CSE370: DATABASE SYSTEMS

Lecture 1: Introduction to Databases

An overview of foundational database concepts

What is a Database?

- Organized collection of related data
- Users store, retrieve, and update data efficiently
- Built for a specific purpose e.g. Airlines Reservation, University Student Information Management etc.

Types of Data

Focus of our course

Structured Data

Structured data is highly organized, has pre-defined formats and data types, can be stored in tabular format, making it easily searchable and analyzable. Examples include data in spreadsheets and relational databases

Unstructured Data

Unstructured data lacks a specific format or organization, making it more challenging to search and analyze. Common examples include text documents, images, audio, social media posts and videos that cannot be stored in a tabular format

Semi-Structured Data

Semi-structured data does not follow a strict tabular structure but still has organizational properties with tags or markers, making it easier to process than unstructured data. Examples include XML or JSON files, data stored in NoSQL databases (like mongoDB)

Database VS File-Based Approach

Database Advantages

📉 Less Data Redundancy – Shared data, no duplicate storage

👥 Multiple Views – Many users, same database but supports different user view

🔐 Access Control – Only authorized users can access

🔄 Concurrency Control – Multiple users update same data, data remains consistent and valid

💾 Backup & Recovery – Restore data after system failures

🔗 Complex Relationships – Relate data easily

✅ Integrity Constraints – Data rules (ex. must be unique or cannot be null) enforced automatically

⚙️ Optimized Queries – Fast and efficient data retrieval

Case Study:

You want to store patient records, doctor schedules, and billing information for a hospital. Different departments (like emergency, radiology, and pharmacy) need access to specific parts of the data. The system must ensure data privacy and allow multiple staff members to use it and update data at the same time.

Database VS File-Based Approach

Database Disadvantages

💰 High Initial Cost – Software + setup cost + developer cost

💻 Extra Hardware May Be Needed

📦 Overhead – Managing security, recovery, concurrency etc. requires computing resources

💰 Maintenance Cost – Need to hire technical support for system maintenance

If your data and application are simple and won't change much, or if only one user or device needs access, or if database facilities such as security, concurrency, etc. are not required, then there is no need to incur the financial cost of developing and maintaining a database, and also there is no need to use up computing resources unnecessarily.

Case Study:

You are keeping track of your daily expenses, recording the date, category, and amount spent. This data is only for your personal use, and you don't need to apply complex queries and high security is not needed.

Database Users (1)

🎨 **Database Designers:** Communicate with end users and stakeholders to understand data requirements. They define the strcuture and constraints of the database, ensuring the design is efficient and scalable, and meets the specific data requirements.

💻 **Software Engineers:** They implement the database using a dbms and develop the applications that interact with the database using different programming languages. They write code, and optimize queries to enhance performance. Their role is to create user-friendly database interfaces.

Database Users (2)

🛠️ **Database Administrators (DBAs):** They aquire software and hardware resources, manage database performance, security, and integrity. They perform backups, system updates, and troubleshooting, DBAs ensure the database is protected from unauthorized access. They monitor and control the use and performance of the database.

👥 **End Users:** These are the individuals who interact with the database's output through reports, dashboards, or applications. They can update the data through web-applications and provide feedback on the application.

DBMS(Database Management System)

A software package or system that facilitates the creation and maintenance of a computerized database. A DBMS provides functionalities to define the database structure; manipulate data; allow data sharing and concurrent processing; and ensure security, protection, recovery, and backup.

Types of DBMS

🔷 **Relational DBMSs (e.g., Oracle db, MySQL, PostgreSQL)**
Used in structured environments where data is stored in tables (relations) with predefined schemas. Ideal for applications needing complex queries and transaction management.

📜 **Document DBMSs (e.g., MongoDB)**
Ideal for handling semi-structured data, store data in document formats (usually JSON or BSON). They offer flexibility in data modeling, making them suitable for applications with evolving schemas.

🔑 **Key-Value Stores (e.g., Redis)**
These databases are designed for simplicity and speed, storing data as a collection of key-value pairs. They are perfect for caching and real-time analytics, where quick data retrieval is essential.

🔗 **Graph DBMSs (e.g., Neo4j)**
Store data as nodes (entities) and edges (relationships), often with properties. Best for relationship-centric use cases such as recommendation systems, and social graphs.

There are many other types of DBMSs. The choice of DBMS depends on the type of data and application requirements

DBMS Languages

SQL (Structured Query Language) is mostly used with Relational DBMS. The language can be divided into the following categories:

Data Definition Language (DDL)

Defines or alters the database structure/schema. Eg. Create a table; add/remove columns; change column names, data types or format etc.

SQL commands → Create Table, Alter Table: impact the structure of a database

Data Manipulation Language (DML)

Modifies/retrieves data by inserting new records/rows; updating existing values; deleting existing records/rows and retrieving stored data.

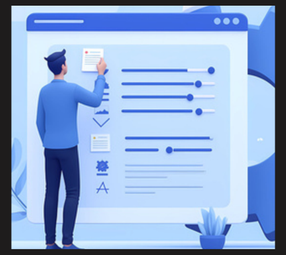
SQL commands → Insert Into, Delete from, Update, Select: manipulate/access the data values stored in the database.

There are other “advanced” language types, such as DCL (data control language) and TCL (transaction control language)

DBMS Interfaces

Standalone Query Language Interface

Programmer interfaces- embed dml in programming languages



User friendly interfaces - menu, forms, natural language, graphics etc

Database System

The DBMS software together with the data itself. Sometimes, the applications are also included.

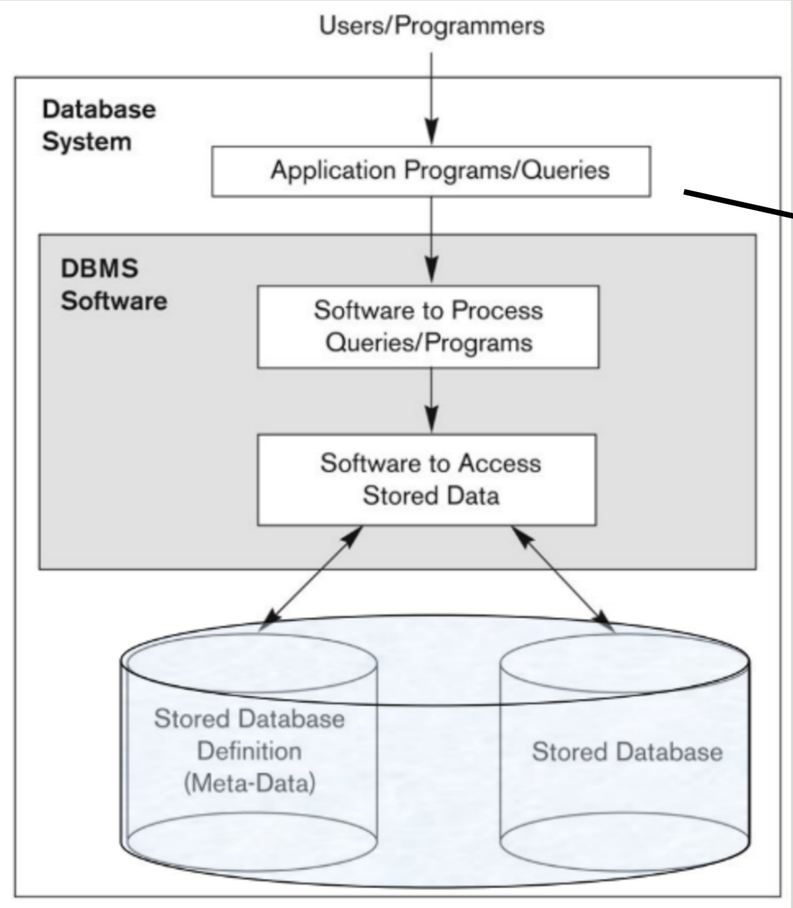


Fig: Simplified Database System Environment

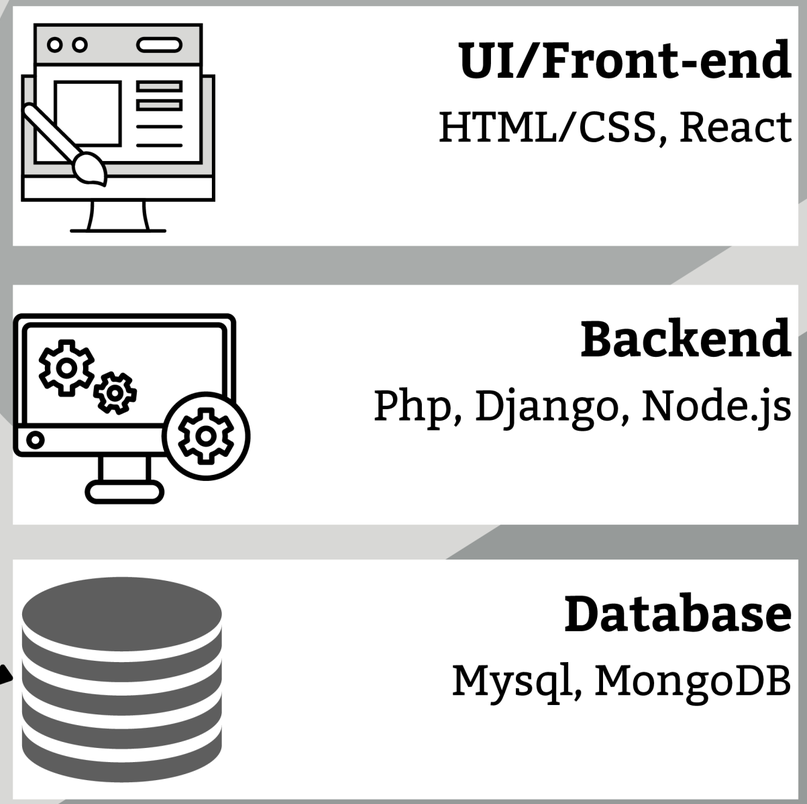
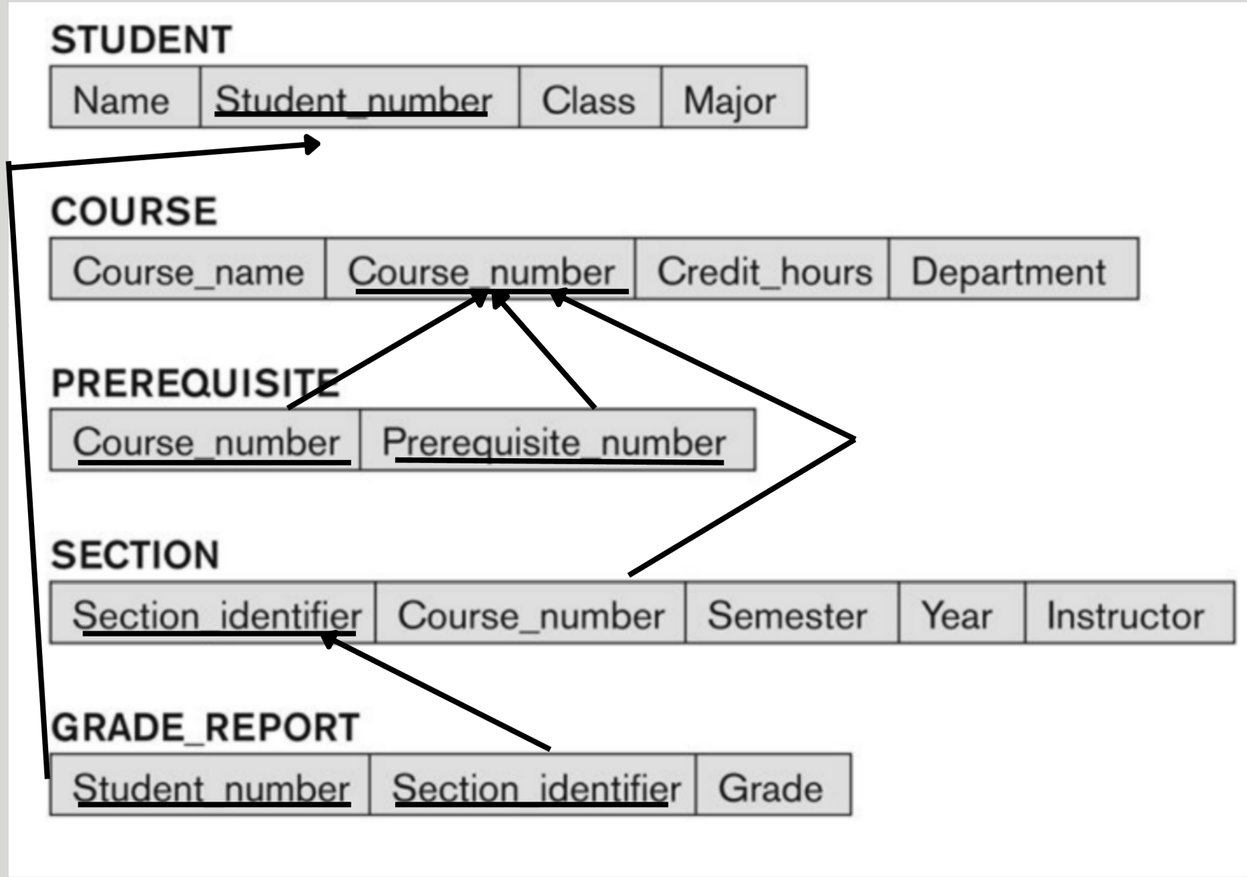


Fig: Full-Stack for a Database Web Application

Database Schema

In a relational database, data is stored in a tabular structure. The Database Schema is the description of a database. It includes descriptions of the database structure, data types, and the constraints on the database. Below is a schema diagram- an illustrative display of (most aspects of) a database schema.



- The database schema should not change very frequently.
- To change/create the schema of a database DDL queries are required.

Database State

Database State is the actual data stored in a database at a particular moment in time. This includes the collection of all the data in the database. Also called database instance (or occurrence or snapshot).

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

COURSE

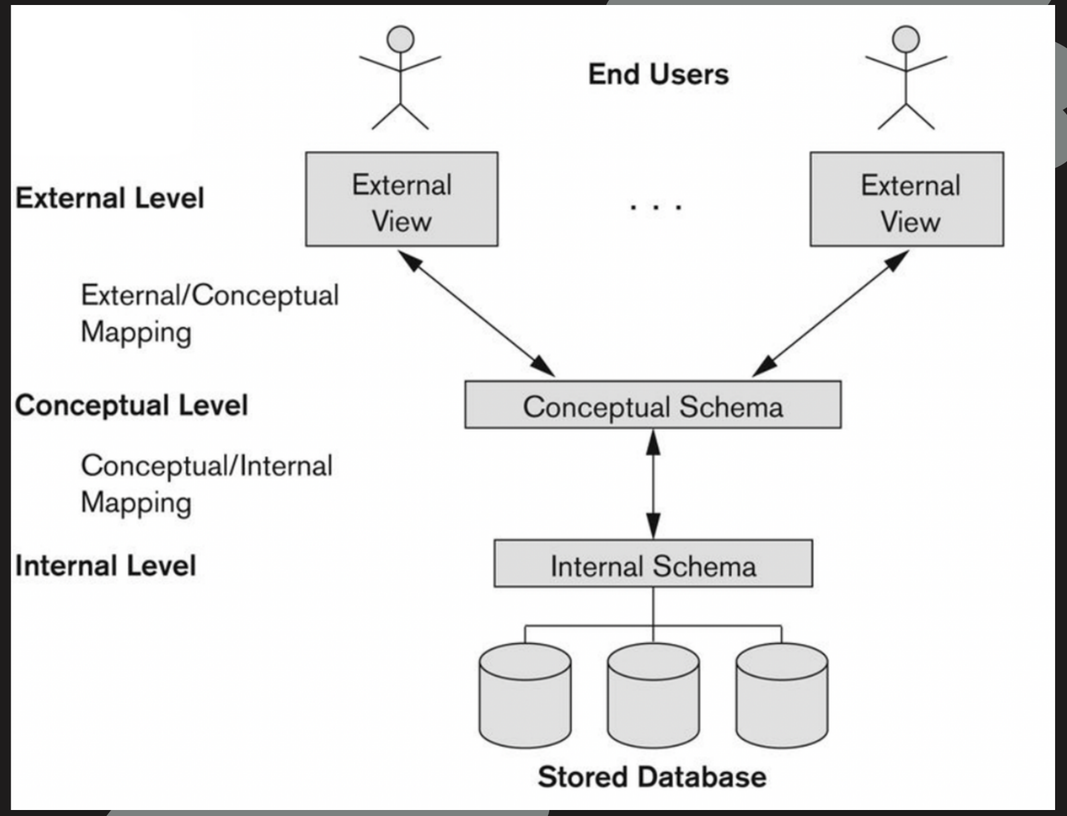
Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

- The database state changes every time the database is updated.
- To change/view the state of a database DML queries are required

Three-Schema Architecture



The Database Schema is defined in 3 levels:

- **Internal Schema:**
  At the internal level to describe physical storage structures and access paths (e.g indexes).
- **Conceptual Schema**
  At the conceptual level to describe the structure and constraints for the whole database for a community of users.
- **External Schema**
  At the external level to describe the various user views.

Data Independence

The ability to change the schema at one level without impacting the schema at the next higher level. There are two types of data independence:

Logical Data Independence

The capacity to change the conceptual schema without having to change the external schemas and their associated application programs.

Example: adding a new column that is not shown in the app or changing the datatype which does not change the external view.

Physical Data Independence

The capacity to change the internal schema without having to change the conceptual schema.

Example: internal schema changes when certain file structures are reorganized (e.g. changing file organization from heap data structure to hash tables) or new indexes are created to improve database performance,

Data Models

Data Abstraction is an important charecteristics of the database approach. It means to hide storage details and present the users with a conceptual view of the database, highligting only the essential features. Data models are used to achieve data abstraction.

Conceptual Data Model

Highest level of abstraction. Provide concepts that are close to the way many users perceive data. It is used to represent conceptual schema of a database. ER/EER diagrams are most commonly used conceptual models. [Note: customized ER diagrams per user group/UI mockups can be used to represent external schema]

Physical Data Model

Lowest level of abstraction. Provide concepts that describe details of how data is stored in the computer. It is used to represent the internat schema of a database. These are usually specified in an ad-hoc manner through DBMS design and administration manuals.

Implementation Data Model

Provide concepts that fall between conceptual and physical, it still hides the physical storage details, but it is close to how a real database will be implemented using a dbms. It is also used to represent the conceptual schema of a database using the relational schema diagram.

What Next?

LECTURE 2: CONCEPTUAL DATA MODELING USING ER

LOADING.....