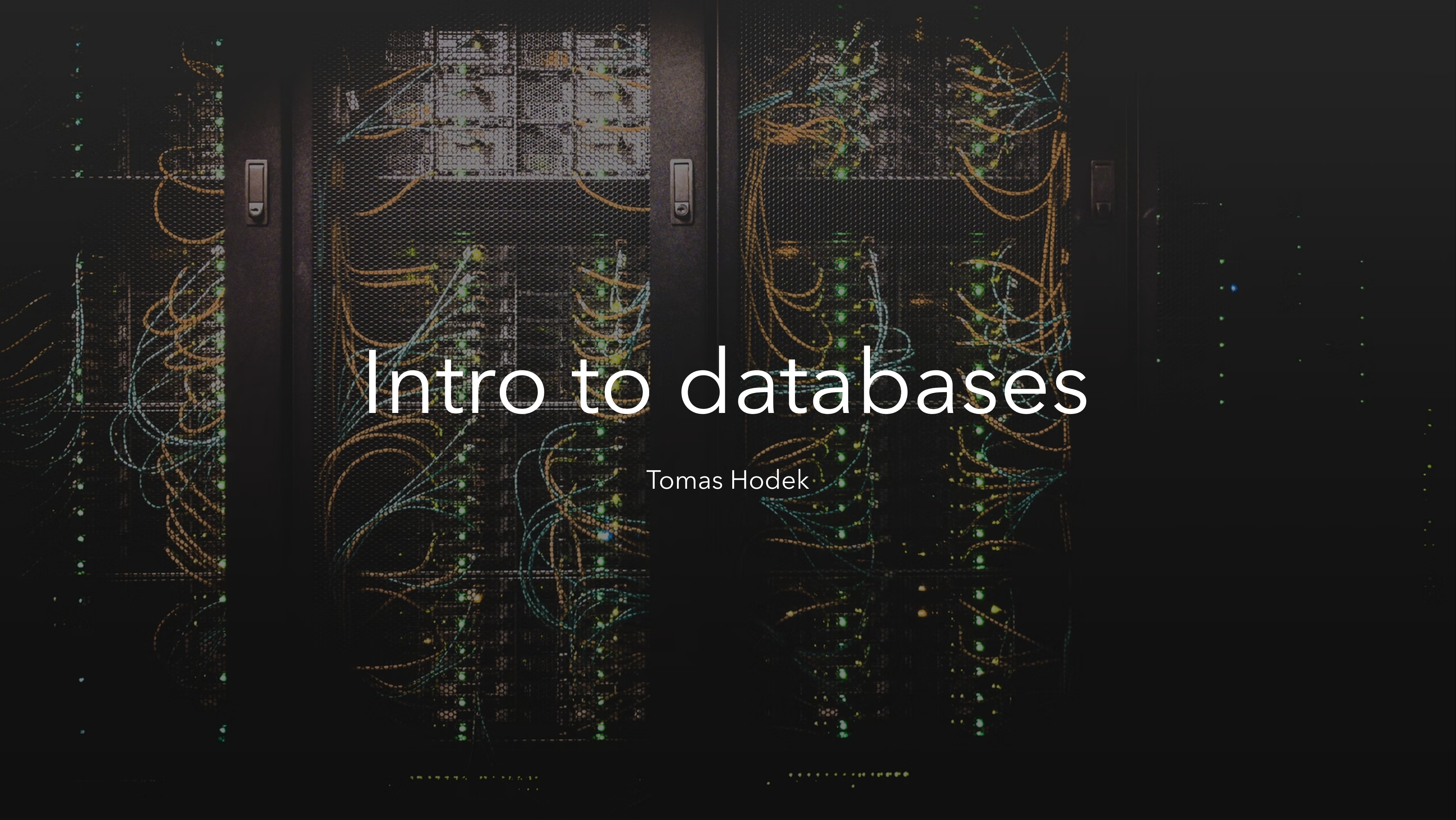


Intro to databases

Tomas Hodek

What is a database?

- Organized collection of structured information

- Simplified => **storage for data**

- Used for storing, backups, searching, filtering,…

What is data? How big it is?

- Data can have any structure
- Base on structure and type of data a DB type is selected
- Data format can be known, uknown or inconsistent
- Data can grow exponentially
- **BIG DATA?**
  - have you heard?
  - do you know how big they are?

BIG DATA

> Big Datais like teenage sex:everyone talks about it, nobodyreally knows how to do it, everyone thinks everyone else isdoing it, so everyone claims they are doing it.

- Actually no standard definition
- Volume, Velocity, Variety, Veracity
- Basic 4V
  - Scale, Complexity, Speed, Uncertainty

Then where to store it?

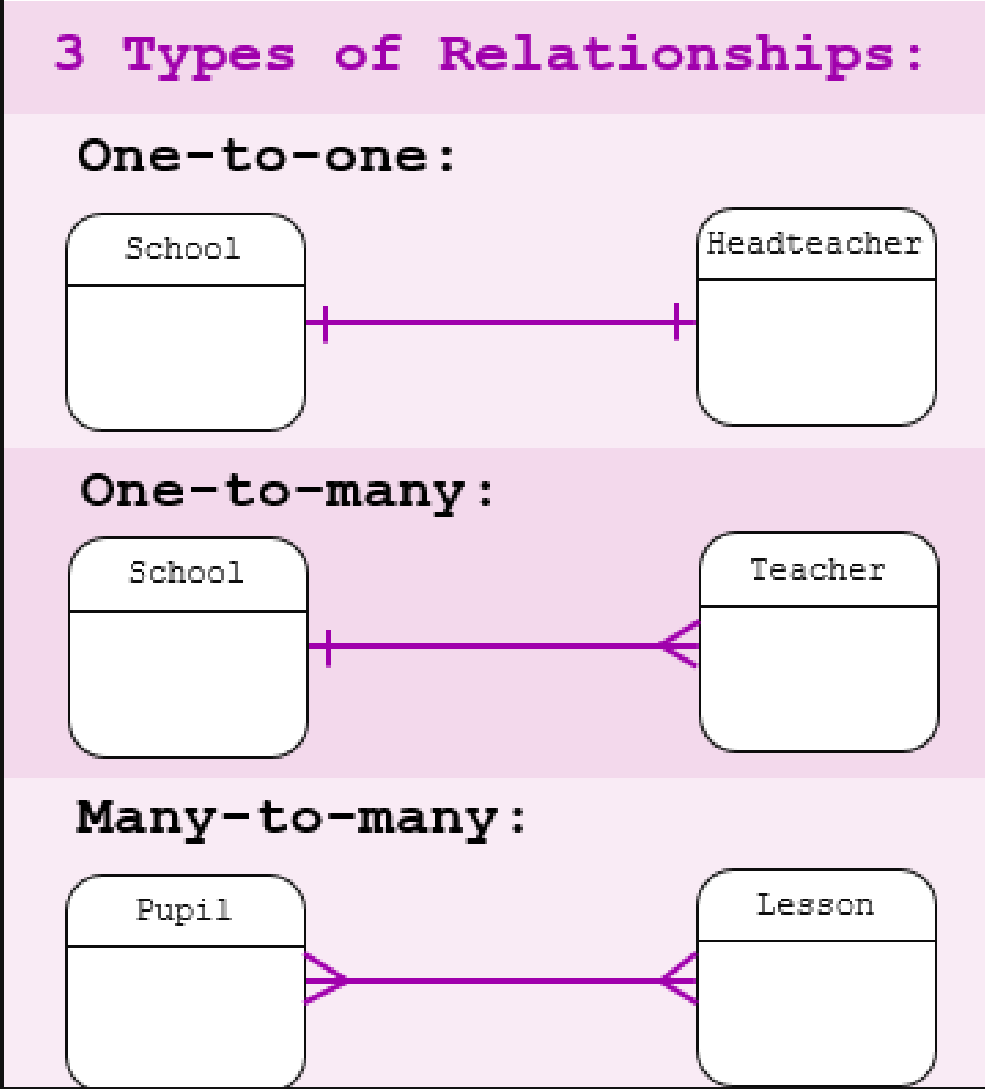
- Relational databases
- Object-oriented databases
- Distributed databases
- Data warehouses
- NoSQL databases
- Graph databases

Then where to store it?

- **Relational databases**
- ~~Object-oriented databases~~
- ~~Distributed databases~~
- ~~Data warehouses~~
- **NoSQL databases**
- ~~Graph databases~~

Relational databases

- Strict data model
- Instance => database => table => row => column
- Relations using unique keys
  - primary key
  - foreign key
- SQL - structured query language
- Queries based on complex conditions
  - projection (views)
  - joins
  - aggregation



Relational databases - bit of theory

- Model shoud be normalized base on normal forms
  - 1NF, 2NF, **3NF**, **BCNF**
- Prevent redudnadncy, update anomalies, etc.
- Creates higher granlularity
- 3NF is (no need to understand or remember now):
  - A relation that is in First and Second Normal Form and in which no non-primary-key attribute is transitively dependent on the primary key, then it is in Third Normal Form (3NF).
  - And is also in 1NF and 2NF
  - Who undersands? - difficult and complex, but important

Relational databases - bit of theory

Transaction

- Flat sequence of database operations that move DB from one state to another
- Should follow ACID properties
  - Atomicity: all or nothing
  - Consistency: from valid state to valid state
  - Isolation: part of transactions are not commited (visible) until whole is done
  - Durability: the result of trancastion is premanent

SQL

- Structured query language

- Declarative and typed

- Covers CRUD - create, read, update, delete

- In SQL

  - *INSERT*
  - *SELECT*
  - *UPDATE*
  - *DELETE*
  - Other: *ALTER*, *TRUNCATE*, *DROP*, *CREATE*

SQL Example

SQL Fiddle

ORM

- Object relational mapping
- Manipulate relational DB using object-oriented programing/paradigm
- Table rows are usually represented as objects, or class instances
- The whole DB schema is represented as classes, linking to each other, no need to write SQL
- Saves time, clearer code, more flexible
- Can be pain if wrongly implemented, hard to learn, performance is poor for complex queries
- Sequelize (NodeJS), Hibernate (Java), Propel, Symfony (PHP), Django (Python)

NoSQL

- Does not mean **no** SQL
- Relational DBs are not enough for Big Data and mostly read access
- Consistency of data is not that critical anymore
- Need for distributed file systems, MapReduce, big computations,…
- Horizontal scaling, high volumes, data distribution (sharding, replication), automated (recovery, distribution, etc.), relaxed consistency and schema

NoSQL types

- Key-Value stores
  - key as an uniquie identifier, value can be anything
  - great performance, not for complex search queries
  - Redis
- Document stores
  - self-describing, hierarchical tree structures (JSON, XML)
  - Documents are organized into collection - extended key-value store
  - queries are complex according to document structures
  - MongoDB, DynamoDB
- Graph Databases (Neo4j)
  - Directed/undircted graphs with relations (edges) and nodes (vertices), uses graph algorithm
  - Social networks, location-based services, chemical and biological databases

MongoDB

- JSON document database
- Instance => databases => collections (similar structure) => documents (one JSON object)
- Primary keys unique inside collections and unchangeable

Collection of movies

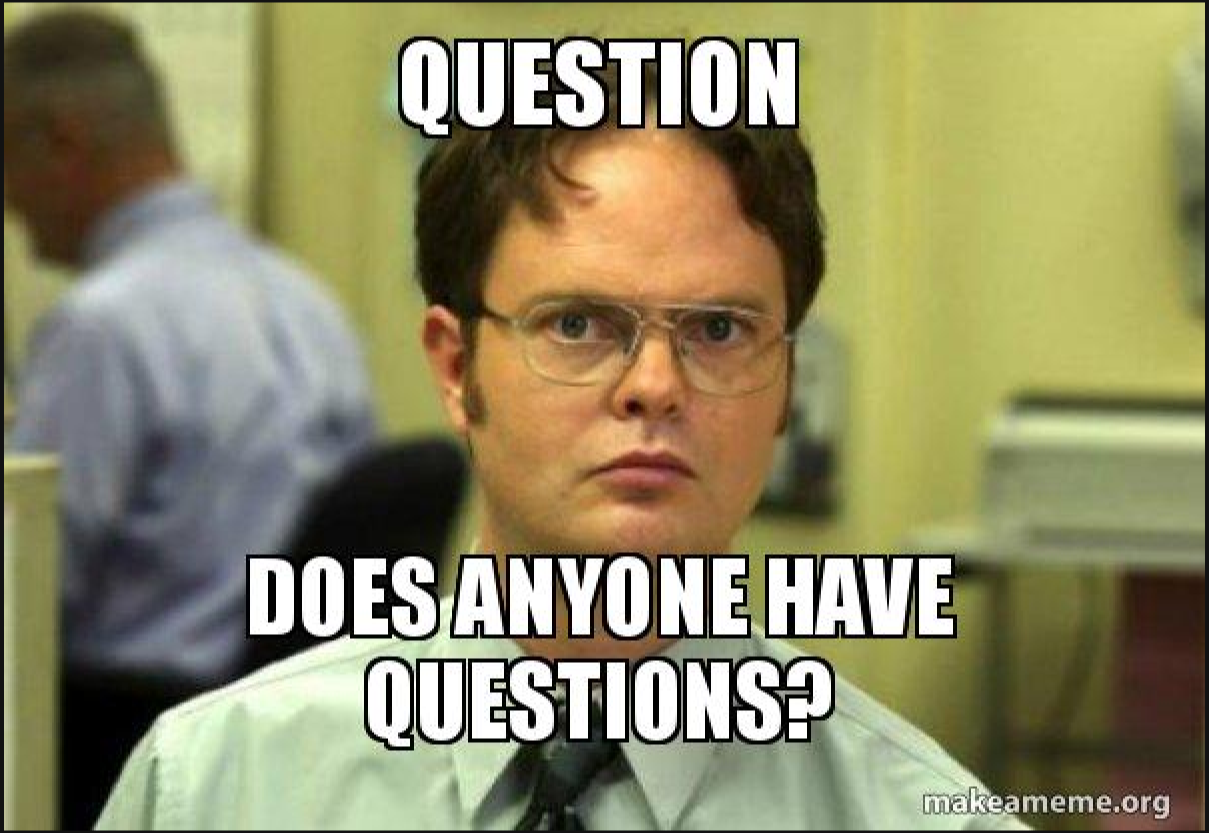
```
{_id: ObjectId("1"),title: "Pulp Fiction",year: 1994},
{_id: ObjectId("2"),title: "Samotáři",year: 2000},
{_id: ObjectId("3"),title: "Medvídek",year: 2007}
```

Query Statement

```
db.movies.find({ year: { $lte: 2005 } },{ _id: false, title: true }).sort({ title: -1 })
```

Result

```
{ title: "Samotáři" }
{ title: "Pulp Fiction" }
```


QUESTION
DOES ANYONE HAVE QUESTIONS?
makeameme.org