CS432: Databases

Introduction to Databases

Instructor
Yogesh K. Meena

Lecture no.
2

CSE, IIT Gandhinagar
January 9, 2025

What is a Database

A database is an organized collection of data, generally stored and accessed electronically from a computer system.

Disclaimer..

- This course is NOT
  - a tutorial on using a specific databases
  - a tutorial on SQL
  - a course on database implementation

Disclaimer..

- This course is NOT
  - a tutorial on using a specific databases
  - a tutorial on SQL
  - a course on database implementation
- But it is about learning
  - the foundations of database design
  - some SQL and relational algebra
  - optimization techniques in database design
  - managing large databases

Course Contents (Racap)

- Introduction to RDBMS.
- Structured Query Language (SQL).
- Relational Algebra, Entity-Relationship Model, Relational Database Design
- Storage and File Structure
- Application Development
- Indexing and Hashing
- Query Processing, Query Optimization - Transactions (Serializability and Recoverability)
- Concurrency Control
- Recovery Systems
- Introduction to no-SQL databases

Why study Databases?

Why study Databases?

File processing system...[data redundancy, inconsistency, difficulty in accessing data, data isolation, integrity problems, atomicity problems, concurrent-access anomalies, security problems]

Why study Databases?

File processing system...[data redundancy, inconsistency, difficulty in accessing data, data isolation, integrity problems, atomicity problems, concurrent-access anomalies, security problems]
Databases are everywhere...

- Academic Database (Students, Faculty, Staff, ...)
- Bank Database (Account holders, Account types, Locations, ...)

Why study Databases?

File processing system...[data redundancy, inconsistency, difficulty in accessing data, data isolation, integrity problems, atomicity problems, concurrent-access anomalies, security problems]
Databases are everywhere...

- Academic Database (Students, Faculty, Staff, ...)
- Bank Database (Account holders, Account types, Locations, ...)
- Youtube Database (User, Video, Comments, ...)
- Twitter Database (User, Tweets, Replies, ...)

Why study Databases?

File processing system...[data redundancy, inconsistency, difficulty in accessing data, data isolation, integrity problems, atomicity problems, concurrent-access anomalies, security problems]
Databases are everywhere...

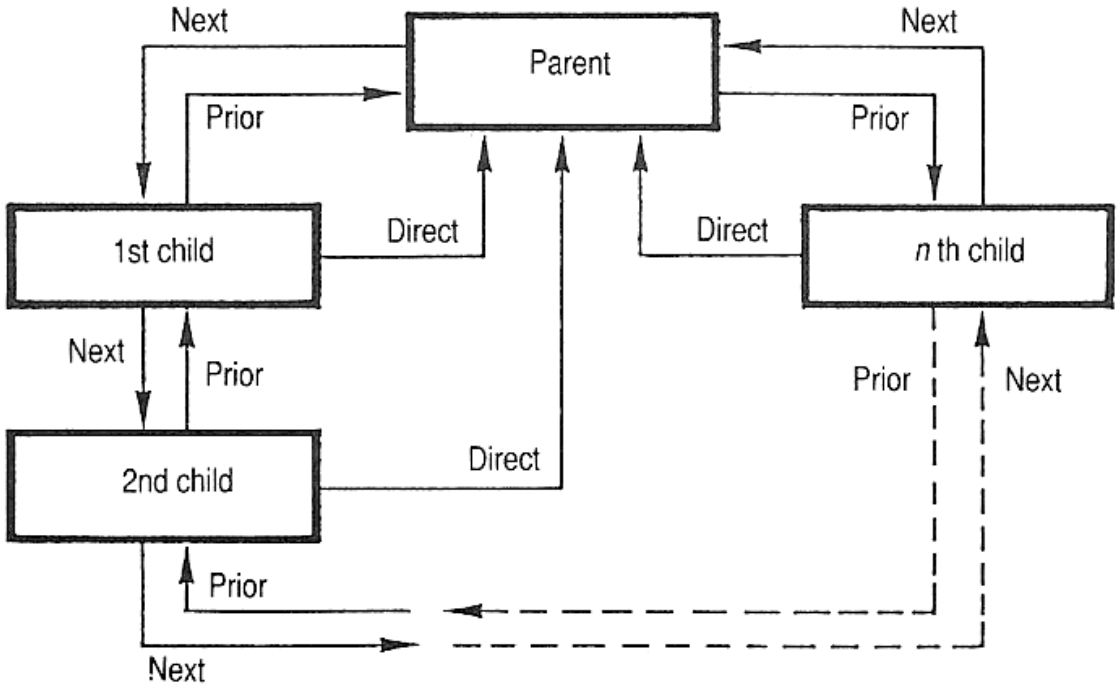
- Academic Database (Students, Faculty, Staff, ...)
- Bank Database (Account holders, Account types, Locations, ...)
- Youtube Database (User, Video, Comments, ...)
- Twitter Database (User, Tweets, Replies, ...)

Sometimes we even don't see them. Can you come up with some examples?

The paradigm shift..

The term "data-base" was coined around 1962.

- Navigational DBMS (1960')



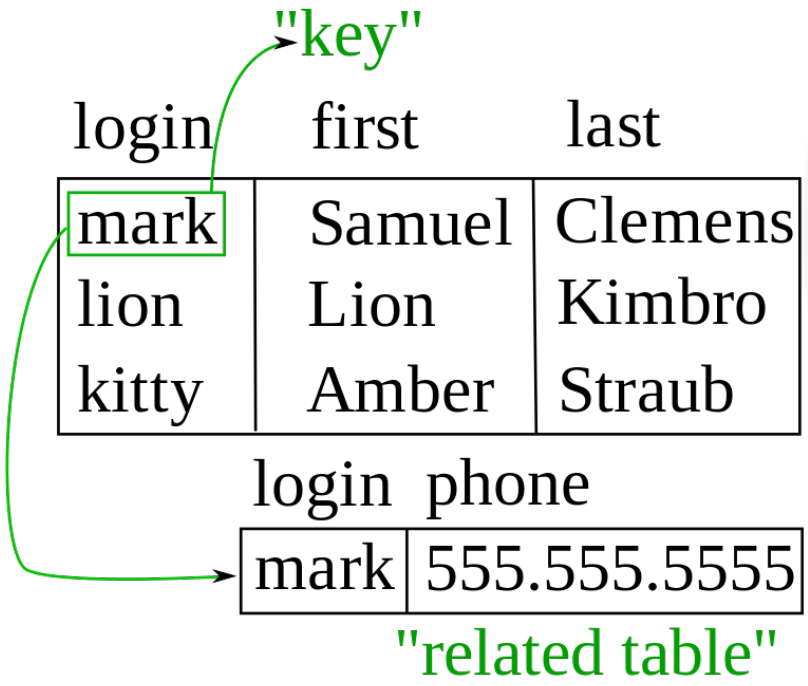
A closed chain of records in a navigational database model (e.g. CODASYL), with ***next pointers***, ***prior pointers*** and ***direct pointers*** provided by keys in the various records.

Source: Wikipedia

The paradigm shift..

The term "data-base" was coined around 1962.

- Relational DBMS (1970')



Edgar Codd, "**A Relational Model of Data for Large Shared Data Banks**"

Source: Wikipedia

Relational Databases: An example

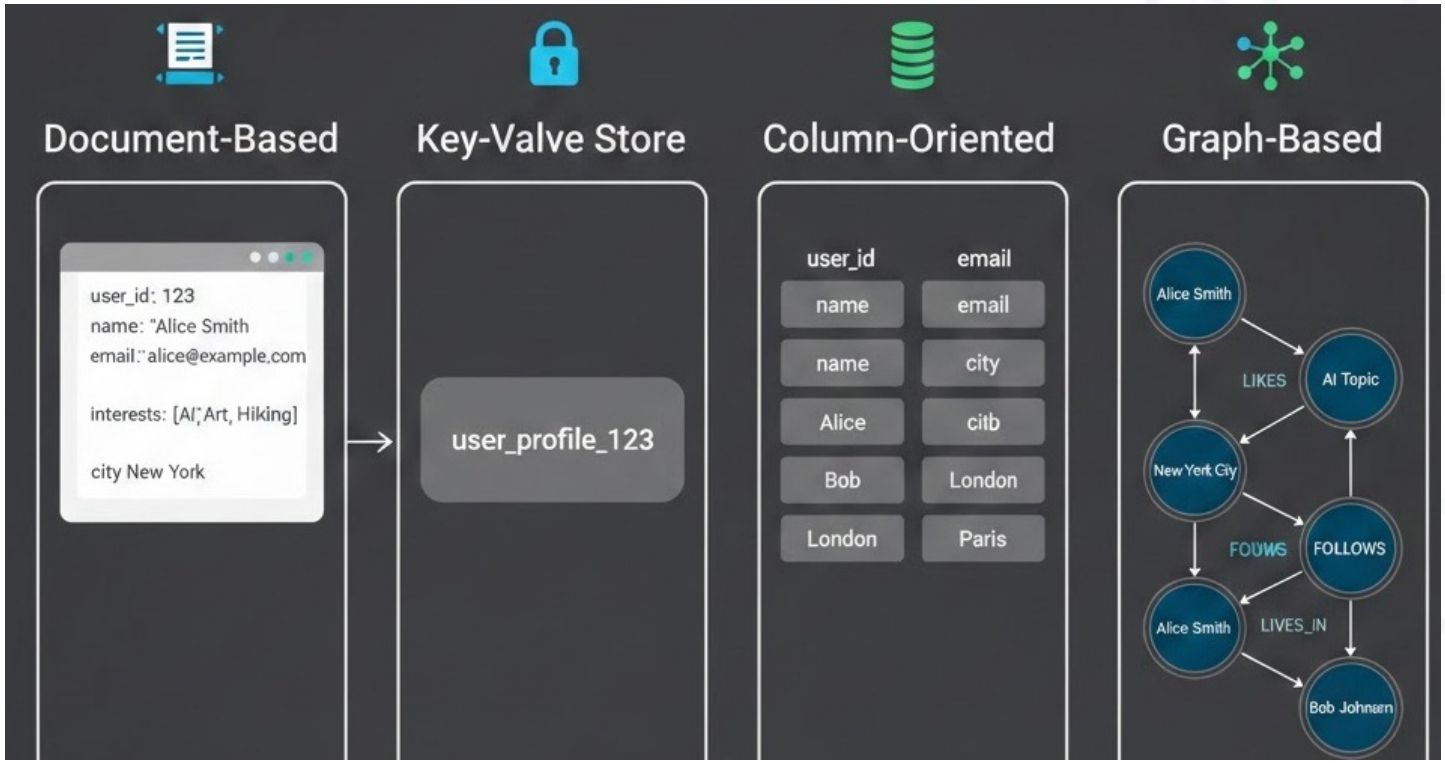
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Source: Silberschatz, Korth, Sudarshan — Database System Concepts

The paradigm shift..

The term "data-base" was coined around 1962.

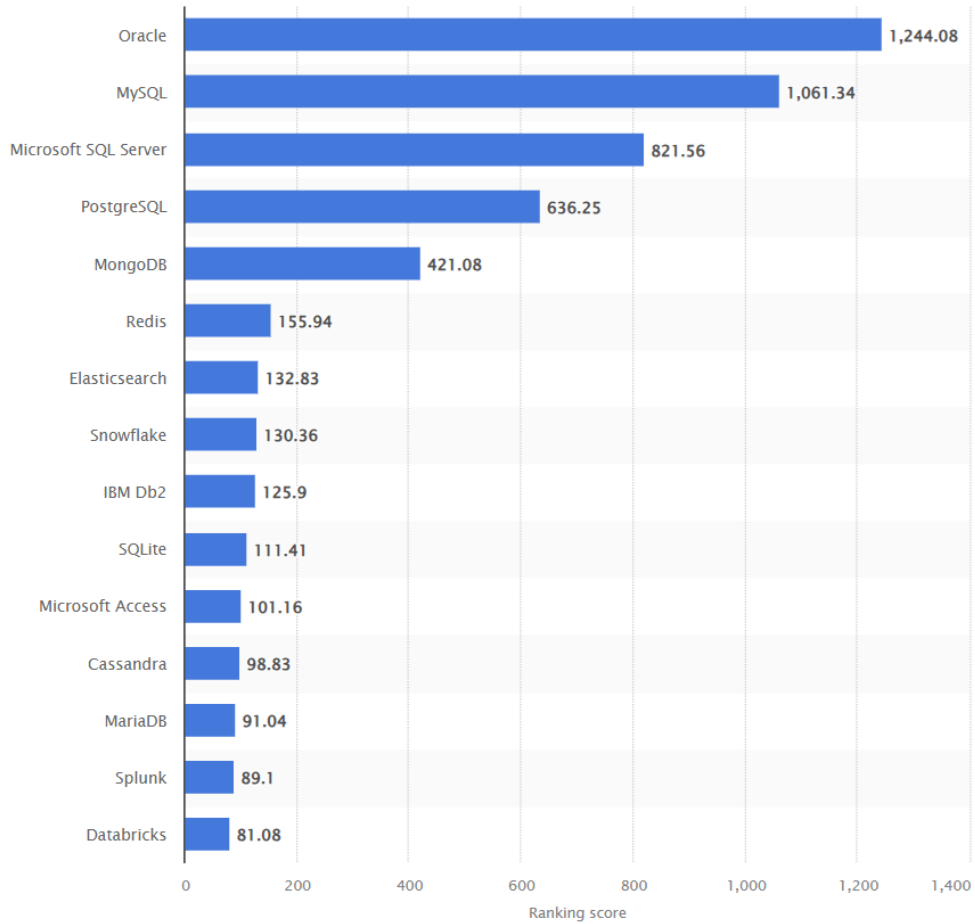
- NoSQL and NewSQL (2000')



NoSQL Database Types

Feature	Document	Key-Value	Columnar	Graph
Data Model	JSON-like documents	Key-Value pairs	Columns instead of rows	Nodes & Relationships
Best Use	Semi-structured data	Fast lookups & caching	Analytics & big data	Relationship-heavy data
Query Perf.	Moderate	Fast	High (Analytics)	Optimized (Links)
Schema	Flexible	Dynamic	Semi-structured	Schema-less
Scalability	Horizontal	High horizontal	Highly scalable	Link-based scaling
Examples	MongoDB, CouchDB	Redis, DynamoDB	Cassandra, HBase	Neo4j, Neptune

DBMS Rankings (June 2024)



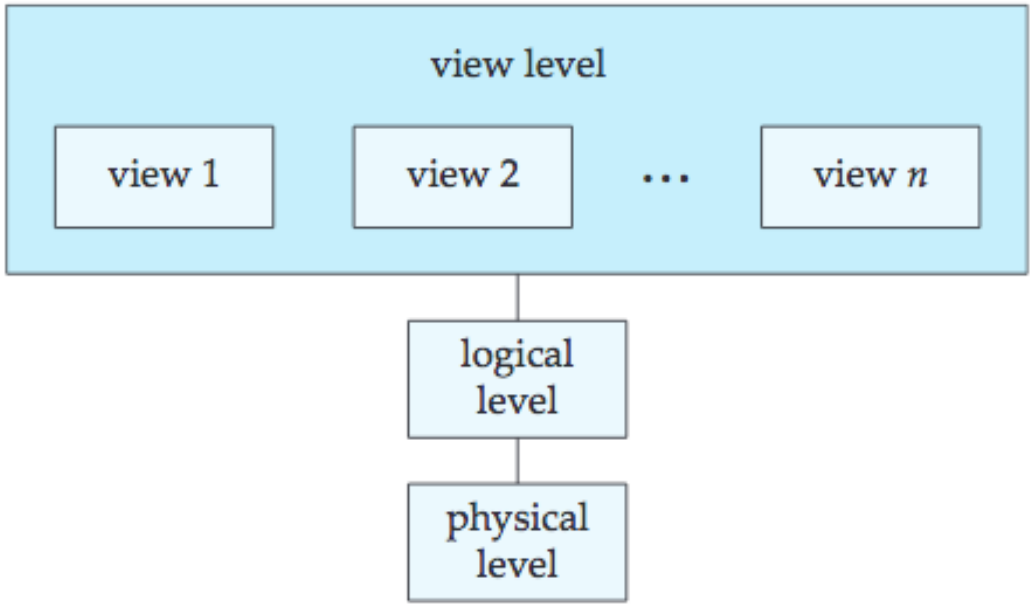
Source: statista.com

DB-Engines (source): https://db-engines.com/en/ranking

Data abstraction

Not all database-system users are not computer trained, we need to hide the complexity

- Physical level
- Logical Level
- View level



Source: Silberschatz, Korth, Sudarshan — Database System

Physical level

- How the data are actually stored in the system.
- Describes complex low-level data structures.

Logical level

- Describes what data are stored in the database.
- What relationships exist among those data.
- Describes the entire database in terms of a small number of relatively simple structures (e.g. Tables).
- **Physical data independence:** User of the logical level does not need to be aware of the complexity at physical layer.

View level

Not everything (the entire database) should be visible to everyone...

- Users need to access only a part of the database.
- Simplifies the interaction of the users with the system.
- Many views for the same database.

An example

type instructor = **record**
ID:**char**(5);
name:**varchar**(20);
deptname:**varchar**(20);
salary:**numeric**(8,2);
end;

Some more record types:

- department: dept name, building, and budget
- course: course id, title, dept name, and credits
- student: ID, name, dept name, and tot cred

How does these can be described at different levels of abstraction?

Instances and Schemas

An analogy to a program written in a programming language.

Schema

Variable declarations corresponds to Schema

Instances

Value of a variable corresponds to Instances

Instances and Schemas

Schema example

type instructor = **record**
ID:**char**(5);
Name:**varchar**(20);
DeptName:**varchar**(20);
Salary:**numeric**(8,2);
end;

Instance example

ID	Name	DeptName	Salary
C0383	Yogesh	CSE	80000.00

Schema Types

- **Logical Schema:** the overall logical structure of the database.
- **Physical schema:** the overall physical structure of the database.

Programmers/Database administrators construct applications by using the logical schema

Data Models

A collection of tools to describe:

- Data
- Data relationships
- Data semantics
- Consistency constraints

Categories of data models

- **Relational Model:** Tables (relations), Columns (fields or attributes)
- **Entity-Relationship Model:** Real objects (entities) and relationships among these objects.
- **Object-Based Data Model:** Extension of ER with encapsulation, methods (functions), and object identity.
- **Semi-structured Data Model:** XML

Some older models include: Network model and Hierarchical model

Database Languages

- **Data-Definition Language**
  - To describe the schema
  - DDL compiler generates a set of table templates stored in a data dictionary
  - Data dictionary contains metadata (i.e., data about data)
    - Database schema
    - Integrity constraints
    - Primary key (ID uniquely identifies instructors)
    - Authorization (Who can access what)

Example

```
create table instructor (
        ID char(5),
        name varchar(20),
        dept_name varchar(20),
        salary numeric(8,2))
```

Database Languages

- **Data-Manipulation Language**
  - It enables users to access or manipulate data (retrieve, insert, delete, and modify)
  - DML that involves information retrieval is called a **query language**.
    - **Procedural DMLs:** What data are needed and how to get those data.
    - **Declarative DMLs:** What data are needed without specifying how to get those data.

SQL

- One of the popular commercial language
- Not as powerful as a universal Turing machine
- It does not support:
  - Input from users
  - Output to displays
  - Communication over the network
- Above actions must be written in a host language, such as C, C++, C#, Java or Python etc.

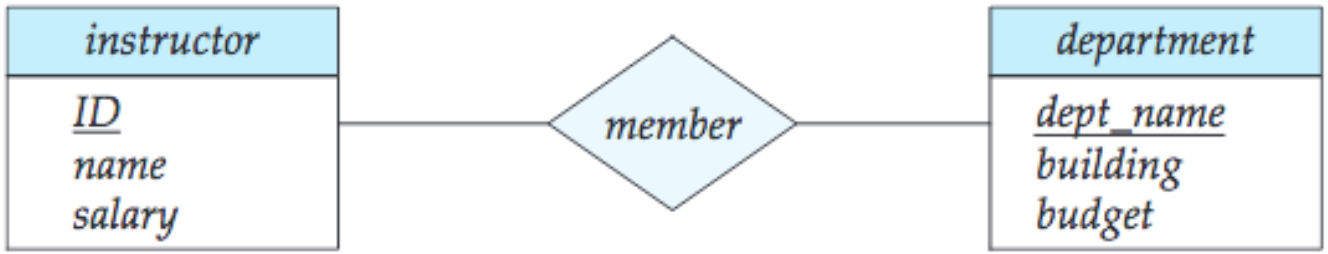
Database Design

It mainly involves the design of the database schema.

- Organizational requirements
- Conceptual schema
- Final design process
  - Logical-design phase
  - Physical-design phase
- Relational model - conceptual design process: (What) attribute want to capture in database and **(How)** to group these attribute to form the various table ("how" part is computer science problem).

Possible designing ideas (for how part)

- Entity-Relationship Model



Possible designing ideas (for how part)

- Normalization

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Repetition of information

Inability to represent certain information

Database Engine

Partitioned into **three modules** to deal with the responsibilities of the overall system.

- **Storage manager**:
  - Huge size of databases from GBs to several TBs of data.
  - Keeping everything in RAM not possible.
- **Query processor**
  - helps the database system to simplify and facilitate access to data
  - translate queries in non-procedural language into an efficient sequence of operations.
- **Transaction manager**
  - What if database fails?
  - How can it supports multiple users concurrently?

Storage manager

An **interface** between low-level data stored in the database and the application programs and queries submitted to the system.

- Interacts with the file system provided by the OS.
- Storing, retrieving, and updating data in the database

Its main component are:

- **Authorization & integrity manager**: authority of users, integrity constraints
- **File manager**: Allocation of disk space & data structures.
- **Buffer manager**: Disks to main memory fetching, what data to cache.

Storage manager

Several data structures

- **Data files**: To store the database
- **Data dictionary**: To store meta data
- **Index**: Provide fast access to data items

The Query Processor

- **DDL interpreter**: Interprets DDL statements and records the definitions in the data dictionary.
- **DML compiler**: Translates a DML statements in a query language into an evaluation plan consisting of low-level instructions [Query optimization]
- **Query evaluation engine**: Executes low-level instructions generated by the DML compiler

Transaction manager

A transaction is a collection of operations that performs a single logical function in a database application

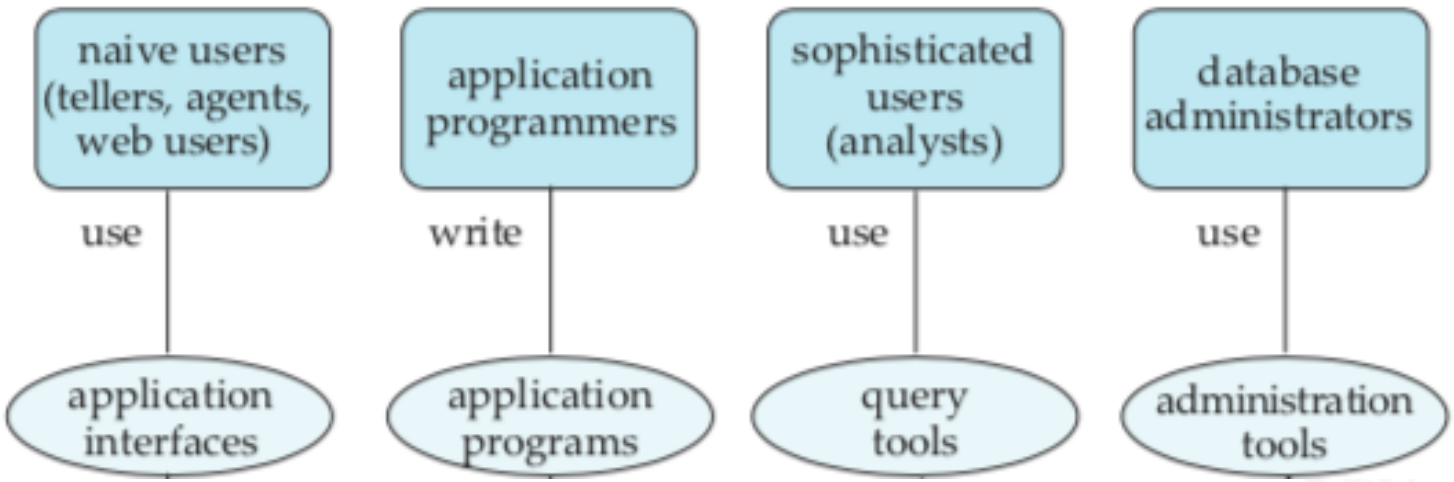
- Each transaction is a unit of both atomicity and consistency
- During the execution of a transaction, it may temporarily allow inconsistency **Any Example?**

Transaction manager

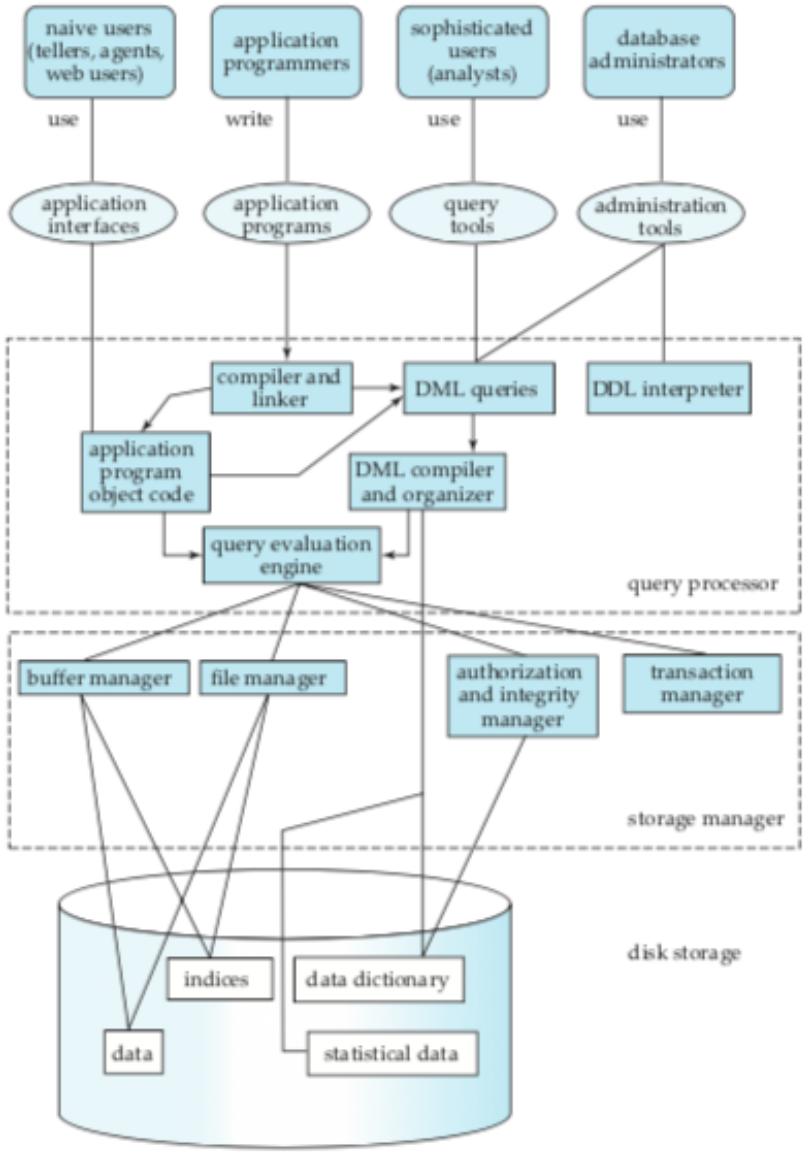
It has two components:

- **Transaction-management**: Database remains in a consistent (correct) state despite system failures
- **Concurrency-control**: It controls the interaction among the concurrent transactions.

Database Users



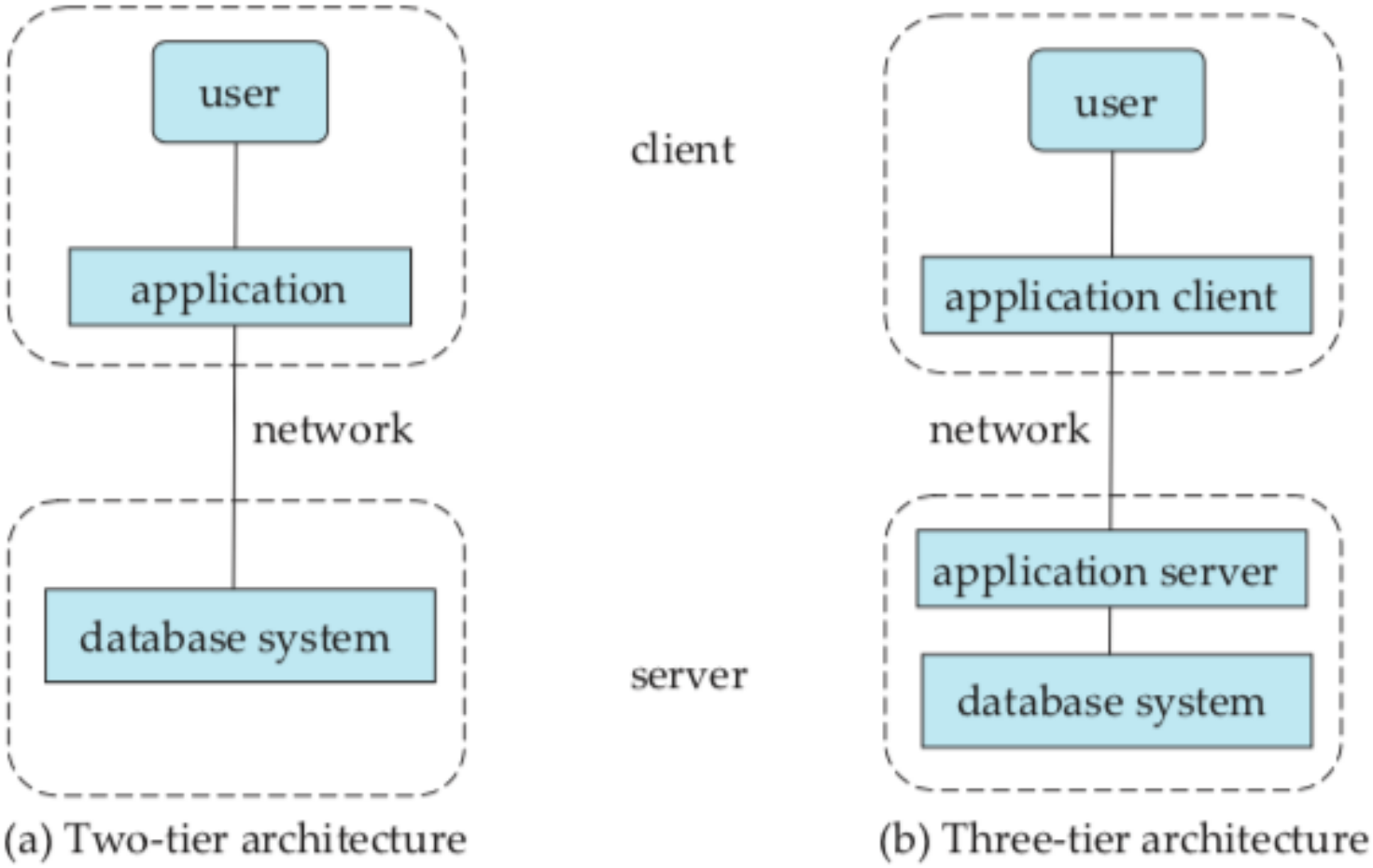
System structure



Database Architecture

- Centralized
- Client-server
- Parallel (multi-processor)
- Distributed

Client-server Architecture



(a) Two-tier architecture

(b) Three-tier architecture

The ACID properties

- **Atomicity**: Either succeeds completely, or fails completely.
- **Consistency**: Any data written to the database must be valid according to all defined rules, including constraints, cascades, triggers, and any combination thereof.
- **Isolation**: Concurrent execution of transactions leaves the database in the same state that would have been obtained if the transactions were executed sequentially.
- **Durability**: Guarantees that once a transaction has been committed, it will remain committed even in the case of a system failure.

Acknowledgments/Contributions

- Some of the images utilized in these slides are subject to copyright - Abraham Silberschatz, Henry Korth, and S. Sudarshan. Database System Concepts. 6th Edition, McGraw-Hill Education
- Contributor 2024-25, 2025-26 - Yogesh K. Meena, IIT Gandhinagar