Big Data Architektúrák és Elemző módszerek

Gombos Gergő, Laki Sándor

források: cognitiveclass.ai

Elérhetőségek

- Előadók:
  - Gombos Gergő
  - Laki Sándor

- honlap:
  - http://ggombos.web.elte.hu
  - http://lakis.web.elte.hu
  - (http://bigdata.elte.hu)

- email:
  - ggombos@inf.elte.hu
  - lakis@inf.elte.hu

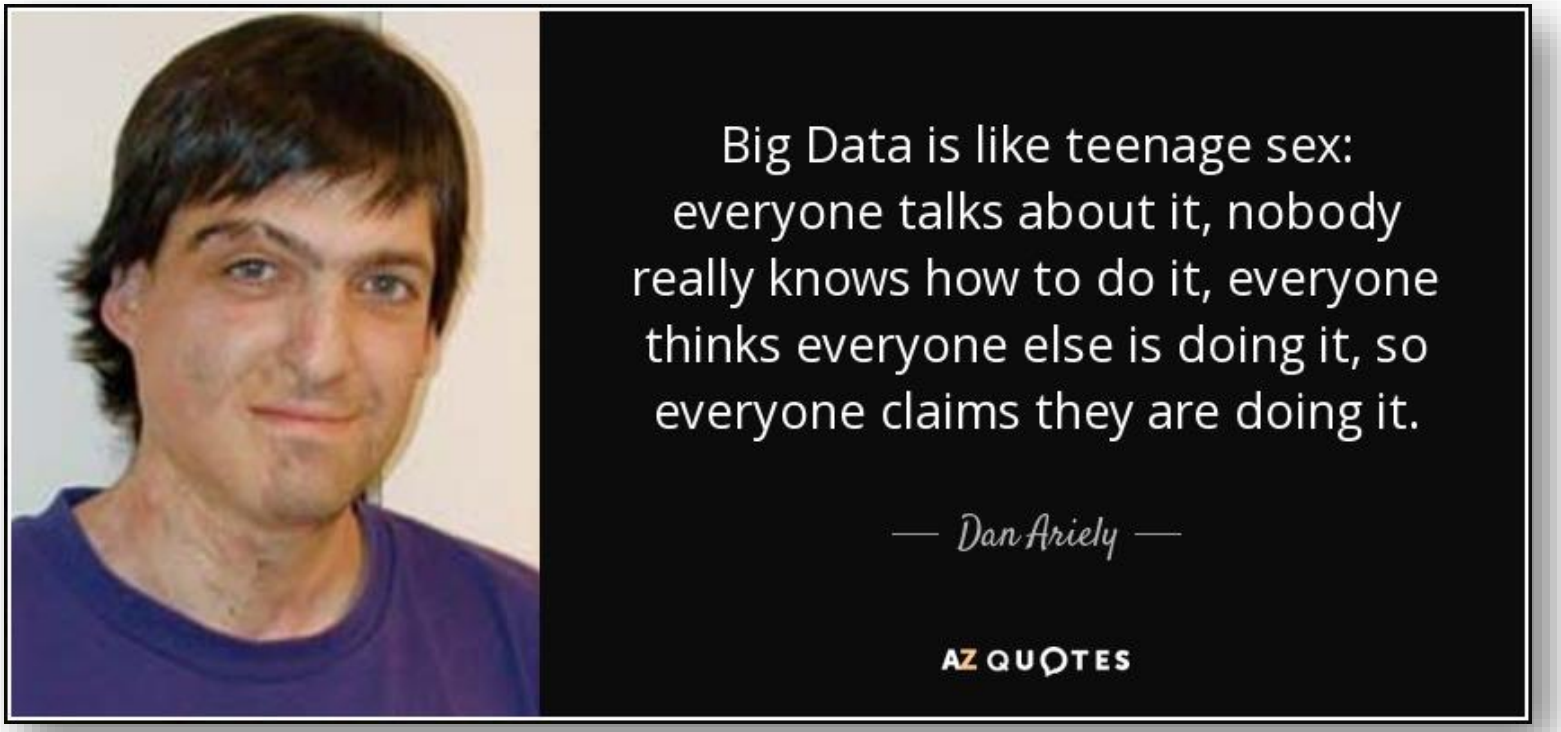
A kurzusról

- Big Data rendszerek, architektúrák megismerése, alap szintű használata
  - Hadoop, HDFS, MapReduce
  - Spark, GraphX, Stream
- Data Science módszerek megismerése, algoritmusok használata
  - pandas, dataframe, numpy, jupyter
  - döntési fák, osztályozás, klaszterezés, dimenzió csökkentés
  - vizualizáció, matplotlib

A követelményekről

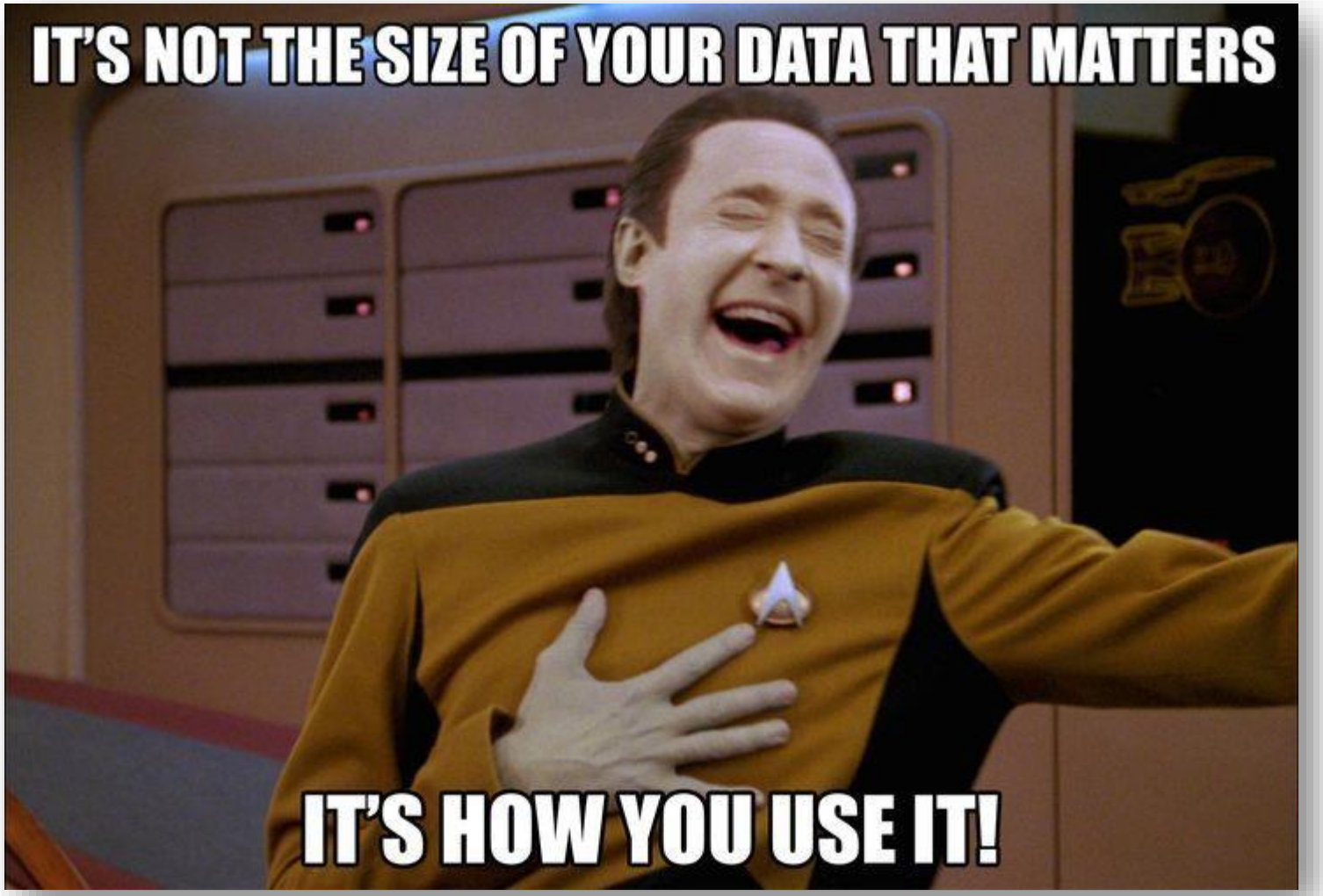
- Előadás:
  - Vizsga a félév végén
- Gyakorlat:
  - 1 beadandó: Hadoop MapReduce
  - (Házifeladatok)
  - 2 ZH a félév során:
    - Spark batch elemzés
    - Adatelemzés, vizualizáció

Big Data

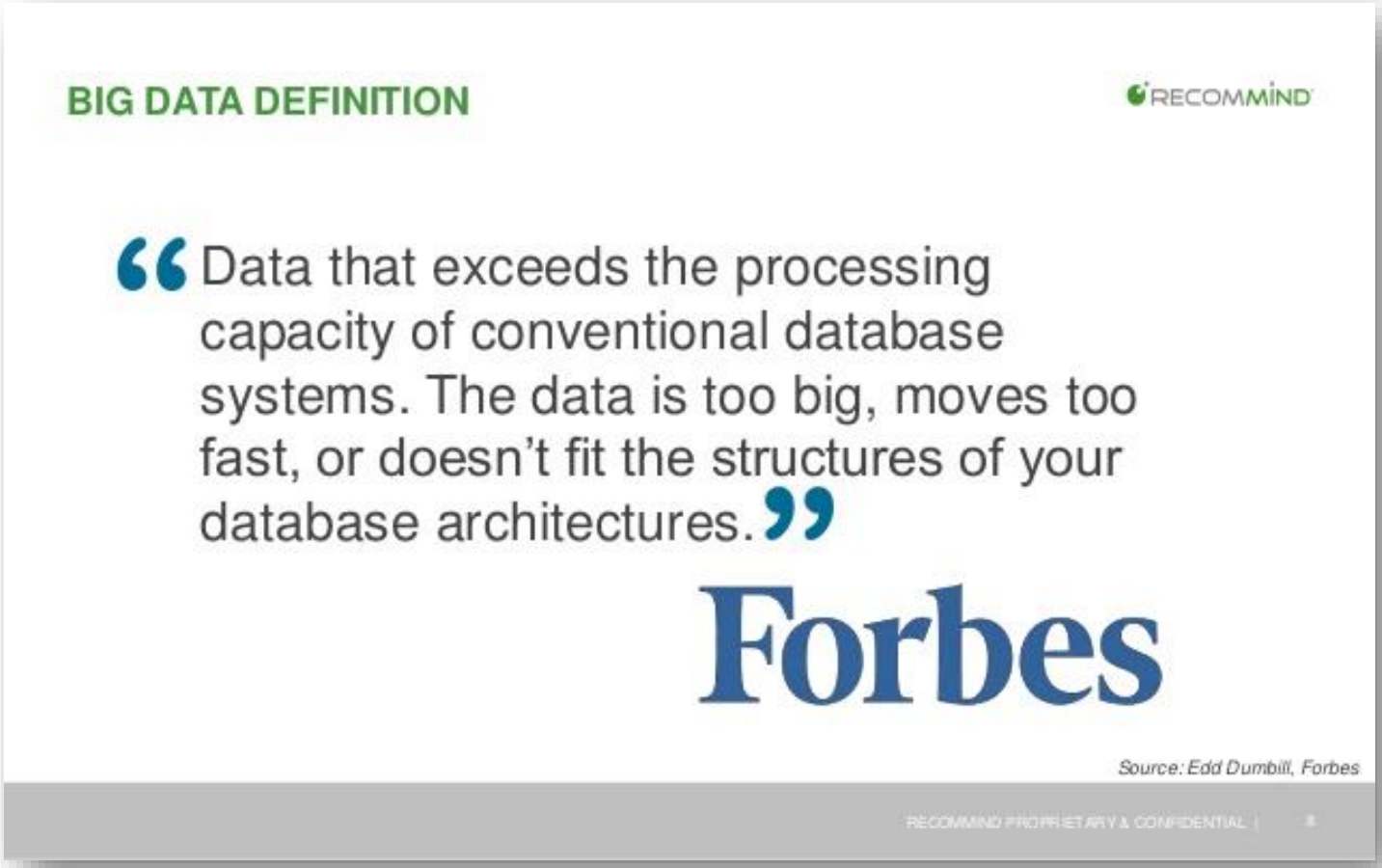

Big Data is like teenage sex: everyone talks about it, nobody really knows how to do it, everyone thinks everyone else is doing it, so everyone claims they are doing it.
— Dan Ariely —
AZ QUOTES

A világ megváltozott!

Nem a méret a lényeg!



Mi az a Big Data?



The FOUR V's of Big Data

From traffic patterns and music downloads to web history and medical records, data is recorded, stored, and analyzed to enable the technology and services that the world relies on every day. But what exactly is big data, and how can these massive amounts of data be used?

As a leader in the sector, IBM data scientists break big data into four dimensions: **Volume, Velocity, Variety and Veracity**

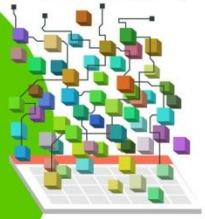
Depending on the industry and organization, big data encompasses information from multiple internal and external sources such as transactions, social media, enterprise content, sensors and mobile devices. Companies can leverage data to adapt their products and services to better meet customer needs, optimize operations and infrastructure, and find new sources of revenue.

By 2015
4.4 MILLION IT JOBS
will be created globally to support big data, with 1.9 million in the United States

Volume
SCALE OF DATA

40 ZETTABYTES
[43 TRILLION GIGABYTES]
of data will be created by 2020, an increase of 300 times from 2005

It's estimated that
2.5 QUINTILLION BYTES
[2.3 TRILLION GIGABYTES]
of data are created each day



6 BILLION PEOPLE
have cell phones

Most companies in the U.S. have at least
100 TERABYTES
[100,000 GIGABYTES]
of data stored

Variety
DIFFERENT FORMS OF DATA

As of 2011, the global size of data in healthcare was estimated to be
150 EXABYTES
[161 BILLION GIGABYTES]

30 BILLION PIECES OF CONTENT
are shared on Facebook every month

By 2014, it's anticipated there will be
420 MILLION WEARABLE, WIRELESS HEALTH MONITORS

4 BILLION+ HOURS OF VIDEO
are watched on YouTube each month

400 MILLION TWEETS
are sent per day by about 200 million monthly active users

Velocity
ANALYSIS OF STREAMING DATA

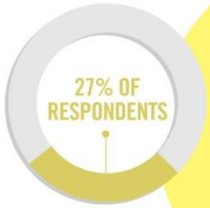
The New York Stock Exchange captures
1 TB OF TRADE INFORMATION
during each trading session

Modern cars have close to
100 SENSORS
that monitor items such as fuel level and tire pressure

By 2016, it is projected there will be
18.9 BILLION NETWORK CONNECTIONS
– almost 2.5 connections per person on earth

Veracity
UNCERTAINTY OF DATA

1 IN 3 BUSINESS LEADERS
don't trust the information they use to make decisions



in one survey were unsure of how much of their data was inaccurate

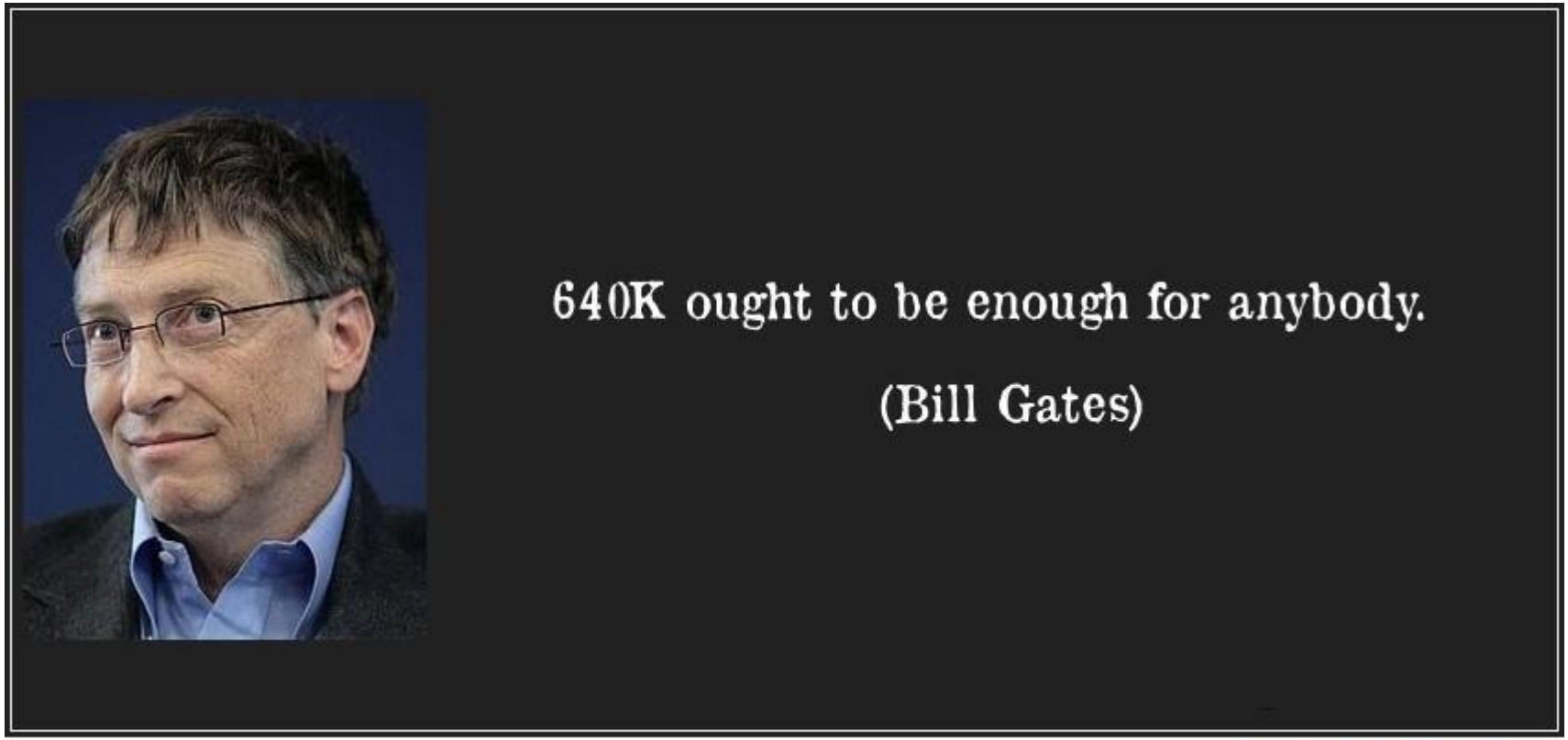
Poor data quality costs the US economy around
$3.1 TRILLION A YEAR

Sources: McKinsey Global Institute, Twitter, Cisco, Gartner, EMC, SAS, IBM, MEPTEC, QAS

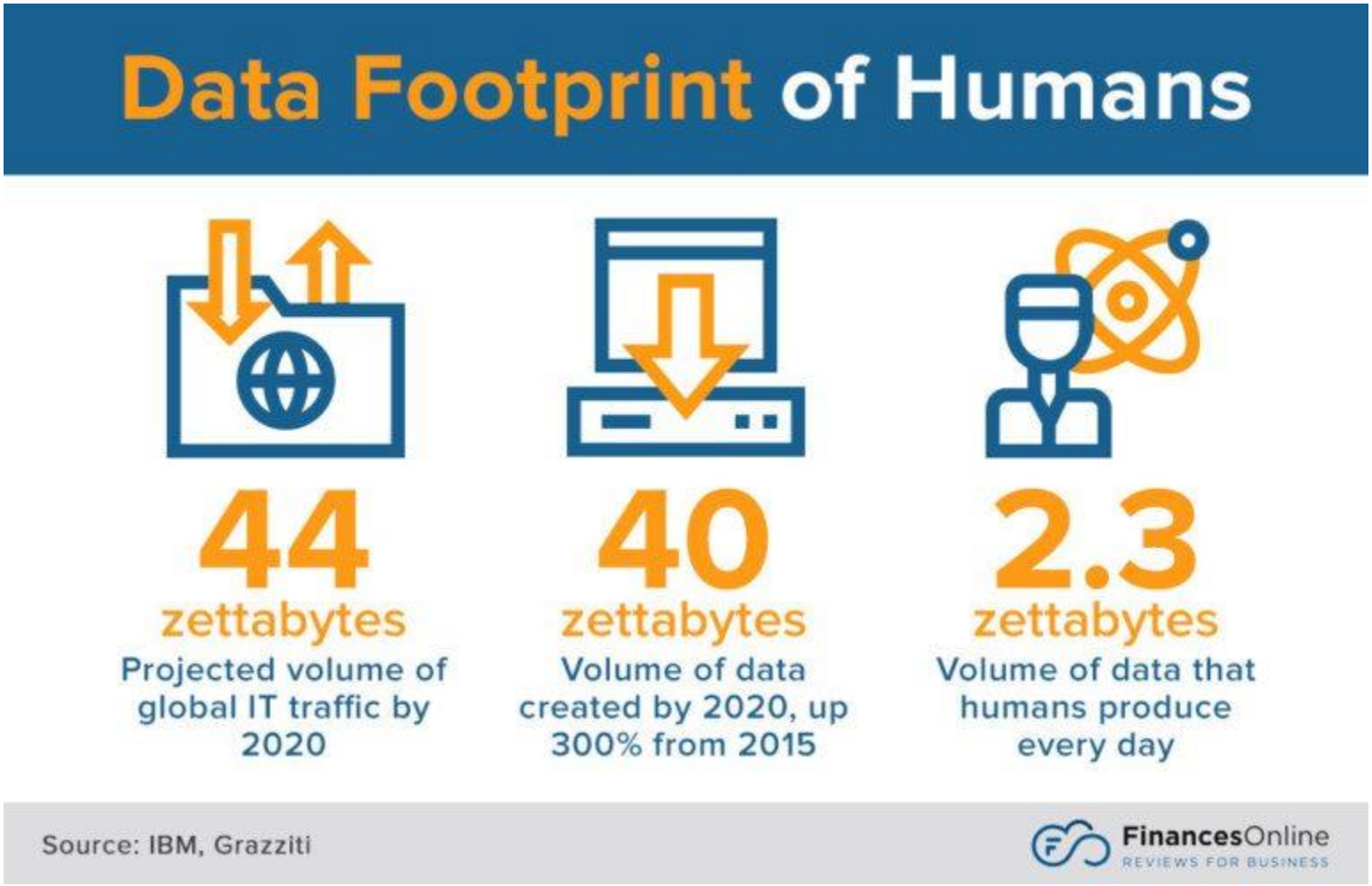
IBM

Volume - 640K Big Data?

1981



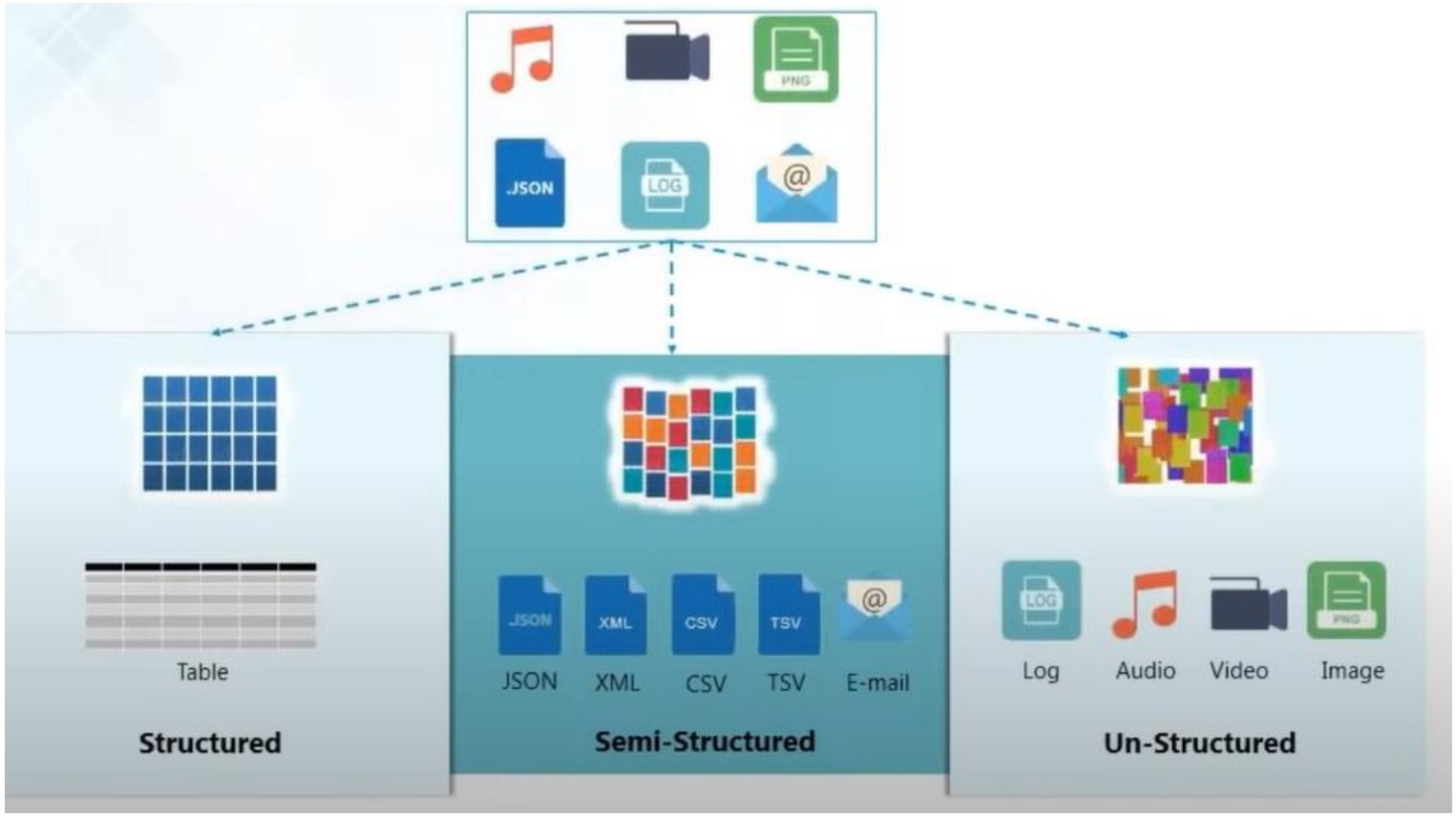
Volume


Data Footprint of Humans
44
zettabytes
Projected volume of global IT traffic by 2020
40
zettabytes
Volume of data created by 2020, up 300% from 2015
2.3
zettabytes
Volume of data that humans produce every day
Source: IBM, Grazziti
FinancesOnline
REVIEWS FOR BUSINESS

Velocity

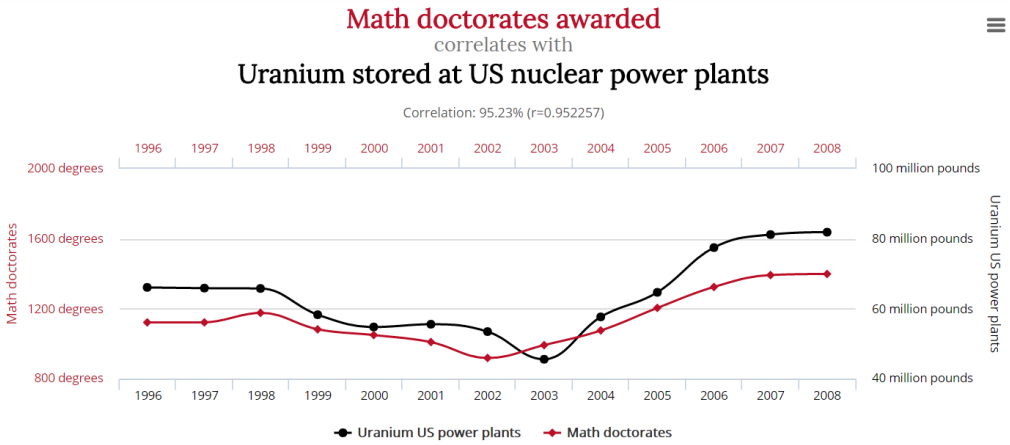

THE INTERNET IN 2023 EVERY MINUTE
22,831
visits to ChatGPT
241.2M
emails sent
18.8M
text messages sent
2.4M
Google searches
694,000
video hours viewed
347,222
tweets
3.47M
snaps created
10.4M
viewing minutes
11,035
fake accounts removed
6.3M
total Zoom meeting minutes
34,247
Slack messages
11,834
chats on Microsoft Teams
6.94M
emoji sent
3.02M
photos created with smartphones
271,309
IOS & Android app downloads
60
SECONDS
Created by: eDiscovery Today & LTMG

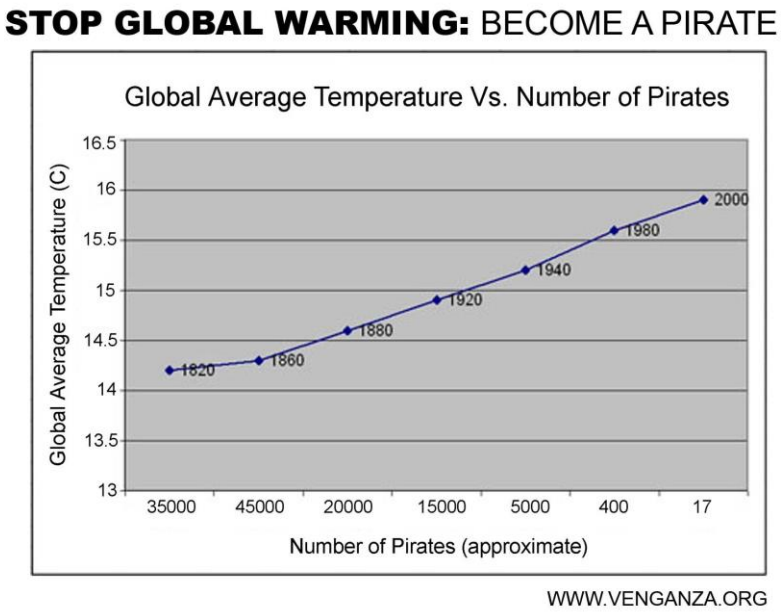
Variety (types of Big Data)

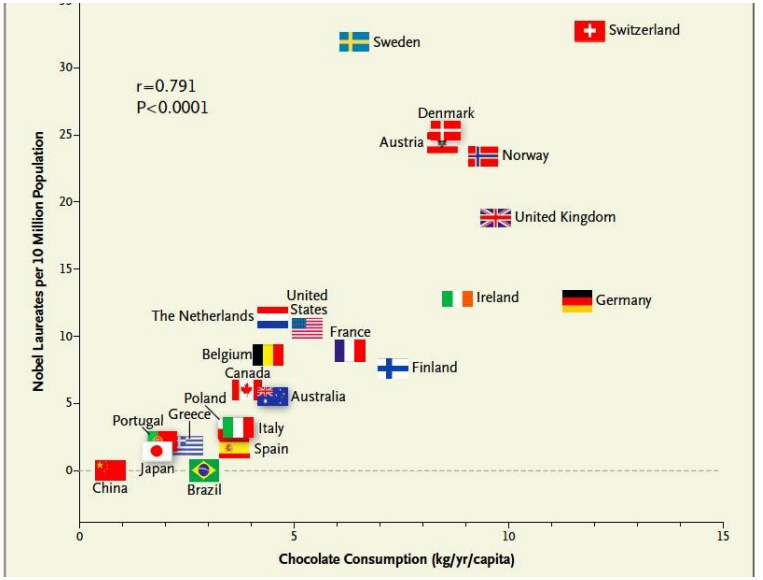


Veracity

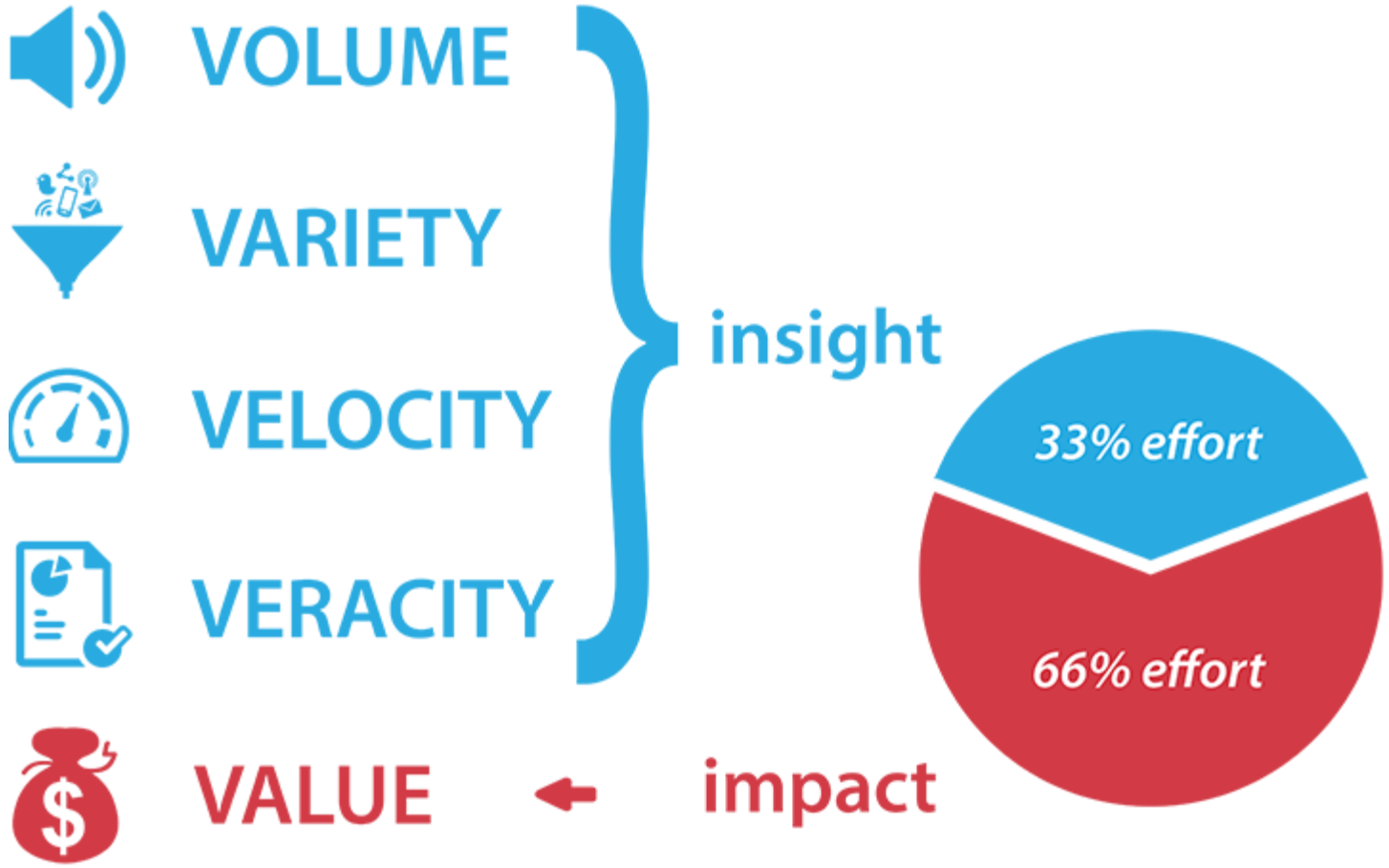
Tényleg ez az összefüggés?







Big Data 5V


VOLUME
VARIETY
VELOCITY
VERACITY
insight
VALUE
impact
33% effort
66% effort

Netflix innovation relies on

Personalized Video Ranking

Video-Video Similarity Ranker

Project Cost Predicting Algorithm

Artwork Visual Analysis

Continue Watching Ranker

Trending Now Ranker

The impact of Big Data on businesses and people

Virtual personal assistants

Google Now

Siri knows what users mean when they ask her questions, she knows where they are, what time they are talking about and can use this information to look for restaurants of a particular type of food and check whether there reservations are available.

Google Now makes recommendations before users ask for them, especially when it is linked up to the user's calendar and location sensing is enabled on the user's phone. Google Now knows where the user is and where he/she needs to be, it can tell users about things like traffic or the weather before you even ask for it.

The V's Evolution of Big Data

9V's (Owais,2016)

Volume
Variety
Velocity
Veracity
Value
Visualization
Variability
Validity
Volatility

10V's (Data Science Central)

Volume
Variety
Velocity
Veracity
Value
Variability
Validity
Venue
Vocabulary
Vagueness

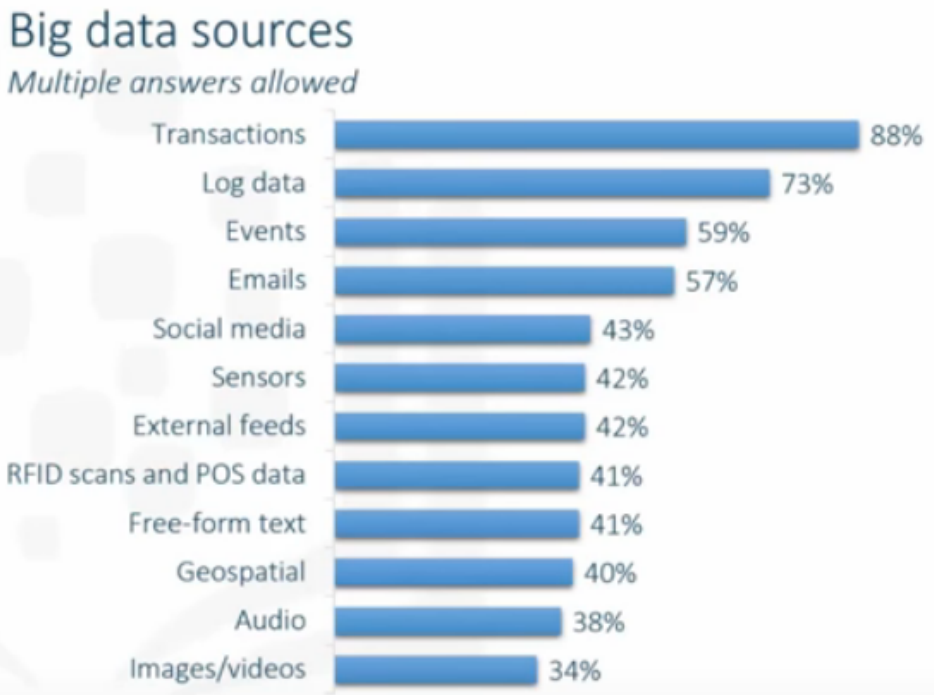
17V's (Panimalar,2017)

Volume
Variety
Velocity
Veracity
Value
Variability
Validity
Venue
Vocabulary
Vagueness
Volatility
Visualization
Viscosity
Virality
Verbosity
Voluntariness
Versality

Where is all the data coming from?

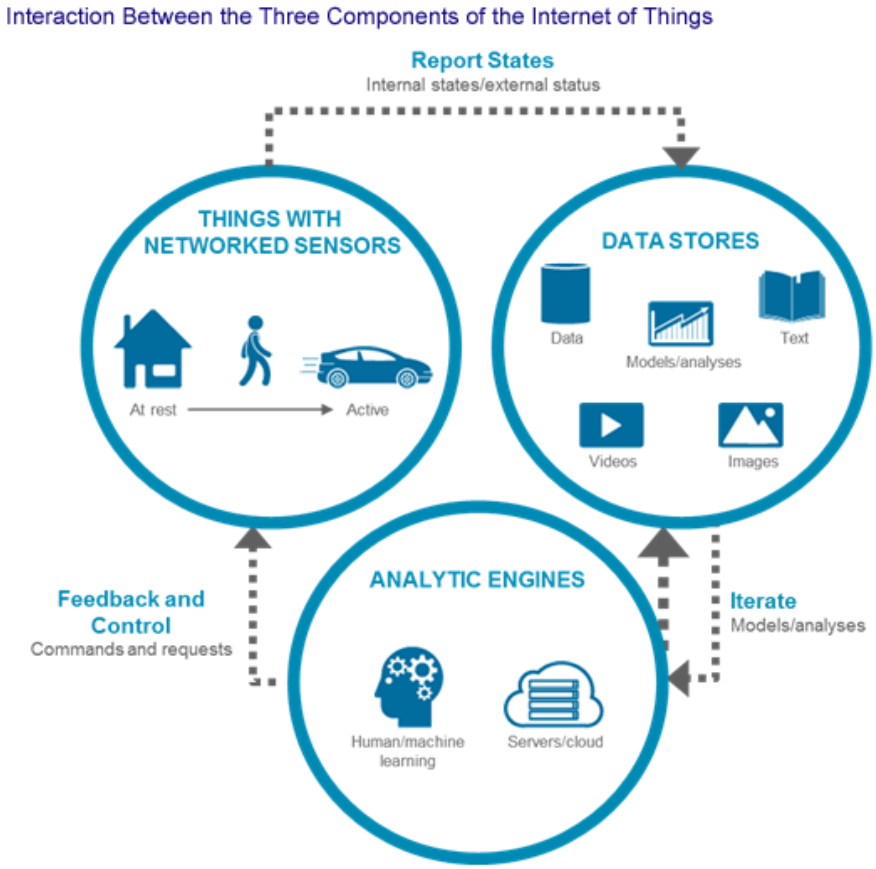
There are three major sources of Big Data:

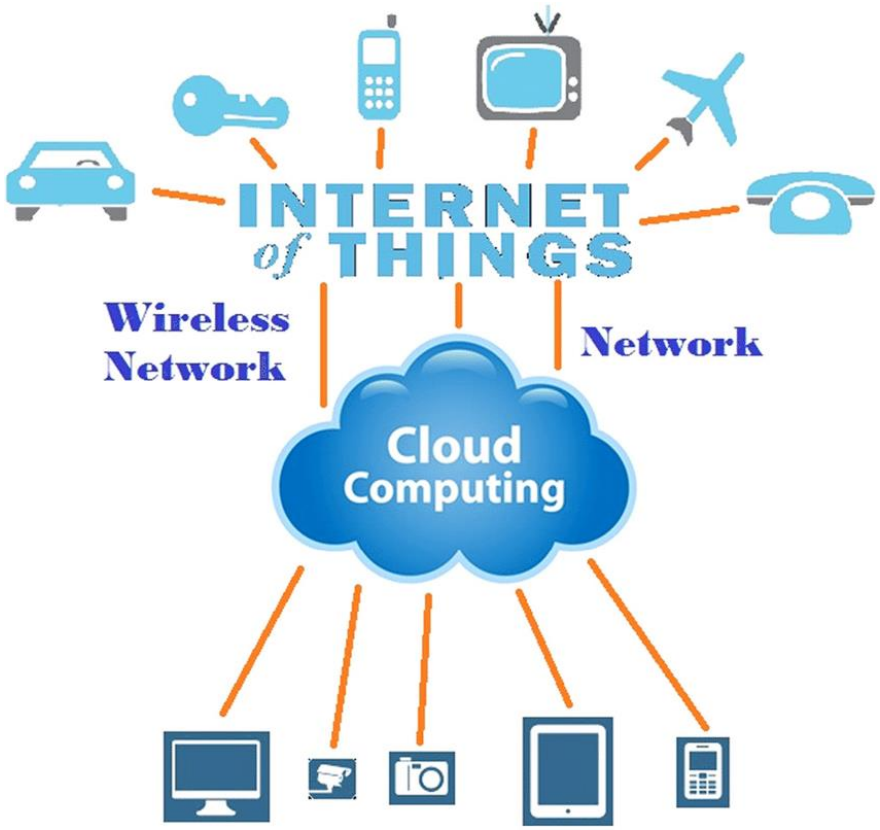
- People-generated data
- Machine-generated data
- Business-generated data



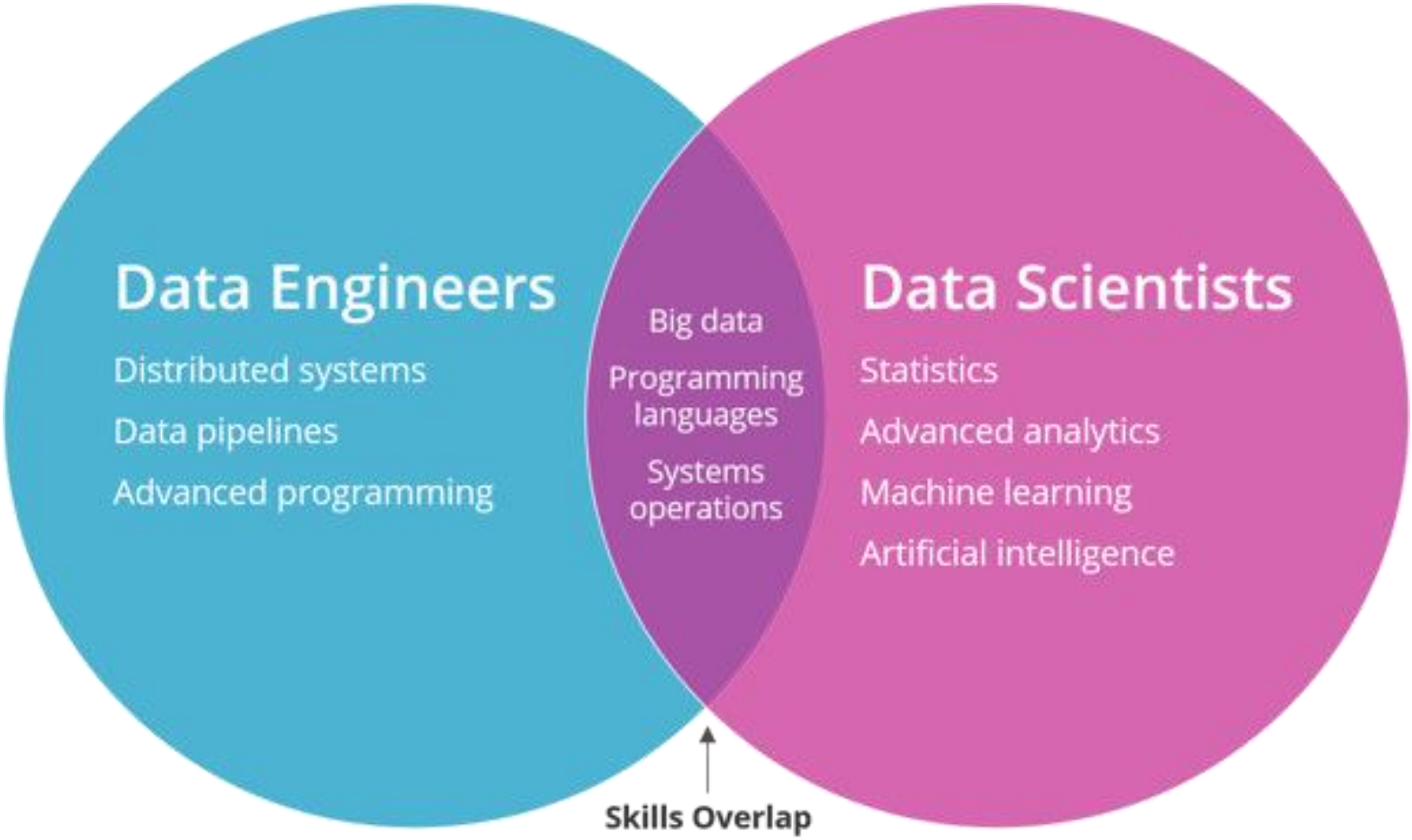
Source: IBM, Said School of Business, 2012

IoT – Internet of Things





Big Data Skillek



Data Engineers

Motiváció – Hagyományos rendszerek

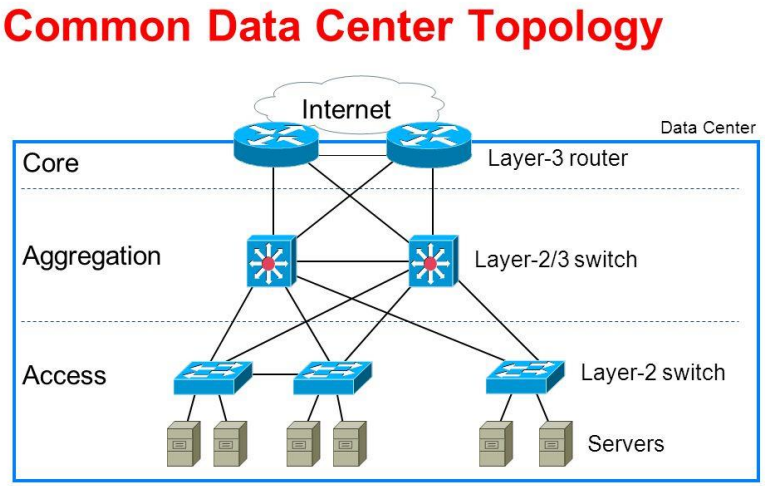
- Hagyományos rendszerek
  - Tipikusan egygépes rendszerek
  - Relatív „kis” méretű statikus adatok
  - Komplex feladatok elvégzése ezeken az adatokon működik
  - Mi történik ha nem fér el az adat? (Google példa)
    - 10 milliárd weblapot dolgoz fel naponta (átlagos méret 20 KB)
      10 milliárd*20 kB = 200TB
    - Lemez olvasási sebesség: 50MB/s => 4 milliárd mp ~= 46+ nap

- Számítási kapacitás növelése
  - Gyorsabb processzor
  - Több memória

Big Data Architektúrák

- Big Data Architektúra
  - Folyamatos adatfolyamok (napi > 1TB)
  - Több gép klaszterben
  - Az adatok elosztva tárolódnak (pl.: HDFS)
  - Számítások párhuzamosítva (MapReduce)

- Kapacitás növelés
  - Új gépek kapcsolása a klaszterhez



Technológiák (nem teljes)

Impala

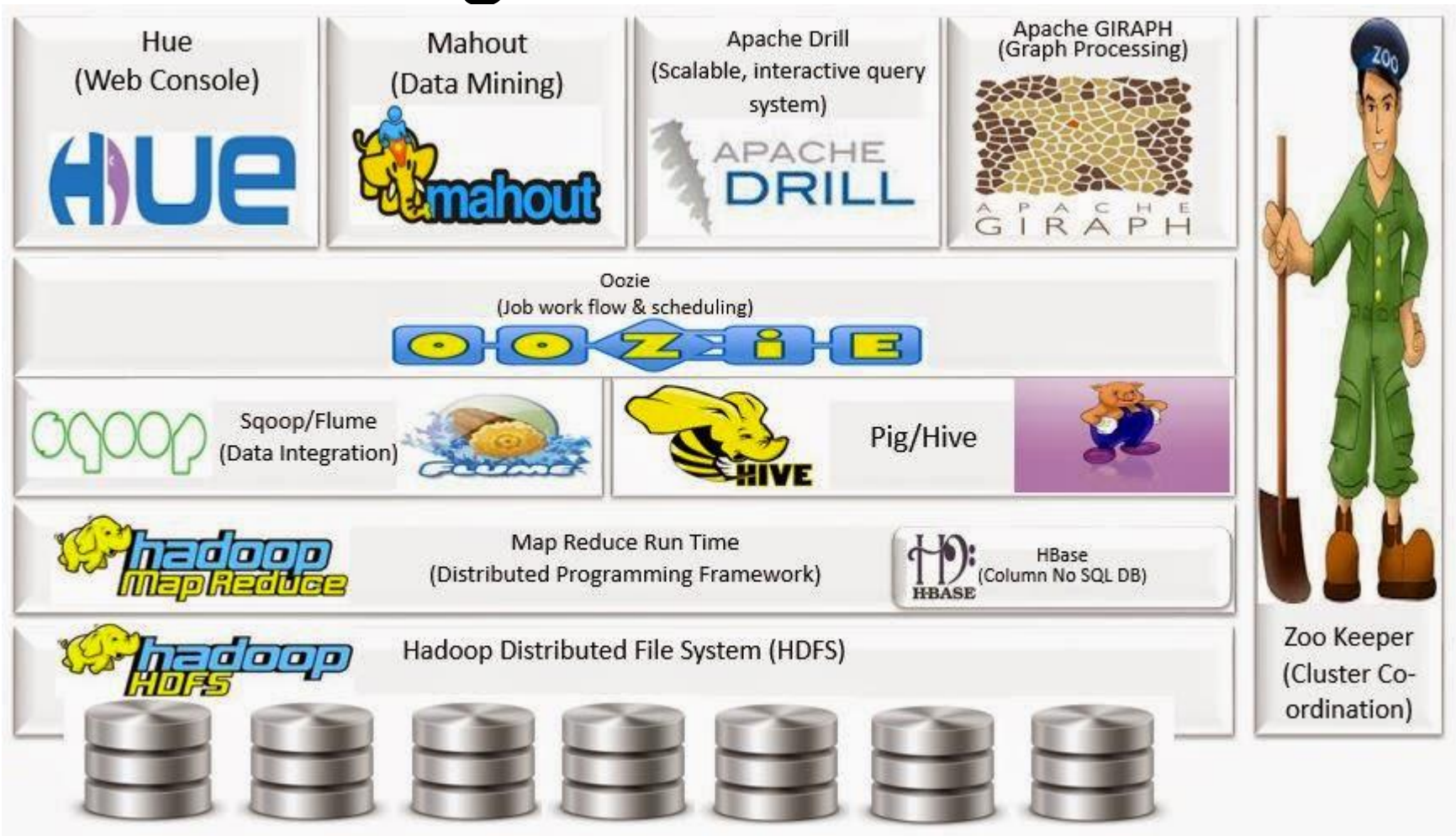
Pig

Zookeeper

https://hadoopecosystemtable.github.io/

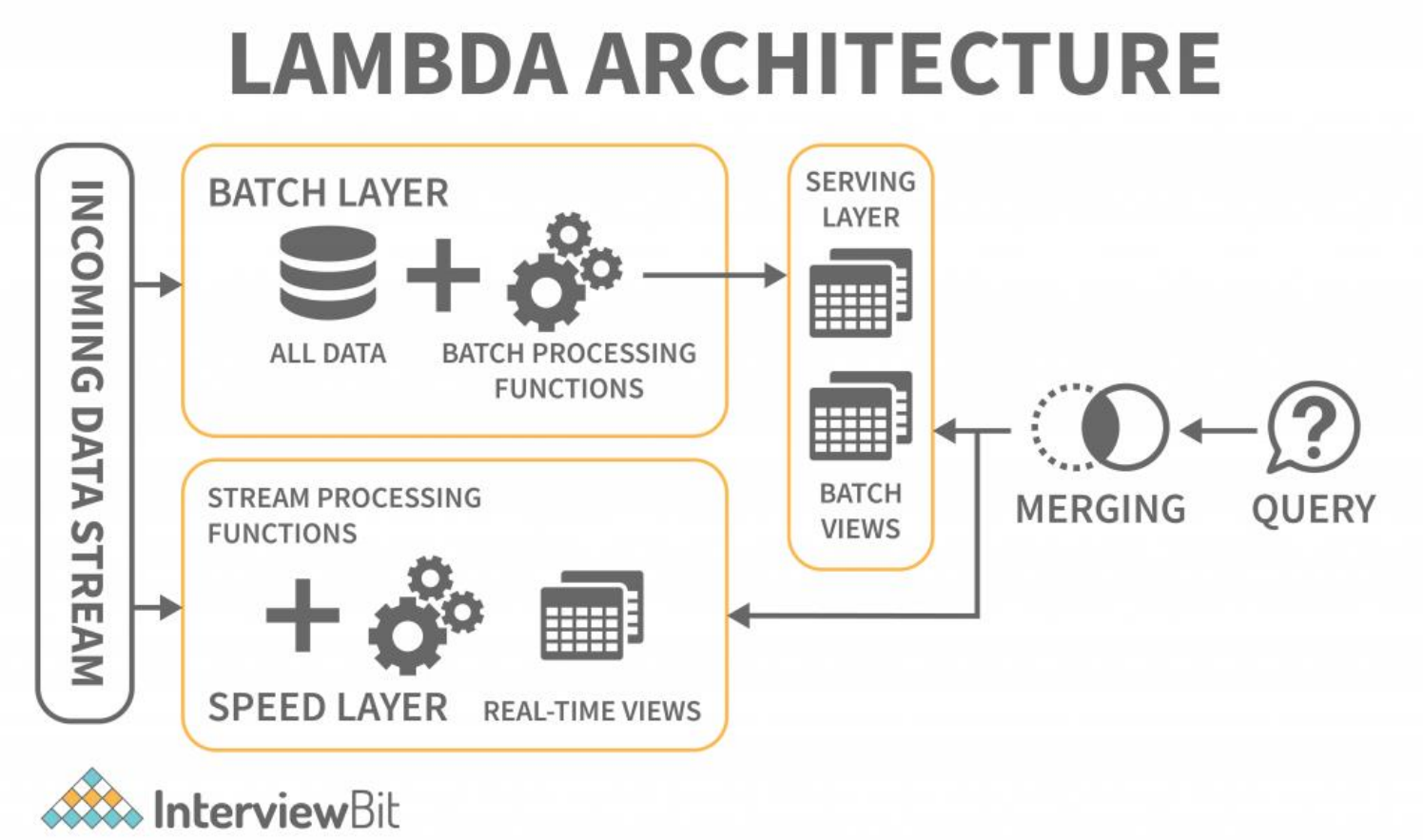
Kik használják?

Példa Big Data Architektúrára



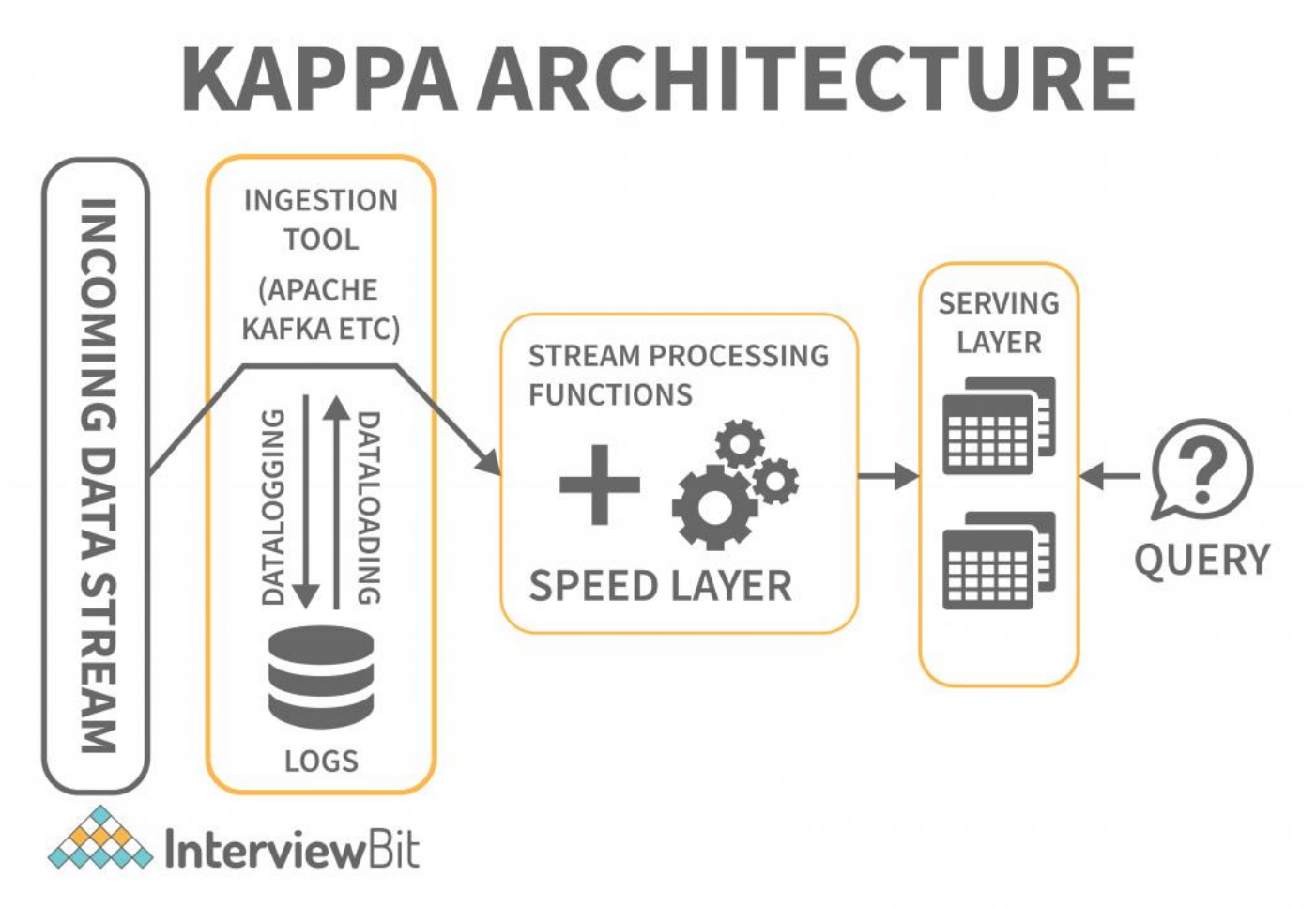
https://hadoopecosystemtable.github.io/

Big Data Architektúra



https://www.interviewbit.com/blog/big-data-architecture/

Big Data Architektúra



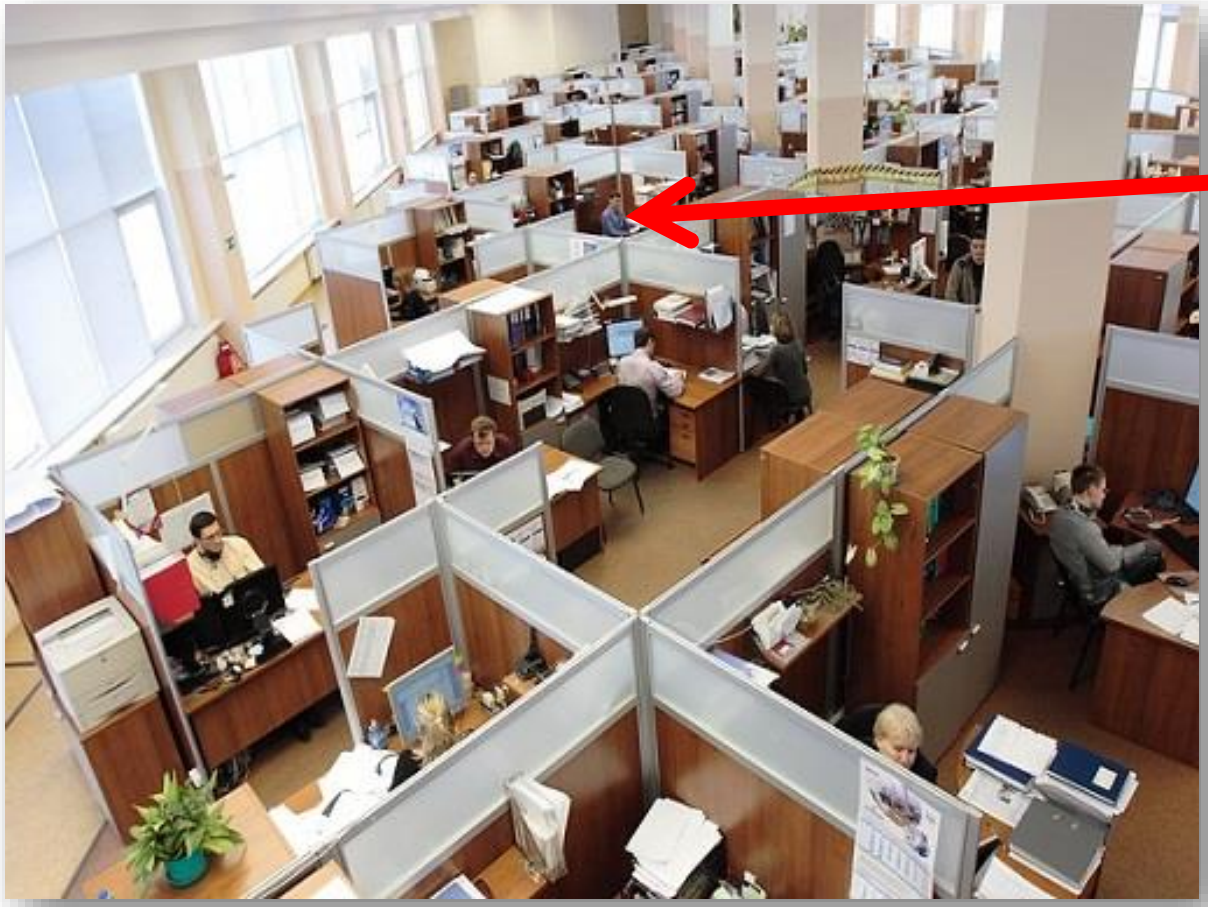
https://www.interviewbit.com/blog/big-data-architecture/

Disztribúciók

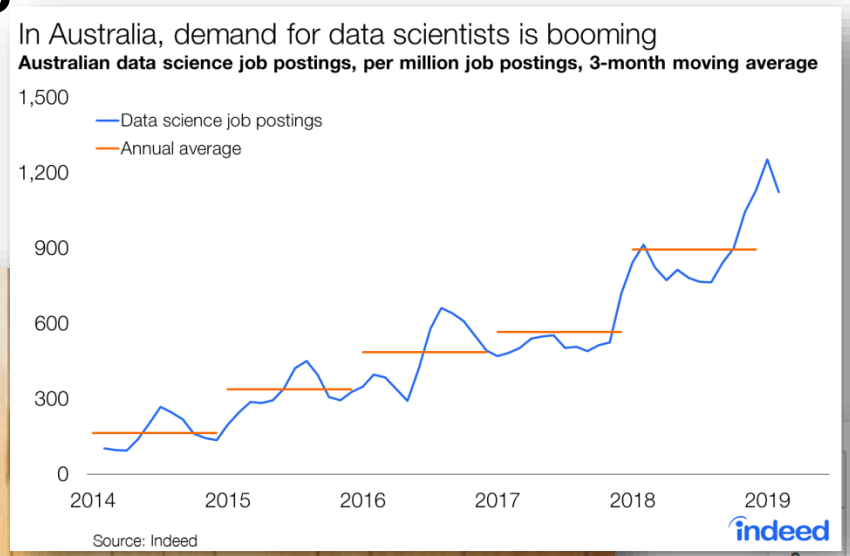
- Apache
  - Az elemek egyesével érhetőek el.
  - Opensource
  - Telepítő: Ambari
- Cloudera, Hortonworks
  - Komplett telepítő rendszer
  - Support (fizetős)
  - Trainingek (fizetős)

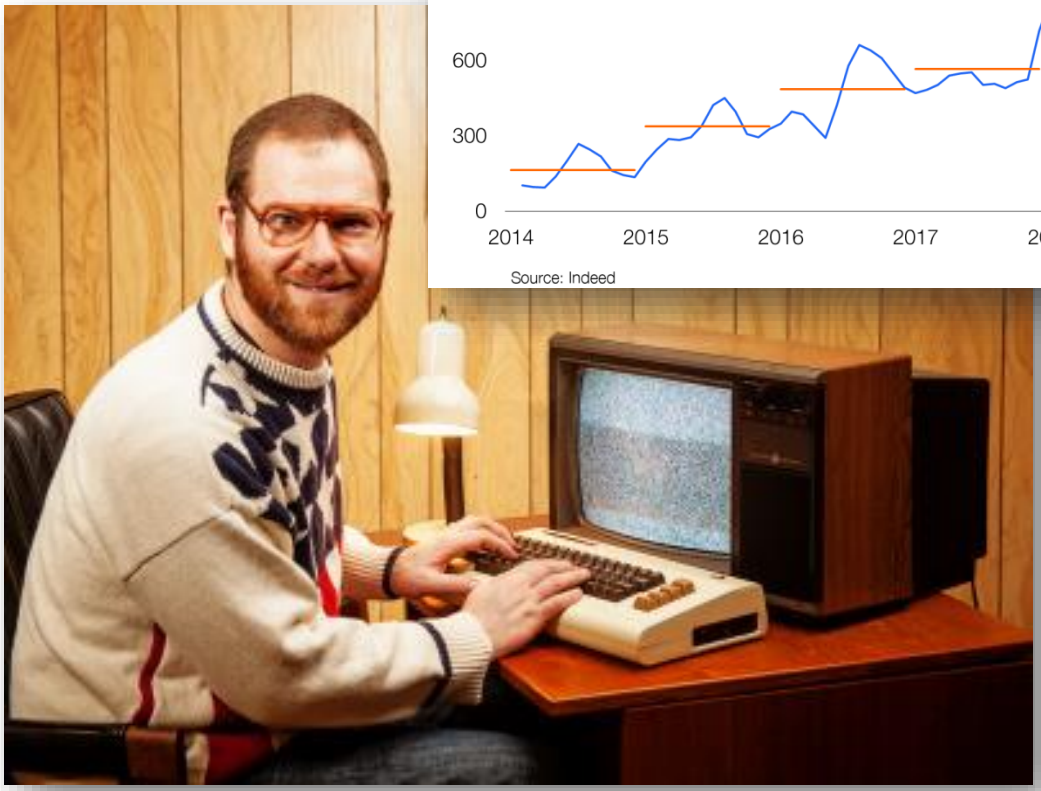
Data Scientist

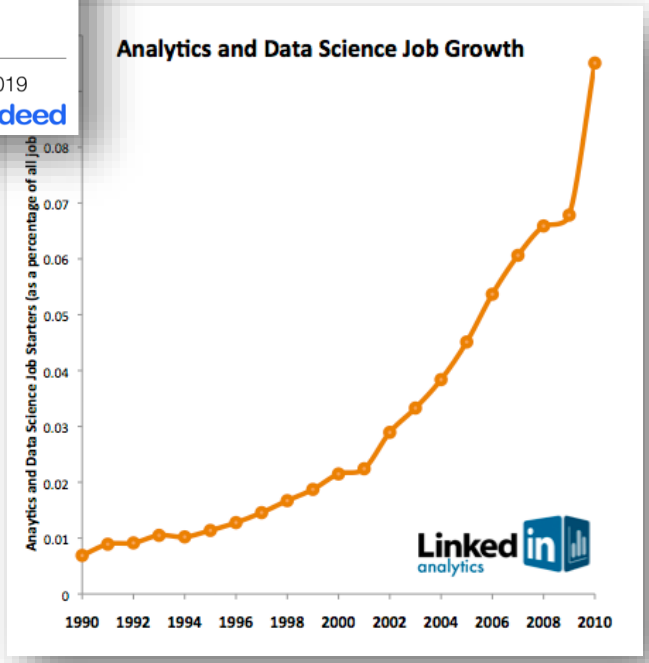
Ki az a Data Scientist?



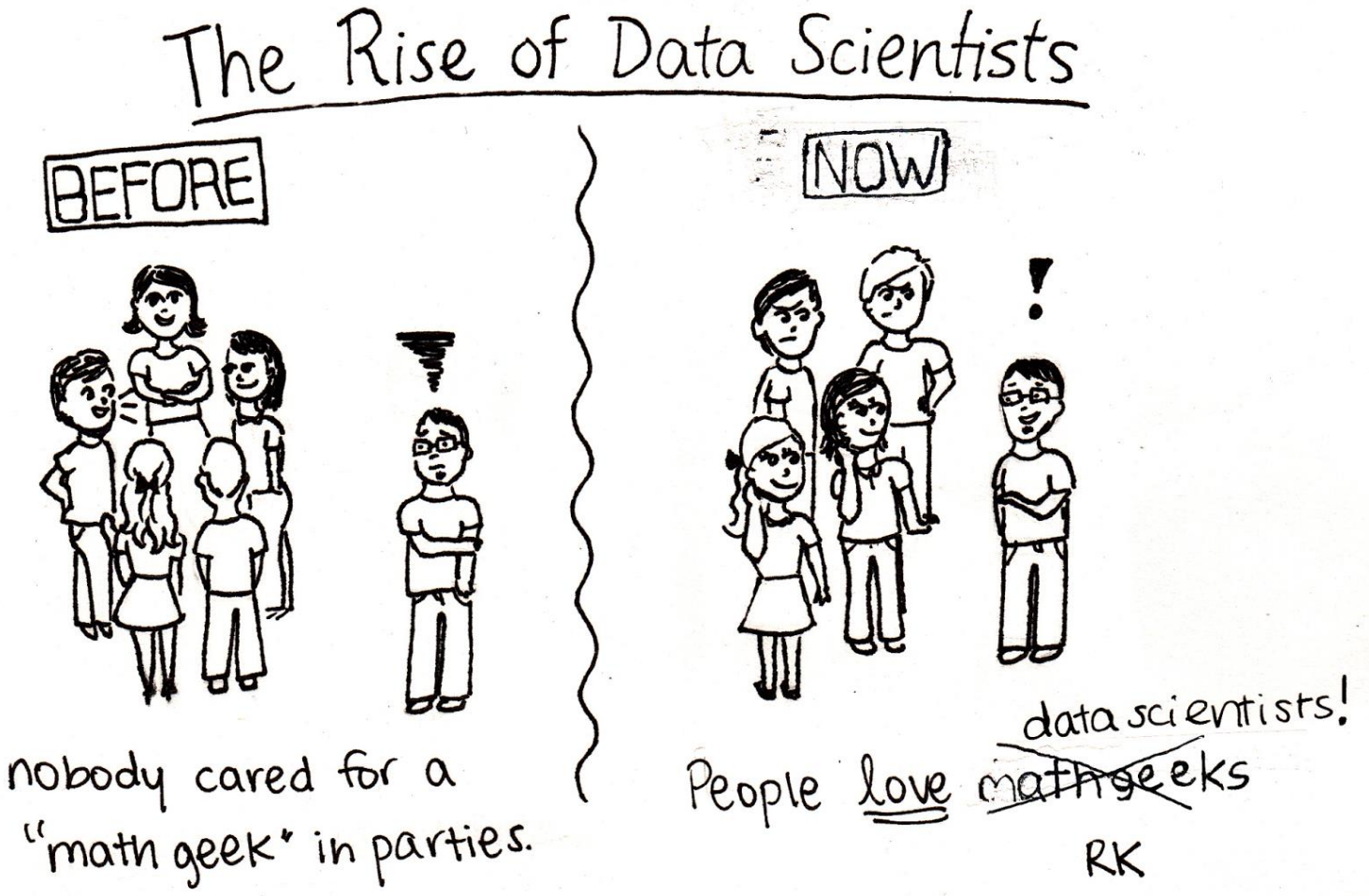
Data Scientist a „legszexibb” foglalkozás a 21. században!



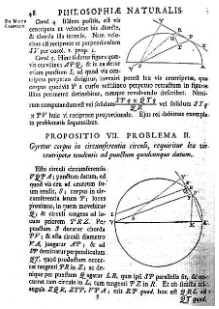


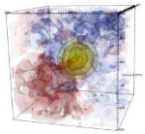


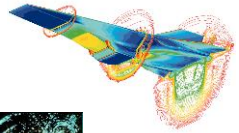
Data Scientist

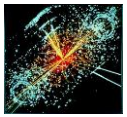


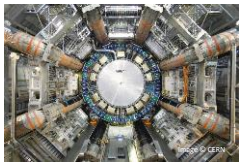
Adat vezérelt világ!

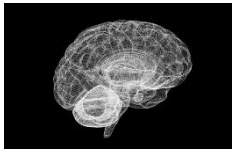












4000 years — 500 years — ~50 years — Today

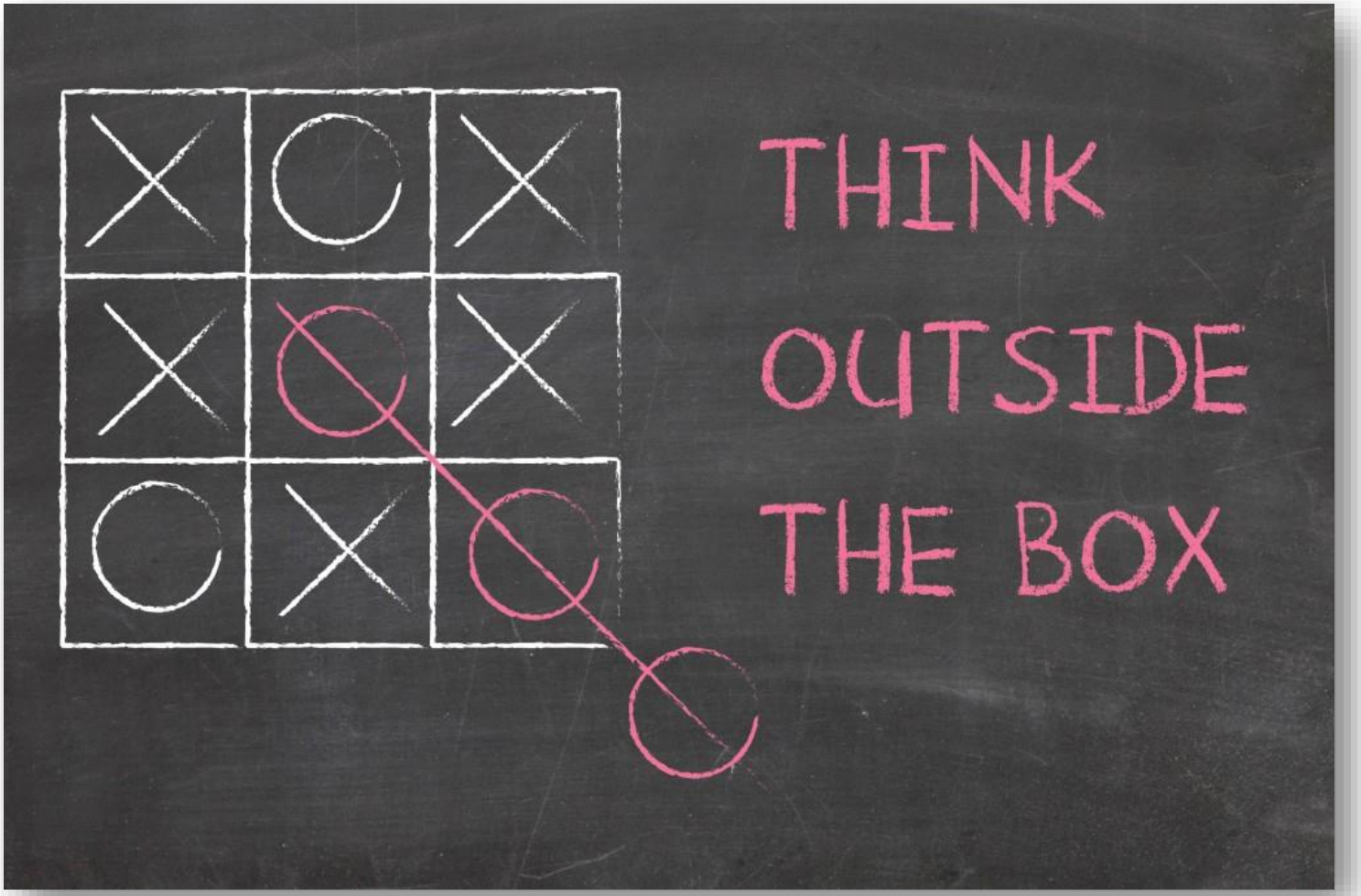
1 – Empirical observations

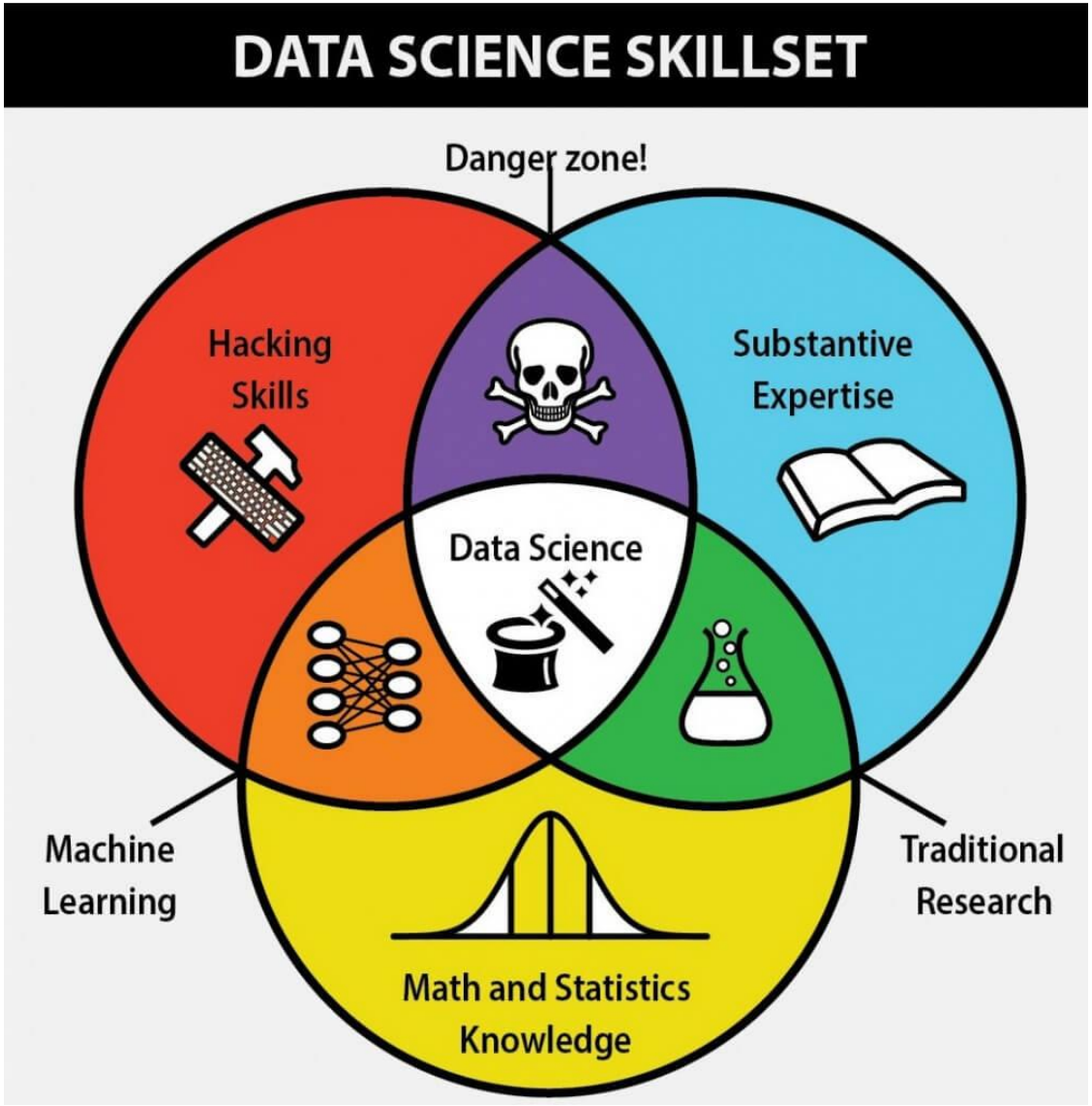
2 - Generalization Theoretical models

3 - Simulations Computational sciences

4 - Data-driven science eScience

Gondolkozz Másképp!




DATA SCIENCE SKILLSET
Danger zone!
Hacking Skills
Substantive Expertise
Data Science
Machine Learning
Traditional Research
Math and Statistics Knowledge

A Data Scientist munkája – a munkatársak szerint

$$\hat{\mathbf{r}} = \frac{x\hat{\mathbf{x}} + y\hat{\mathbf{y}} + z\hat{\mathbf{z}}}{\sqrt{x^2 + y^2 + z^2}}$$

$$\hat{\boldsymbol{\theta}} = \frac{(x\hat{\mathbf{x}} + y\hat{\mathbf{y}})z - (x^2 + y^2)\hat{\mathbf{z}}}{\sqrt{x^2 + y^2 + z^2}\sqrt{x^2 + y^2}}$$

$$\hat{\boldsymbol{\varphi}} = \frac{-y\hat{\mathbf{x}} + x\hat{\mathbf{y}}}{\sqrt{x^2 + y^2}}$$

A Data Scientist munkája – az anyukája szerint

A Data Scientist munkája – valójában

```
import pandas as pd
```

```
import numpy as np
```

```
SELECT * FROM Data WHERE…
```

@beckerfuffle #pyDataNYC

The Data Science is OSEMN

Source: A Taxonomy of Data Science

Big Data kihívások

hadoop történelem

- 2004 – Google publikálja a MapReduce technikát
- 2006 – Apache projekt lett Yahoo! támogatással
- További támogatók:

Kiknek jó a hadoop?

"... to create building blocks for programmers who just happen to have ***lots of data to store, lots of data to analyze, or lots of machines to coordinate***, and who don't have the time, the skill, or the inclination to become distributed systems experts to build the infrastructure to handle it."

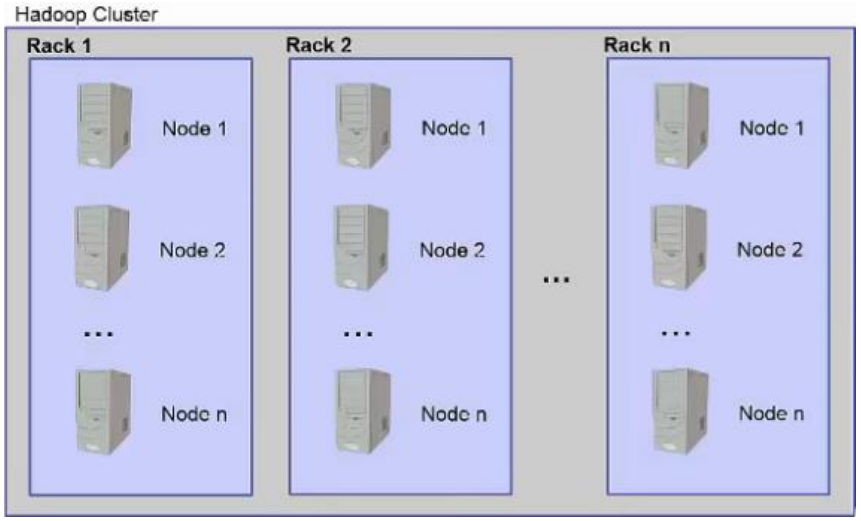
Tom White

Hadoop: The Definitive Guide

Kik haszálják?

Terminológia

- Klaszter:
  - Összetartozó számítógépek halmaza
- Rack:
  - Egy fizikai vagy virtuális csoportba tartozó gépek, amelyek jellemzően azonos áram és hálózati elérésen keresztül érhetőek el.
- Node:
  - Számító egységek, tipikusan a számítógépek



Hadoop Sorts a Petabyte in 16.25 Hours and a Terabyte in 62 Seconds

By aanand – Mon, May 11, 2009 11:00 AM EDT

We used Apache Hadoop to compete in Jim Gray's Sort benchmark. Jim's Gray's sort benchmark consists of a set of many related benchmarks, each with their own rules. All of the sort benchmarks measure the time to sort different numbers of 100 byte records. The first 10 bytes of each record is the key and the rest is the value. The **mInute sort** must finish end to end in less than a minute. The **Gray sort** must sort more than 100 terabytes and must run for at least an hour. The best times we observed were:

Bytes	Nodes	Maps	Reduces	Replication	Time
500,000,000,000	1406	8000	2600	1	59 seconds
1,000,000,000,000	1460	8000	2700	1	62 seconds
100,000,000,000,000	3452	190,000	10,000	2	173 minutes
1,000,000,000,000,000	3658	80,000	20,000	2	975 minutes

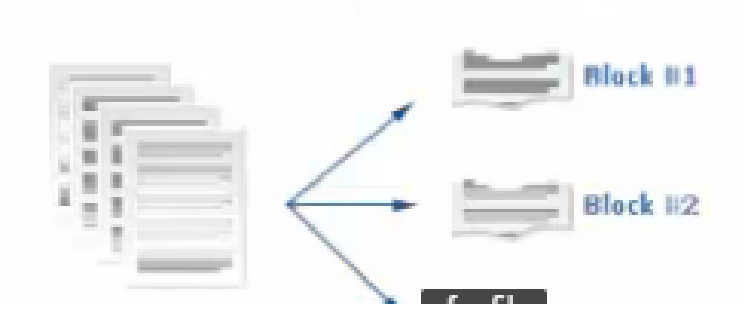
approximately 3800 nodes

Hadoop Architektúra

- Két fő komponens:
  - Distributed File System
    - Hadoop Distributed File System (HDFS)
    - Google File System (GFS)
  - MapReduce Engine / Resource Manager (Yarn)
    - Adat feldolgozó keretrendszer
    - Beépített erőforrás manager és ütemező

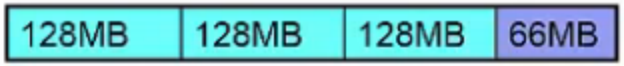
Hadoop Distributed File System (HDFS)

- Meglévő fájlrendszeren működik
  - Nagyobb mennyiségű hiba kezelésére tervezték
    - replikák miatt
- Nagy méretű fájlok tárolására tervezték
  - Folytonos adat olvasására hatékony
    - Véletlen elérés nincs
- A fájlokat blokkokra bontja és azokat tárolja



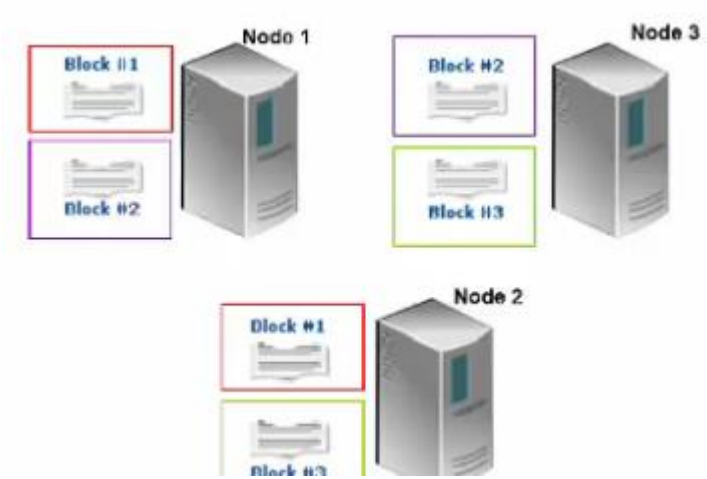
HDFS fájl blokkok

- Nem ugyanaz mint az op. rendszer fájl blokkjai
  - HDFS több operációs rendszeren tárolja a blokkokat
- Blokk méret: 128MB (régebben: 64MB)
- Fájl mérete nagyobb lehet mint bármely lemez kapacitása a klaszterben
  - A fájlok több nodeon tárolódnak blokként
- Ha kisebb a fájl mint a blokkméret
  - Akkor csak a szükséges részt tárolja



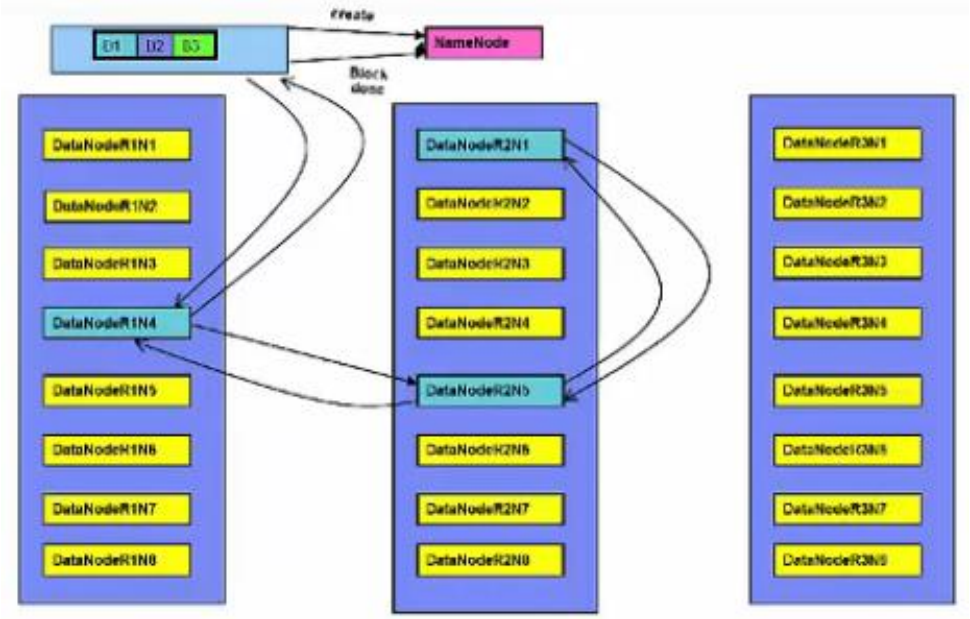
HDFS replika

- A blokkok több node-on vannak tárolva: replikálva
- Ez biztosítja hogy ne legyen adatvesztés hiba esetén
- Default replikaszám: 3
- Szabály: 1 replikának mindig másik rackben kell lennie



Replika készítés folyamata

1. Blokk elküldése egy node-nak
2. A node elküldi a blokkot egy másik rackben lévő node-hoz
3. Ez a node elküldi az adatot nodenak a saját rackjében
4. Minden node visszaigazolja, hogy letárolta az adatot



Klaszter állapotának ellenőrzése

- *hdfs dfsadmin –report* vagy Ambari felület
- Megmutatja a HDFS állapotát
  - Hiányzó blokkok száma
  - Nem replikált blokkok száma

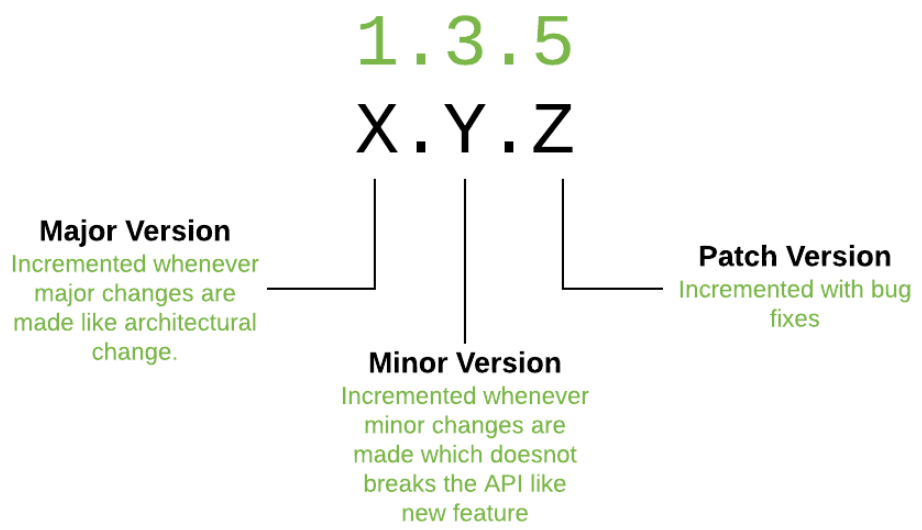
```
C:\Users\Gogo>hdfs dfsadmin -report
A fájlnév, a könyvtárnév vagy a kötetcímke szintaxisa nem megfelelő.
Configured Capacity: 499739779072 (465.42 GB)
Present Capacity: 63237525417 (58.89 GB)
DFS Remaining: 63236362240 (58.89 GB)
DFS Used: 1163177 (1.11 MB)
DFS Used%: 0.00%
Under replicated blocks: 12
Blocks with corrupt replicas: 0
Missing blocks: 0
Missing blocks (with replication factor 1): 0
Pending deletion blocks: 0

-------------------------------------------------
Live datanodes (1):

Name: 127.0.0.1:50010 (wit-ams-cloudservice.cloudapp.net)
Hostname: Gogo-PC
Decommission Status : Normal
Configured Capacity: 499739779072 (465.42 GB)
DFS Used: 1163177 (1.11 MB)
Non DFS Used: 436502253655 (406.52 GB)
DFS Remaining: 63236362240 (58.89 GB)
DFS Used%: 0.00%
DFS Remaining%: 12.65%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 1
Last contact: Thu Sep 10 13:34:08 CEST 2020
Last Block Report: Thu Sep 10 13:33:27 CEST 2020
```

Verziók

Download

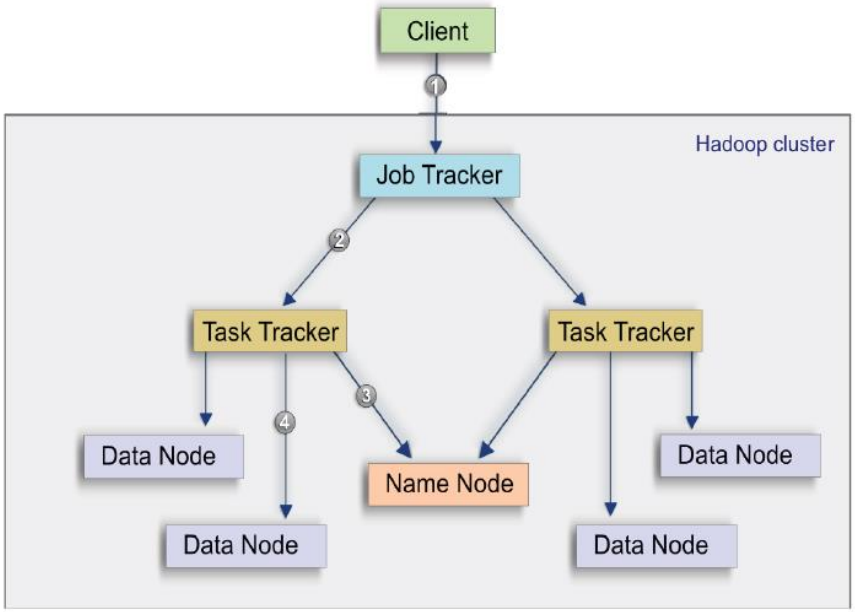
Hadoop is released as source code tart using GPG or SHA-512.

Version	Release date
3.1.4	2020 Aug 3
3.3.0	2020 Jul 14
2.10.0	2019 Oct 29
3.2.1	2019 Sep 22
2.9.2	2018 Nov 19



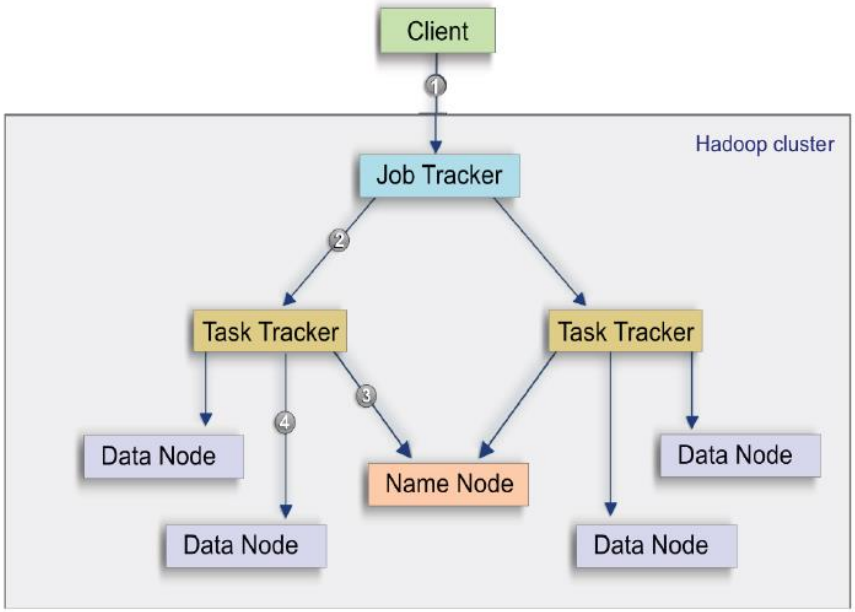
Architektúra (Hadoop 1.x)

- JobTracker:
  - MapReduce Jobok kezelése
  - Klienssel való kommunikáció
  - Feladat kiosztása a TaskTrackereknek
- Task Tracker
  - Adott task folyamat elvégzése:
    - Map
    - Reduce
    - Shuffle



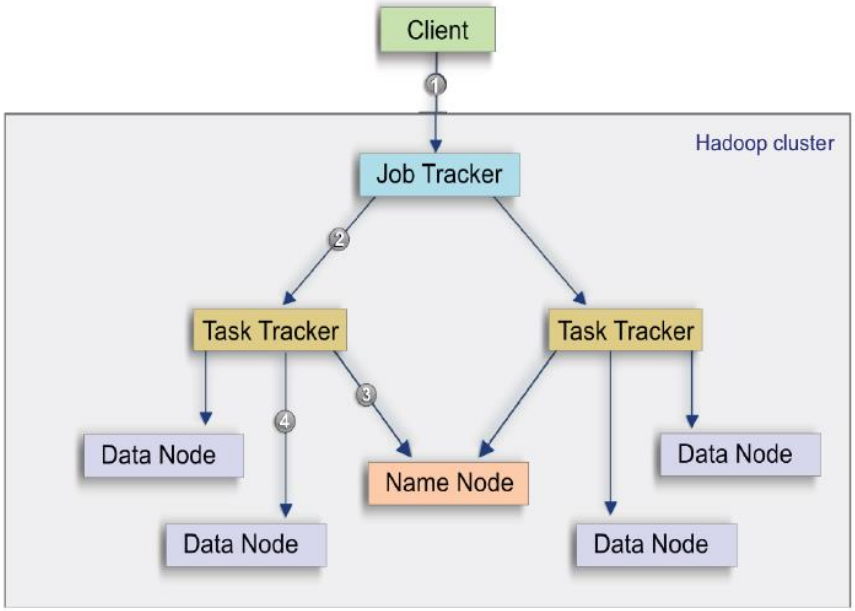
Architektúra (Hadoop 1.x)

- NameNode:
  - HDFS Könyvtár struktúra kezelés
  - Fájlok meglétének ellenőrzése
    - replikaszám
    - hiányzó blokkok
  - Memóriában tartja az információkat
  - Lehet belőle kettő.
- DataNode
  - Tárolja a fájl blokkokat a HDFS-en.



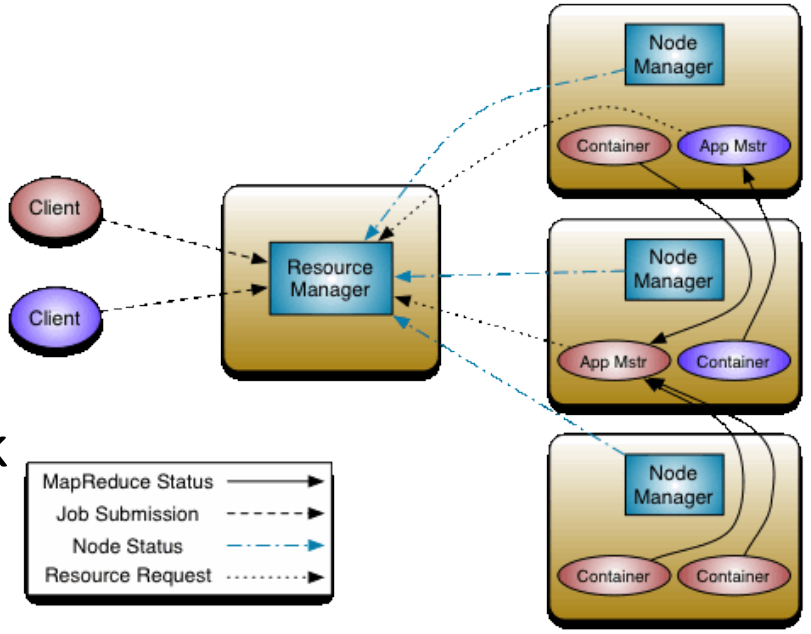
Architektúra (Hadoop 1.x)

- Probléma:
  - Job Tracker szűk keresztmetszet
    - Több kliens kiszolgálásakor túlterhelt
  - Csak MapReduce Jobokat képes kezelni

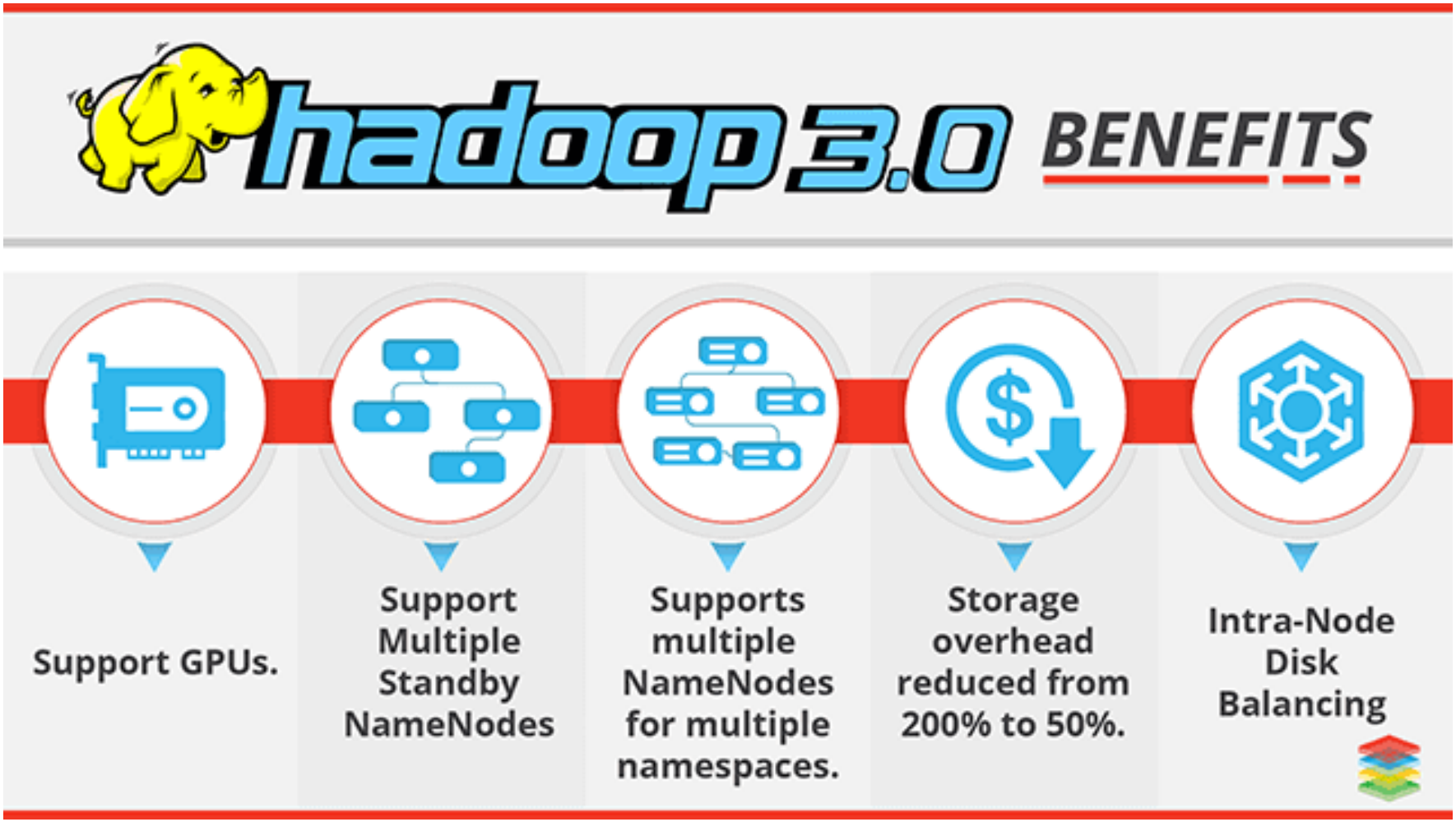


Architektúra (Hadoop 2.x)

- Resource Manager (YARN)
  - Kapcsolat a kliensekkel
  - Jobok felügyelete
  - A klaszter erőforrásainak ismerete
  - Containerek készítése
- Node Manager
  - Adott node erőforrásainak ismerete
- Application Master
  - Tetszőleges alkalmazás felügyelete és kezelése



Hadoop 3.x


hadoop 3.0 BENEFITS
Support GPUs.
Support Multiple Standby NameNodes
Supports multiple NameNodes for multiple namespaces.
Storage overhead reduced from 200% to 50%.
Intra-Node Disk Balancing

MapReduce

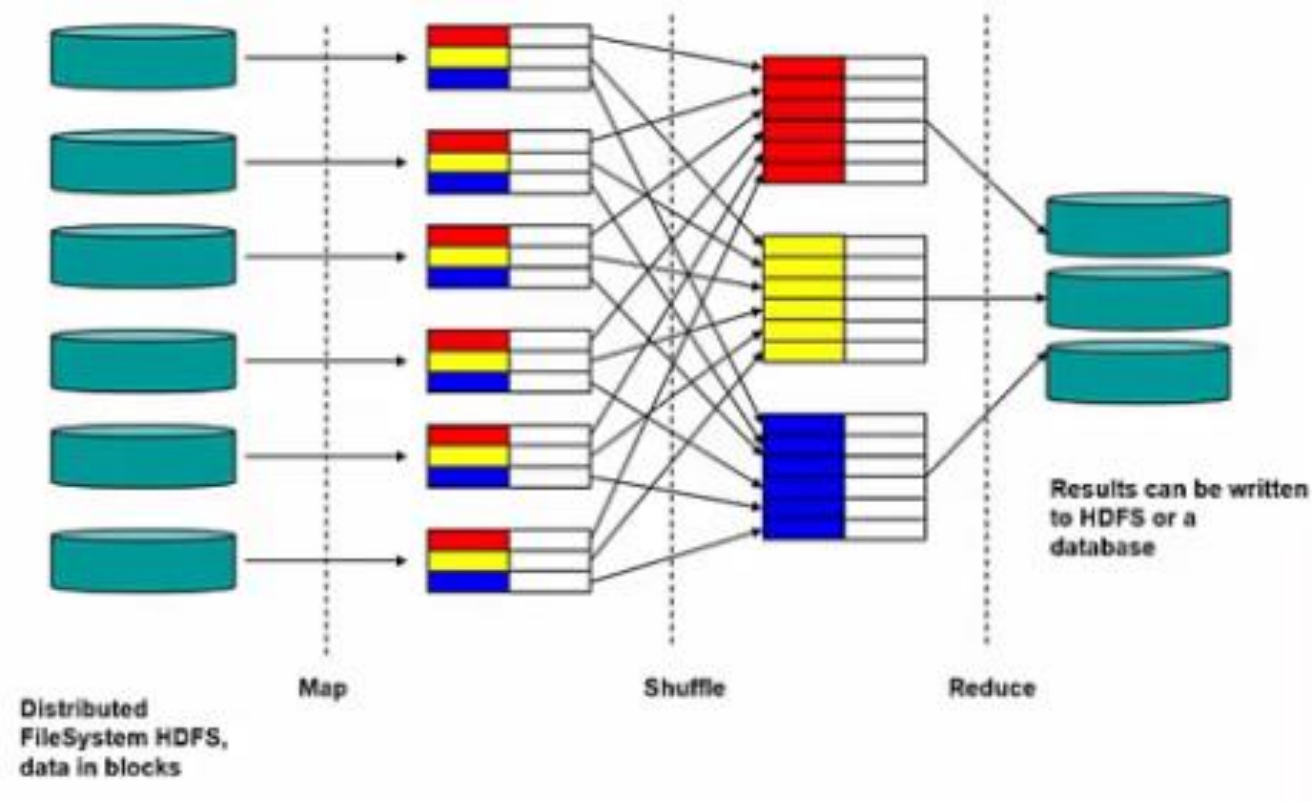
- Nagy mennyiségű adat feldolgozása
- Map
  - Master node szétbontja a problémát kisebb feladatokra
  - Kiosztja ezeket a feladatokat a worker node-oknak
- Reduce
  - Master node összegyűjti a részproblémák eredményét
  - Összegzi a részeredményeket
- A Map és a Reduce folyamatok is képesek párhuzamosan futni

Alap adattípusok

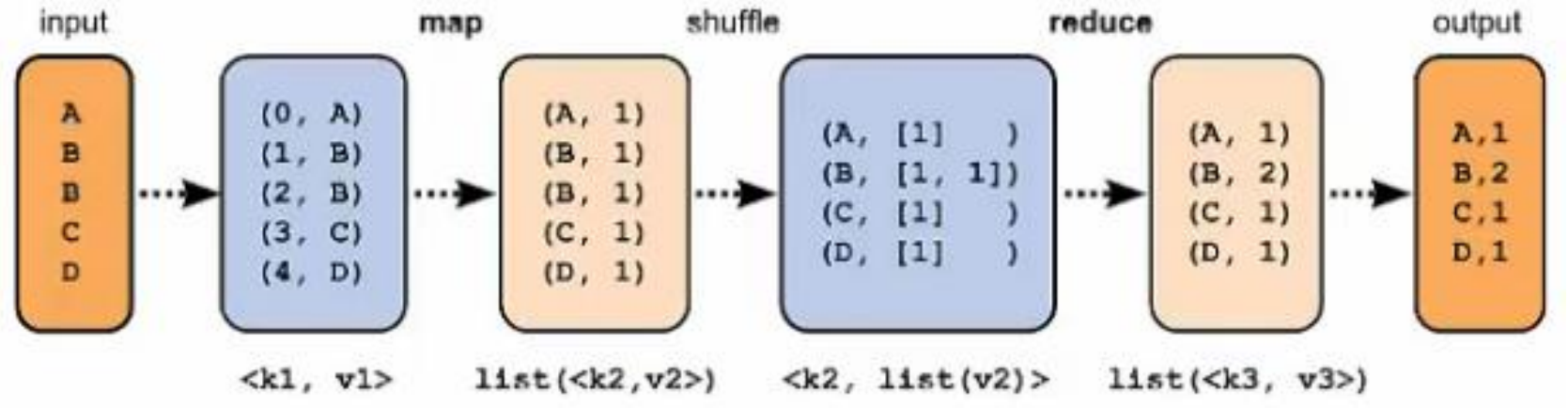
- Kulcs-érték párok
- Listák

	Input	Output
map	<k1, v1>	list(<k2, v2>)
reduce	<k2, list(v2)>	list(<k3, v3>)

MapReduce overview

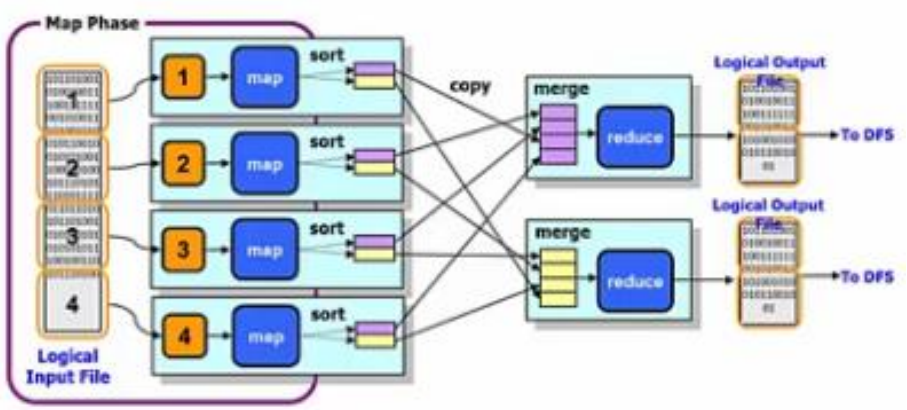


Egyszerű adatfolyam példa



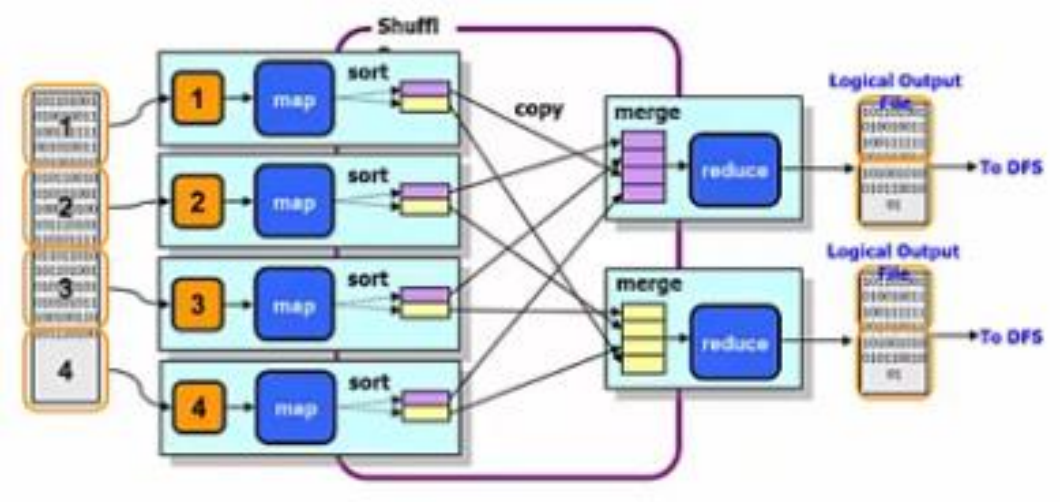
MapReduce – Map fázis

- Mapperek
  - kis program, elosztva a klaszterben, lokális adaton fut
  - az input fájl csak egy kis részén dolgozik
  - minden mapper értelmezi, szűri vagy átalakítja a bemenetet
  - <kulcs,érték> párokat hoz létre



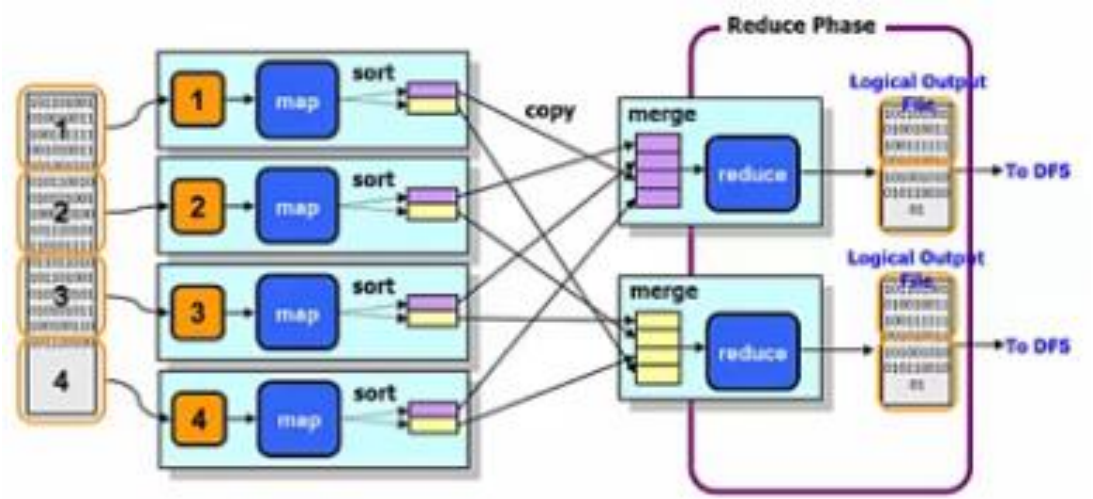
MapReduce – Shuffle fázis

- Minden mapper kimenete lokálisan csoportosítva van kulcs alapján
- Minden egyes kulcshoz kiválasztunk egy node-t
- Minden ilyen adatmozgatást a MapReduce végez



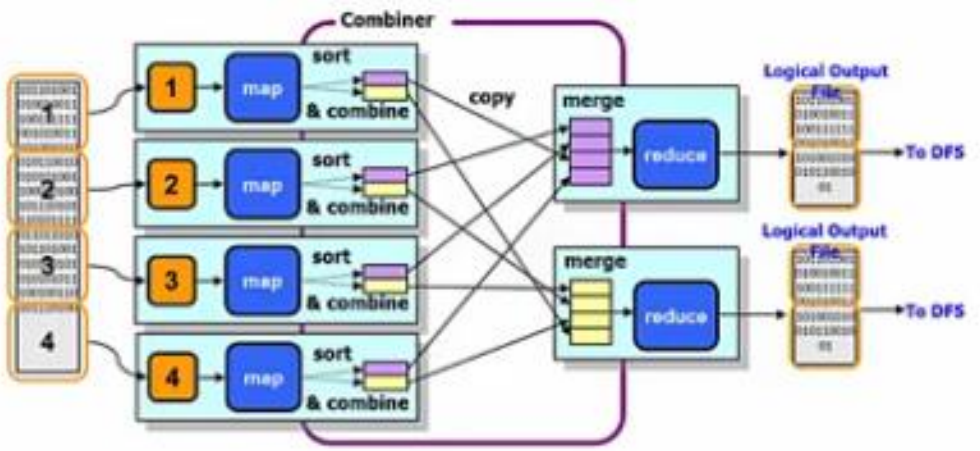
MapReduce – Reduce fázis

- Reducerek
  - kis programok, amelyek aggregálják a hozzájuk tartozó kulcshoz rendelt értékeket
  - Minden reducer saját fájlba írja a kimenetét



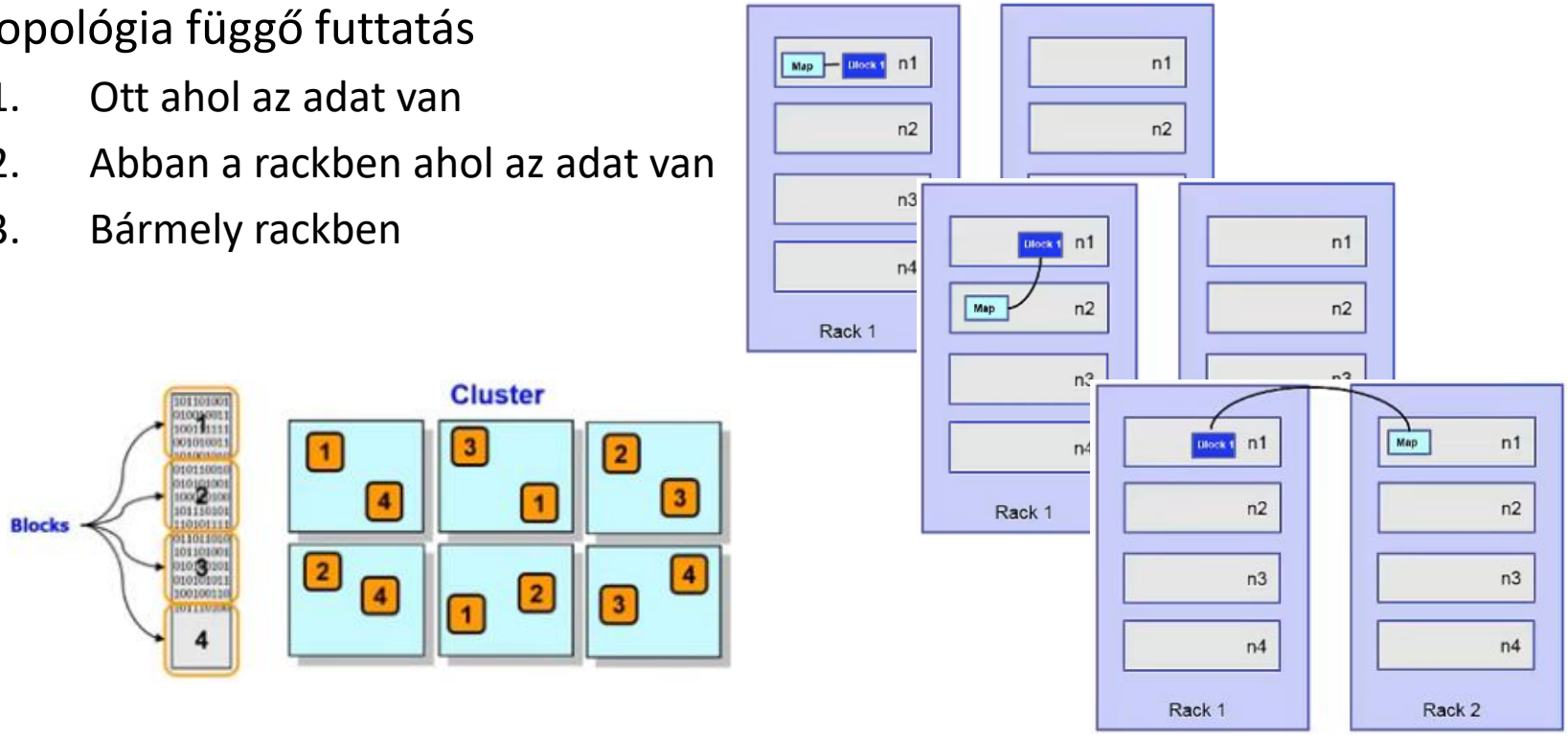
MapReducer – Combiner (opcionális)

- Minden adatot lerendezünk és csoportosítunk mielőtt eljut a reducer node-hoz. Ahhoz hogy csökkentük a hálózati forgalmat, a map oldalon már tudunk végezni egy előzetes redukálást.



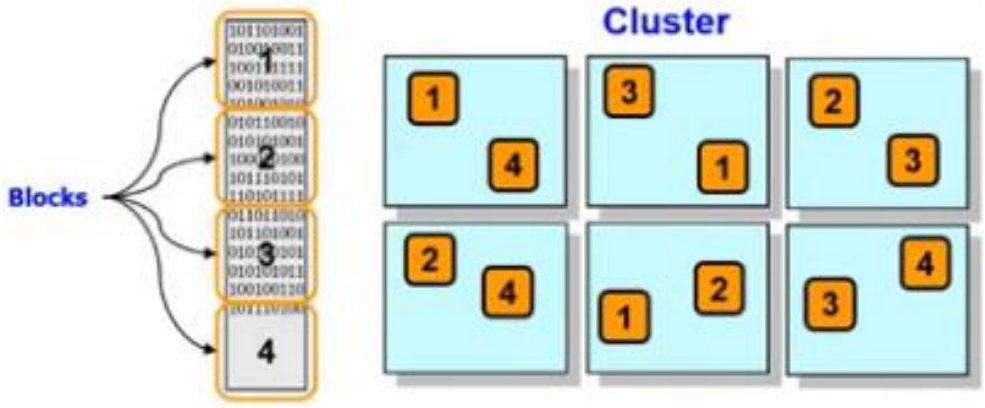
MapReduce – Elosztott fájlrendszer

- Vezérlési szabály
  - az adat a teljes klaszterben van tárolva
  - a programot mozgassuk az adathoz, ne az adatot a programhoz
- Topológia függő futtatás
  1. Ott ahol az adat van
  2. Abban a rackben ahol az adat van
  3. Bármely rackben



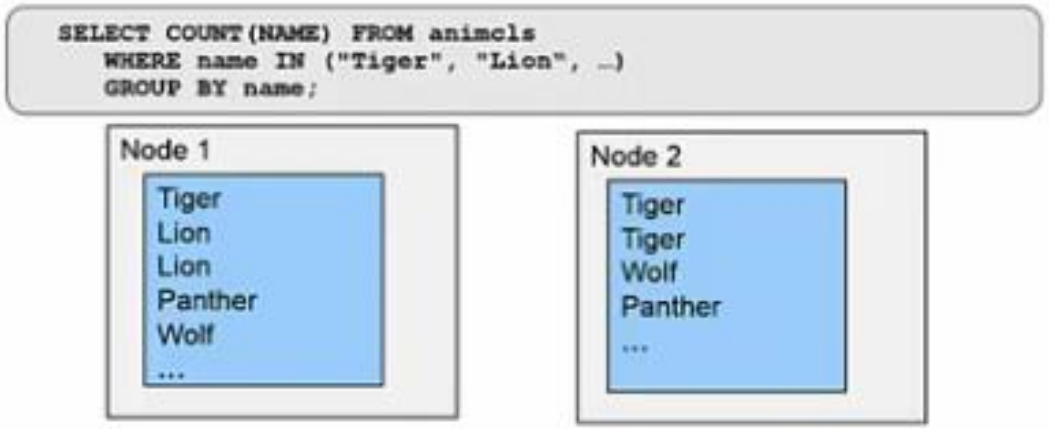
Speculative Execute

- Egyes node-ok lassúak lehetnek
- Ugyanaz az adat megtalálható több node-on is.
- Futtassuk ugyanazt a feladatot ezeken a node-okon.
- A leggyorsabb eredményét fogadjuk el a többit elvetjük.



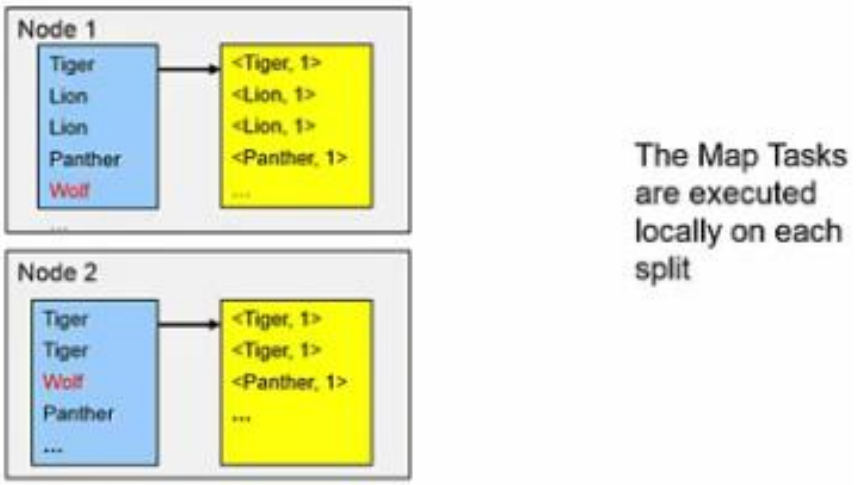
Word Count példa

- Ebben a példában állatnevek vannak
  - MapReduce automatikus darabolja a fájlt sortörések alapján
  - A fájl két részre lett bontva, és két node-on tárolódik
- Meg szeretnénk számolni milyen gyakran fordulnak elő a nagymacska állatnevek
  - SQL-ben így nézne ki:



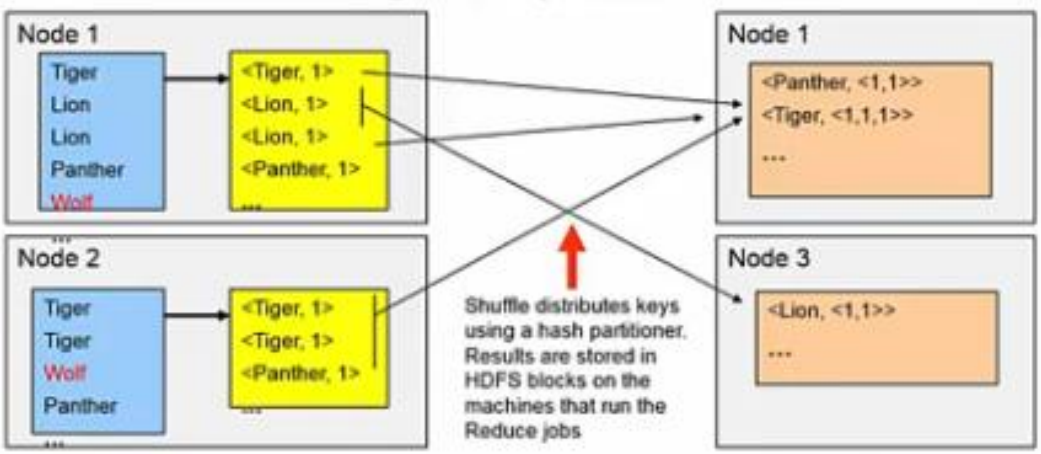
Map task

- Két feladtunk lesz a Map folyamatnál:
  - szűrje ki a nem nagy macskákat
  - készítse elő s számoláshoz az adatot:
    - <Text(name), Integer(1)>



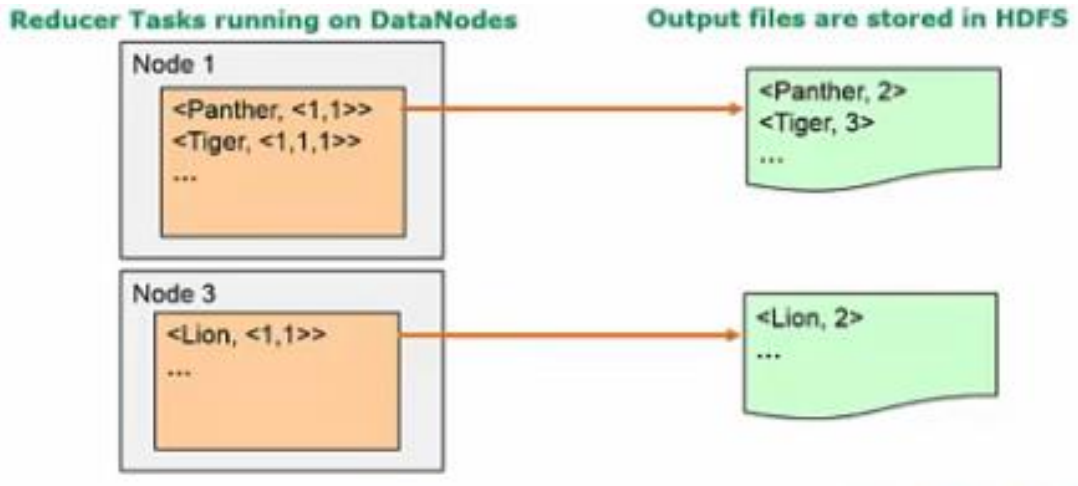
Shuffle

- Shuffle fázis mozgatja az értékeket egy adott kulcs alapján a megfelelő node-okhoz
- Elosztást a Partioner Class végzi (hash elosztás)
- Reducer bármely node-on képes futni (itt node1, node3)
  - Különböző számú Mapper és Reducer task is lehet



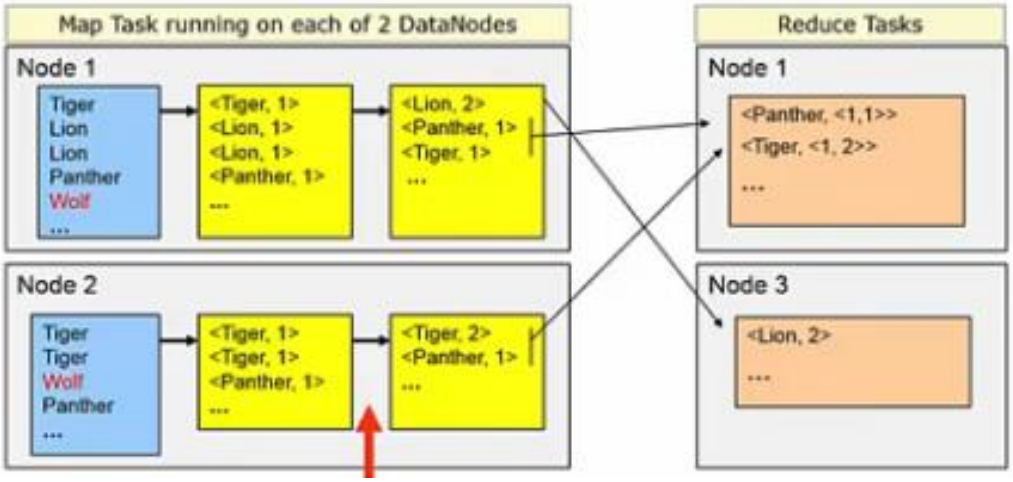
Reduce

- A reducer aggregálja a kulcshoz tartozó értékeket
  - Az eredmény a HDFS-re lesz visszaírva
  - Alapból egy file / reducer
  - Reducer aggregálja az értékeket egy adott kulcshoz (ebben az esetben az állatnevekhez)



Combiner (Optinal)

- Combiner
  - opcionális, de gyorsítani tudja a futást
  - aggregációt végez a map folyamat után
  - kevesebb adatot küldünk át a hálózaton a reducereknek
  - Shuffle fázis előtt fut



Writeables

Java Class	Writeable
boolean	BooleanWriteable
byte	ByteWriteable
int	IntWriteable
float	FloatWriteable
long	LongWriteable
String	Text
double	DoubleWriteable

Mapper

```java
package hu.elte;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class WCMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
    private final Text wordKey = new Text("");
    private final IntWritable wordNum = new IntWritable(1);

    @Override
    public void map(LongWritable key, Text value, Context context)
        throws IOException, InterruptedException {
        String[] fields = value.toString().split(" ");
        for (String s : fields) {
            wordKey.set(s);
            context.write(wordKey, wordNum);
        }
    }
}
```

Reducer

```java
package hu.elte;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class WCReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
    private final IntWritable wordNum = new IntWritable(1);

    public void reduce(Text _key, Iterable<IntWritable> values, Context context)
            throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        wordNum.set(sum);
        context.write(_key, wordNum );
    }

}
```

Driver

```java
package hu.elte;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WCDriver {

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "WCTest");

        job.setJarByClass(hu.elte.WCDriver.class);
        job.setMapperClass(hu.elte.WCMapper.class);

        job.setReducerClass(hu.elte.WCReducer.class);

        // TODO: specify output types
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        System.out.println(args[0]+"  ----  "+args[1]);

        // TODO: specify input and output DIRECTORIES (not files)
        FileInputFormat.setInputPaths(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        if (!job.waitForCompletion(true))
            return;
    }

}
```

Köszönöm a Figyelmet!