Module 3 – Types of Database Model

What is a Database Model?

A **database model** is a way to **organize data** inside a database. It defines **how data is stored, connected, and accessed.**
Just like you use different ways to organize your room (books on shelves, clothes in drawers), databases use different models to organize data.

Main Types of Database Models:

1. Hierarchical Model
2. Network Model
3. Relational Model
4. Object-Oriented Model
5. Entity-Relationship (ER) Model

1. Hierarchical Model

Idea:
Data is arranged like a **tree**. One parent has one or more children, but each child has only **one parent**.

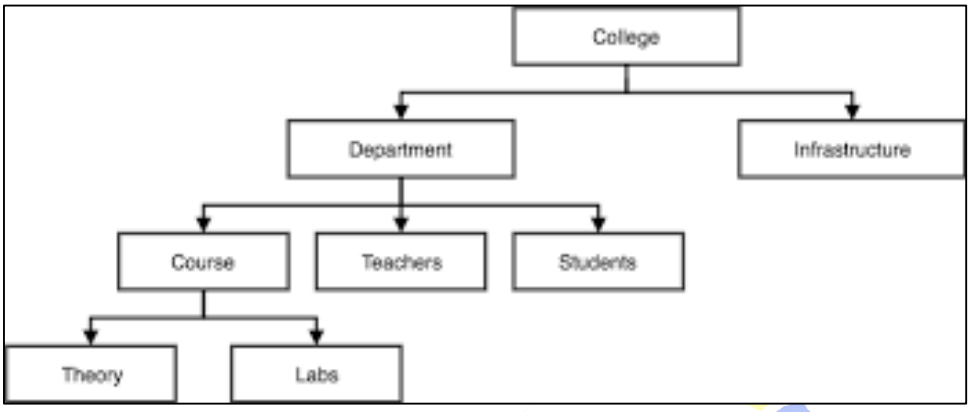
Structure:
Like folders and subfolders on your computer.

Example:
A company database:
- Company (Parent)
  - Departments (Child)
    - Employees (Child of Department)

Limitations:
- Can't handle complex relationships (e.g., if an employee works in two departments).
- Difficult to change structure.

Data and Database Management System

Real-life Analogy:
Family tree — one parent, many children.



2. Network Model

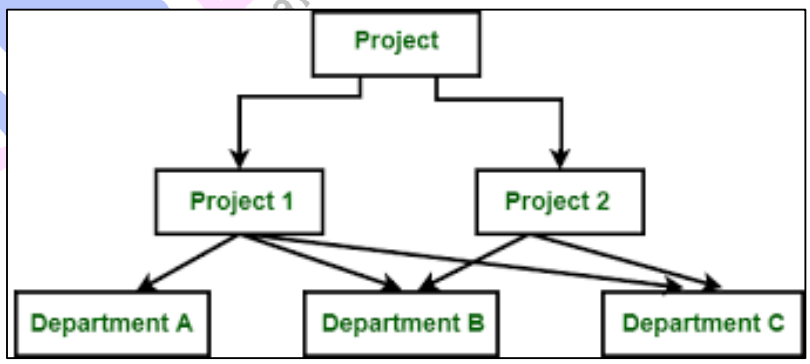
Idea:
More flexible than hierarchical. A child can have **more than one parent**.
Data is organized as records connected by **pointers** (links).

Structure:
Like a **graph** with multiple connections.

Example:
In a university:
- Students can enroll in many courses
- Courses can have many students

Advantages:
- Represents many-to-many relationships well.

Limitations:
- Complex to manage
- Harder to design than relational



3. Relational Model

Idea:
Data is stored in **tables (rows and columns)**.
Each table is called a **relation**.

Structure:
Simple and organized like spreadsheets.

Example:
A table of students:

ID	Name	Age
1	Ravi	20
2	Sita	22

Another table for courses:
A third table links students and courses:

CourseID	CourseName
101	Math
102	Physics

StudentID	CourseID
1	101
2	102

Advantages:
- Easy to use SQL
- Popular and widely used (MySQL, Oracle, PostgreSQL)

Limitations:
- Not the best for very large or unstructured data

4. Object-Oriented Model

Idea:
Stores data as **objects** (like in object-oriented programming).
Each object contains both data and behavior (methods).

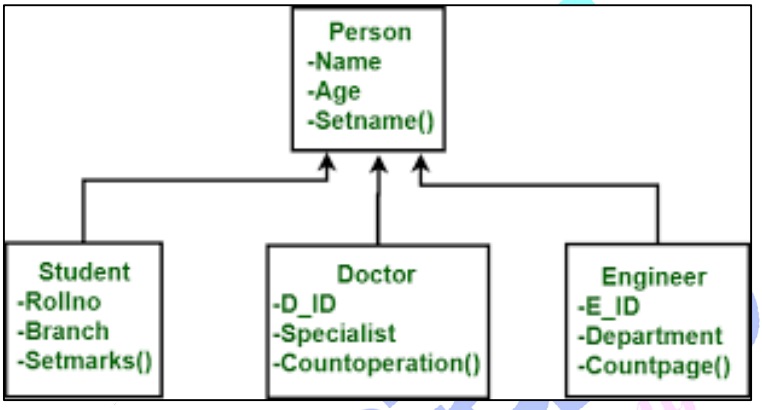
Structure:
Similar to how Java or C++ defines classes.

Example:
An object for a "Student" might contain:
- Data: Name, Age, ID

• Methods: enroll(), updateMarks()

Advantages:
• Works well with complex data
• Good for multimedia, CAD, etc.

Limitations:
• Less common in commercial DBs
• Harder to learn than relational model



5. Entity-Relationship (ER) Model

Idea:
Used for **database design**. Not used to store data, but to **plan and visualize** relationships before building the database.

Structure:
• Entities: objects (like Student, Course)
• Attributes: properties (like Name, ID)
• Relationships: connections (like "enrolls in")

Example:
Student —(enrolls in)—> Course

Advantages:
• Helps design before creating the real database

Data and Database Management System

Limitations:

- Not used for storage – only for planning