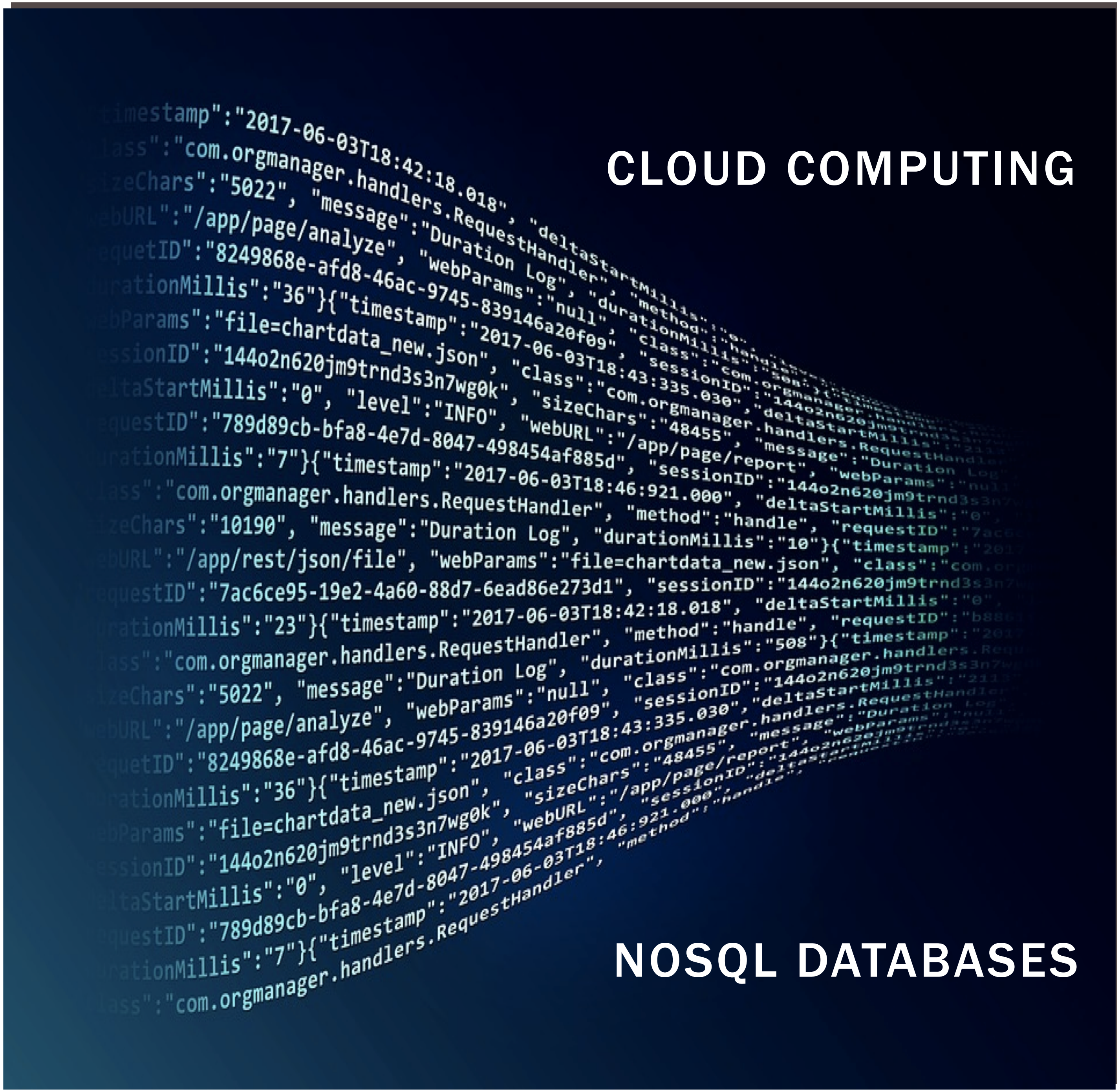


CLOUD COMPUTING

NOSQL DATABASES

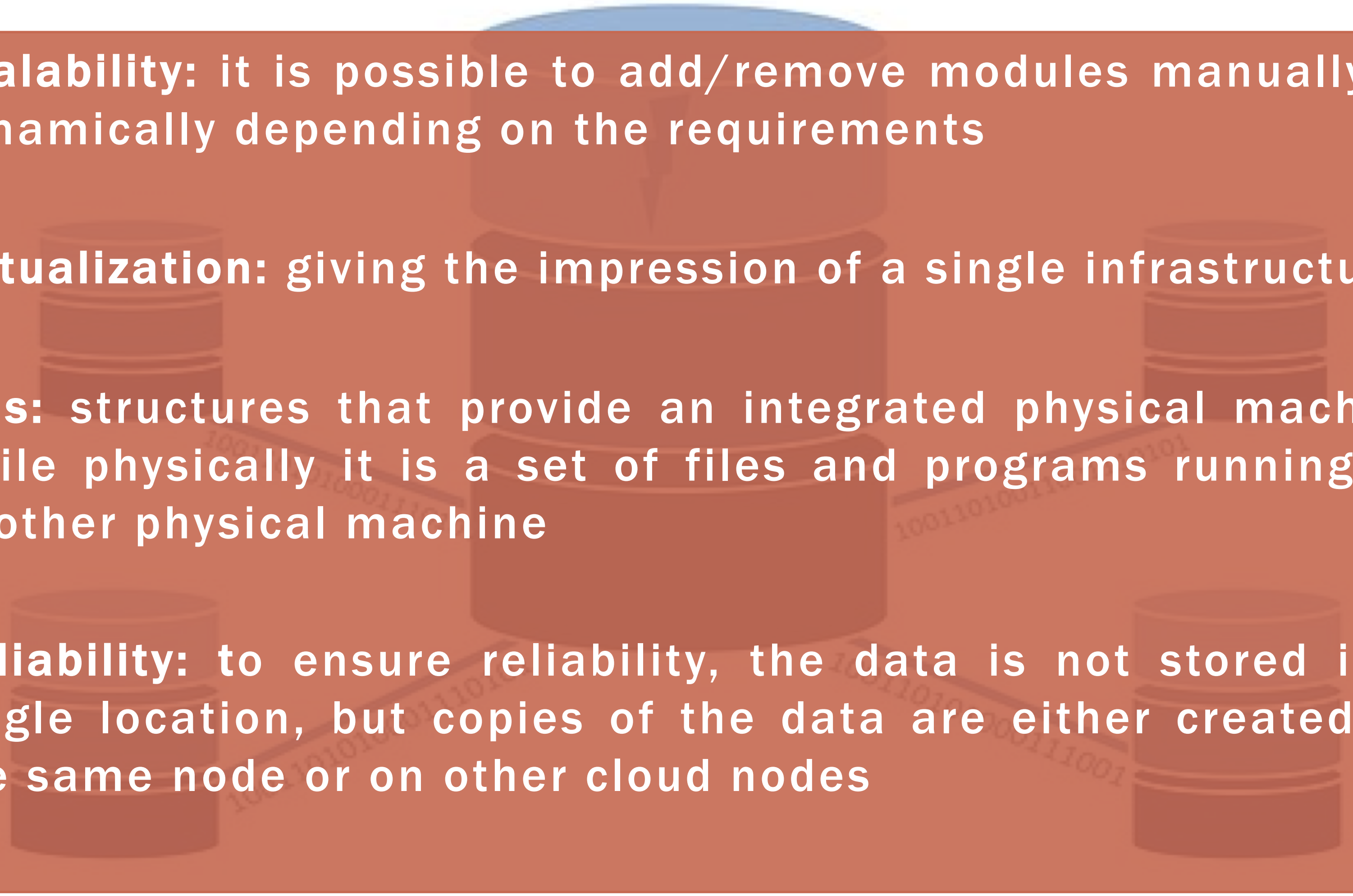
INTRODUCTION TO CLOUD COMPUTING

- Cloud Computing: providing computing resources (storage, time to CPU, memory, files, networks, servers, applications and services) and / or services (remote execution)

- Upgrading and maintaining the infrastructure and provided applications is the responsibility of the provider and not the user

- Users just need to connect to the network, from any device, PC, tablet, smartphone, and can instantly use applications

CLOUD COMPUTING CHARACTERISTICS 1/3



- **Scalability:** it is possible to add/remove modules manually or dynamically depending on the requirements
- **Virtualization:** giving the impression of a single infrastructure
- **VMs:** structures that provide an integrated physical machine while physically it is a set of files and programs running on another physical machine
- **Reliability:** to ensure reliability, the data is not stored in a single location, but copies of the data are either created on the same node or on other cloud nodes

CLOUD COMPUTING CHARACTERISTICS 2/3

- **Maintenance:** correcting applications from programming errors and installing new versions of the application centralized

- **Performance:**
  - Low response time
  - High data transfer
  - Low use of computing resources
  - High availability of the system
  - Fast data processing

- **On demand self-service:** fast and easy hardware engagement and customization for application development, without intermediate human interaction

CLOUD COMPUTING CHARACTERISTICS 3/3

- **Security:** mechanisms that protect data and machines from accidental and unauthorized access, change or destruction, and from natural disasters

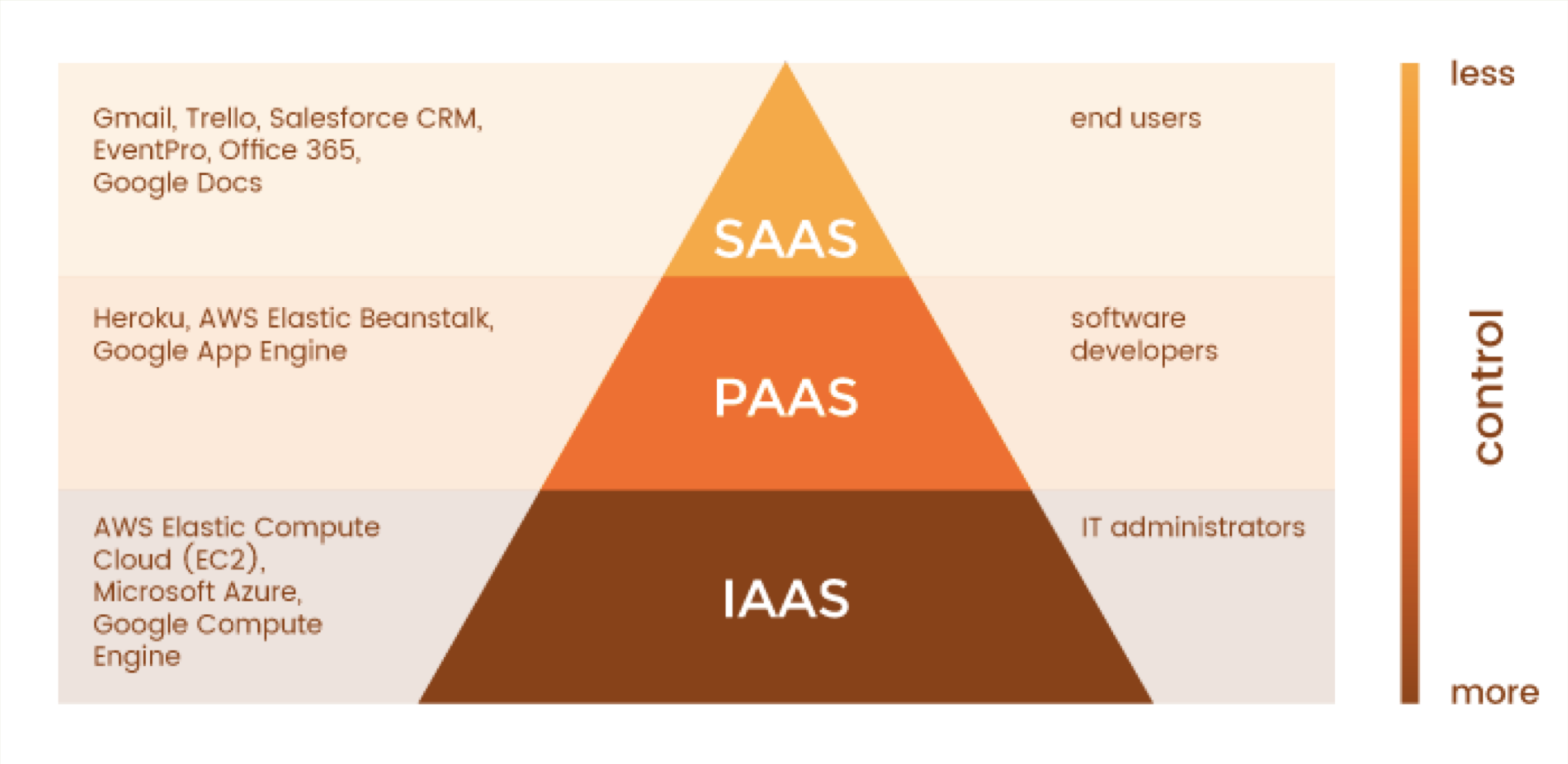
- **Multitenancy:** every user has a single instance of the application tailored to his needs

- **Cloud API:** providing REST architectures to communicate with other applications and machines

- **Reduced costs:** pricing the use of computing resources according to the type of commitment (pay-as-you-go charging policy)

CLOUD COMPUTING SERVICES

The basic idea of Cloud Computing is to provide services to the end users. These services can be separated into the following three layers


Gmail, Trello, Salesforce CRM, EventPro, Office 365, Google Docs
SAAS
end users
Heroku, AWS Elastic Beanstalk, Google App Engine
PAAS
software developers
AWS Elastic Compute Cloud (EC2), Microsoft Azure, Google Compute Engine
IAAS
IT administrators
less
control
more

Choosing the Right Cloud Service: IaaS, PaaS, or SaaS, (https://rubygarage.org/blog/iaas-vs-paas-vs-saas)

IAAS (INFRASTRUCURE AS A SERVICE) 1/2

- In this service, users are **provided with infrastructure**, namely computers, servers and computing resources, either as machines or more often as virtual machines

- The environment must be **capable** of being used to implement **PaaS and SaaS services**

The service provider is responsible for:

- - **Maintenance of machines** in which users' VMs run
- - **Safety** of user data
- - **Reliability** of the machines so that in the event of failure the performance of the service will not be altered
- - **Ensure expandability** and **dynamic commitment** of resources

Users can:

- - **Define partially or fully** the characteristics of the VMs they handle (CPU, RAM, hard disk, operating system, network and more)
- - **Management** of VMs with absolute freedom

IAAS (INFRASTRUCURE AS A SERVICE) 2/2

- An example of IaaS is the **Rackspace Cloud**, which was one of the first IaaS providers

Has the following **3 main services:**

- **- Cloud Files:** They are the storage space for users. Access is through an open source API. The data is kept in three copies
- **- Cloud Servers:** Deploy and run servers through a server API. Dynamic machinery delivery is supported in abrupt changes in workload
- **- Cloud Sites:** Users keep an unlimited number of web pages, emails and databases

PAAS (PLATFORM AS A SERVICE) 1/2

It provides users with **computational platforms on which they implement applications**

Current benefits include:

- Installed operating system
- Planning environment
- Web server
- Design tools and testing
- Security and reliability of data
- Automatic scaling

PAAS (PLATFORM AS A SERVICE) 2/2

At the PaaS level, users:

- They develop applications or manage applications they have acquired but have **no control over the infrastructure**
- You do not need to keep machines and software, which greatly **reduces application development costs**

PaaS platform examples:

- **Citrix Xen Desktop,**
- **Windows Azure Compute**

SAAS (SOFTWARE AS A SERVICE) 1/2

It provides users with **ready-made** applications **running on the cloud that are accessible through a network**

Both software and data are centrally located in the cloud and available to users through a web browser

The provider is responsible for:

- Setting applications that they offer on-demand users
- Scaling resulting in the workload being allocated according to the requirements
- Develop compatible software versions as the supplied software can be used by various devices, computers, tablets, smartphones
- Reliability and security of user data

SAAS (SOFTWARE AS A SERVICE) 2/2

- Users **have no control over the infrastructure** or platforms on which applications run

- SaaS **no longer requires** the installation and constant updating of the software or the storage commitment to store the data on each user's device

- Example SaaS:
  - **Google Apps (Gmail)**

NAAS (NETWORK AS A SERVICE)

Network as a Service (NaaS) provides **network infrastructure and services**

The service includes:

- Knowledge of network topology so users can choose storage nodes
- Checking the packet switches on network switches,
- Set firewalls by users and
- Edit the data inside the network
- API for easy user access to various services of the NaaS provider

CLOUD COMPUTING DEPLOYMENT MODELS 1/2

When transferring a service into a cloud computing environment, there are many issues need to be addressed according to the needs.

There are four main types of clouds, each with its advantages and disadvantages:

Private cloud:

- - for exclusive use by a single organization with specific users or only a particular user and can be developed either by the organization itself or by external partners.

Public cloud:

- - can grow exponentially relative to the size of a company’s private cloud and hosted more often outside the customer’s premises

CLOUD COMPUTING DEPLOYMENT MODELS 2/2

Hybrid cloud:

- - a combination of two or more cloud types
- - combines fault-tolerant architecture with immediate availability of data - - without the necessary network connection
- - both private and remote server-based infrastructure is required

Community cloud:

- - for use by a specific community of users with a defined and shared purpose.
- - infrastructure can be owned and managed by one or more of the participating organizations, an external provider or even a combination of them

RELATIONAL AND NOSQL DATABASES

The real meaning of this generic term NoSQL is that this type doesn’t follow the principles of the traditional relational databases.

Instead, they have their principles, and they can handle efficiently the output of modern web scale databases like Twitter, Facebook, Google.

twitter

facebook

Google

WHAT DOES NOTSQL MEAN

NoSQL describes a method of data retrieval that is modeled in a different way than traditional **relational databases** like **MySQL** and **PostgreSQL.**

- **NoSQL** databases are a growing industry and is commonly used in **big data** and **real-time web applications.**
- **NoSQL** stands for **“Not Only SQL”** as there may be times where SQL is used.
- A better name (in my opinion) would be **“Non-Relational Database”**

RELATIONAL AND NOSQL DATABASES

Nowadays it is stated that NoSQL databases have a lot more to offer than just offering solutions to scale problems.
In particular, they provide the following advantages:

- The data representation is **schema-less**, and there is no need to define a certain structure from the beginning since you can add new fields at runtime
- The speed even with a small amount of data can be processed in **milliseconds** rather than hundreds of milliseconds
- The **elasticity** of the applications due to the scalability features that NoSQL databases offer
- **Reduce in development time**, and developers don’t have to deal with complex SQL queries and difficult joints to collate the data from different tables into a new view

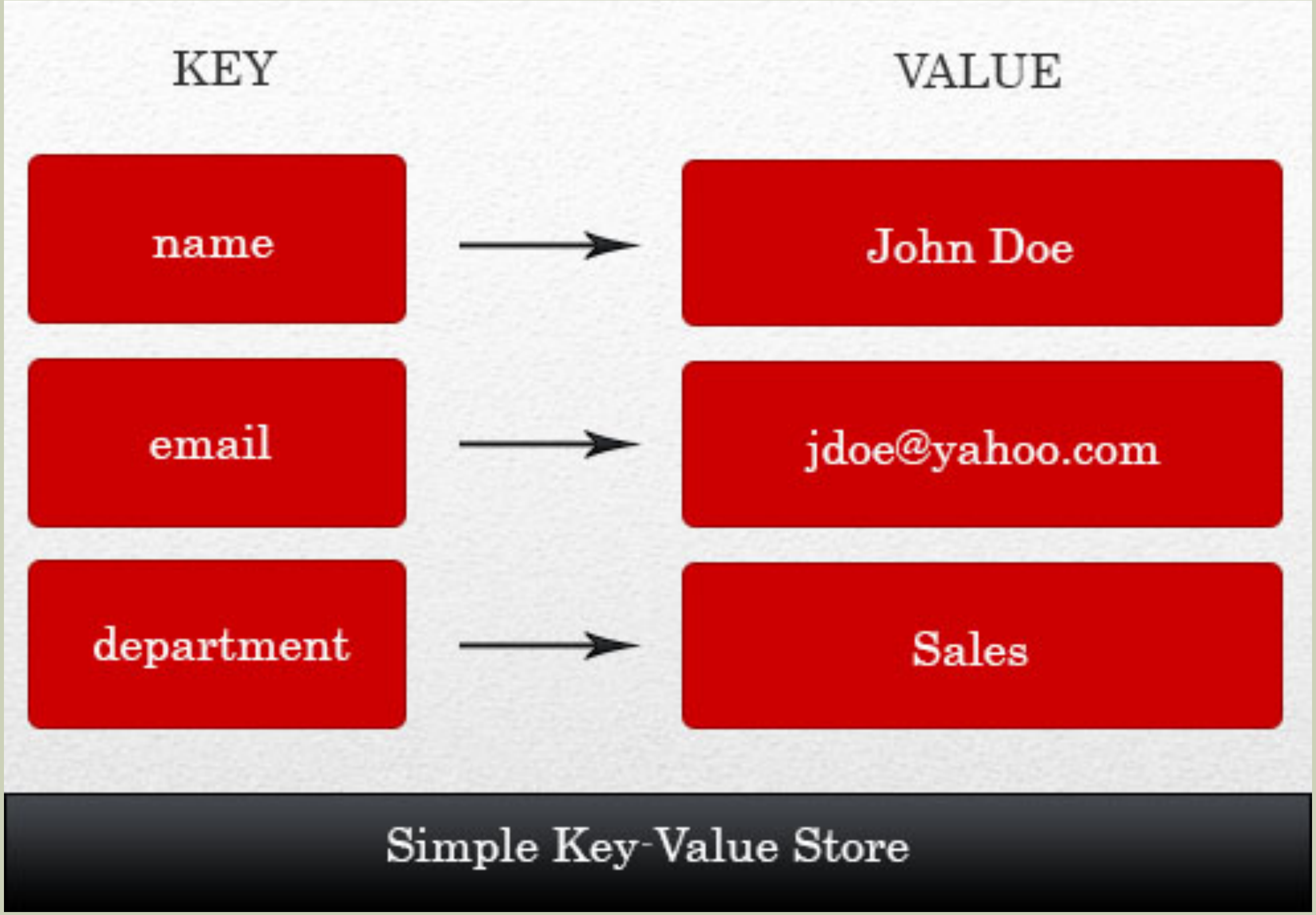
TYPES OF NOSQL DATABASES

4 main types

- Key-Value Stores
- Document Databases
- Column Family Stores
- Graph Databases

APACHE HBASE
Cassandra
riak
CouchDB relax
mongoDB
HYPERTABLE INC
Neo4j
redis

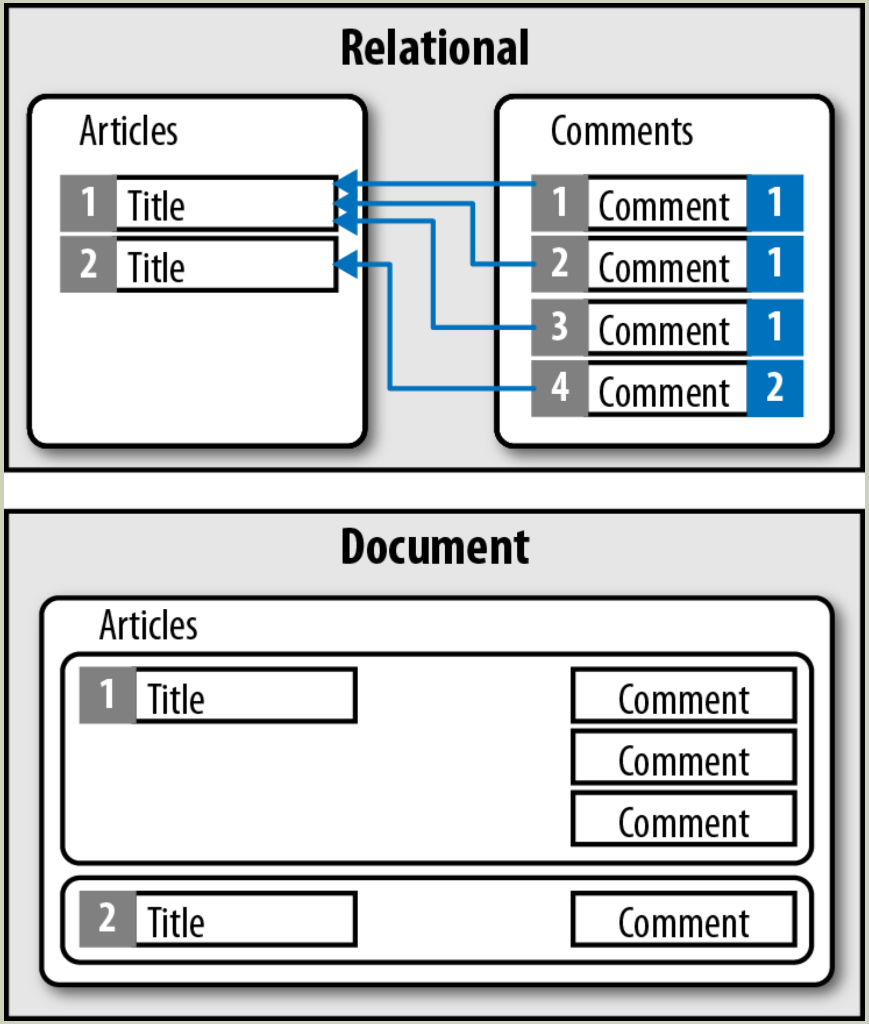
KEY VALUE STORES



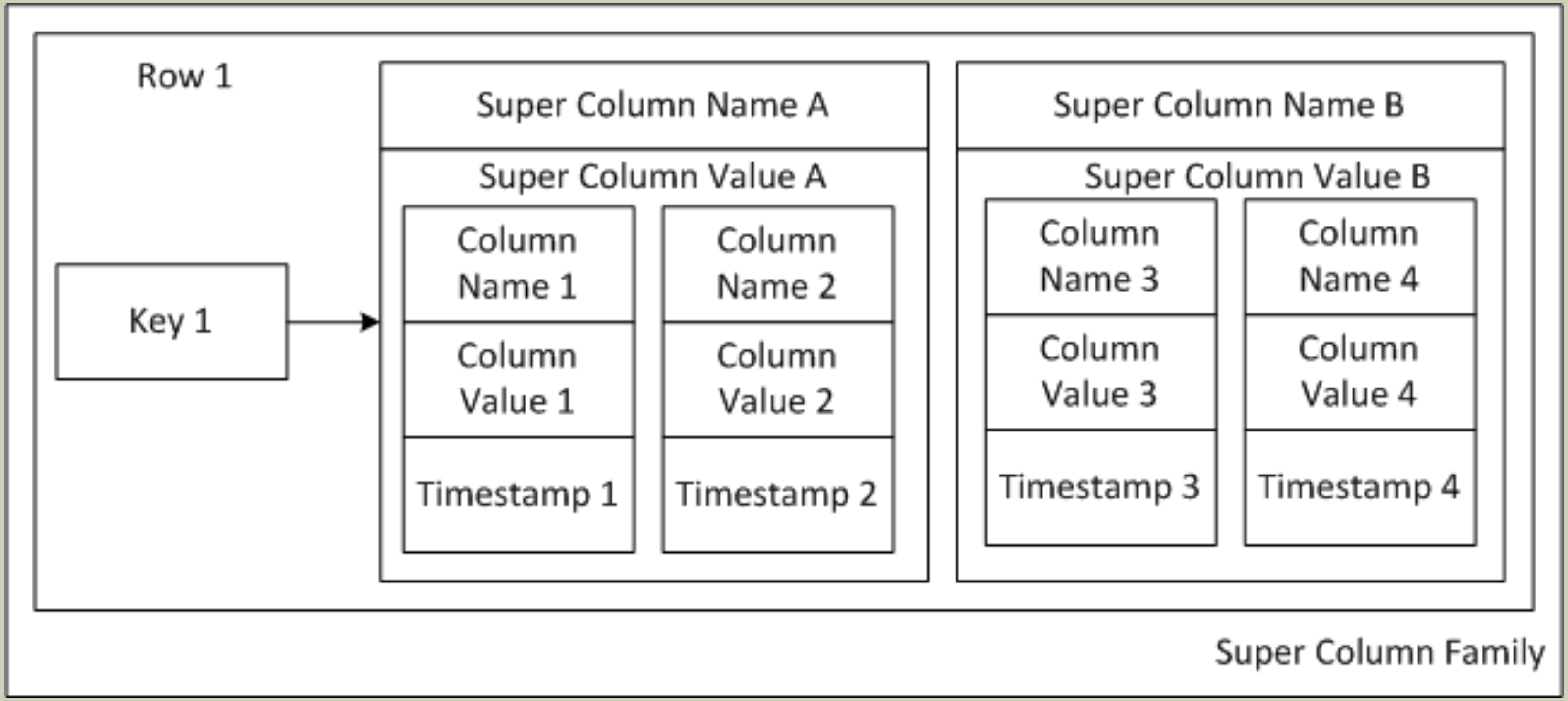
KEY	VALUE
name	John Doe
email	jdoe@yahoo.com
department	Sales

Simple Key-Value Store

DOCUMENT DATABASES



COLUMN STORES



Row 1

Key 1

Super Column Name A

Super Column Value A

Column Name 1

Column Value 1

Timestamp 1

Column Name 2

Column Value 2

Timestamp 2

Super Column Name B

Super Column Value B

Column Name 3

Column Value 3

Timestamp 3

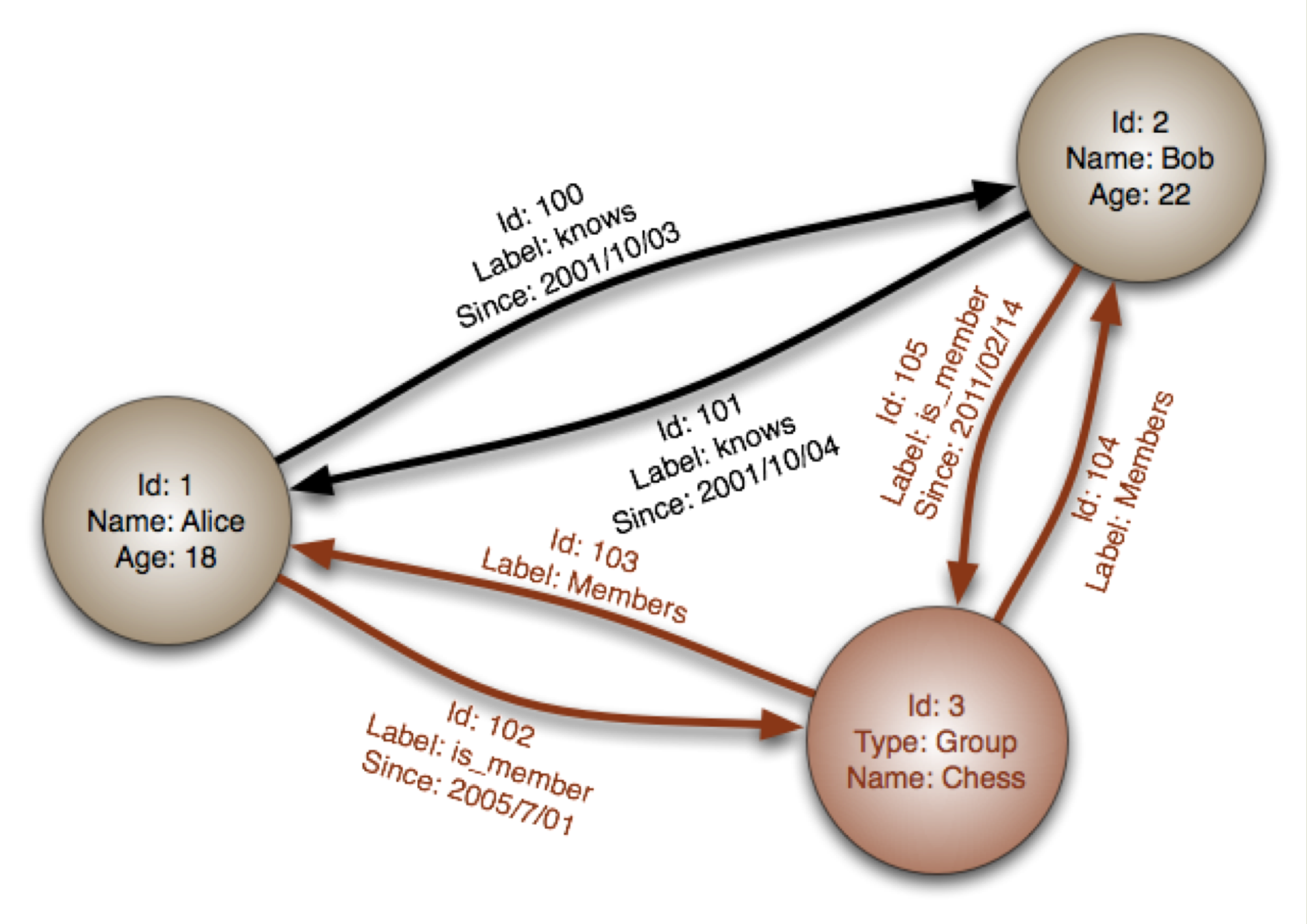
Column Name 4

Column Value 4

Timestamp 4

Super Column Family

GRAPH DATABASES

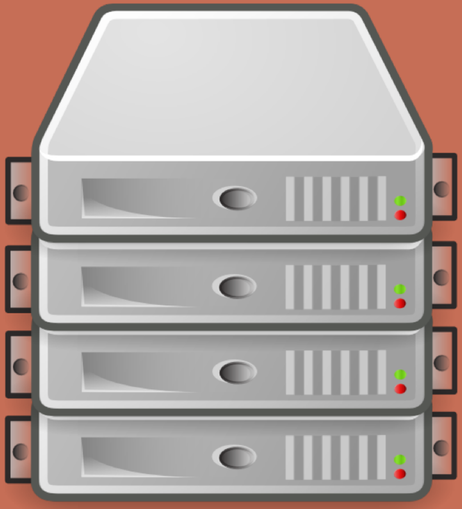


POPULAR NOSQL DATABASES

Name	Year	Type	Developer
MongoDB	2009	Document	10Gen
CouchDB	2005	Document	Apache
Cassandra	2008	Column Store	Apache
CouchBase	2011	Document	Couchbase
Riak	2009	Key-value	Basho Technologies
SimpleDB	2007	Document	Amazon

ADVANTAGES OF NOSQL

- Better Scaling
- Significantly Cheaper
- Schema-less (Partially)
- Less Management
- Cheaper Hardware
- Heterogeneous Data



APACHE CASSANDRA

Apache Cassandra is an open source NoSQL database, which is extensively scalable.

Some of its characteristics are:

- - the provision of availability which is continuous,
- - its scalability which is linear and the simplicity in operating on multiple servers without any single point of failure.

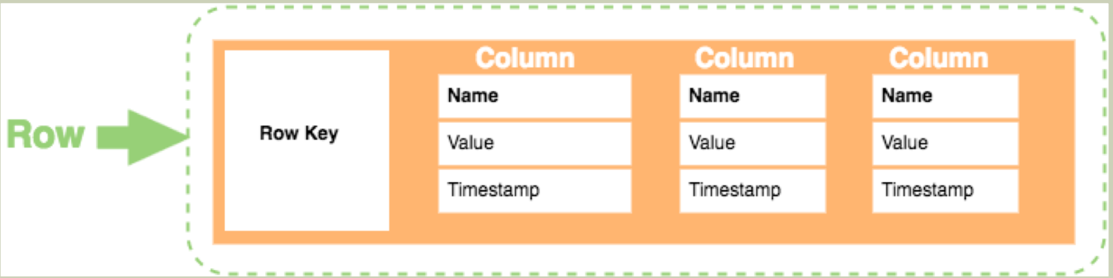
The data model of Cassandra is **extremely flexible** and provides very fast response times.

Cassandra was originally built for Facebook, ensuring zero point of failure and it was designed without master or named nodes, but symmetric peer to peer.

APACHE CASSANDRA

The NoSQL database Cassandra is a row oriented database. The data objects of the schema of Cassandra are the following:

- **Keyspace**: A group of one or more column families
- **Column family:** A collection of rows where each row is an ordered collection of columns.
- **Column (cell):** It is the smallest data model element and storage unit and consists of three values, the name, the value and a timestamp
- **Row:** an upper level compared to columns, it is a group of columns with a unique row key, called a primary key



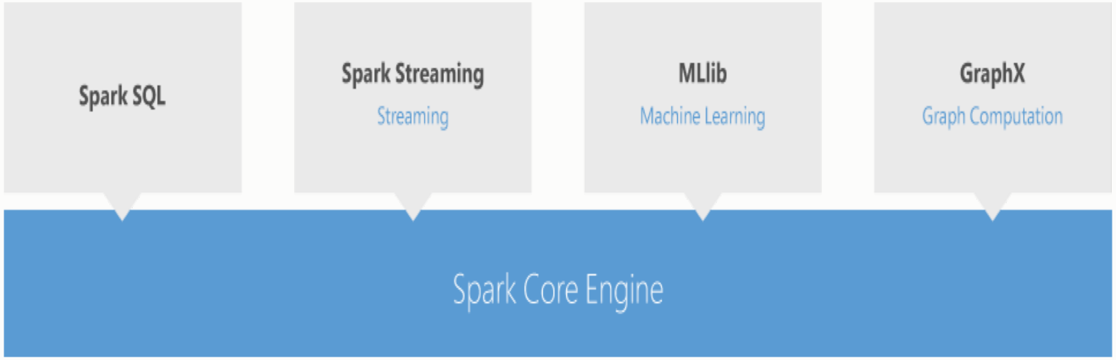
APACHE CASSANDRA

RDBMS	NoSQL Cassandra
Handles incoming data in moderate speed	Handles incoming data in high speed
The source of the data is one or few locations	The source of data is from many locations
Mainly manages data which are structured	Can manage all types of data
Supports complex transactions	Support very simple transactions
Single points of failure exist	No single points of failure exist
Supports a modest volume of data	Supports a huge amount of data
The implementation is centralized	The implementation is descentralized
Data is primarily written in one location	Data can be written in multiple locations
Supports scalability in read, sacrificing consistency	Supports scalability in read and write
Grows in vertical scaling	Grows in horizontal scaling

APACHE SPARK

Apache Spark was developed at UC Berkeley's AMPLab, and later the Spark codebase was donated to the Apache Software Foundation

The core system of Spark consists of different libraries and components that provide a rich set of higher tools including Spark SQL for SQL and structured data processing, MLlib for machine learning, GraphX for processing graphs and Spark Streaming.



APACHE SPARK

- **Spark Core:** The basic engine of the platform, with all the functionality built-in. It includes components capable of task scheduling, management of memory, fault recovery, interaction with the data storage systems and The main block of Spark Core Engine is the Resilient Distributed Dataset (RDD)

- **Spark Streaming:** Live streams of data can be processed with the functions of this Spark component. The importance of this component lays on the ability to analyze not only batch data but also data arriving in real time

- **Machine Learning (MLib):** a framework for machine learning including multiple type of machine learning algorithms like regression, classification, neural network processing and other. This library is based on the RDD API and can be used with R and Python programming languages.

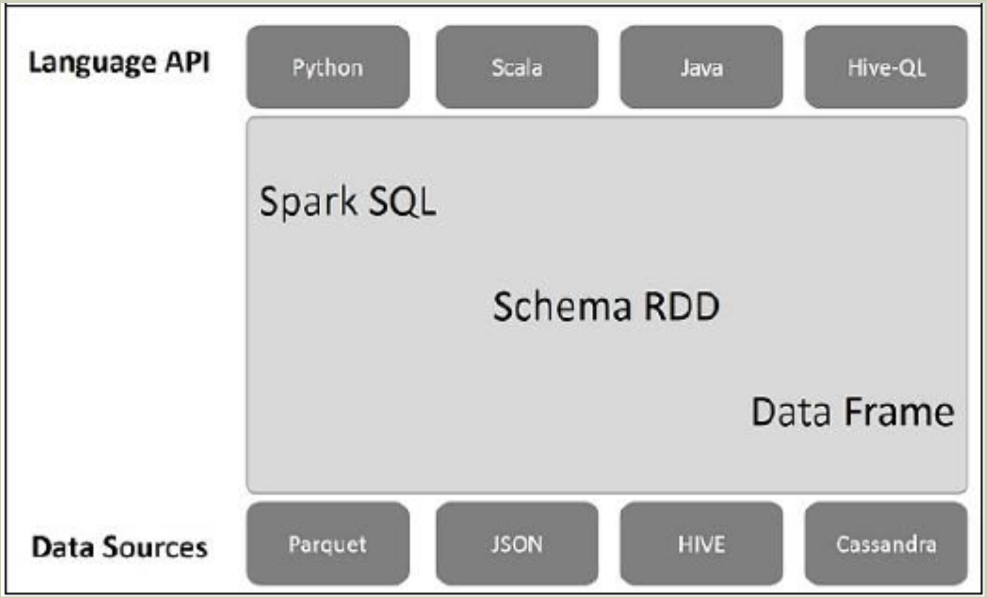
- **Graph Processing (GraphX):** This library is built on top of Spark and is its specialized graph processing system. It enables the composition of graphs with data that are unstructured and tabular

APACHE SPARK

Spark SQL: an increasingly important module of Apache Spark that is used for data processing providing structured data formats manipulation.

Data scientists can perform SQL queries and operations directly on the DataFrames even 100x faster than before.

Datasources can be directly imported to DataFrames, as long as they have their data in structured format.


Language API
Python
Scala
Java
Hive-QL
Spark SQL
Schema RDD
Data Frame
Data Sources
Parquet
JSON
HIVE
Cassandra