BIG DATA MANAGEMENT

CE/CZ4123

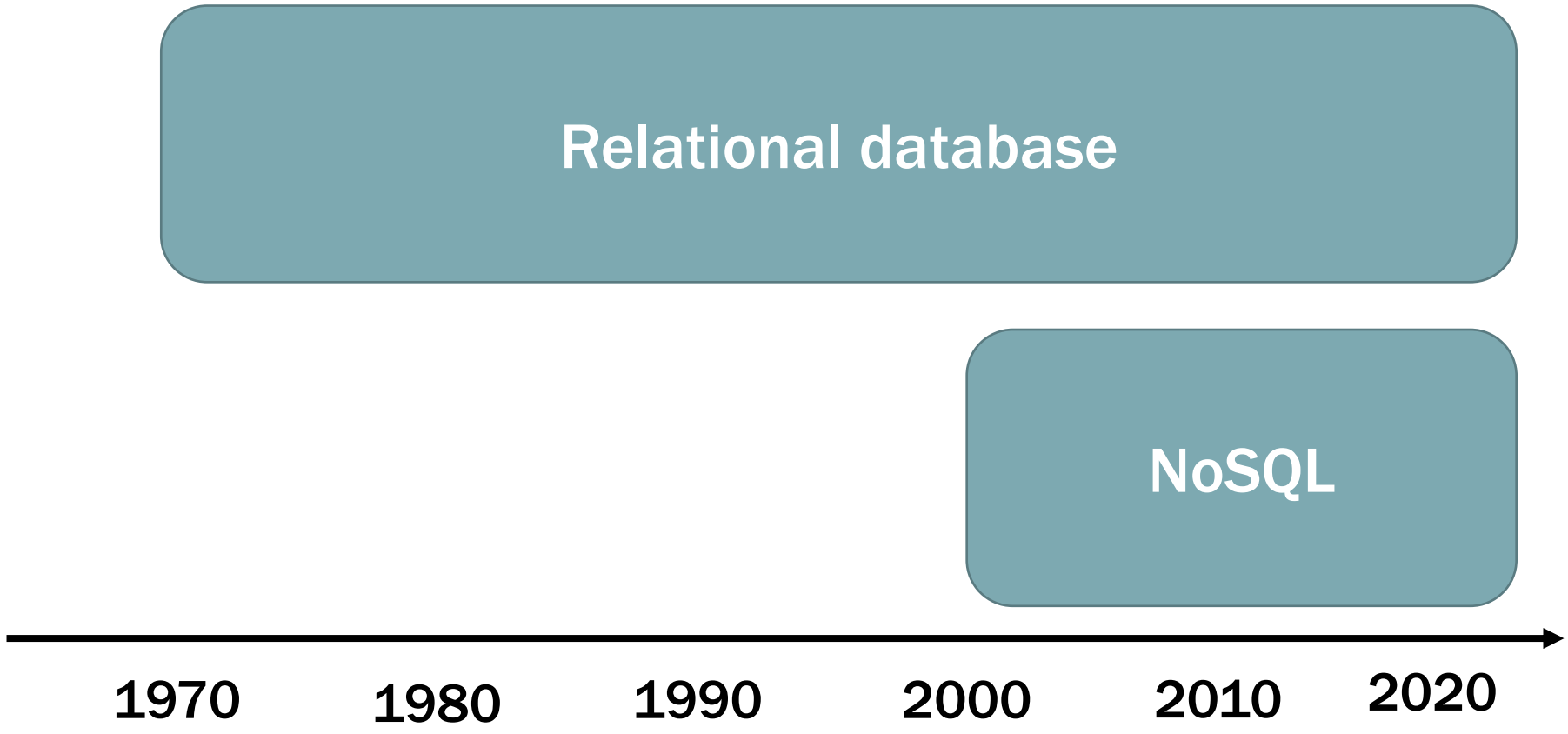
NOSQL

Siqiang Luo

Assistant Professor

BASIC CONCEPT

You may have heard about the word “NoSQL”



RELATIONAL DATABASES (RECAP)

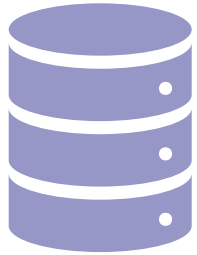
Query patterns	• Selection, Projection, Join, Aggregation
Query languages	• SQL (Structured Query Language) • Relational algebra
Systems	• Oracle Database, Microsoft SQL Server, IBM DB2, MySQL, PostgreSQL

RELATIONAL DATABASES (RECAP)

Normal Forms

Functional dependencies

1NF, 2NF, 3NF, BCNF



Purpose

Good: Remove data redundancy, prevent update anomalies

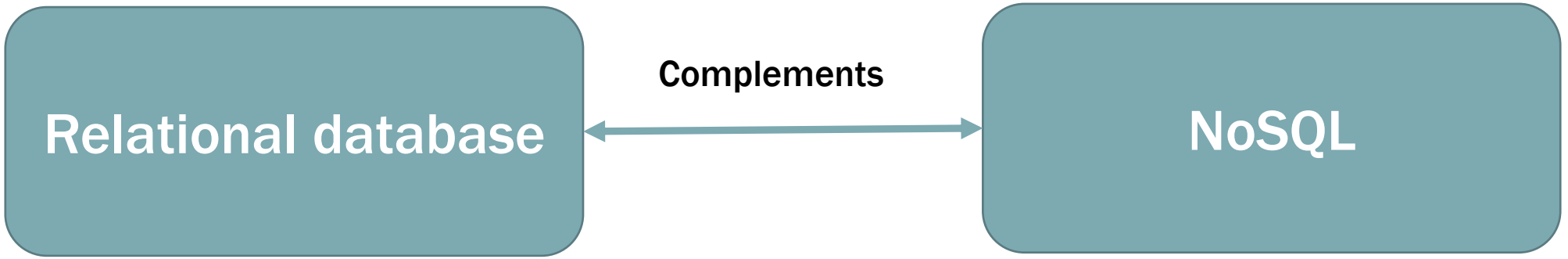
Bad: Data is divided into small pieces and so queries involve joining them (costly).

CURRENT TRENDS

Big data	• Volume, Variety, Velocity, ... • Various data formats • Strong consistency is no longer mission-critical
Extensive user base	• Population online, hours spent online, devices online • Growing companies / web applications • Even millions of users within a few months
Cloud computing	• On-demand services of data storage and computing power
Real-time analytic processing	• Quality of services becomes more and more important.

RELATIONAL AND NOSQL DATABASES

- NoSQL also means “Not only SQL”, where SQL refers to the relational database (not exactly the SQL language).
- A **NoSQL** database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.
- NoSQL databases are increasingly used in big data applications.



PROPERTIES OF NOSQL DATABASE

Well designed schema

e.g., Relations are well decomposed and connected by foreign keys

Benefit: standardized data model designs (normal forms)

Relational database

Flexible schema (schemaless)

e.g., documents, key-value pairs

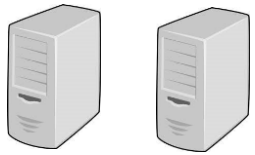
Benefit: programmers are more flexible in designing data models.

NoSQL

PROPERTIES OF NOSQL DATABASE

Not that easy to scale

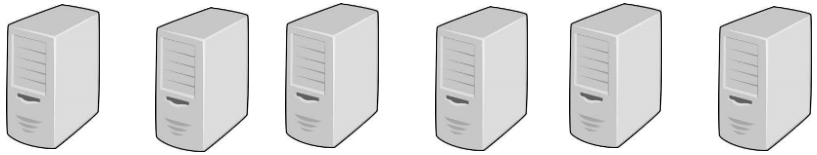
Reason: usually relational database requires strict consistency



Relational database

Easier to scale

Reason: usually requires eventual consistency

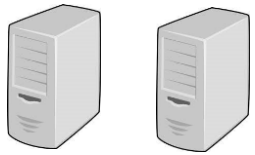


NoSQL

PROPERTIES OF NOSQL DATABASE

Better supports query languages

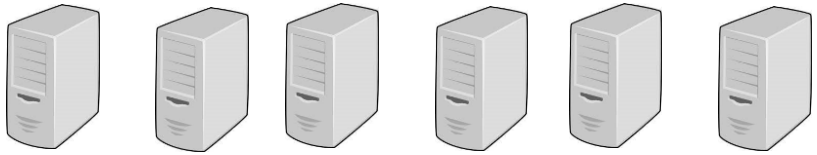
e.g., SQL



Relational database

Partially supports query language

Lack of standardized interface



NoSQL

PROPERTIES OF NOSQL DATABASE

Support all kinds of relational queries

Queries are less flexible, but can have higher performance

Relational database

NoSQL

TYPES OF NOSQL DATABASES

Key-Value Stores

Wide-Column Database

Document Database

Graph Database

KEY-VALUE STORE

- Data model
  - key-value pairs
  - The simplest NoSQL database type
  - Works as a simple hash table (mapping)

- 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

KEY-VALUE STORE

- Very fundamental database, and widely used in Web applications with billion users (e.g., Facebook, Amazon)
- Use cases: Session data, user profiles, user preferences, shopping carts, i.e., when values are only accessed via keys
- Representatives:
  - RocksDB, LevelDB, Redis, Memcache,...

WIDE-COLUMN DATABASE

Data model: 2-dimensional key-value models.

- The names and format of the columns can vary across rows, even within the same table.
- Columns are not separately stored, but some of them can be grouped as a “column family”. Given a column family, it can be stored row-by-row.
- It can have very large number of columns

Representatives:

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

DOCUMENT DATABASE

- Data model: data are stored as documents
  - A document describes an object
  - Can be described in JSON format (i.e., a list of key-value pairs to describe the object attributes)
  - Example: { "FirstName": "Bob", "Address": "5 Oak St.", "Hobby": "sailing" }
  - A document is addressed by a “key”, so also regarded as a subclass of key-value store. Differences is that the value is a document which can contain very rich information.
- Representatives:
  - MongoDB, Couchbase, Amazon DynamoDB, CouchDB, RethinkDB, RavenDB, Terrastore

GRAPH DATABASE

Data model: graphs

- A graph consists of nodes and edges
- Nodes represent entities; edges for relationships
- Easier to model data that contain many entities and interconnected relationships, e.g., social networks

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

WHEN TO CHOOSE ONE OVER THE OTHER (RDB VS NOSQL)

There is NOT a golden rule that you should choose one over the other. Both systems are improving themselves, and some modern systems have integrated their advantages as a mixed system in some sense.

However, there are some guidelines one can consider

1) When the data scale is small, relational database is great
2) When the project is always about relational queries and have various access patterns, relational database is great
3) When you want to better enforce field constraints, relational database is preferred.
4) When the data scale is very large, and you need very high performance for some specific type of queries, NoSQL is great.

SCENARIOS

Suppose a company with 10000 employees want to build an employee management system. It would require to query the salaries of employees, the managers of an employee etc.

Would you choose a relational database or NoSQL database?

SCENARIOS

Suppose a company with 10000 employees want to build an employee management system. It would require to query the salaries of employees, the managers of an employee etc.

Would you choose a relational database or NoSQL database?

Relational database is good for this scenario.

SCENARIOS

Suppose we aim to build a stock database that records all the stock price changes every 5 seconds. The query pattern is given a stock id, search the lowest and highest stock prices of a given time period. The results should be returned in a short time when searched.

Would you choose a relational database or NoSQL database?

SCENARIOS

Suppose we aim to build a stock database that records all the stock price changes every 5 seconds. The query pattern is given a stock id, search the lowest and highest stock prices of a given time period. The results should be returned in a short time when searched.

Would you choose a relational database or NoSQL database?

NoSQL key-value store is good choice for this scenario.

NOTE

When we emphasize the benefit of NoSQL systems, do NOT take for granted that NoSQL databases are always better than Relational databases.

Relational databases are great in many aspects

Both relational databases and NoSQL databases are improving

Both relational databases and NoSQL databases can be chosen for different applications.

ACKNOWLEDGEMENT

Part of the slide contents are inspired by the course materials of Prof. **Martin Svoboda**

DISCUSSION

Debate between SQL and NoSQL

https://www.youtube.com/watch?v=rRoy6I4gKWU&list=RDCMUC_x5XG1OV2P6uZZ5FSM9Ttw&start_radio=1&rv=rRoy6I4gKWU&t=774