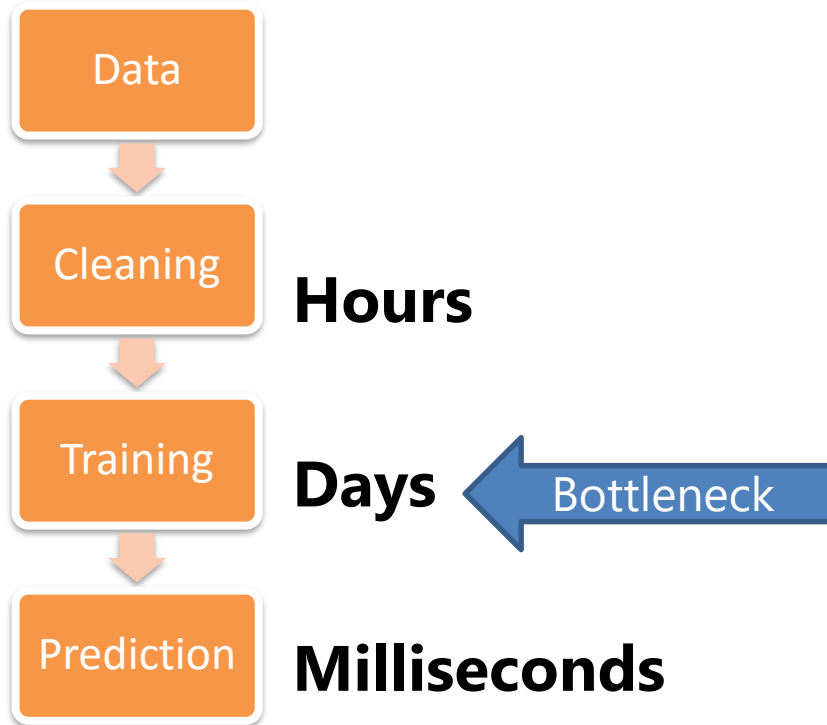
Example:

```
rxSetComputeContext("local") # Default, runs on edge node
```

```
myContext <- RxSpark()  
rxSetComputeContext(myContext)  
# Computes using Spark Engine
```

Other compute context: <https://azure.microsoft.com/en-us/documentation/articles/hdinsight-hadoop-r-server-compute-contexts/>

Processing Times - Machine Learning



- Large scale systems are only needed for training
- Phones can use models outputted by mahout to predict new data
- After a model is trained, save the model to any IO file type and reload it where you want

QUESTIONS