SQL, NoSQL, NewSQL

Introduction to NoSQL: Lecture 1

Piotr Fulmański

FACULTY OF MATHEMATICS
AND COMPUTER SCIENCE
University of Lodz

NoSQL Theory and examples

by Piotr Fulmański

Piotr Fulmański, 2021

PIOTR FULMAŃSKI

NoSQL
Theory and examples

SIMPLE INTRODUCTION SERIES

General overview of *SQL databases

- Why there are so many different families of databases?
- What factors were the main reason for NoSQL to appear?
- Is NoSQL going to replace SQL?

Data and database

Database definition

- **Database** is an *organized collection of data stored in accordance with specific rules*.

In practice we used to think about databases in much more narrower and thus more useful sense.

Data and database

First, **we think about pure immaterial data** - we store numbers, texts, images and songs but not real objects.

We define **data** as a *set of values of qualitative or quantitative variables (properties) describing some object or phenomenon*.

Data and database

Second, **we pay a great attention to automatic way of processing.**

Data and database

Database definition

- **Database** is a *digital data collected in accordance with the rules adopted for a given computer program specialized for collecting, storing and processing this data.*

Such a program (often a package of various programs) is called a *database management system* (DBMS).

Data and database

DATA, INFORMATION, KNOWLEDGE, WISDOM PYRAMID

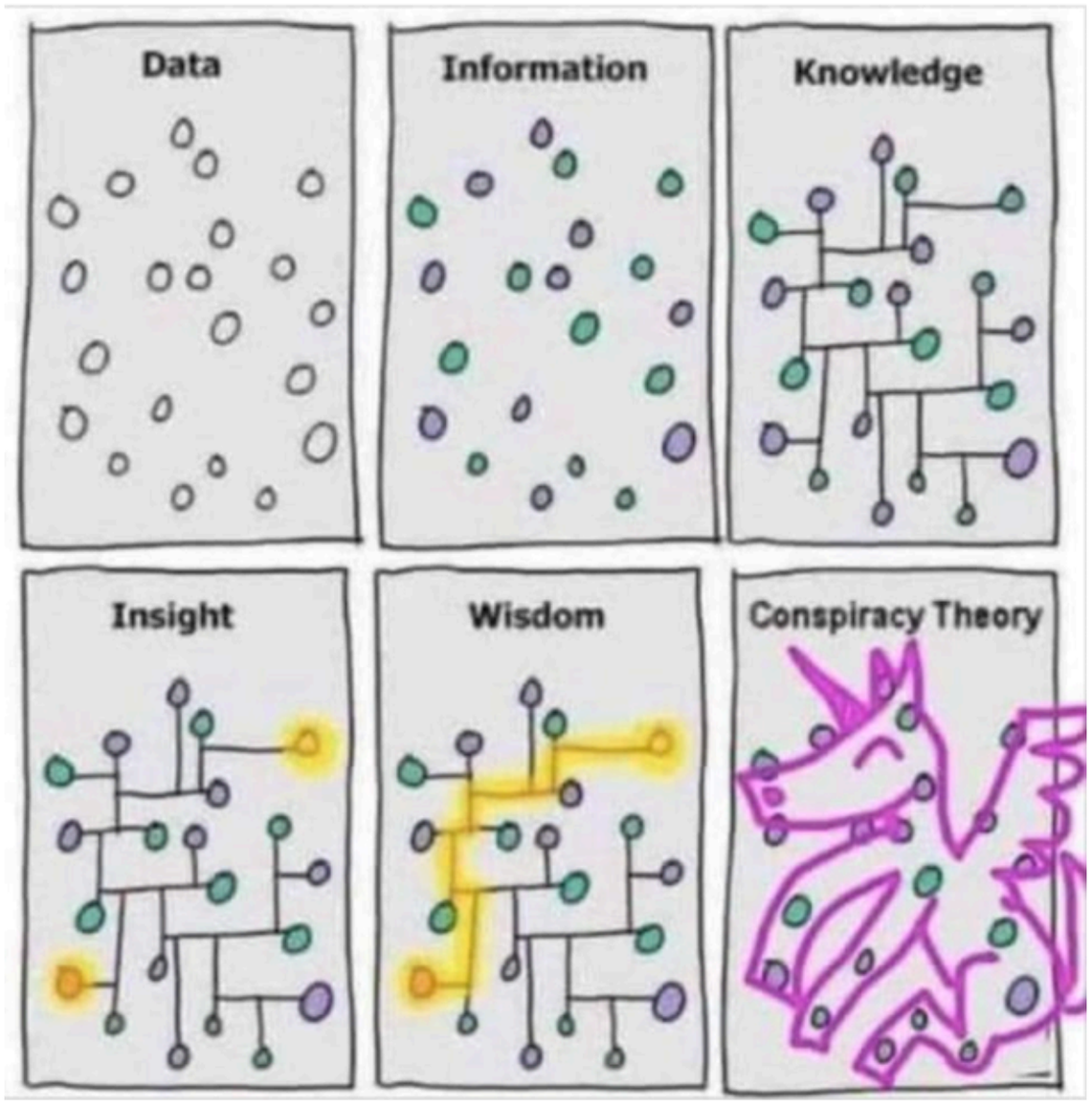
- *Events* produce **data**
- *Context* enriches data to create **information**
- *Meaning* supply information to create **knowledge**
- *Integrated* knowledge form **wisdom**

Knowledge is knowing a tomato is a fruit.

Wisdom is not putting it in a fruit salad.

Data and database

DATA, INFORMATION, KNOWLEDGE, WISDOM PYRAMID



Data and database

DATA, INFORMATION, KNOWLEDGE, WISDOM PYRAMID

We have different forms of data and the question is if we have right tools to store all of them?

Data can be stored in...

even flat file.

Information can be stored in...

most SQL and NoSQL databases.

Knowledge can be store in...

specific databases.

Wisdom?

I don't think so.

Data and database

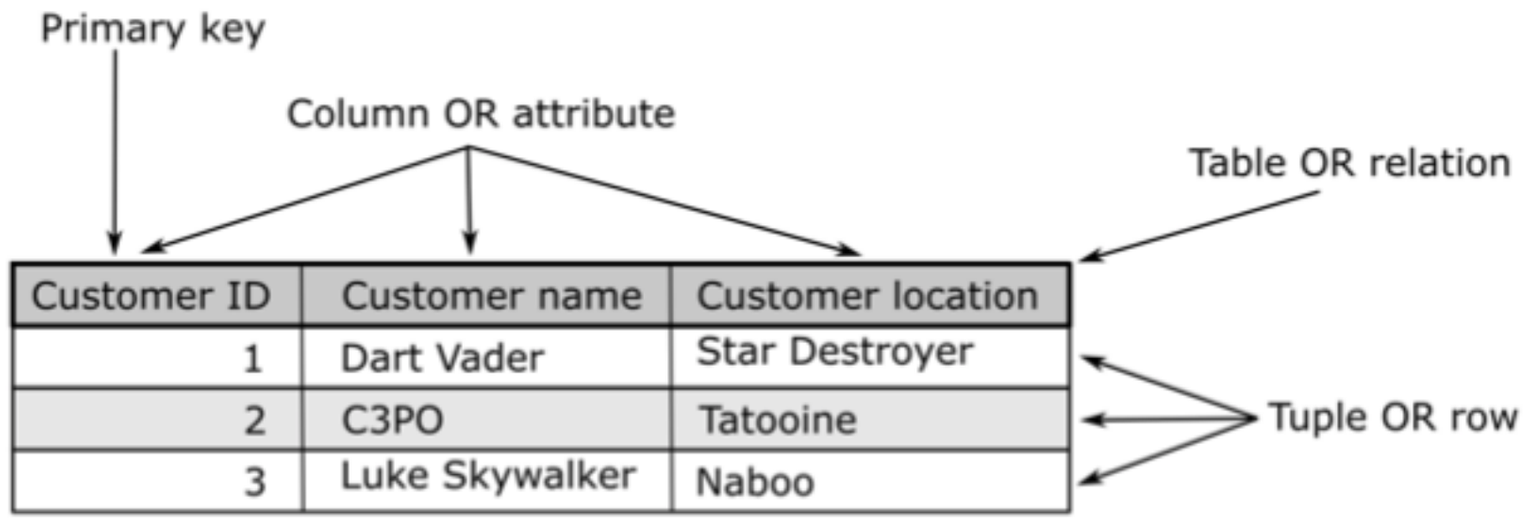
Database or store?

A datastore (store) is, as the name indicates, a place where data is stored. We can store data for example in a hard disk simply using a file storage system or in a database, in which the data are stored physically in files, but those files are managed by some, very often sophisticated, management system. Viewed from this perspective, database are a special type of datastore. Not all NoSQL databases have a builtin "manager", or their functionality is very limited, so the management is done in the application level. That is why we may see them just as an another one storage system. Simply speaking, simple NoSQL databases (for example key-value) are very often referred as a store while those more complicated (graph for example) as a database, but this is not rule of the thumb.

SQL

THE 1970s

SQL — **S**tructured **Q**uery **L**anguage



SQL

SQL — **S**tructured **Q**uery **L**anguage

Customer table

ID	Name
1	Al
2	Betty
3	Carolina
4	Diana
5	Emma
6	Fiona

Order table

ID	CustomerID	Total
1	1	100.00
2	1	33.00
3	4	5.00
4	2	250.00
5	3	64.00
6	6	172.00

```
SELECT Name
FROM Customer
INNER JOIN Order ON Customer.ID = Order.CustomerID
WHERE Total > 100;
```

Result

```
Betty
Fiona
```

SQL

NORMAL FORMS

The main goal of all normal forms is to force user to keep data in a form limiting data redundancy and helping to avoid troubles while data is inserted, updated or deleted.

SQL

NORMAL FORMS — CONSEQUENCES

A relational database is like a garage that forces you to take your car apart and store the pieces in little drawers, every time you drive into it.

SQL

ACID — THE SOURCE OF THE POWER… AND PROBLEMS

The relational model does not itself define the way in which the database handles concurrent data change requests named transactions. To ensure consistency and integrity of data an ACID transaction model is used and became de facto the standard for all serious relational database implementations. An ACID transaction should be

• ***A****tomic*. The transaction can not be divided - either all the statements in the transaction are applied to the database or none are.

• ***C****onsistent*. The database remains in a consistent state before and after transaction execution.

• ***I****solated*. While multiple transactions can be executed by one or more users simultaneously, one transaction should not see the effects of other in-progress transactions.

• ***D****urable*. Once a transaction is saved (committed) to the database, its changes are expected to persist even if there is a failure of operating system or hardware.

Object-oriented databases

THE LATE 1980s AND EARLY 1990s

- Mismatch between the relational model (the way we store data) and real object.

Databases don’t exist to make programmers life simpler. They represent significant assets that must be accessible to those who want to mine the information for decision making and business intelligence.

SQL

SUMMARY — PROS

- ACID transactions at the database level makes development and usage easier.
- Most SQL code is portable to other SQL databases.
- Typed columns and constraints helps validate data before it’s added to the database which increase consistency of the data stored in database.
- Build in mechanism like views or roles prevents data to be changed or viewed by unauthorized users.

SQL

SUMMARY — CONS

- ACID transactions may block system for a short time which may be unacceptable.
- The object-relational mapping is possible but can be complex and add one more intermediate layer.
- RDBMSs don’t scale out. Sharding over many servers can be done but requires new or tuned application code and will be operationally inefficient.
- It is difficult to store high-variability data in tables.
- It is difficult to store data in real time and make real time processing.
- Too much overhead in using a full-featured DBMS as a “dumb” data store for may web-based application.
- SQL is an overkill for simple look-up queries.
- Consistency and correctness in the price of availability and performance.

Big Data

BIG PROBLEM WITH DATA

Story about two tasks: stamping task and numbering task.

Need for:

- task parallelization,
- system scaling.

When there is relatively little data, non-scalable (or at most vertically scalable) systems are sufficient.

Data may be a problem when there is a lot of it.

Big Data

HOW MUCH DATA DO WE PROCESS NOWADAYS

Every minute (in 2018):

- 4.166.667 likes made by Facebook users,
- 347.222 tweets on Tweeter,
- 100.040 cals on Skype,
- 77.166 hours of movies from Netflix,
- 694 Uber users take ride,
- 51.000 application are downloaded from AppStore.

data lake — we collect all data we can get regardless whether we really need them or not

Big Data

BIG DATA Vs

What is worse, **volume** is not the only problem with data. Along with that, there are more factors playing also very important role: **velocity** and **variety**. All of them together constitutes something we called nowadays big data.

SQL solutions were not able to cope with the current needs of working with large data sets in real time.

NoSQL

THE 2000s

Lots of joins pain One of the most important negative feature of SQL databases is it relational, forced by normal forms, nature. Every time we have to get some information, we have to combine data distributed among many tables into something united.

NoSQL is a set of concepts that allows the rapid and efficient processing of data sets with a focus on performance, reliability, and agility.

In the NoSQL solutions, we focus primarily on processing which is:

- fast,
- efficient,
- reliable,
- agile.

NoSQL

MOTIVATIONS

Flexibility

One drawback when working with relational database is that we have to know many things (or all of them) in advance before we start using this database.

Agile approach to data is the flexibility to collect, store and processing any type of data any time its appear.

NoSQL
MOTIVATIONS

Availability

Availability	Unavailability time	
	within a month	within a year
99%	7 hours	3.5 days
99.5%	3.5 hours	2 days
99.9%	43m i 12s	8h i 45m
99.99%	4m i 19s	52m i 36s
99.999%	25s	5m i 15s

NoSQL

MOTIVATIONS

Speed

For the customer's point of view, availability seems to be the most important factor. Relational architecture with lots of constraints set on data (tables) and additional rules (e.g. databases triggers) and needs of splitting every object into the smallest pieces (normal forms) does not support fast data saving. It would be much useful to allow store immediately anything we want at the time we want deferring all rules checking to some later point in the time.

NoSQL
MOTIVATIONS

Cost

NoSQL
MOTIVATIONS

Scalability

NoSQL

NoSQL FAMILY

Column family

The key difference is that the names and format of the columns can vary from row to row in the same table.

NoSQL

NoSQL FAMILY

Key-value

Key-value systems treat the data (identified by a key) as a single opaque collection or better say object (value). There is no way to search by object's properties for that simple reason that every object can be different. It's up to the user to handle it properly.

NoSQL

NoSQL FAMILY

Document

Document databases contrast with the traditional relational database where a single object may be spread across several tables and to get it joins may be needed. Document databases store all information for a given object as a single instance (object), and every stored object can be different from every other.

NoSQL

NoSQL FAMILY

Graph

Relational databases, despite their name, are rather poor suited at representing rich relationships. All the relationships we have in relational databases are foreign keys, which are a kind of pointers to primary keys in other tables. These pointers are not objects we can observe and manipulate easily – in most cases they are abstract, meaningless numbers.

- Sometimes the way we organize our data is also a kind of information. Relations between entities are as important, or more important, than the entities within data.

- Sometimes more than data themselves we care about information hidden in a way we organize them.

NoSQL

NoSQL FAMILY

Column

Paradoxically we can say that the more columns we have, the less probable is that we will process all the columns of a single row.

NoSQL

NoSQL FAMILY

Time series

The key feature of time series databases is the ability to perform efficiently some actions on timed series of data like tracking, monitoring, downsampling, and aggregating over time.

NoSQL

BASE

While ACID systems focus on **high data integrity**, NoSQL systems focus on something significantly different: the **availability**. BASE stands for these concepts:

- **B**asic **A**vailability means that the database appears to work most of the time. It allows systems to be temporarily inconsistent so that transactions are manageable.

- **S**oft-state means that stores don’t have to be write-consistent, nor do different replicas have to be mutually consistent all the time. Some inaccuracy is temporarily allowed and data may change while being used. State of the system may change over time, even without input. This is because of eventual consistency.

- **E**ventual consistency means that there may be times when the database is in an inconsistent state. Eventually, when all service logic is executed, the system is left in a consistent state.

NewSQL

FROM 2011

- Like NoSQL, NewSQL is used to describe a loosely-affiliated group of companies developing some type of software.

- If it is really *new* justifying *new* prefix before SQL term is questionable.

- All of NewSQL companies tries to develop (new) relational database products and services designed

    - to bring the benefits of the relational model to distributed architectures,

    - or to improve the performance of relational databases to the extent that horizontal scalability is no longer a necessity.

NewSQL

IS IT A BIG FAKE?

- NewSQL is a shorthand for the various (new) scalable and high-performance SQL databases. Given that relational DBMSs have been around for over four decades, it is justifiable to ask whether the claim of NewSQL's superiority is actually true or whether it is simply a marketing term.
- If they are indeed able to get better performance, then the next question is
    - whether there is anything scientifically new about them that enables them to achieve these gains,
    - or is it just that hardware has advanced so much that now the bottlenecks from earlier years are no longer a problem.
- If NewSQL is only a marketing terms do we really need it? Wouldn't our time be better invested in trying to understand what the fundamental issues are and how to overcome them.

Bibliography

- [Ful] Piotr Fulmański, *NoSQL. Theory and examples*, Piotr Fulmański, 2021