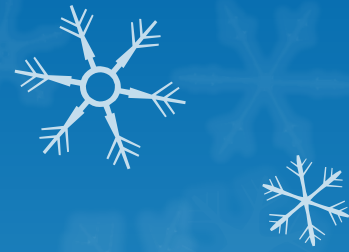


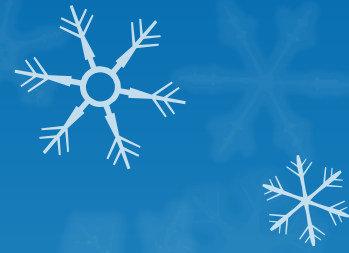
Course Name:
**Database Management
Systems**



Lecture 3






Topics to be covered

- ❑ Data Models
- ❑ DBMS Languages







Data Models



- **Data Model:** A set of concepts to describe the *structure* of a database, and certain *constraints* that the database should obey.
- **Data Model Operations:** Operations for specifying database retrievals and updates by referring to the concepts of the data model. Operations on the data model may include *basic operations* and *user-defined operations*.

Categories of data models



- **Conceptual (high-level, semantic)** data models: Provide concepts that are close to the way many users *perceive* data. (Also called **entity-based** or **object-based** data models.)
 - **Physical (low-level, internal)** data models: Provide concepts that describe details of how data is stored in the computer.
 - **Implementation (representational)** data models: Provide concepts that fall between the above two, balancing user views with some computer storage details.
- 

History of Data Models

- Relational Model: proposed in 1970 by E.F. Codd (IBM), first commercial system in 1981-82. Now in several commercial products (DB2, ORACLE, SQL Server, SYBASE, INFORMIX).
- Network Model: the first one to be implemented by Honeywell in 1964-65 (IDS System). Adopted heavily due to the support by CODASYL (CODASYL - DBTG report of 1971). Later implemented in a large variety of systems - IDMS (Cullinet - now CA), DMS 1100 (Unisys), IMAGE (H.P.), VAX -DBMS (Digital Equipment Corp.).
- Hierarchical Data Model: implemented in a joint effort by IBM and North American Rockwell around 1965. Resulted in the IMS family of systems. The most popular model. Other system based on this model: System 2k (SAS inc.)

History of Data Models

- Object-oriented Data Model(s): several models have been proposed for implementing in a database system. One set comprises models of persistent O-O Programming Languages such as C++ (e.g., in OBJECTSTORE or VERSANT), and Smalltalk (e.g., in GEMSTONE). Additionally, systems like O₂, ORION (at MCC - then ITASCA), IRIS (at H.P.- used in Open OODB).
- Object-Relational Models: Most Recent Trend. Started with Informix Universal Server. Exemplified in the latest versions of Oracle-10i, DB2, and SQL Server etc. systems.

Hierarchical Model



- **ADVANTAGES:**

- Hierarchical Model is simple to construct and operate on
- Corresponds to a number of natural hierarchically organized domains - e.g., assemblies in manufacturing, personnel organization in companies
- Language is simple; uses constructs like GET, GET UNIQUE, GET NEXT, GET NEXT WITHIN PARENT etc.

- **DISADVANTAGES:**

- Navigational and procedural nature of processing
- Database is visualized as a linear arrangement of records
- Little scope for "query optimization"

Network Model

- **ADVANTAGES:**






- Network Model is able to model complex relationships and represents semantics of add/delete on the relationships.
- Can handle most situations for modeling using record types and relationship types.
- Language is navigational; uses constructs like FIND, FIND member, FIND owner, FIND NEXT within set, GET etc. Programmers can do optimal navigation through the database.

- **DISADVANTAGES:**

- Navigational and procedural nature of processing
- Database contains a complex array of pointers that thread through a set of records.
Little scope for automated "query optimization"

Schemas versus Instances



- **Database Schema:** The *description* of a database. Includes descriptions of the database structure and the constraints that should hold on the database.
 - **Schema Diagram:** A diagrammatic display of (some aspects of) a database schema.
 - **Schema Construct:** A component of the schema or an object within the schema, e.g., STUDENT, COURSE.
 - **Database Instance:** The actual data stored in a database at a *particular moment in time*. Also called **database state** (or **occurrence**).
- 

Database Schema Vs. Database State


- **Database State:** Refers to the content of a database at a moment in time.
- **Initial Database State:** Refers to the database when it is loaded
- **Valid State:** A state that satisfies the structure and constraints of the database.
- **Distinction**
 - The **database schema** changes *very infrequently*. The **database state** changes *every time the database is updated*.
 - **Schema** is also called **intension**, whereas **state** is called **extension**.

DBMS Languages

- **Data Definition Language (DDL)**: Used by the DBA and database designers to specify the *conceptual schema* of a database. In many DBMSs, the DDL is also used to define internal and external schemas (views). In some DBMSs, separate **storage definition language (SDL)** and **view definition language (VDL)** are used to define internal and external schemas.

DBMS Languages



- **Data Manipulation Language (DML):** Used to specify database retrievals and updates.
 - DML commands (**data sublanguage**) can be *embedded* in a general-purpose programming language (**host language**), such as COBOL, C or an Assembly Language.
 - Alternatively, *stand-alone* DML commands can be applied directly (**query language**).

DBMS Languages



- **High Level or Non-procedural Languages:** e.g., SQL, are *set-oriented* and specify what data to retrieve than how to retrieve. Also called *declarative* languages.
- **Low Level or Procedural Languages:** record-at-a-time; they specify *how* to retrieve data and include constructs such as looping.