Database and its Applications

Data Models and Mathematical Foundations

Pooja T S

Computer Applications

Database and its Applications

Data Models and Mathematical Foundations

Data Models

Pooja T S

Computer Applications

Data Models in Databases

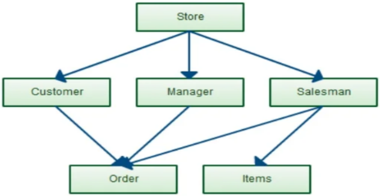
- Data models define how data is structured, stored, and manipulated.
- They provide a conceptual framework that determines:
  - How relationships among data are represented
  - How operations can be performed on data
  - How constraints and rules are applied
- We will explore five classical models: Network, Hierarchical, Hybrid, Object-Oriented, and Document.

Introduction

- Definition: A data model is a **conceptual framework** that determines how data is organized and accessed.
- Purpose:
  - Provides abstraction from physical storage
  - Defines consistency and constraints
  - Enables communication between developers, DBAs, and end-users
- Evolution:
  - Early models focused on rigid structures (hierarchical, network).
  - Later models introduced flexibility (document, object-oriented).
  - Today, hybrid and multi-model databases combine strengths.

Network Model

- Organizes data into records connected by explicit links.
- Each record may have multiple parent and child relationships (many-to-many).
- Based on **pointers** to establish connections.
- Advantages:
  - Efficient traversal of complex relationships
  - Suited for applications with interconnected data



Network Model- Example

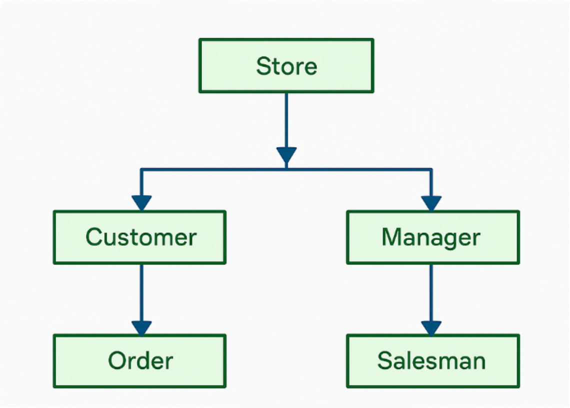
- Data is organised as records (nodes) connected by links (edges).
- Entities can have many-to-many relationships.
- Structure:
  - Store is linked to Customer, Manager, Salesman.
  - Customer, Manager, Salesman connect to Order.
  - Order further links to Items.
- Implications:
  - A customer can place many orders.
  - A salesman can handle many orders/items.
  - Orders can include multiple items.

Hierarchical Model

- Organizes data into a tree-like structure (parent → child).
- Each child has exactly one parent; supports one-to-many relationships.
- Advantages:
  - Simple, intuitive model
  - Fast navigation along hierarchy
- Limitations:
  - Difficult to model many-to-many relationships
  - Rigid schema makes evolution hard
- Example: Company database (Company → Departments → Employees).

Database and its Applications

Hierarchical Model-Example



Hierarchical Model – Example

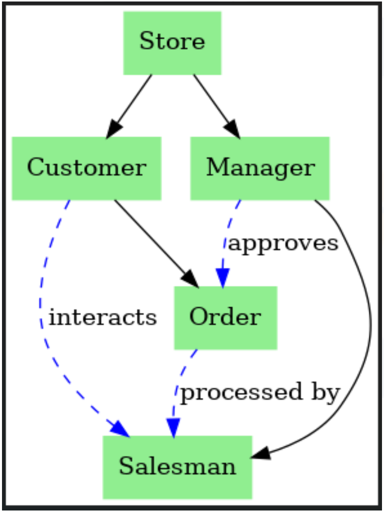
- Data is organised in a **tree structure** (parent–child).
- Each child has only one parent, but a parent can have multiple children.
- Structure:
  - Store is the root node.
  - Store connects to Customer and Manager.
  - Customer connects to Order.
  - Manager connects to Salesman.
- Implications:
  - A store can have many customers and managers.
  - Each customer can place multiple orders.
  - Each manager supervises multiple salesmen.
  - Data must be accessed by navigating parent → child links.

Hybrid Model

- Combines features of **network and hierarchical** models (sometimes also relational).
- Supports both tree-like navigation and graph-like cross-links.
- Provides flexibility for enterprise applications with diverse data types.
- Example: Telecom billing systems (customers, accounts, call records, services).
- Used in **early ERP systems** and remains relevant in modern multi-model databases.

Database and its Applications

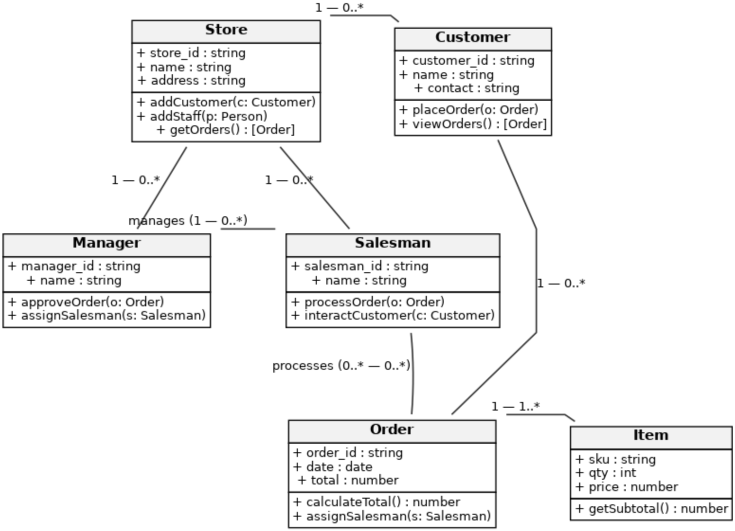
Hybrid Model-Example



Object-Oriented Model

- Extends the idea of data storage by integrating concepts of object-oriented programming.
- Data represented as **objects** containing attributes (data) and methods (behavior).
- Key Features:
  - Inheritance – reuse properties across classes
  - Encapsulation – hide implementation details
  - Polymorphism – methods operate differently based on objects
- Example: CAD/CAM, multimedia databases, scientific research repositories.

Object Oriented Model- Example



1 — 0..*

Store

+ store_id : string
+ name : string
+ address : string

+ addCustomer(c: Customer)
+ addStaff(p: Person)
+ getOrders() : [Order]

Customer

+ customer_id : string
+ name : string
+ contact : string

+ placeOrder(o: Order)
+ viewOrders() : [Order]

1 — 0..*

1 — 0..*

manages (1 — 0..*)

Manager

+ manager_id : string
+ name : string

+ approveOrder(o: Order)
+ assignSalesman(s: Salesman)

Salesman

+ salesman_id : string
+ name : string

+ processOrder(o: Order)
+ interactCustomer(c: Customer)

1 — 0..*

processes (0..* — 0..*)

1 — 1..*

Order

+ order_id : string
+ date : date
+ total : number

+ calculateTotal() : number
+ assignSalesman(s: Salesman)

Item

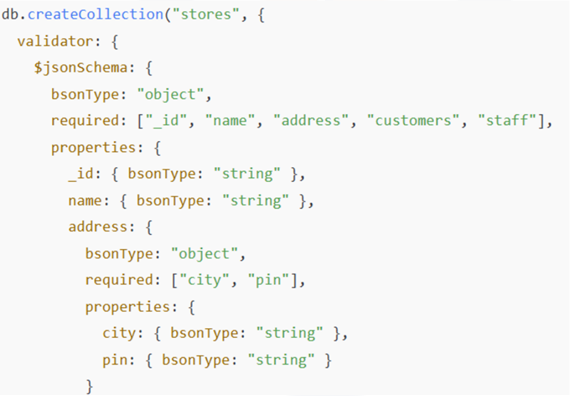
+ sku : string
+ qty : int
+ price : number

+ getSubtotal() : number

Document Model

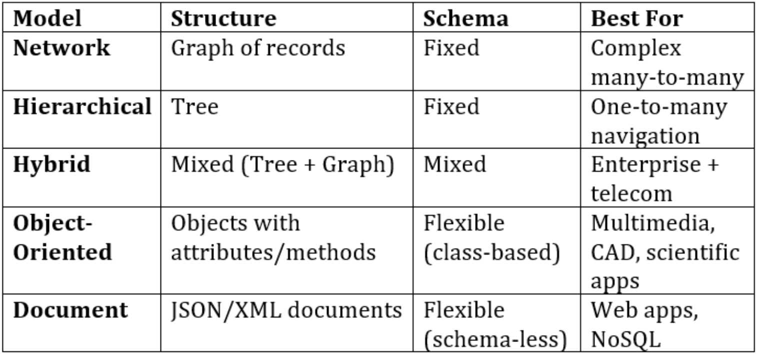
- Stores information as documents, typically JSON, XML, or BSON.
- **Schema-less** design – records in the same collection may differ in fields.
- Advantages:
  - Flexibility – easy to evolve structure
  - Natural representation of nested or complex data
  - Optimized for modern web/mobile applications
- Example: MongoDB, CouchDB used in e-commerce catalogs, social media feeds.

Document Model- Example



```
db.createCollection("stores", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["_id", "name", "address", "customers", "staff"],
      properties: {
        _id: { bsonType: "string" },
        name: { bsonType: "string" },
        address: {
          bsonType: "object",
          required: ["city", "pin"],
          properties: {
            city: { bsonType: "string" },
            pin: { bsonType: "string" }
          }
```

Comparison Table- Example



Model	Structure	Schema	Best For
Network	Graph of records	Fixed	Complex many-to-many
Hierarchical	Tree	Fixed	One-to-many navigation
Hybrid	Mixed (Tree + Graph)	Mixed	Enterprise + telecom
Object-Oriented	Objects with attributes/methods	Flexible (class-based)	Multimedia, CAD, scientific apps
Document	JSON/XML documents	Flexible (schema-less)	Web apps, NoSQL

Figure: Comparision of Models

Summary

- Each data model emerged to solve specific storage and representation needs.
- Early models (network, hierarchical) laid foundation for consistency and efficiency.
- Object-oriented and document models provide flexibility for modern applications.
- No single “best” model – selection depends on data type, complexity, and use case.

Interactive Activity

- Identify and Match the real-world scenario to the data model:
  - Airline booking
  - Organization chart
  - Telecom billing
  - Multimedia database
  - E-commerce catalog
- Justify your answers in 1–2 lines.

Thank You

Pooja T S
Assistant Professor
Department of Computer Applications

poojats@pes.edu
080-26721983 Extn: 233