Introduction to Database Concepts

Prof. Dr. Adnan YAZICI
Department of Computer Engineering,
Faculty of Engineering
Middle East Technical University

(Fall 2021)

Some Definitions

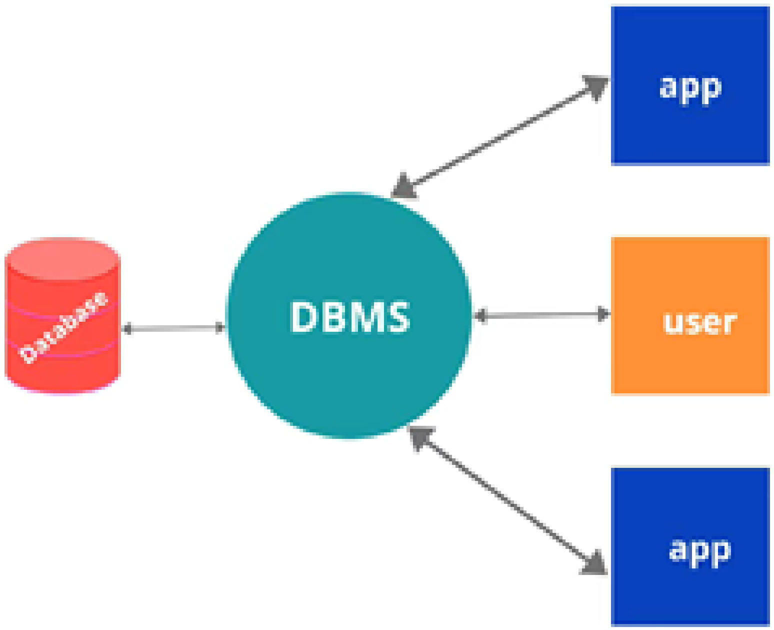
- Data: Raw, unprocessed facts.

  Ex: 32, 0.001, John, an image.
- Information: Processed data.

  Ex: John in the image is 32 years old.
- Database: A large and integrated collection of related data

  Ex: Online banking systems and library management system.
- Meta Data: The database definition
- Database Systems (DBSs): model real-world enterprise
  - Entities (e.g., students, courses)
  - Relationships (e.g., John is taking the DB course)
- Database Management System (DBMS): a software package (a collection of programs) designed to create and maintain databases.

DBMS $\neq$ DATABASE ?



MySQL

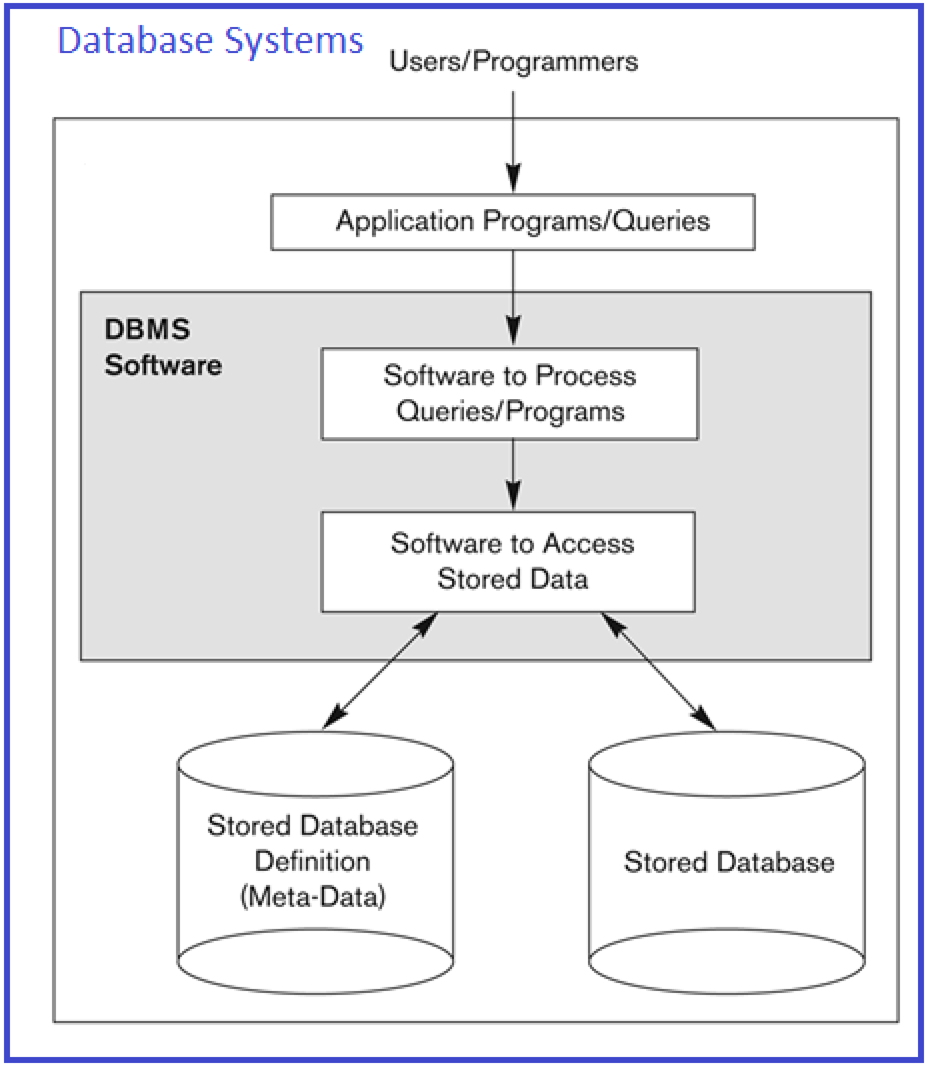
PostgreSQL

MongoDB

Neo4j

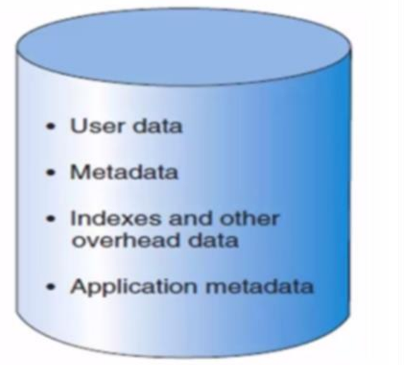
Cassandra

Database system environment (Simplified)



Database System:

{Users, Application Programs, DBMS, Database, Meta-data}



Functionalities of DBMS

- Define: Specifying the data types, structures and constraints for the data to be stored in DB
- Construct: Process of storing data on some storage medium.
- Manipulate: Querying the database to retrieve specific data, updating database and generating reports.
- Share: Allows multiple users and programs to access the database concurrently.

Functionalities of DBMS (cont.)

- *Define* (*create*) a particular database in terms of its data types, structures, and constraints (creating a model)
- *Construct* or *Load* the initial database contents on a secondary storage medium
- *Manipulate* the database:
  - Retrieval: Querying to retrieve specific data, generating reports
  - Modification: Insertions, deletions and updates to its content
  - Accessing the database through Web applications
- Enforce rules
- Control concurency
- Provide security and data privacy
- Perform data backup and recovery
- Optimize queries
- *Share* a database allowing multiple users and programs to access the database simultaneously

Functionality of a DBMS (cont.)

The programmer sees SQL, which has two components:

- Data Definition Language - DDL
- Data Manipulation Language - DML
  - query language

Behind the scenes the DBMS has:

- Query optimizer
- Query engine
- Storage management
- Transaction Management (concurrency, recovery)
- Many others…

Properties of Database

- A database represents some aspects of the real World (miniworld)
- A database is a logically coherent collection of data with some inheret meaning
- A database is designed, built and populated with data for a specific purpose.
- In short, a database is a collection of information that is organized so that data can be easily stored, managed, updated, and retrieved.

Evolution of Databases

Motivation for studying DBS

- Most computers are used for data processing (over $500 billion/year). A big growth area in the “information age”
- This course covers data processing from a computer science perspective:
  - Modeling of data
  - Storage of data
  - Organization of data
  - Querying and access of data
  - Processing of data

Why Study Databases??

- Shift from **computation** to **information**
- Datasets increasing in diversity and volume.
  - Digital libraries, interactive video, Human Genome project, ERPs, many more…
  - ... need for DBMS exploding
- DBMS encompasses most of CS
  - Data structures, OS, languages, theory of computation, “AI”, multimedia, logic, algorithms

Why Use a DBMS?

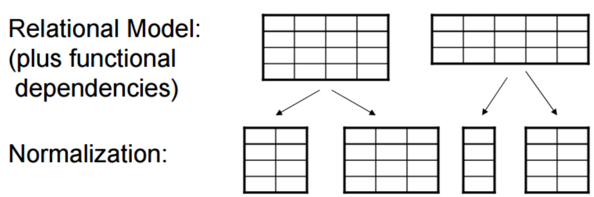
- Data independence
- Efficient access.
- Reduced application development time.
- Data integrity, privacy and security.
- Uniform data administration.
- Multiuser and concurrent access,
- Recovery from crashes.

Database Models

- A **database model** is a collection of concepts for describing data.
- A **schema** is a description of a particular collection of data, using the given database model.
- The **relational model of data** is the most widely used model today.
  - Main concept: **relation**, basically a table with **rows** and **columns**.
  - Every relation has a **schema**, which describes the columns, or fields.

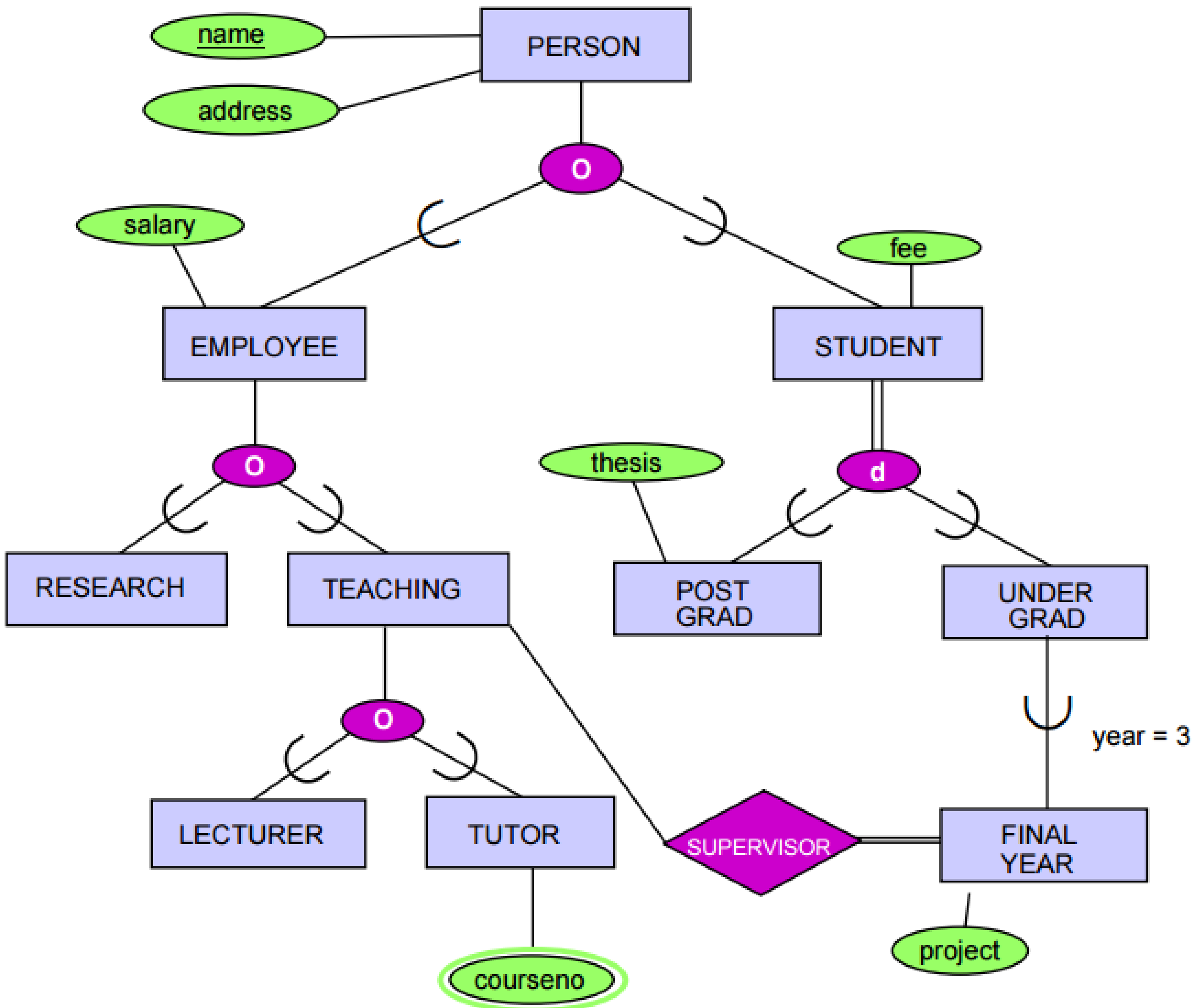
Example from UNIVERSITY Database

- **Conceptual schema:**
  - Students (<u>*sid*</u>: string, *name*: string, *login*: string, *age*: integer, *gpa*: real)
  - Courses (<u>*cid*</u>: string, *cname*: string, *credits*: integer)
  - Enrolled (<u>*sid*: string, *cid*: string</u>, *grade*: string)
- **Logical schema:**



- **Physical schema:**
  - Relations stored as unordered/ordered files.
  - B+-Index on first column of Students.
- **External Schema (View):**
  - Course_info (<u>*cid*: string, *enrollment*: integer</u>)

University Database (conceptual design)



Structure of Relational Databases

TABLE 1					
	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN 5
ROW 1	data	data	data	data	data
ROW 2	data	data	data	data	data
ROW 3	data	data	data	data	data
ROW 4	data	data	data	data	data

NAME	ID	DATE_OF_BIRTH	ADDRESS	GENDER	PHONE
Aaron Paul	D1	05-Jul-86	Kuala Lumpur	M	60169990102
Lara Croft	D2	01-Oct-98	Bangalore	F	9774755019
Ruth Langmore	D3	23-May-01	Singapore	F	6545459898

TABLE 1					
	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN 5
ROW 1	data	data	data	data	data
ROW 2	data	data	data	data	data
ROW 3	data	data	data	data	data
ROW 4	data	data	data	data	data

NAME	ID	DATE_OF_BIRTH	ADDRESS	GENDER	PHONE
Aaron Paul	D1	05-Jul-86	Kuala Lumpur	M	60169990102
Lara Croft	D2	01-Oct-98	Bangalore	F	9774755019
Ruth Langmore	D3	23-May-01	Singapore	F	6545459898
Em					

Instance of Students Relation

Students(<u>*sid*</u>: string, *name:* string, *login:* string, *age:* integer, *gpa:* real)

<u>sid</u>	***name***	***login***	***age***	***gpa***
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8

Structure of Relational Databases

Office Database

EMPLOYEE			
ID	NAME	AGE	MANAGER_ID
E101	Libinus Xavier	37	M123
E102	Gautham Bhonsle	35	M555
E103	Aravind	45	M404
E104	Shazil	28	M800
E105	Manisha Shah	34	M555

SQL

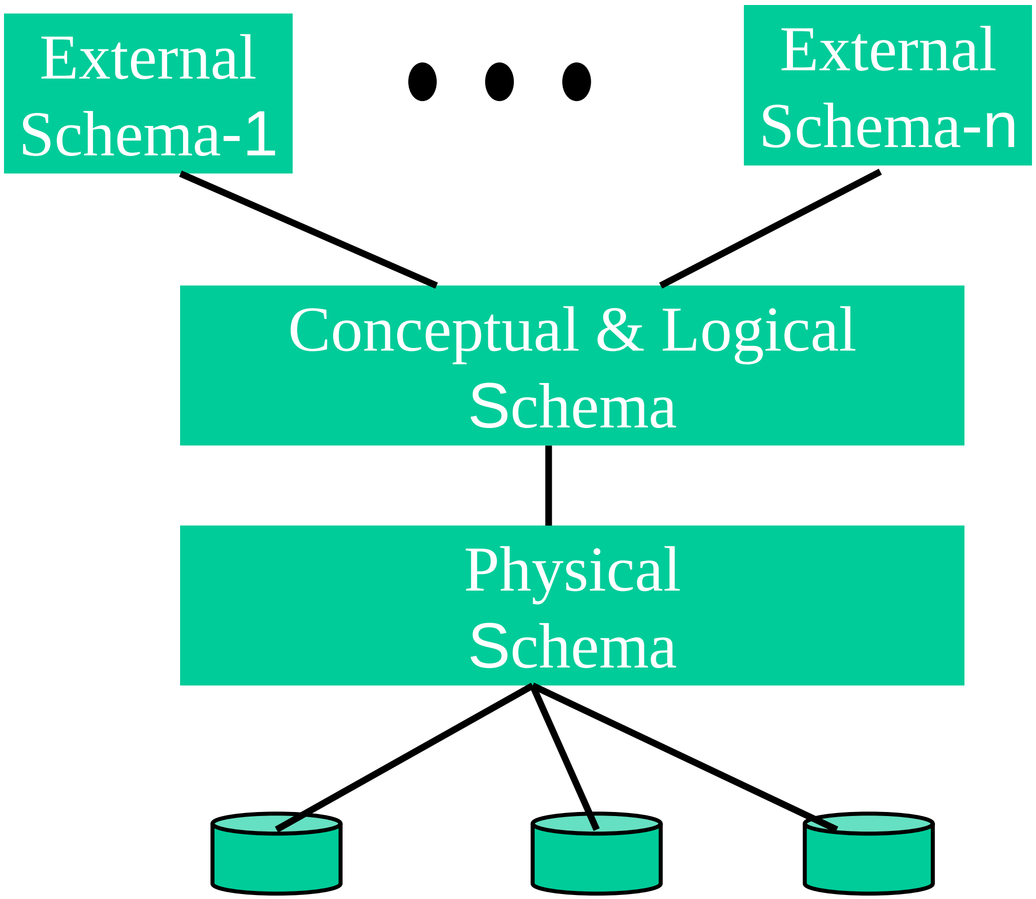
(Structured Query Language)

MANAGER		
ID	NAME	DEPT_ID
M123	Ravindranadh	D1011
M404	Shripad Karambelkar	D1011
M555	Meenu Dutta	D2022
M800	James Xavier	D1099
M999	Ibrahim Sheik	D1099

DEPARTMENT			
ID	NAME	DESCRIPTION	LOCATION
D1011	FINANCE	Finance Operations	Mumbai
D1099	HR	Human Resource	Bangalore
D2022	IT	Information Technology	Bangalore
D3033	ADMIN	Administrative Operations	Bangalore

Oracle Microsoft SQL Server MySQL PostgreSQL

Three-schema Architecture (Levels of Abstraction)



Describe how a specific user or groups view the data (database)

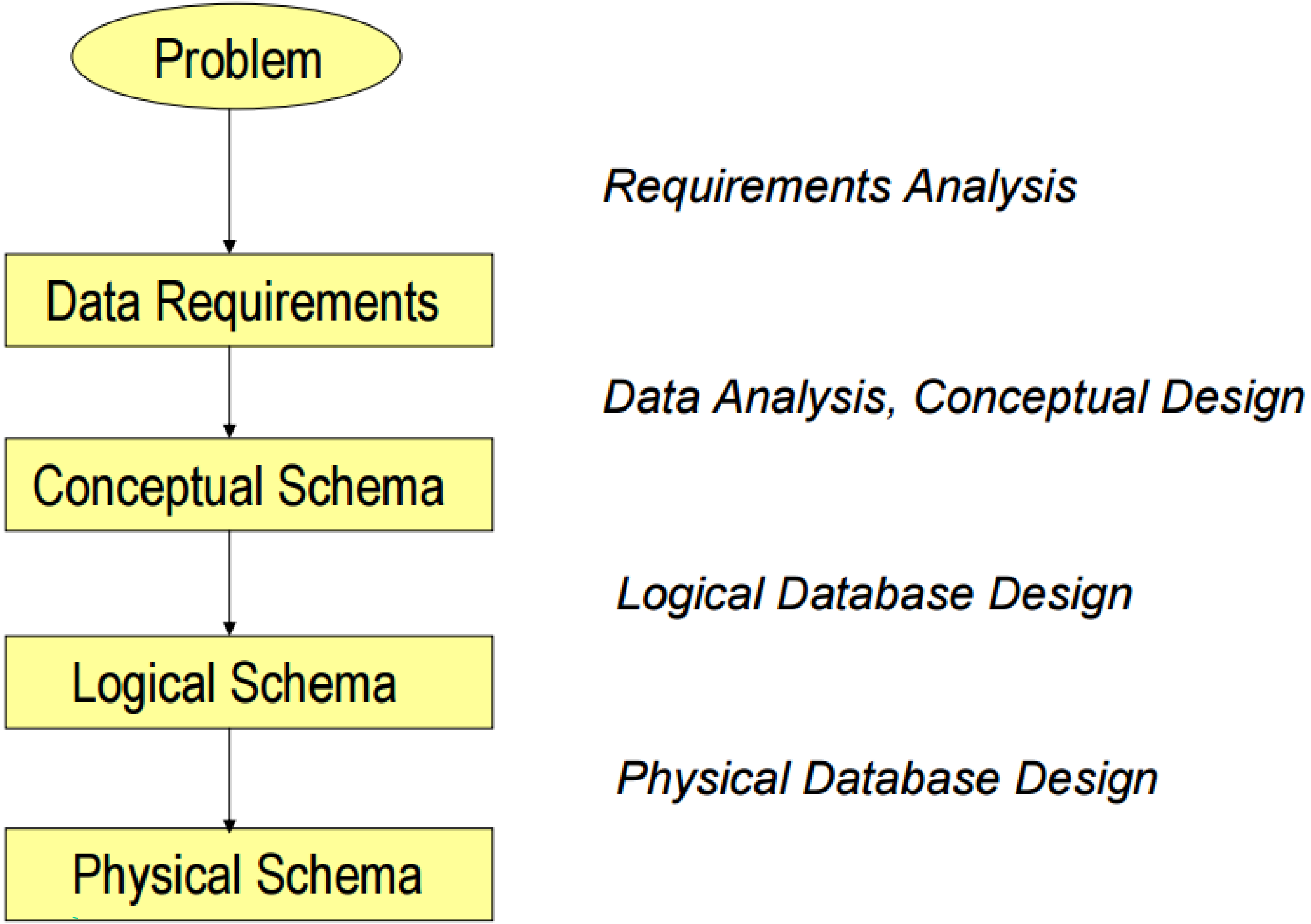
Defines conceptual and logical structure, that is, describes the whole database for all users

Physical storage structures and details, describes the files and indexes used.

Data residing in disks

* Schemas are defined using DDL; data is modified/queried using DML.

Database Design Goes Through Stages



Data Independence

- Applications are insulated from how data is structured and stored, which means the ability to change the database at one level with no impact to the next higher level.

  - **Physical data independence:** The ability to change the physical schema without affecting conceptual / logical schema.
  - *Example*: adding a new index structure to database.

  - **Logical data independence:** The ability to change the conceptual/logical structure without affecting existing external views.

    Example: adding a new attribute to the schema

Data Structures vs File Structures

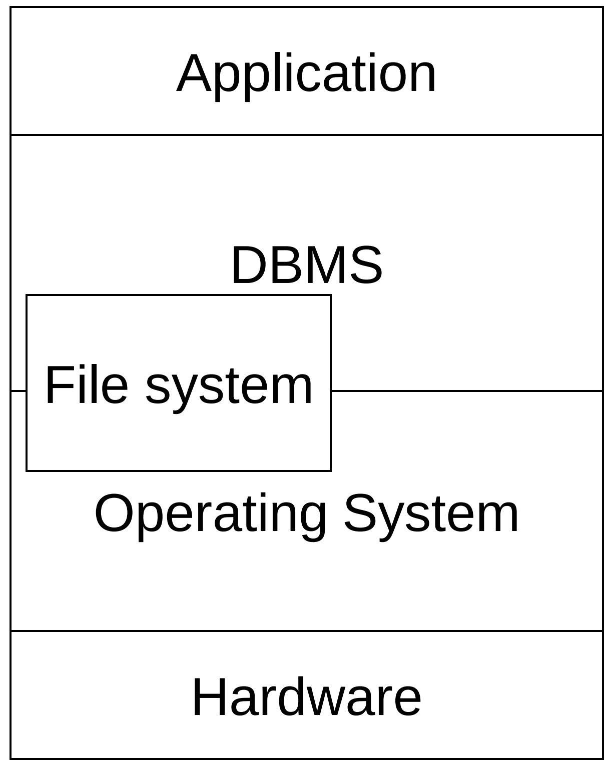
- Both involve:
  - Representation of Data

    +

  - Operations for accessing data

- Difference:
  - Data structures: deal with data in main memory
  - File structures: deal with data in secondary storage

Where do File Structures fit in Computer Science?



Computer Architecture (simple version)

data is manipulated here

Main Memory (RAM)

- Semiconductors
- Fast, expensive, volatile, small

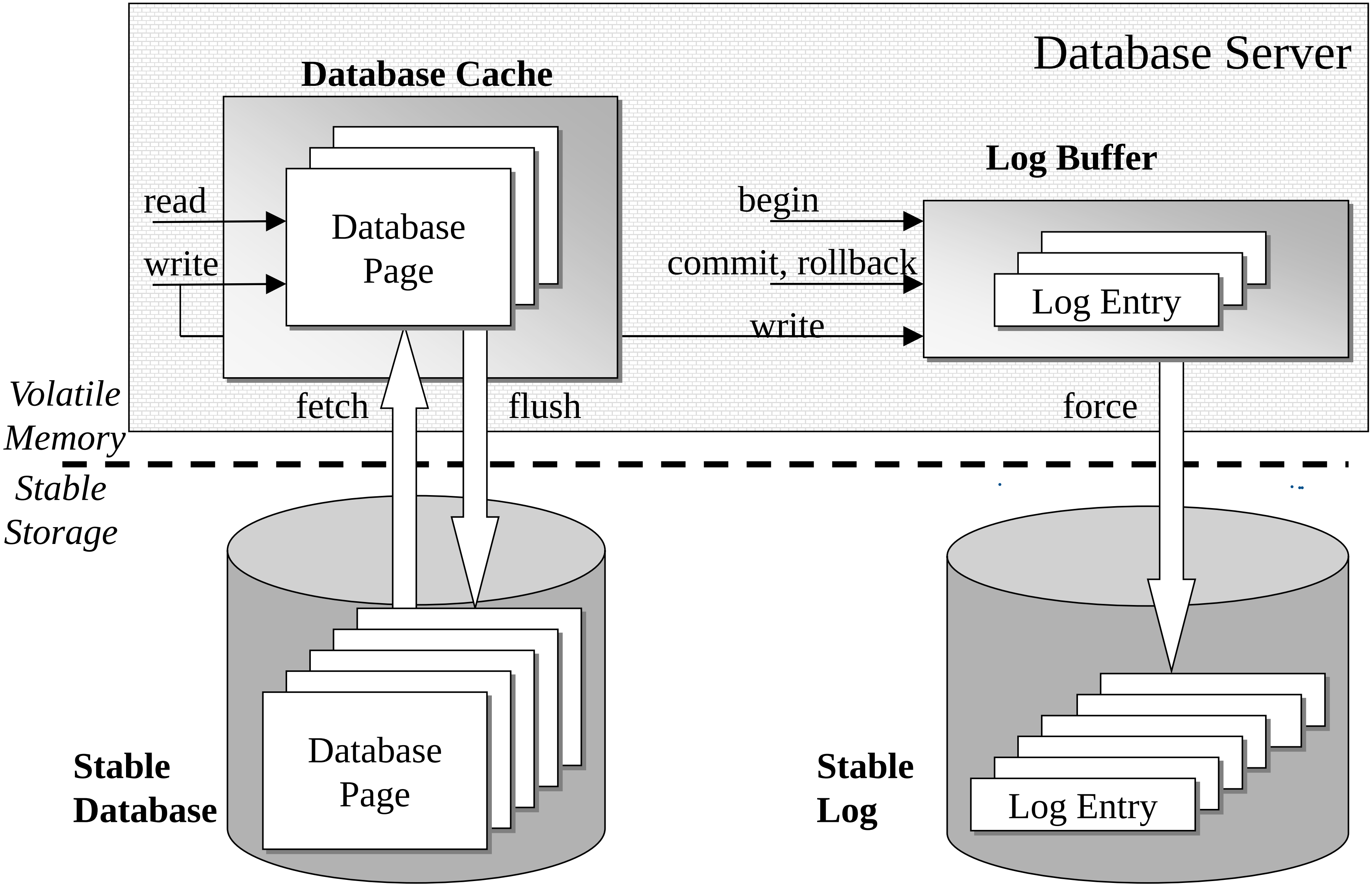
data transfer

data is stored here

Secondary Storage

- disks, tape
- Slow,cheap, stable, large

Overview of System Architecture (different perspective)



Computer Architecture (simple version)

Advantages:

- Main memory is fast
- Secondary storage is big (because it is cheap)
- Secondary storage is stable (non-volatile) i.e. data is not lost during power failures.

Disadvantages:

- Main memory is small. Many databases are too large to fit in MM.
- Main memory is volatile, i.e. data is lost during power failures.
- Secondary storage is slow (10,000 times slower than MM).

How fast is main memory?

- Typical time for getting info from:

  Main memory: ~12 nanosec = 120 x 10^{-9} sec

  Magnetic disks: ~30 milisec = 30 x 10^{-3} sec

- An analogy keeping same time proportion as above:

  Looking at the index of a book : 20 sec

  versus

  Going to the library: 58 days

Normal Arrangement

- **Secondary storage (SS)** provides reliable, long-term storage for large volumes of data
- At any given time, we are usually interested in only a small portion of the data
- This data is loaded temporarily into **main memory (MM)**, where it can be rapidly manipulated and processed.
- As our interests shift, data is transferred automatically between **MM** and **SS**, so the data we are focused on is always in MM.

Goal of the file structures

- Minimize the number of trips to the disk in order to get desired information
- Grouping related information so that we are likely to get the requested data we need with only one trip to the disk.

File Systems

- Data is not scattered hither and thither on disk.
- Instead, it is organized into **files**.
- Files are organized into **database pages (blocks)**.
- Database pages are organized into **records**
- Records are organized into **fields**.

Example

- A **student** file may be a collection of student records, one record for each student
- Each student record may have several fields, such as
  - *Name*
  - *Address*
  - *Student number*
  - *Gender*
  - *Age*
  - *GPA*
- Typically, each record in a file has the same fields.

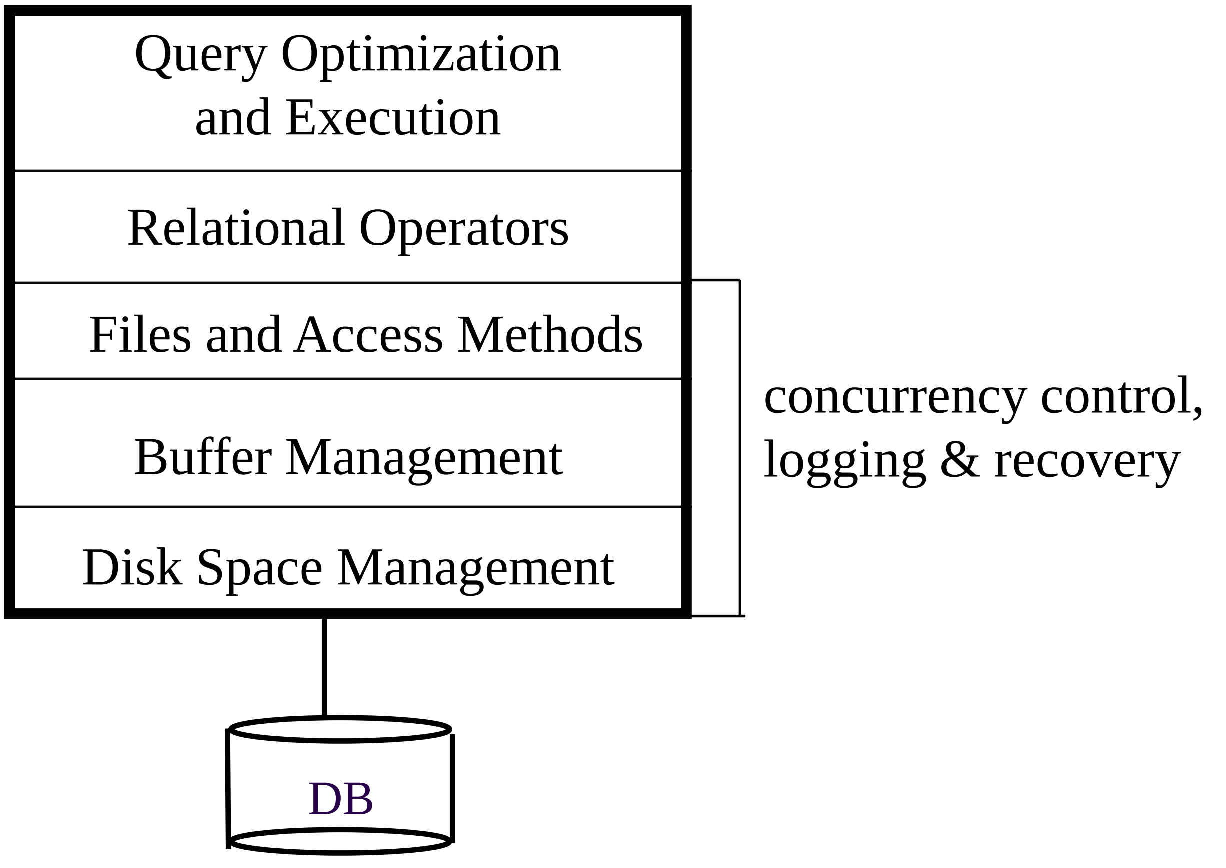
Properties of Files

1) **Persistence:** Data written into a file persists after the program stops, so the data can be used later.
2) **Shareability:** Data stored in files can be shared by many programs and users simultaneously.
3) **Size:** Data files can be very large. Typically, they cannot fit into MM.

What a DBMS provides for Applications?

- **Convenience** (simple data model, physical/logical data independence, declarative high-level query languages (SQL), transaction guarantees)
- **Multi-user** (concurency control, transaction guarantees)
- **Safety** (autorization and attacker stand points, consistent data even w/failures, related to hardware, software, power, user)
- **Persistence** (multiple application programs work on data and kept)
- **Security** (controling access to DB)
- **Reliablity** (DBMS must do the job, 99.9999%)
- **Massiveness** (millions of terabytes even per day, now going to exabytes, even to zetabytes)
- **Efficiency** (performance, thousands of queries/updates per second)

Typical DBMS architecture



- A typical DBMS has a layered architecture.
- The concurrency control and recovery components are not shown here.
- There exist other possible architectures; each DBMS has its own variations.

Key people?

- **DBMS implementer** (builds the DBMS)

- **Database designer** (establishes the schema)

- **Database application developer** (develops the programs that operate on DB)

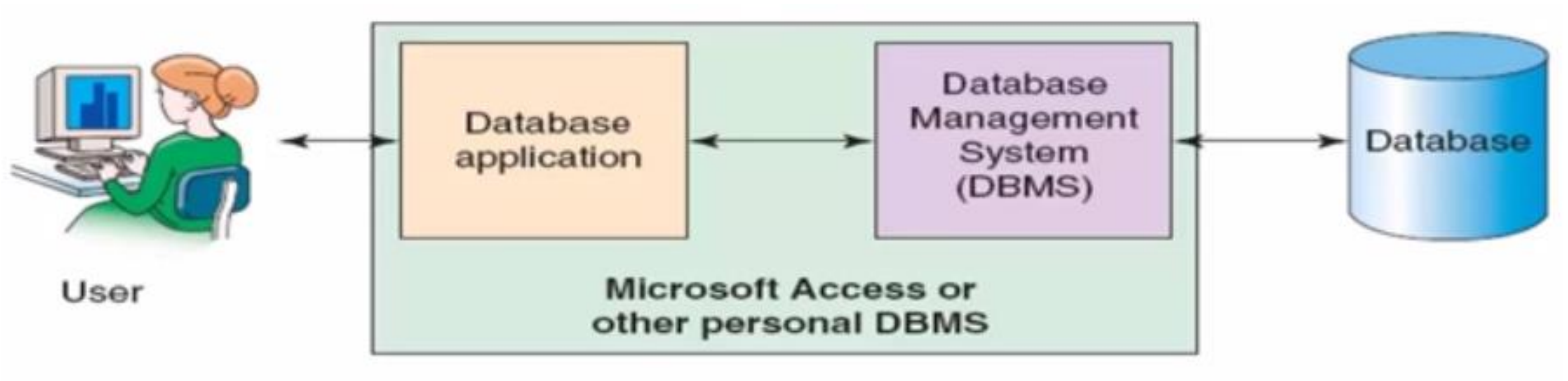
- **Database administrator** (loads data, keeps DB system running smoothly and securely)

Types of Databases

- **Databases** can be classified according to:
  - Number of users
  - Database location(s)
  - Expected type and extent of use
- **Single-user (personal) database** supports only one user at a time
  - Desktop database: single-user; runs on PC
- **Multiuser database** supports multiple users at the same time
  - **Workgroup** and **Enterprise** databases

Types of Databases

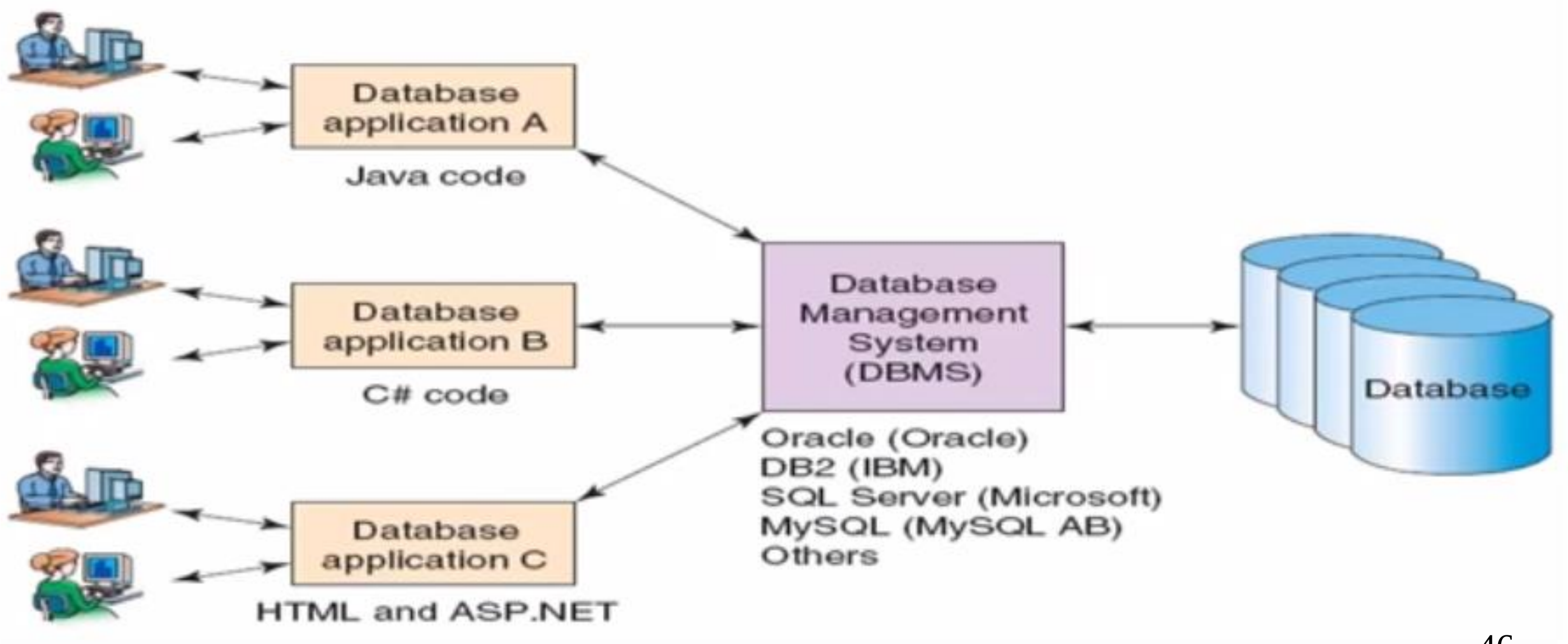
- **Personal database systems** typically:
  - Support one application
  - Have only a few tables
  - Are simple in design
  - Involve only one computer
  - Support one user at a time
  - Example Desktop DBMS Products: Microsoft Access



A **database application** is a set of one or more computer programs or websites that serve as an intermediary between the user and the DBMS.

Types of Databases

- **Enterprise-Level database systems** typically:
  - Support several users simultaneously
  - Support more than one application
  - Involve multiple computers
  - Are complex in design
  - Have many tables
  - Have many databases



Types of Databases (cont’d.)

- **Centralized database**: data located at a single site
- **Distributed database**: data distributed across several different sites
- **Operational database**: supports a company’s day-to-day operations
  - Transactional or production database
- **OLAP systems**: store historical data to be used for tactical or strategic decisions and for any type of data analtics.

Types of Databases (cont'd.)

- **Structured data** result from formatting
  - Structure applied based on type of processing to be performed
- **Unstructured data** exist in their original state
- **Semi-structured data** have been processed to some extent
- **Extensible Markup Language (XML)** represents data elements in textual format
  - **XML database** supports semi-structured XML data

Types of Databases Based on Data Models

- In the **relational database model,** the data and the database is thought of as a set of records. The relational model of data is the most widely used model today.

  Ex: MySQL, PostgreSQL, MS ACCESS

  – **Main concept**: relation, basically a table with rows and columns.
  – Every relation has a schema, which describes the columns, or fields.

- In the **object-relational database** model the data and the database is a set of complex (nested) objects.

  Ex: ORACLE, SYBASE, DB2, SQLServer

- In the **object-oriented database model** the data and the database is thought of as a set of objects.

  Ex: db4o, GemStone, Objectstore, Versant Object.

Comparison of DBMS Based on Relation or Object-Relational Data Models

TABLE 1.1 Types of Databases

PRODUCT	NUMBER OF USERS			DATA LOCATION		DATA USAGE		XML
	SINGLE USER	MULTIUSER					DATA WAREHOUSE	
		WORKGROUP	ENTERPRISE	CENTRALIZED	DISTRIBUTED	OPERATIONAL		
MS Access	X	X		X		X		
MS SQL Server	X[3]	X	X	X	X	X	X	X
IBM DB2	X[3]	X	X	X	X	X	X	X
MySQL	X	X	X	X	X	X	X	X*
Oracle RDBMS	X[3]	X	X	X	X	X	X	X

* Supports XML functions only. XML data are stored in large text objects.

Types of Databases Based on Data Models

- In **XML data model**, an XML document captures data, as a hierarchical structure of labeled values.
    - **XML Enabled;**

    Ex: IBM DB2 (pureXML), Microsoft SQL Server, Oracle DB, PostgreSQL and
    - **Native XML DBs;**

    Ex: Tamino, Sedna, Ozon, eXist, MarkLogic Server

- **Main Memory Data Models**: is a database management system that primarily relies on main memory for computer data storage. Backup copy is on disk.
    - Main memory databases are faster than disk-optimized databases because the disk access is slower than memory access, the internal optimization algorithms are simpler and execute fewer CPU instructions.
    - Applications where response time is critical, such as those running telecommunications network equipment and mobile advertising networks, sensors, and mobile devices, often use main-memory databases.

Ex: eXtremeDB, TimesTen (Oracle), Velocity (SAP), CSQL, ArrayDB

Types of Databases Based on Data Models

- **Bigdata** is data sets so large or complex that they are difficult to process using traditional data processing applications like RDBMSor ORDBMS. It has some properties, 3Vs (velocity, volume, variety), 5Vs, 8 and more Vs.
- A **NoSQL Data Model** (often interpreted as Not-only SQL) provides a mechanism for storage and retrieval of data that is modeled with non-tabular relations. NoSQL databases are increasingly used in bigdata and real-time web applications. The main NoSQL data models:
  - **Key-value Store:** is a type of nonrelational database that stores data as a collection of key-value pairs in which a key serves as a unique identifier. Both keys and values can be anything, ranging from simple objects to complex compound objects.

    Ex: CouchDB, Dynamo, MemcacheDB, Redis, Aerospike, etc.
  - **Document Store:** store all information for a given object in a single instance in the database, and every stored object can be different from every other. Collections and documents.

    Ex: Apache CouchDB, MongoDB, OrientDB, etc.
  - **Column Store:** A wide-column store uses tables, rows, and columns, but unlike a relational database, the names and format of the columns can vary from row to row in the same table.

    Ex: Google's Bigtable, Apache Cassandra, Amozon DynamoDB, HBase, Vertica, etc.
  - **Graph Databases:** uses semantic graph structures for semantic queries with notes, edges, and properties to represnet and store data.

    Ex: Neo4J, InfiniteGraph, Titan, OrientDB, Allegro, etc.
  - **Multi-model:** is a database system that can store, index and query data in more than one model. A database system that combines many of DB models is multi-model.

    Ex: OrientDB, FoundationDB, ArangoDB, Alchemy Database, CortexDB, Cassandra.

Data Model	Performance	Scalability	Flexibility	Complexity	Functionality
Key–Value Store	high	high	high	none	variable (none)
Column-Oriented Store	high	high	moderate	low	minimal
Document-Oriented Store	high	variable (high)	high	low	variable (low)
Graph Database	variable	variable	high	high	graph theory
Relational Database	variable	variable	low	moderate	relational algebra

Data Models

- **NewSQL database** is a class of modern RDBMSs that provides the similar scalable performance of NoSQL systems for **OLTP read-write workloads** while still maintaining the ACID properties of a traditional data base system.
  - A type of a **NewSQL** system is a completely new database platform that designed to operate in a distributed cluster of shared-nothing nodes, in which each node owns a subset of the data.

    Ex: Google Spanner, SAP Hana, VoltDB.
  - The other type of **NewSQL** systems are **highly optimized storage engines** for SQL. These systems provide the same programming interface as SQL, but scale better than built-in engines.

    Ex: InnoDB, InfiniDB.

- In a **Spatial Databases, Spatiotemporal Databases, Multimedia Databases, Information Retrieval Systems, Fuzzy Databases, Probabilistic Databases, Bioinformatics Databases, etc.**

Database Tendencies and Research

- Sensors record data
  - ➔ DBs grow in size
  - ➔ DBs become more widespread
  - ➔ date may be less reliable, i.e., uncertain
- Multimedia data
  - ➔ Requirements for larger storage
  - ➔ New query operations
    (e.g., find a song by humming the melody,
    find pictures with a given face)
- Data on the Web
  - ➔ Accessed/changed by many people (Facebook,…)
  - ➔ Speed up access, loosen consistency (NoSQL)