Database Overview

Types of data

- Structured Data

  Structured data is generally **tabular** data that is represented by columns and rows in a database.
  Databases that hold tables in this form are called relational databases.
  The mathematical term "**relation**" specify to a formed set of data held as a table.
  In structured data, all row in a table has the same set of columns.
  SQL programming language used for structured data.

- Unstructured Data

Unstructured data is information that either does not organize in a pre-defined manner or does not have a pre-defined data model.
Videos, **audio**, and **binary data** files might not have a specific structure. They're assigned to unstructured data.

- Semi-structured Data

  Two of the key attributes that distinguish semi-structured data from structured data are **nested data structures** and the **lack of a fixed schema**:

  > Unlike structured data, which represents data as a flat table, semi-structured data can contain n-level hierarchies of nested information.
  >
  > Structured data requires a fixed schema that is defined before the data can be loaded and queried in a relational database system. Semi-structured data does not require a prior definition of a schema and can constantly evolve, i.e. new attributes can be added at any time.
  > In addition, entities within the same class may have different attributes even though they are grouped together, and the order of the attributes is not important.

  Semi-structured data consist of documents held in **JSON** format. It also includes **key-value** stores and **graph** databases.

Types of databases

	SQL Databases	NoSQL Databases
Data Storage Model	Tables with fixed rows and columns	Document: JSON documents Key-value: key-value pairs Wide-column: tables with rows and dynamic columns

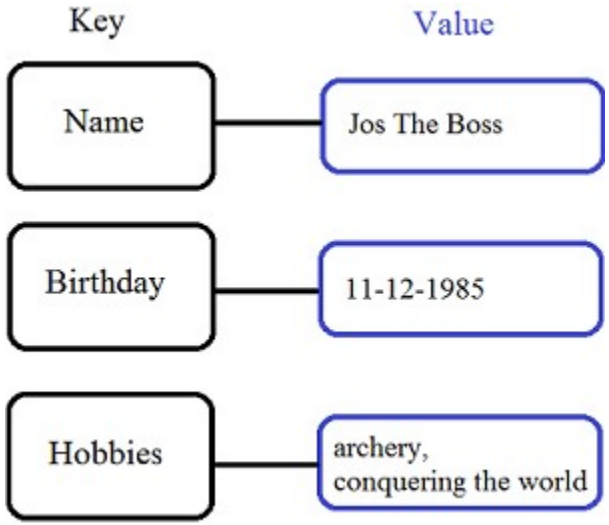
		Graph: nodes and edges
Development History	Developed in the 1970s with a focus on reducing data duplication	Developed in the late 2000s with a focus on scaling and allowing for rapid application change
Examples	Oracle, MySQL, Microsoft SQL Server, and PostgreSQL	Document: MongoDB Key-value: Redis and DynamoDB Wide-column: Cassandra and HBase Graph: Neo4j
Primary Purpose	General purpose	Document: general purpose Key-value: large amounts of data with simple lookup queries Wide-column: large amounts of data with predictable query patterns Graph: analyzing and traversing relationships between connected data
Schemas	Fixed	Flexible
Scaling	Vertical (scale-up with a larger server)	Horizontal (scale-out across commodity servers)
CAP Tradeoffs	Prioritize strong consistency over everything else	Prioritize availability and partition tolerance and offer only eventual consistency

Types of NoSQL Databases

Key-value

Key-value stores are the least complex of the NoSQL databases. They are, as the name suggests, a collection of key-value pairs, and this simplicity makes them the most scalable of the NoSQL database types, capable of storing huge amounts of data.

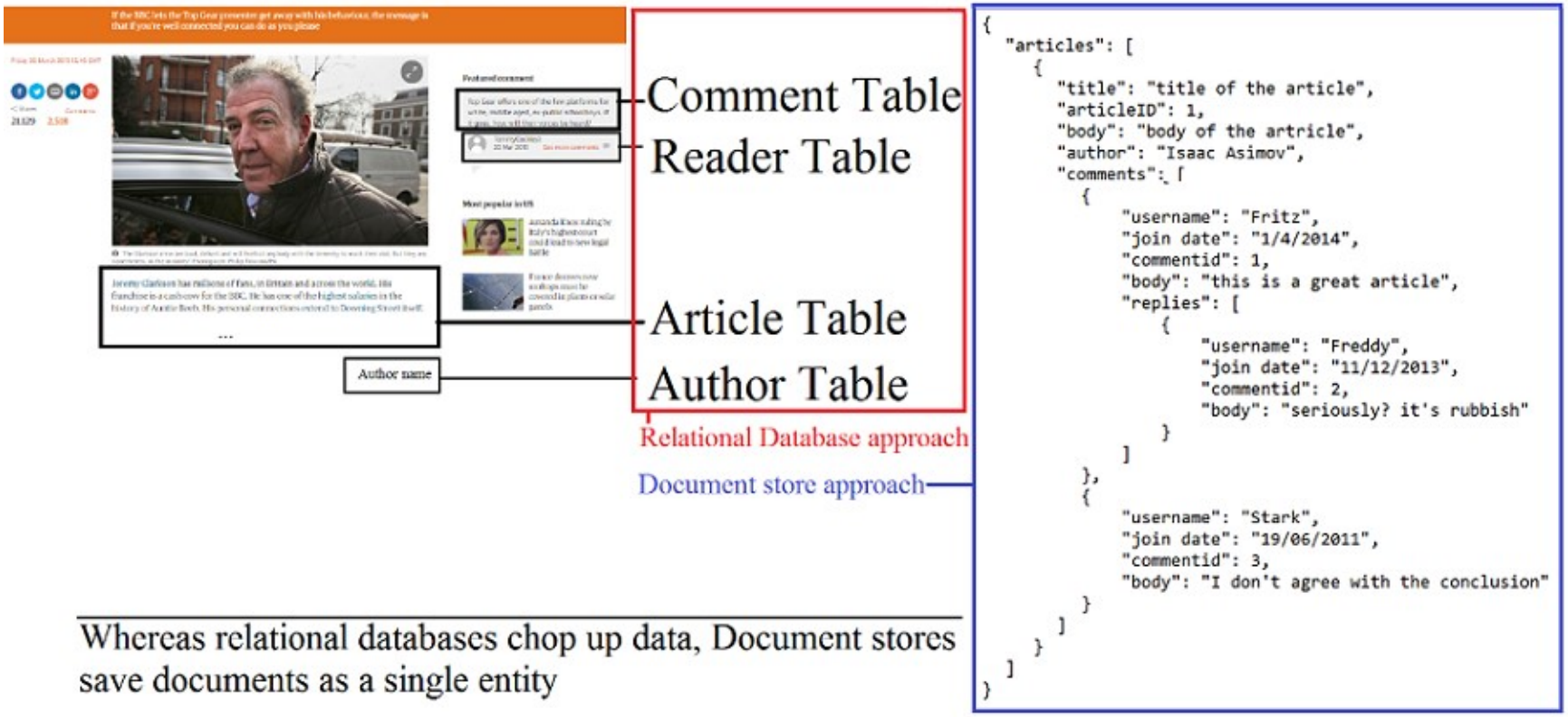
The value in a key-value store can be anything: a string, a number, but also an entire new set of key-value pairs encapsulated in an object.



Document

Document stores are one step up in complexity from key-value stores: a document store does assume a certain document structure that can be specified with a schema.

The way things are stored in a relational database makes sense from a normalization point of view: everything should be stored only once and connected via foreign keys. Document stores care little about normalization as long as the data is in a structure that makes sense.



Column-oriented

Traditional relational databases are row-oriented, with each row having a row-id and each field within the row stored together in a table.

ROWID	Name	Birthday	Hobbies
1	Jos The Boss	11-12-1985	archery, conquering the world
2	Fritz von Braun	27-1-1978	building things, surfing
3	Freddy Stark		swordplay, lollygagging, archery
4	Delphine Thewiseone	16-9-1986	

Row-oriented databases layout. Every entity (person) is represented by a single row, spread over multiple columns.

A database index is a data structure that allows for quick lookups on data at the cost of storage space and additional writes (index update).
An index maps the row number to the data, whereas a column database maps the data to the row numbers.

Name	ROWID
Jos The Boss	1
Fritz Schneider	2
Freddy Stark	3
Delphine Thewiseone	4

Birthday	ROWID
11-12-1985	1
27-1-1978	2
16-9-1986	4

Hobbies	ROWID
archery	1, 3
conquering the world	1
building things	2
surfing	2
swordplay	3
lollygagging	3

A column-oriented database stores each column separately

Storing the columns separately also allows for optimized compression because there's only one data type per table.

In a column-oriented database, it's easy to add another column because none of the existing columns are affected by it. But adding an entire record requires adapting all tables. This makes the row-oriented database preferable over the column-oriented database for online transaction processing (OLTP) because this implies adding or changing records constantly.

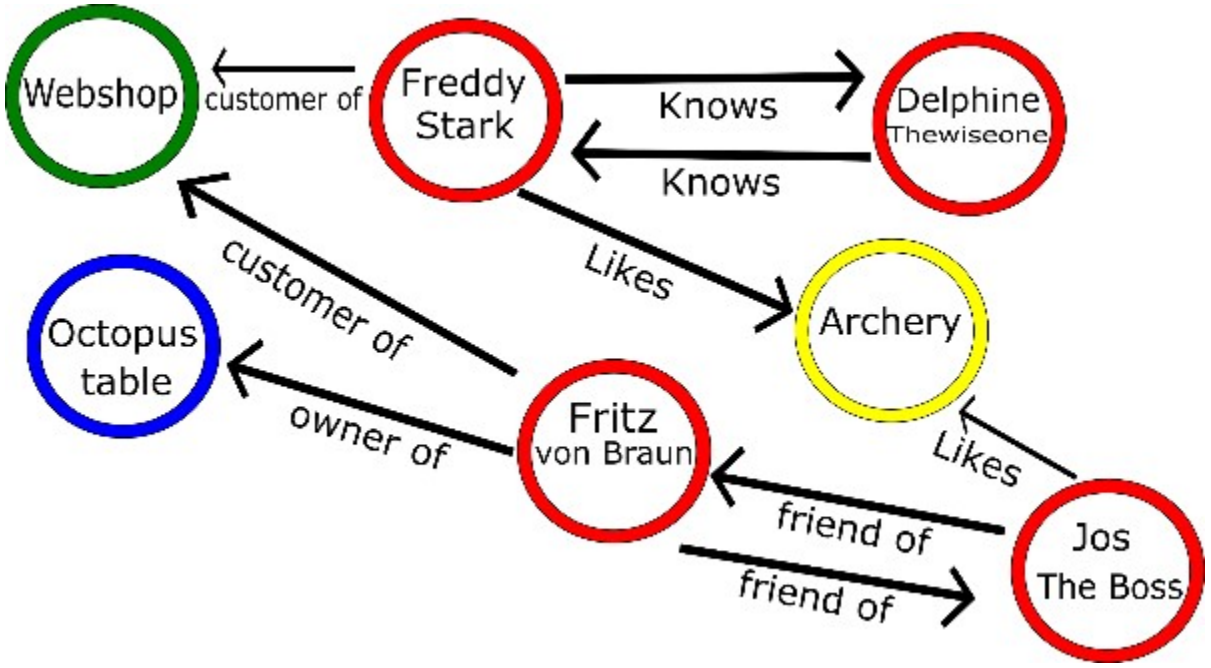
Graph

The last big NoSQL database type is the most complex one, geared toward storing relations between entities in an efficient manner. When the data is highly interconnected, such as for social networks or scientific paper citations, graph databases are the answer.

Graph or network data has two main components:

Node: The entities themselves. In a social network, this could be people.

Edge: The relationship between two entities. This relationship is represented by a line and has its own properties. An edge can have a direction, for example, if the arrow indicates who is whose boss.



References*

https://k21academy.com/microsoft-azure/dp-900/structured-data-vs-unstructured-data-vs-semi-structured-data/
https://docs.snowflake.com/en/user-guide/semistructured-intro.html
https://www.mongodb.com/nosql-explained/nosql-vs-sql
https://dzone.com/articles/nosql-database-types-1
https://towardsdatascience.com/datastore-choices-sql-vs-nosql-database-ebec24d56106#17c3

*Ranking of database management systems: https://db-engines.com/en/ranking