

1. INTRODUCTION

1.1 Data and Database

- **Data** represent facts obtained from observation and/or measurement.
- A **database** is a collection of **interrelated computer data files**, a set of application programs, and a user interface, which are all controlled by a **database management system**.

Example:

P (Professor)

ID	name	...
1	Smith	...
2	James	...
3	Jones	...
...

G (Grad-student)

ID	name	P_ID	...
1	Brown	2	...
2	White	1	...
3	Green	2	...
...

Query:

List information about each professor and the graduate students s/he advises.

- A **database management system** (DBMS) is a collection of programs that carry out many activities including:
 - setting up storage structures,
 - loading data,
 - responding data requests from programs and users, and
 - performing updates.

1.2 Types of Database systems

- Centralized database systems:

All data and processing functions are located at a single site.

- A **personal computer database** normally has a single user.
- A **central computer database** is often very large and is accessed by a large number of users.
- In a **client/server database**, computers are linked together in a communication network. They share printers, files and facilities for database management, etc..

- Distributed databases:

A single logical database spreads physically across networked computers at multiple locations.

- In a **homogeneous database**, the database technology and data at different locations are compatible with each other.
- In a **heterogeneous database** the database technology and data at different locations are not compatible with each other.

1.3 Brief History of Database Systems

- First generation (50's and 60's): Files on tapes or cards.
 - Major activity was to process data that were kept on magnetic tapes or punched paper cards.
 - The file systems used batch sequential processing only.
- Second generation (60's): Files on disks.
 - Batch mode was supplemented with user interactive modes.
 - The file systems allowed multiple direct access to random locations of a disk.
- Third generation (mid 60's and 70's): Early database systems
 - Systems started to separate **logical** and **physical** data structures.
 - Systems started to use **implementation oriented** data models, mainly including the hierarchical model and the network model.

- Applications were first batch oriented, on-line support added later, transaction management (concurrency control, recovery) added, access control facilities provided.
- Fourth generation (80's): Relational systems
 - Systems with simple, solid theoretical model.
 - Systems provided powerful, set-oriented query languages.
 - Database systems stored data without redundancy (or with controlled redundancy).
 - distributed databases emerged.
- Fifth generation (90's and 00's): Post-relational systems
 - Designers are developing new systems that have enhanced functionality, more complex data, and can serve a broader class of applications.
 - * Object-oriented database systems.
 - * Extended relational database systems.
 - * Knowledge-based database systems.
 - * XML-based database systems.

1.4 Data Abstraction

- A database system hides certain details of how data are stored and manipulated and provides users with an abstract view of the data.
- The three levels of schema:

A **schema** is a description of the data contents, structure, and possibly other aspects of a database.

- A **logical level schema** is a specification of the data contents of the entire database, including the types of the data stored and the relationships among the data.
- A **view level schema** is a description of the part of the database seen by an application program or a user.

- A **physical level schema** is a specification of the database physical storage structure and available access paths.

1.5 Data Modeling and Data Models

- **Database modeling** is the *process* to generate an abstract (logical) view of data. — data components, data types, relationships, constraints, etc..
- A **database model** is a collection of *formal languages* for describing data, data relationships, data semantics, and consistency constraints.
- The most commonly used database models:
 - Entity-Relationship model
 - Hierarchical model
 - Network model
 - Relational model
 - Object-oriented model
 - Object-relational model

1.6 Data Independence

- An objective of DBMSs is to achieve **data independence**.
- With data independence, application programs are unaffected by changes in storage structures and access strategy.
- **Physical data independence** is the ability to modify the physical schema without causing application programs to be rewritten.
- **Logical data independence** is the ability to modify the logical schema without causing application programs to be rewritten.

1.7 Data Definition Language (DDL)

- A DDL is used for writing schemas.
- A system may have different DDLs for
 - (1) view level schemas,
 - (2) logical level schemas, and
 - (3) physical level schemas.
- A description in a DDL is compiled to create a database structure.
- A result of compiling a DDL “program” is a set of tables which are stored in a data dictionary.
- A **data dictionary** is a file that contains metadata, that is, “data about data”.

1.8 Data Manipulation Language (DML) or Query Language

- A DBMS provides a DML or a query language which includes facilities for storing, deleting, modifying, and retrieving data from the database.
- There are two types of access to a database:
 - Queries or database commands embedded within application programs,
 - Queries in a “stand alone” query language.
- There are two types of languages:
 - **procedural** (navigational) – the query specifies (to some extent) the strategy (path) used to find the desired data.
 - **declarative** (non-navigational or non-procedural) – the query only specifies **what** data are wanted, not **how** to find them.

1.9 A Course Overview

- The Entity-Relationship model
 - The model and its visual representation
 - Database analysis and design in the E-R model
- The relational model
 - The model and its mathematical fundamentals
 - Translation from the E-R model to relational model
 - Theoretical relational query language: relational algebra and relational calculus
 - Advanced structures and functions in SQL.
- The formal design methodology in the relational model
 - The problems with a poor design
 - The criteria for a good design
 - The normalization techniques for generating a good design
- Transaction management
 - Concurrency control
 - Database recovery
- Query Optimization
- Other models
 - The object database model
 - The object-relational model

Reading: Chapter 1.