FAKULTA ELEKTROTECHNICKÁ

České vysoké učení technické v Praze

B4M36DS2 – Database Systems 2

MSc. **Yuliia Prokop**, Ph.D.

prokoyul@fel.cvut.cz

Telegram **@Yulia_Prokop**

ČVUT FEL CourseWare Wiki **https://cw.fel.cvut.cz/b231/courses/b4m36ds2/start**

Basic course information

Lectures: Monday, 9:15 – 10:45

Practical classes: Monday, 12:45 – 14:15, 14:30 – 16:00, 16:15 – 17:45

Homework – maximum 120 points

Course credit – minimum 100 points

Exam – maximum 100 points

- **written exam** (mandatory) + **oral exam** (optional)

CourseWare Wiki – course materials

BRUTE – upload reports on the homework

NoSQL Server – submit and execute homework

FAKULTA ELEKTROTECHNICKÁ

České vysoké učení technické v Praze

B4M36DS2 – Database Systems 2

Lecture 1 - Introduction: Big Data, NoSQL Databases

25. 9. 2023

Yuliia Prokop

prokoyul@fel.cvut.cz, Telegram **@Yulia_Prokop**

Based on **Martin Svoboda**'s materials (https://www.ksi.mff.cuni.cz/~svoboda/courses/211-B4M36DS2/)

ČVUT FEL CourseWare Wiki **https://cw.fel.cvut.cz/b231/courses/b4m36ds2/start**

Lecture Outline

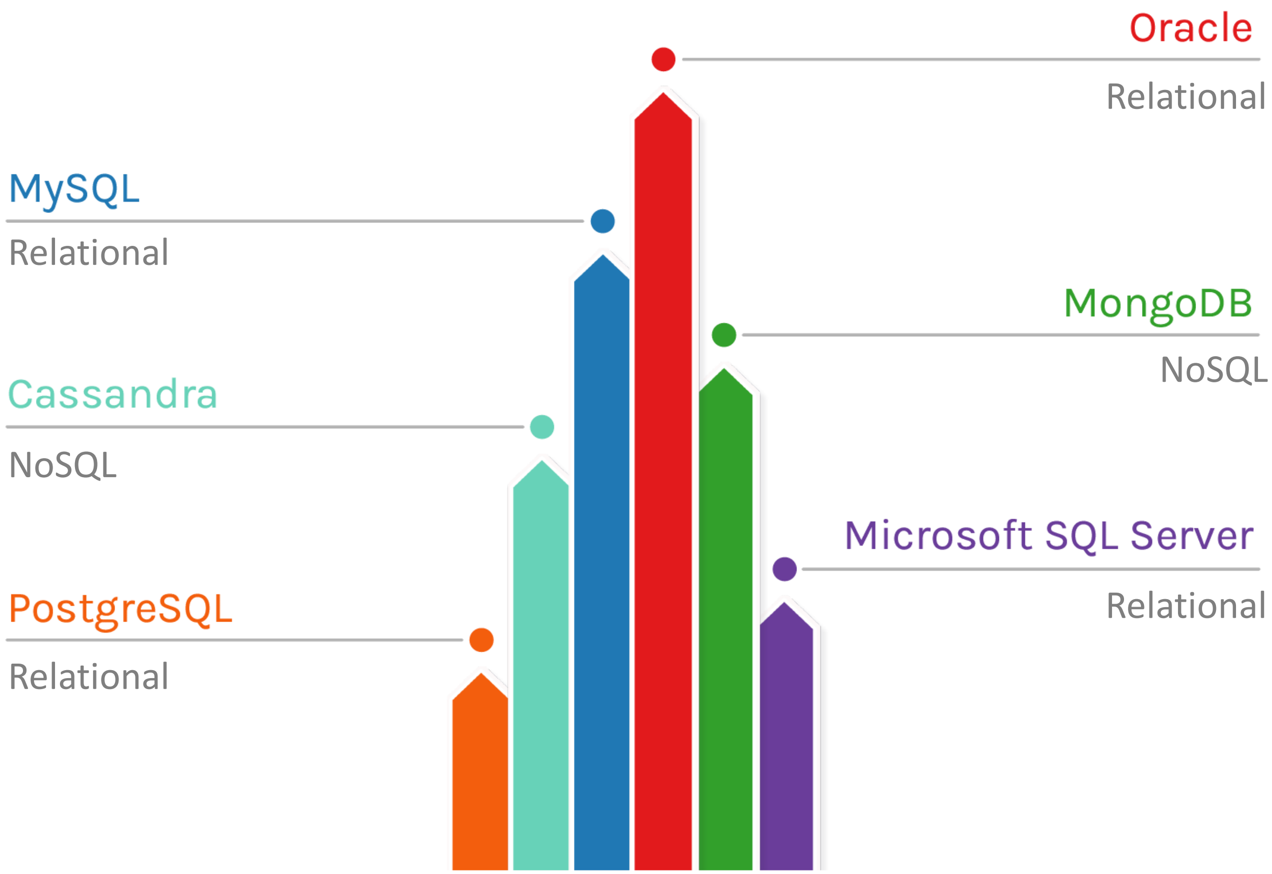
- ✓ History of database models
- ✓ DBMS ranking 2023
- ✓ Big Data and its characteristics
- ✓ Relational DBS features
- ✓ Types of data stores
- ✓ NoSQL DBS features

Historical trends of Database Management System

A database management system (DBMS) allows a person to organize, store, and retrieve data from a computer.

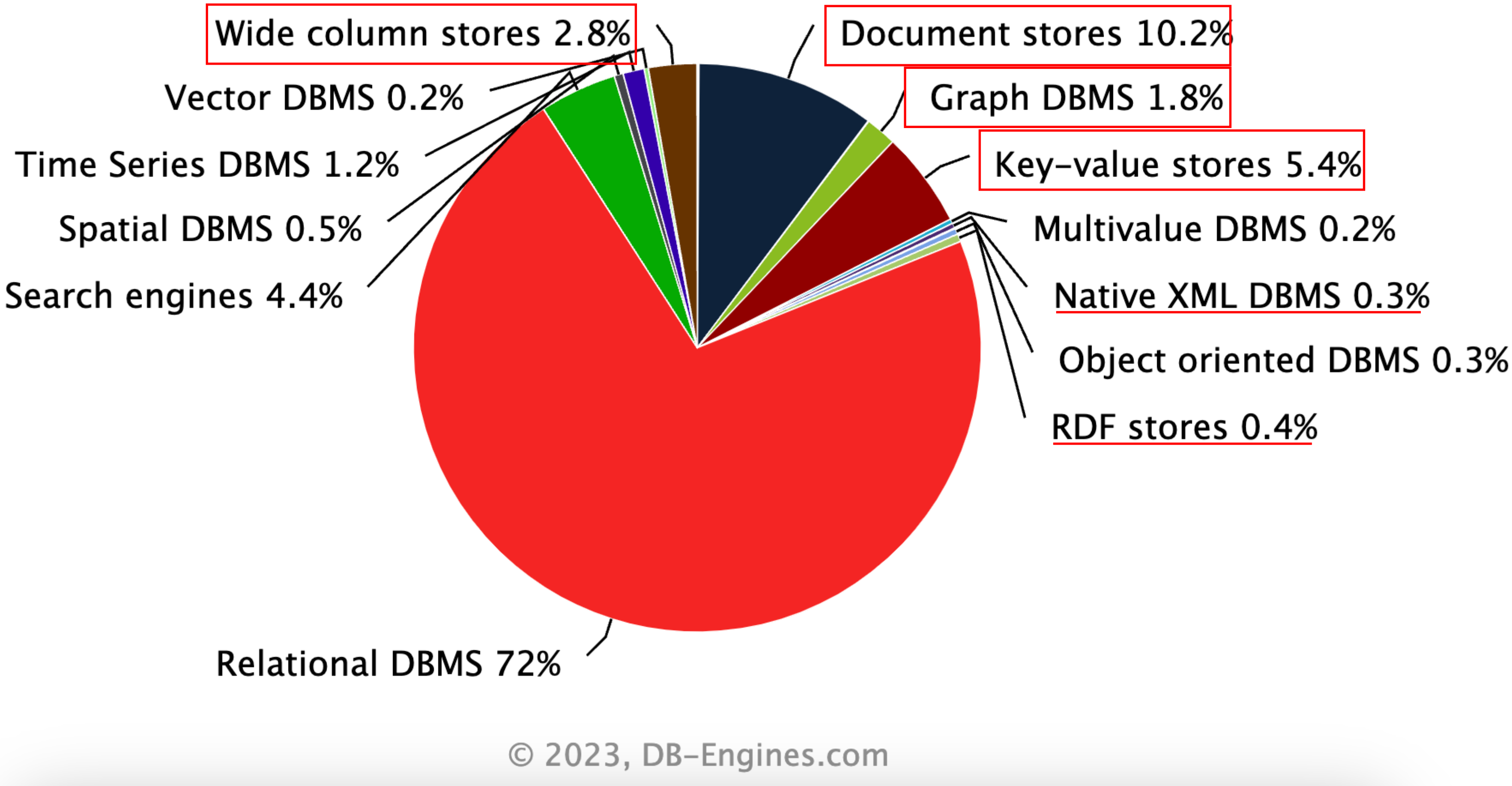
- **2008** NoSQL, Big Data
- **2000s** Relational database model
- **1990s** Object database model
- **1980s** Structured Query Language (SQL)
- **1970s** Relational database model
- **1960s** Network and hierarchical models

Top Database Management Systems In July, 2023



Source: https://red9.com/database-popularity-ranking/

Database ranking 2023

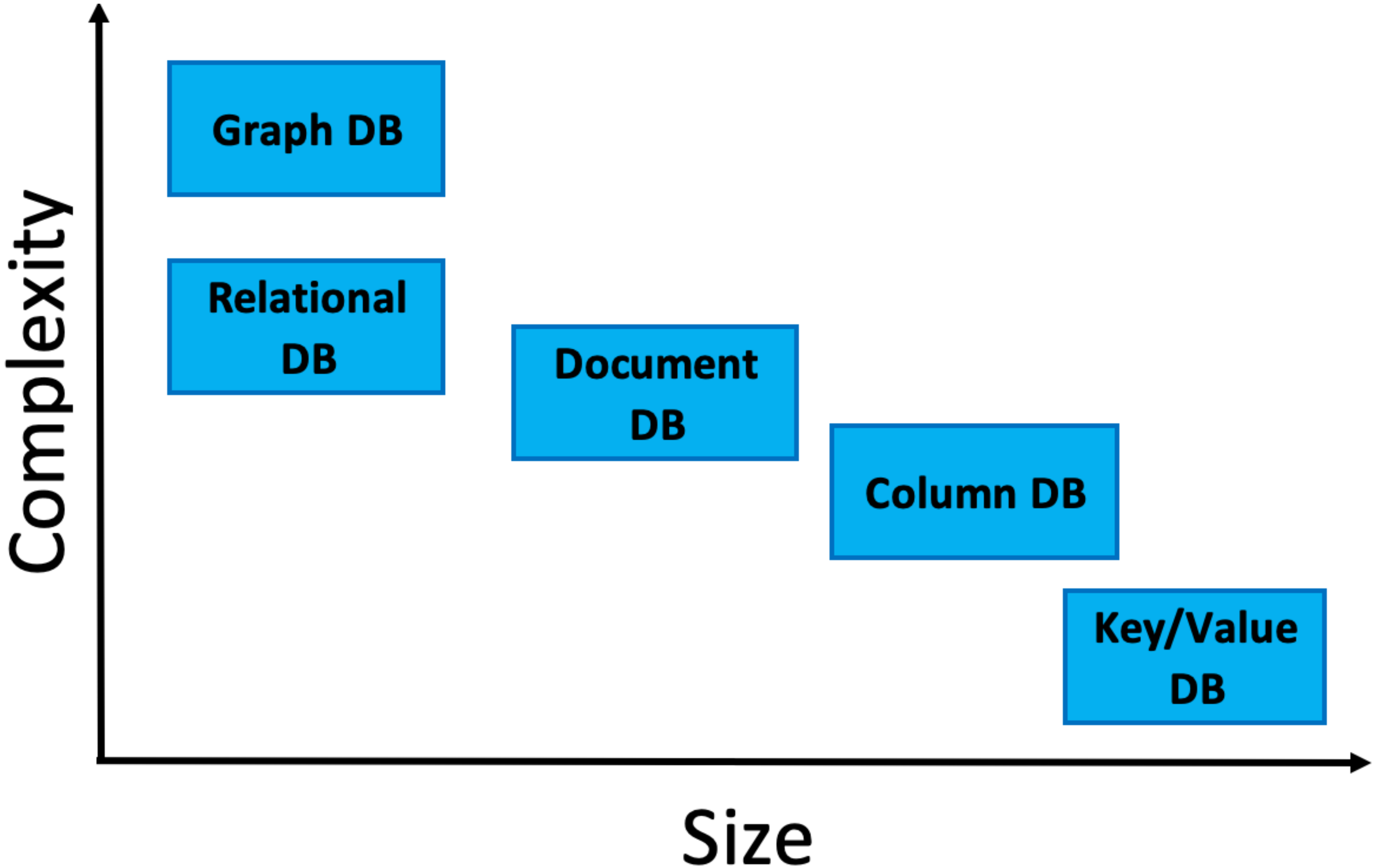


Source: https://db-engines.com/en/ranking_categories

Database ranking 2023 & NoSQL DBS in the course

- **Document** stores (**MongoDB**, CouchDB)
- **Key-value** stores (**Redis**, DynamoDB)
- **Wide column** stores (**Cassandra**, HBase)
- **Grapf** DBMS (**Neo4j**, RDF)
- Hybrid systems (**HADOOP**)
- Native XML DBMS

Size / Complexity of data stores



Big Data

What is Big Data?

Big Data primarily refers to data sets that are **too large** or complex to be dealt with by traditional data-processing application software. It is characterized by the three Vs: volume, variety, and velocity

Where is Big Data?

- **Social media and networks**

  …all of us are generating data
- **Scientific instruments**

  …collecting all sorts of data
- **Mobile devices**

  …tracking all objects all the time
- **Sensor technology and networks**

  …measuring all kinds of data

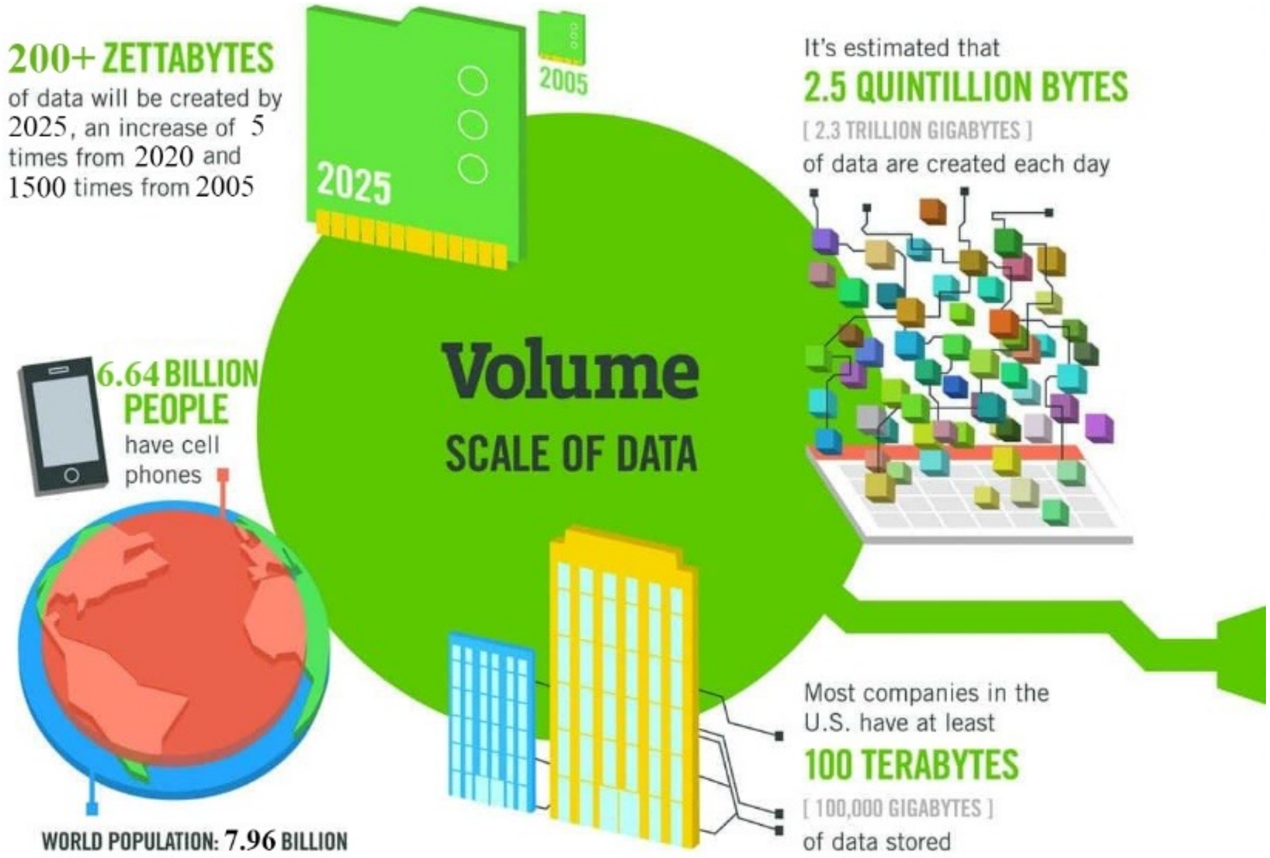
Key Big Data Statistics 2023

- **In 2022 the global big data industry is worth $274.3 billion.**
- **Google receives more than 9 billion searches every day.**
- 100 billion messages are exchanged on WhatsApp every day.
- 95% of businesses struggle to manage unstructured data.
- Big data in healthcare will be worth $67 billion by 2025.
- **79 zettabytes of data were generated in 2021.**
- Data interactions have increased by 5000% since 2010.
- More than 1.2 billion years have been spent online.
- **Internet users generate 2.5 quintillion bytes of data each day.**
- In the year 2020, each internet user generated 1.7MB of data per second.
- 95% of businesses said that managing unstructured data is a significant problem.
- More than 91% of organizations are investing in artificial intelligence and big data today.
- With the help of big data, Netflix saves over $1 billion annually with customer retention.

Source: https://earthweb.com/big-data-statistics/

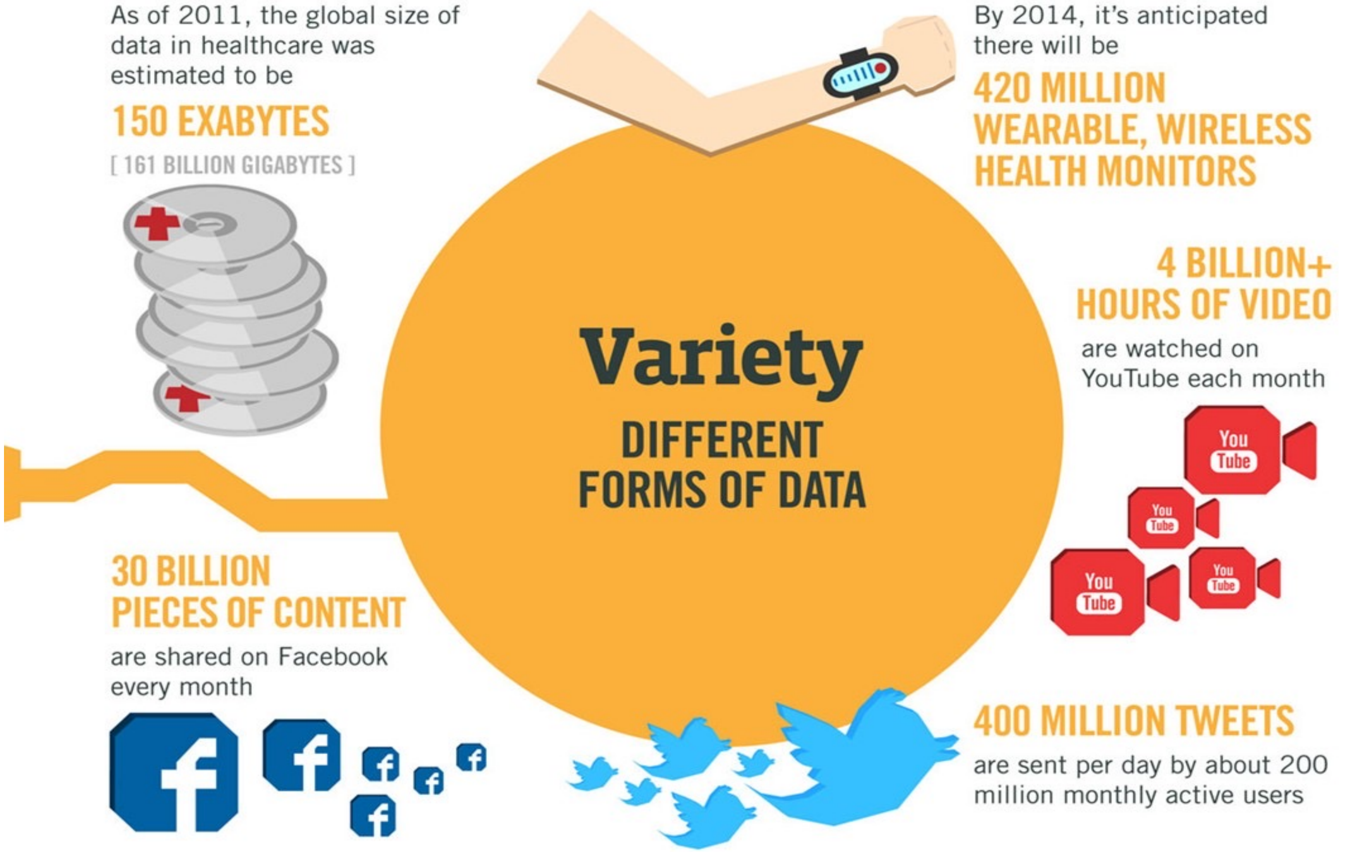
Big Data Characteristics : Volume

Volume refers to the scale of the data



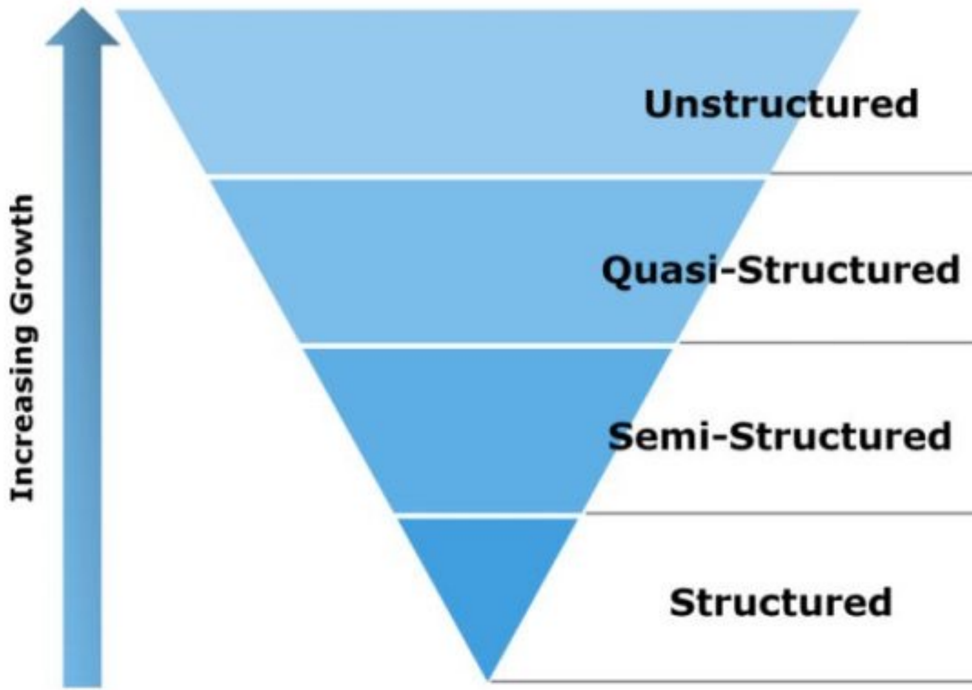
Source: http://www.ibmbigdatahub.com/; https://www.bankmycell.com; https://techjury.net

Variety refers to the different formats that data comes in



Source: http://www.ibmbigdatahub.com/

Types of Data



Unstructured
- Data that has no inherent structure and is usually stored as different types of files.
- E.g. Text documents, PDFs, images, and videos

Quasi-Structured
- Textual data with erratic formats that can be formatted with effort and software tools
- E.g. Clickstream data

Semi-Structured
- Textual data files with an apparent pattern, enabling analysis
- E.g. Spreadsheets and XML files

Structured
- Data having a defined data model, format, structure
- E.g. Database

Source: https://www.mycloudwiki.com/san/data-and-information-basics/

Structured Data

- Types of data: **structured**, **unstructured semi-structured**
- Any data that can be stored, accessed and processed in the form of fixed format is termed as a ‘**structured**’ data.

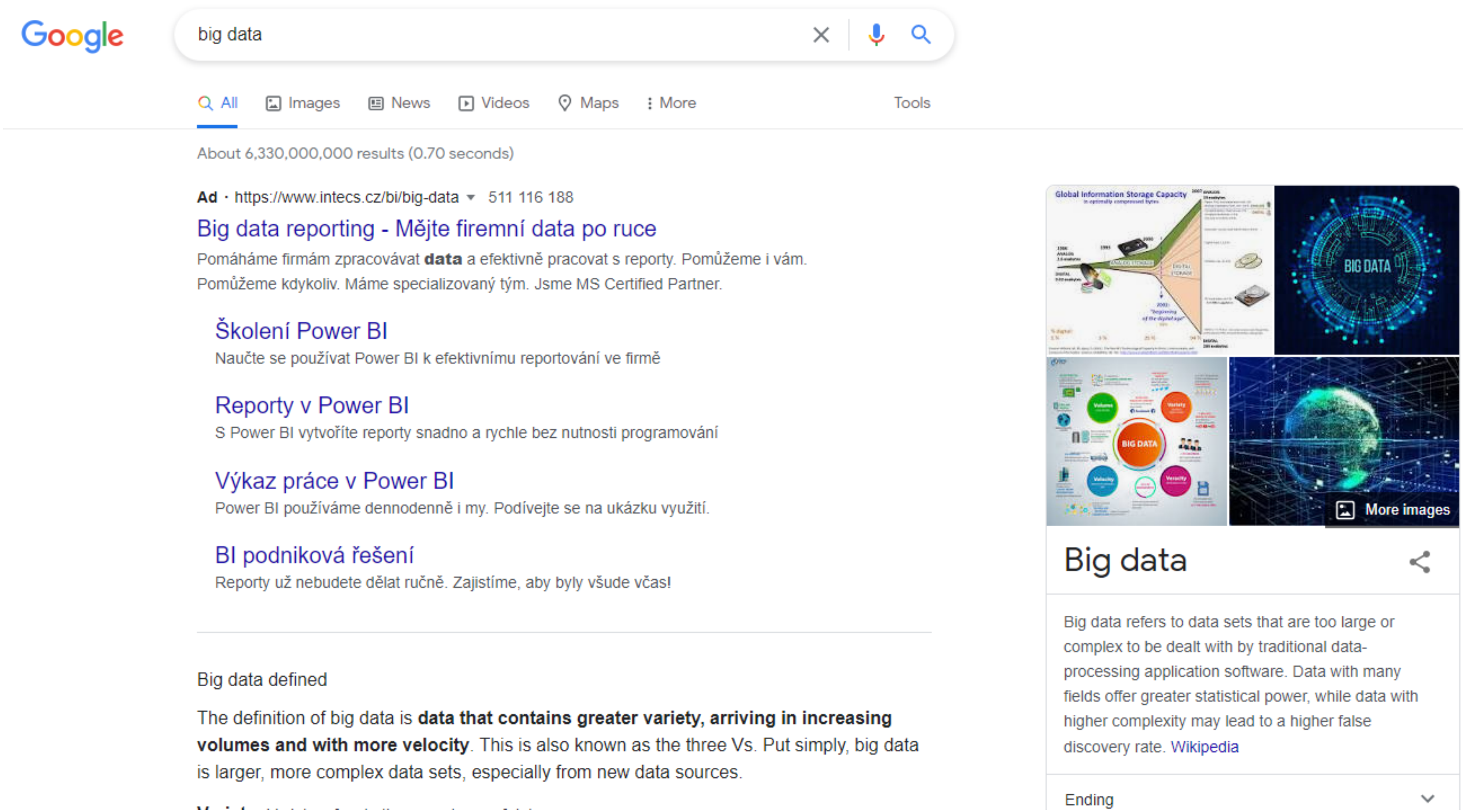
Employee_ID	Employee_Name	Gender	Department	Salary_In_lacs
2365	Rajesh Kulkarni	Male	Finance	650000
3398	Pratibha Joshi	Female	Admin	650000
7465	Shushil Roy	Male	Admin	500000
7500	Shubhojit Das	Male	Finance	500000
7699	Priya Sane	Female	Finance	550000

Semi-structured Data (no or little schema)

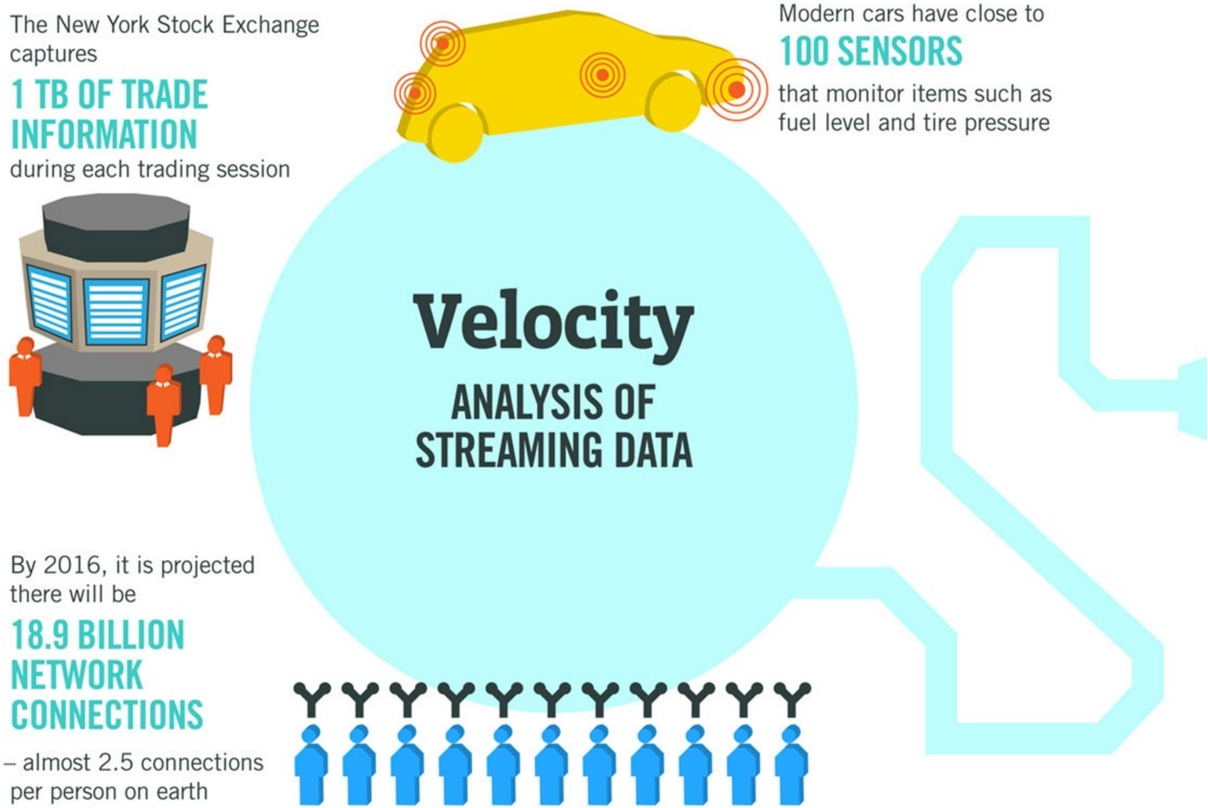

```
{
    {
        "Employee_ID" : 2365,
        "Employee_Name" : "Rajesh Kulkarni",
        "Gender" : "Male",
        "Department" : "Finance”,
        "Salary" : 650000,
        "Phone" : "666555444"
    },
    {
        "Employee_ID" : 3398,
        "Employee_Name" : "Pratibha Joshi",
        "Gender" : "Female",
        "Department" : "Admin",
        "Salary" : 650000,
    }
}
```

Unstructured Data

- Any data with unknown form or structure is classified as unstructured data.



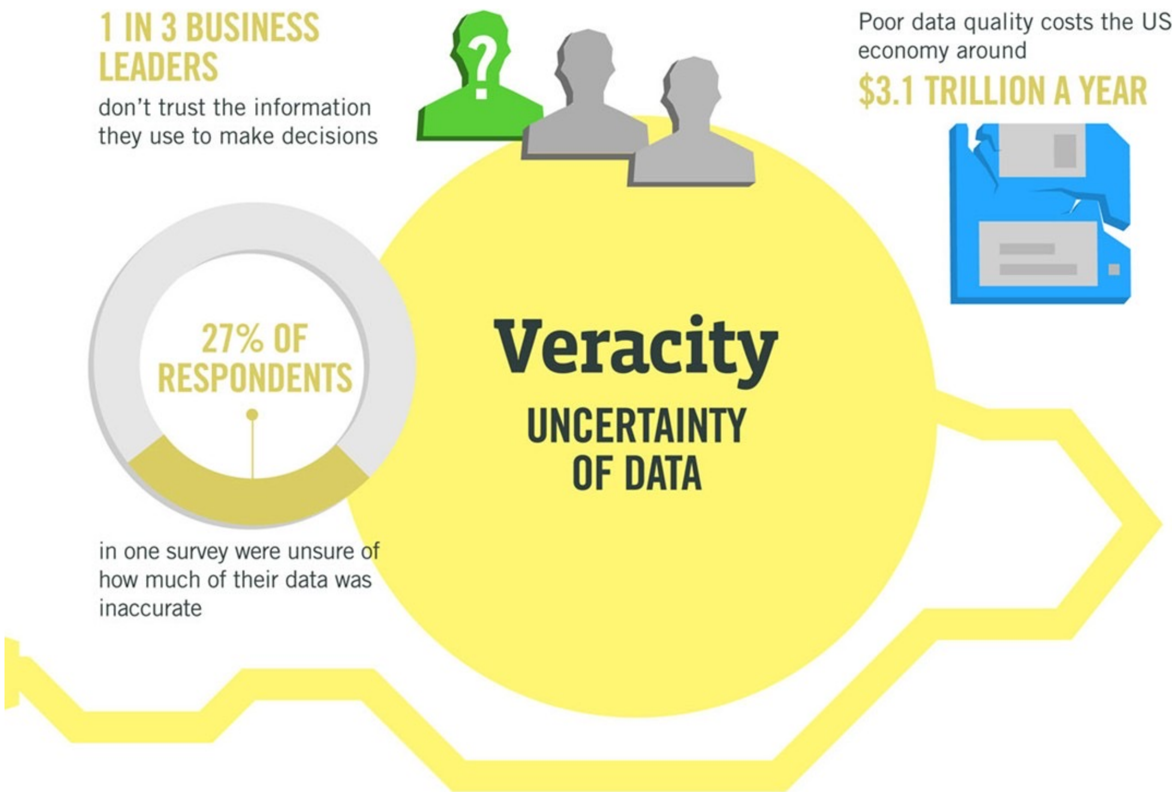
Velocity refers to the speed at which large datasets are acquired, processed, and accessed



Source: http://www.ibmbigdatahub.com/

Big Data Characteristics : Veracity (Uncertainty)

Veracity refers to the quality and accuracy of data. Big data can be noisy and uncertain, full of biases, abnormalities, and imprecision. Low veracity can greatly damage the accuracy of your results.



Source: http://www.ibmbigdatahub.com/

Big Data Characteristics : Three C

- **Cardinality**
  - the number of records in the dynamically growing dataset at a particular instance
- **Continuity**
  - two characteristics and they are: (i) representation of data by continuous functions, and (ii) continuous growth of data size with respect to time
- **Complexity**
  - three characteristics and they are: (i) large varieties of data types, (ii) high dimensional dataset; and (iii) the speed of data processing is very high

Big Data Characteristics : Additional V

- **Value**

  The business value of the data (needs to be revealed)

- **Validity**

  Data correctness and accuracy with respect to the intended use

- **Volatility**

  Period of time the data is valid and should be maintained

Big Data Characteristics : Additional V

Source: https://www.xenonstack.com/blog/big-data-engineering/ingestion-processing-big-data-iot-stream/

Comparison between traditional data and big data

	Traditional data	Big data
Volume	In GBs	TBs and PBs
Data generation rate	Per hour; per day	More rapid
Data structure	Structured	Semi-structured or Unstructured
Data source	Centralized	Fully distributed
Data integration	Easy	Difficult
Data store	RDBMS	HDFS, NoSQL
Data access	Interactive	Batch or near real-time

Source: Furht, Borko, and Flavio Villanustre. "Introduction to big data."

Relational databases : data model

✓ **Relational model**: Structured data is stored in **tables** with **rows** and **columns**

- Each row represents a record with a **unique key**
- Columns hold attributes of data.

Students

ID	Name	Phone	DateOfBirth	Sex
500	Alexander	666-555-444	06.03.2000	M
501	Roman	777-666-555	23.08.1999	M
502	Tereza	555-666-777	14.05.2000	F

All data must follow this schema.

Relational databases : relationships

- Relational databases allow you to define **relationships** between different data sets.
- **Foreign keys** are used to define the relationships among the tables.

Students

ID	Name	Phone	DateOfBirth	Sex
500	Alexander	666-555-444	06.03.2000	M
501	Roman	777-666-555	23.08.1999	M
502	Tereza	555-666-777	14.05.2000	F

ID	CourseID	Grade
500	1001	B
501	1002	A
501	1003	B
502	1002	C

Takes_Course

CourseID	Title	Credits
1001	Data Mining	5
1002	Artificial Intelligence	6
1003	Database systems	5

Courses

Relational databases : SQL

Relational databases use **Standard Query Language (SQL)** as the standard interface for querying and manipulating data.

SELECT id, name, price **FROM** products

Representatives

- Oracle Database, Microsoft SQL Server, IBM DB2
- MySQL, PostgreSQL

Relational databases provide powerful tools for querying and analysing data, which can be used to generate reports, discover trends, and make informed decisions.

Selection is based on complex conditions, **projection**, **joins**, **aggregation**, derivation of new values, recursive queries, …

Relational databases : Normal forms

Model

- Functional dependencies
- 1NF, 2NF, 3NF, BCNF (Boyce-Codd normal form)

Objective

- **Normalization of database schema** to BCNF or 3NF
- Algorithms: decomposition or synthesis

Motivation

- Diminish **data redundancy**, prevent update anomalies
- However:

  Data is scattered into small pieces (high granularity), and so these pieces have to be joined back together when querying!

Relational databases : Transactions

- **Transaction** = flat sequence of database operations (READ, WRITE, COMMIT, ABORT)

Objectives

- Enforcement of ACID properties
- **Efficient parallel / concurrent execution** (slow hard drives, …)

ACID properties

- **A**tomicity – partial execution is not allowed (all or nothing)
- **C**onsistency – transactions turn one valid database state into another
- **I**solation – uncommitted effects are concealed among transactions
- **D**urability – effects of committed transactions are permanent

Current Trends

Big Data

- **Volume**: terabytes → zettabytes
- **Variety**: structured → semi-structured and unstructured data
- **Velocity**: batch processing → streaming data

Big users

- Population online, hours spent online, devices online, …
- Rapidly growing companies / web applications
  - Even millions of users within a few months

Current Trends

Everything is in the **cloud**

- **SaaS:** Software as a Service
- **PaaS:** Platform as a Service
- **IaaS:** Infrastructure as a Service

Processing paradigms

- **OLTP:** Online Transaction Processing
- **OLAP:** Online Analytical Processing
- *…but also…*
- **RTAP:** **Real-Time Analytical Processing**

Current Trends

Data assumptions

- **Data format** is becoming unknown or inconsistent
- Linear growth → **unpredictable exponential growth**
- **Read requests** often prevail **write requests**
- Data updates are no longer frequent
- Data is expected to be replaced
- Strong **consistency** is no longer mission-critical

Current Trends

⟹ **New approach is required**

- Relational databases simply do not follow the current trends

Key technologies

- Distributed **file systems**
- **MapReduce** and other programming models
- Grid computing, cloud computing
- **NoSQL databases**
- Data warehouses
- Large scale machine learning

What does **NoSQL** actually mean?

- Not: *no to SQL*
- Not: *not only SQL*
- NoSQL is an **accidental term with no precise definition**

NoSQL Databases

What does **NoSQL** actually mean?

NoSQL movement = The whole point of **seeking alternatives** is that you need to solve a problem that **relational databases are a bad fit for**

NoSQL databases = Next generation databases mostly addressing some of the points: being

- ✓ **non-relational**,
- ✓ **distributed**,
- ✓ **open-source**,
- ✓ **horizontally scalable**.

The original intention has been modern web-scale databases. Often more characteristics apply as: **schema-free**, **easy replication support**, **simple API**, **eventually consistent**, a **huge data amount**, and more.

Source: http://nosql-database.org/

NoSQL: typical applications

Some typical applications that use NoSQL:

– Social media (Facebook, etc.)

– Web links (Google search)

– Marketing and sales (Amazon, etc.)

– Interactive maps (Google maps, etc.)

– Email (Gmail, etc.)

– Ontologies and Knowledge Graphs (Equinor, Bosch, etc.)

Types of NoSQL Databases

Core types

- **Key-value** stores
- **Wide column** (column family, column oriented, …) stores
- **Document** stores
- **Graph** databases

Non-core types

- **Object** databases
- Native **XML** databases
- **RDF** stores
- …

Types of NoSQL Databases: Key-Value Stores

Data model

- The most simple NoSQL database type
  - Works as a simple hash table (mapping)
- **Key-value pairs**
  - Key (id, identifier, primary key)
  - Value: binary object, black box for the database system

Query patterns

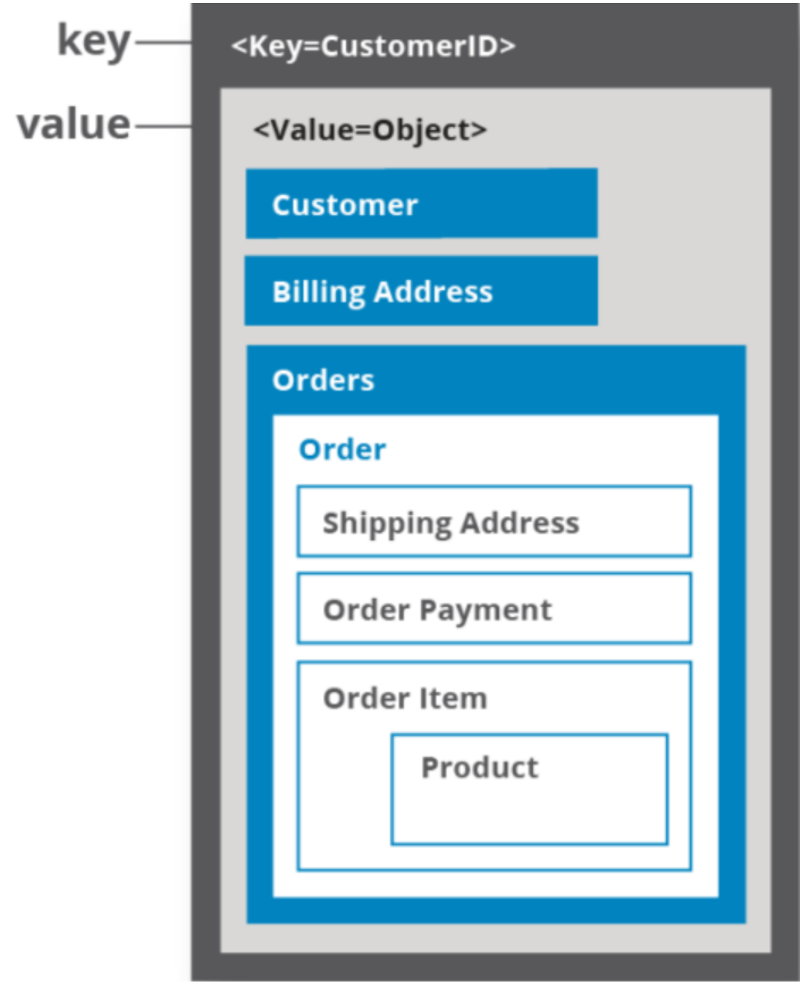
- Create, update or remove value for a given key
- **Get value** for a given key

Characteristics

- Simple model ⇒ **great performance, easily scaled, …**
- Simple model ⇒ **not for complex queries nor complex data**

Types of NoSQL Databases: Key-Value Stores

key	value
123	123 Main St.
126	(805) 477-3900



Source: https://hazelcast.com

Types of NoSQL Databases: Key-Value Stores

Suitable use cases

- Session data, user profiles, user preferences, shopping carts, …

  I.e. **when values are only accessed via keys**

When not to use

- **Relationships among entities**
- Queries requiring **access to the content of the value part**
- **Set operations** involving multiple key-value pairs

Representatives

- **Redis**, **MemcachedDB**, **Riak KV**, Hazelcast, Ehcache, Amazon SimpleDB, Berkeley DB, Oracle NoSQL, Infinispan, LevelDB, Ignite, Project Voldemort
- *Multi-model*: OrientDB, ArangoDB

Types of NoSQL Databases: Document Stores

Data model

- **Documents**
  - Self-describing
  - **Hierarchical tree structures** (JSON, XML, …)
    - Scalar values, maps, lists, sets, nested documents, …
  - Identified by a **unique identifier** (key, …)
- Documents are **organized into collections**

Query patterns

- Create, update or remove a document
- **Retrieve documents according to complex query conditions**

Observation

- Extended key-value stores where the value part is examinable!

Types of NoSQL Databases: Document Stores

```
{
  "title" : "Medvídek",
  "year" : 2007,
  "actors" : [
    {
      "firstname" : "Jiří",
      "lastname" : "Macháček"
    },
    {
      "firstname" : "Ivan",
      "lastname" : "Trojan"
    }
  ],
  "director" :
    { "firstname" : "Jan",
      "lastname" : "Hřebejk"
    }
}
```

Types of NoSQL Databases: Document Stores

Suitable use cases

- Event logging, content management systems, blogs, web analytics, e-commerce applications, …
  - I.e. **for structured documents with similar schema**

When not to use

- **Set operations** involving multiple documents
- The design of document structure is constantly changing
  - I.e. when the required level of granularity would outbalance the advantages of aggregates

Types of NoSQL Databases: Document Stores

Representatives

- **<u>MongoDB</u>**, **Couchbase**, Amazon **DynamoDB**, **CouchDB**, RethinkDB, RavenDB, Terrastore
- *Multi-model*: **MarkLogic**, **OrientDB**, OpenLink Virtuoso, ArangoDB

Types of NoSQL Databases: Wide Column Stores

Data model

- Column family (table)
  - Table is a collection of **similar rows** (not necessarily identical)
- Row
  - Row is a collection of **columns**
    - Should encompass a group of data that is accessed together
  - Associated with a unique **row key**
- Column
  - Column consists of a **column name** and **column value** (and possibly other metadata records)
  - Scalar values, but also **flat sets, lists or maps** may be allowed

Types of NoSQL Databases: Wide Column Stores

Query patterns

- Create, update or remove a row within a given column family
- **Select rows according to a row key or <u>simple</u> conditions**

Warning

- Wide column stores are <u>not just a special kind of RDBMSs</u> with a variable set of columns!

Types of NoSQL Databases: Wide Column Stores

Suitable use cases

- Event logging, content management systems, blogs, …
  - I.e. **for structured flat data with similar schema**

When not to use

- **ACID transactions** are required
- **Complex queries**: aggregation (SUM, AVG, …), joining, …
- Early prototypes: i.e. when **database design may change**

Representatives

- Apache **Cassandra**, Apache **HBase**, Apache Accumulo, Hypertable, **Google Bigtable**

Types of NoSQL Databases: Graph Databases

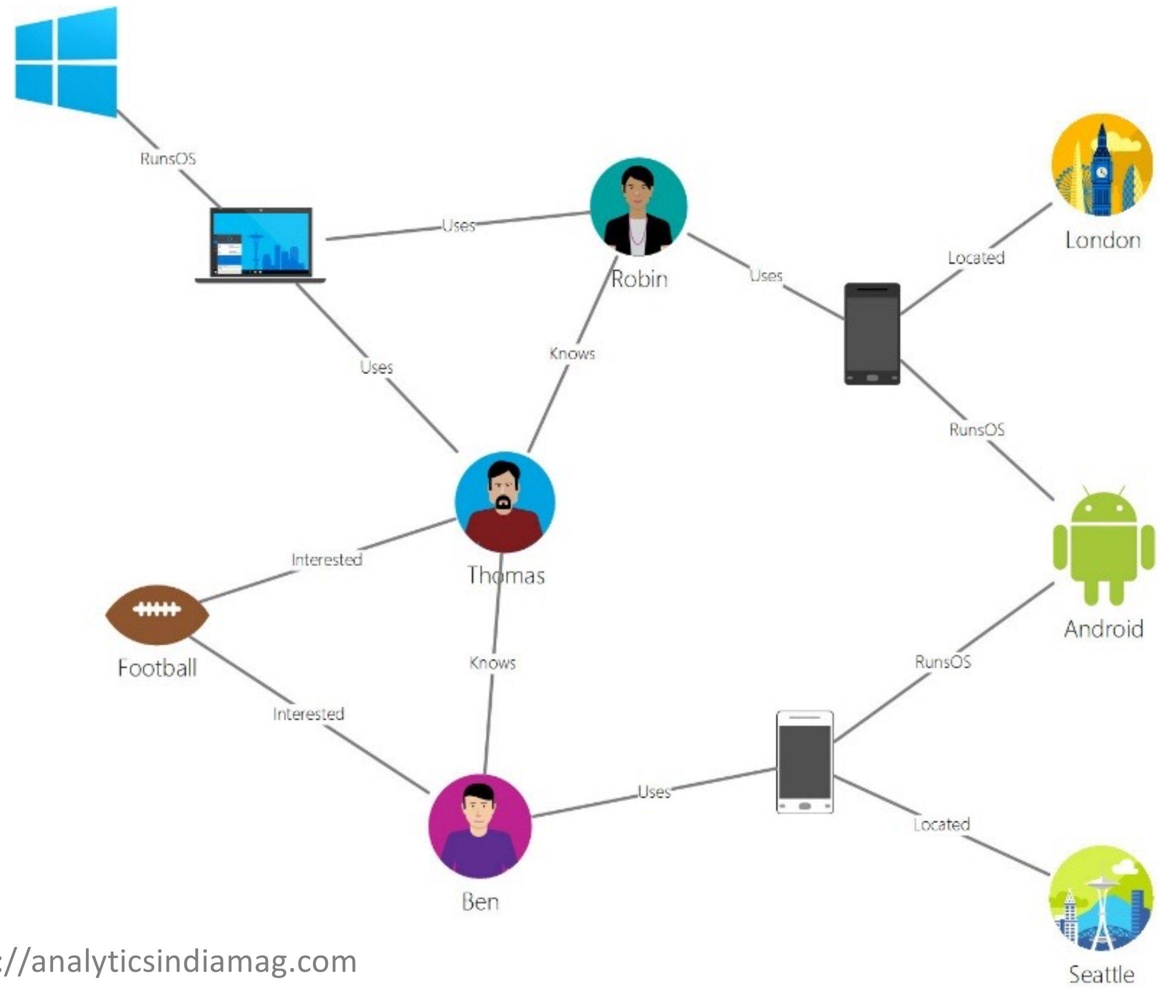
Data model

- Property graphs
  - **Directed / undirected graphs**, i.e. collections of …
    - nodes (vertices) for real-world entities, and
    - relationships (edges) between these nodes
  - Both the nodes and relationships can be associated with additional properties

Types of databases

- **Non-transactional** = small number of very large graphs
- **Transactional** = large number of small graphs

Types of NoSQL Databases: Graph Databases



Source: https://analyticsindiamag.com

Types of NoSQL Databases: Graph Databases

Query patterns

- Create, update or remove a node / relationship in a graph
- **Graph algorithms** (shortest paths, spanning trees, …)
- General **graph traversals**
- **Sub-graph** queries or **super-graph** queries
- Similarity based queries (approximate matching)

Representatives

- **Neo4j**, **Titan**, Apache Giraph, InfiniteGraph, FlockDB
- *Multi-model*: **OrientDB**, OpenLink **Virtuoso**, **ArangoDB**

Types of NoSQL Databases: Graph Databases

Suitable use cases

- Social networks, routing, dispatch, and location-based services, recommendation engines, chemical compounds, biological pathways, linguistic trees, …
    - I.e. simply **for graph structures**

When not to use

- **Extensive batch operations** are required
    - Multiple nodes / relationships are to be affected
- **Only too large graphs** to be stored
    - Graph distribution is difficult or impossible at all

Types of NoSQL Databases: Native XML Databases

Data model

- **XML documents**
  - Tree structure with nested elements, attributes, and text values (beside other less important constructs)
  - Documents are organized into collections

Query languages

- **XPath**: *XML Path Language* (navigation)
- **XQuery**: *XML Query Language* (querying)
- **XSLT**: *XSL Transformations* (transformation) Representatives
- **Sedna**, **Tamino**, BaseX, eXist-db
- *Multi-model*: **MarkLogic**, OpenLink **Virtuoso**

Types of NoSQL Databases: Native XML Databases

```
<?xml version = "1.0"?>
<contact-info>
   <contact1>
      <name>Martin Novotny</name>
      <company>ABC group</company>
      <phone>(420) 555-6667</phone>
   </contact1>

   <contact2>
      <name>Filip Vesely</name>
      <company>ABC group</company>
      <phone>(420) 666-5667</phone>
   </contact2>
</contact-info>
```

MarkLogic™

OPENLINK VIRTUOSO UNIVERSAL SERVER

sedna Native XML Database System

Tamino XML Server

existdb

Types of NoSQL Databases: RDF Stores

Data model

- **RDF triples**
  - Components: **subject**, **predicate**, and **object**
  - Each triple represents a statement about a real-world entity
- Triples can be viewed as **graphs**
  - **Vertices** for subjects and objects
  - **Edges** directly correspond to individual statements

Query language

- **SPARQL**: *SPARQL Protocol and RDF Query Language*

Representatives

- Apache **Jena**, **rdf4j** (Sesame), Algebraix
- *Multi-model*: **MarkLogic**, OpenLink **Virtuoso**

Features of NoSQL Databases

Data model

- Traditional approach: relational model
- (New) possibilities:
  - **Key-value**, **document**, **wide column**, **graph**
  - Object, XML, RDF, …
- Goal
  - Respect the real-world nature of data
    (i.e. data structure and mutual relationships)

NoSQL Databases: Aggregate structure

- Aggregate definition
  - Data unit with a complex structure
  - **Collection of related data pieces we wish to treat as a unit** (with respect to data manipulation and data consistency)
- Examples
  - **Value** part of key-value pairs in key-value stores
  - **Document** in document stores
  - **Row** of a **column family** in wide column stores
- Types of systems
  - **Aggregate-ignorant**: relational, graph
    - It is not a bad thing, it is a feature
  - **Aggregate-oriented**: key-value, document, wide column
- Design notes
  - No universal strategy how to draw **aggregate boundaries Atomicity** of database operations: just a single aggregate at a time

Features of NoSQL Databases

Elastic scaling

- Traditional approach: scaling-up
  - Buying bigger servers as database load increases
- New approach: scaling-out
  - Distributing database data across multiple hosts
    - Graph databases (unfortunately): difficult or impossible at all

Data distribution

- Sharding
  - Particular ways how database data is split into separate groups
- Replication
  - Maintaining several data copies (performance, recovery)

Features of NoSQL Databases

Automated processes

- Traditional approach

  Expensive and highly trained database administrators

- New approach: **automatic recovery, distribution, tuning, …**

Relaxed consistency

- Traditional approach

  Strong consistency (ACID properties and transactions)

- New approach

  Eventual consistency only (BASE properties)

  I.e. we have to make trade-offs because of the data distribution

Features of NoSQL Databases

Schemalessness

- Relational databases
  - Database schema present and **strictly enforced**
- NoSQL databases
  - **Relaxed schema** or **completely missing**
  - Consequences: **higher flexibility**
    - Dealing with **non-uniform data**
    - **Structural changes** cause no overhead
  - However: there is (usually) an **implicit schema**
    - We must know the data structure at the application level anyway

Open source

- Often community and enterprise versions (with extended features or extent of support)

Simple APIs

- Often state-less application interfaces (HTTP)

Current State: Five advantages

- **Scaling**
  - Horizontal distribution of data among hosts
- **Volume**
  - High volumes of data that cannot be handled by RDBMS
- **Administrators**
  - No longer needed because of the automated maintenance
- **Economics**
  - Usage of cheap commodity servers, lower overall costs
- **Flexibility**
  - Relaxed or missing data schema, easier design changes

Current State: Five challenges

- Maturity
  - Often still in the pre-production phase with key features missing
- Support
  - Mostly open source, limited sources of credibility
- Administration
  - Sometimes relatively difficult to install and maintain
- Analytics
  - Missing support for business intelligence and ad-hoc querying
- Expertise
  - Still a low number of NoSQL experts available in the market

Conclusion

The end of relational databases?

- Certainly no
  - They are still suitable for most projects
  - Familiarity, stability, feature set, available support, …
- However, we should also consider different database models and systems
  - Polyglot persistence = **usage of different data stores in different circumstances**

Lecture Conclusion

Big Data

- 4V characteristics: volume, variety, velocity, veracity

NoSQL databases

- (New) **logical models**
  - Core: key-value, wide column, document, graph Non-core: XML, RDF, …
- (New) **principles and features**
  - Horizontal scaling, data sharding and replication, eventual consistency, …