Database Concepts

Georgia Tech

Database Concepts

- Data models, schemas, instances
- Three-schema architecture and data independence
- Database languages and interfaces
- Database systems
- DBMS Architectures
- Classification of DBMSes

Data Models

- Abstraction: suppression of details
  - Essential attributes of an entity for a particular application ("selective ignorance")
- Data model: collection of concepts describing a database
  - Structure of database: entities, attributes, data types, relationships
  - Operations on the data: updates and retrievals

Categories of Data Models

- High level conceptual, e.g., (E)ER
  - The end users' conception of their data, understood by end users and database developers
  - A tool for understanding user data in enough detail to derive an implementaion model from it
- Representational (implementation), e.g., Relational
  - Understood by database developers
  - Rigorous, mechanically translatable to physical model
- Low-level physical
  - How data are stored on disk (the code inside a DBMS)

Conceptual Data Model: Entity-Relationship

- Entity: a real world object or concept that will be modeled in the database
- Attribute: a property of interest of some entity
- Relationship: an association between two or more entities

Representational (Implementation) Models

- Most common: relational data model (focus of this class)
- Others:
  - Legacy: network, hierarchical
  - Object data models: never gained widespread adoption
  - Self-describing: XML, JSON (e.g., MongoDB) - a.k.a. NOSQL (Not Only SQL)
- Graph models: major emphasis today, e.g., social networks

Schemas and Databases

- A schema is a description of the data in a database (metadata), typically depicted in a schema diagram
  - Constructs, e.g., STUDENT, COURSE, that specify elemets of the data model
  - Constraints, e.g., STUDENT.GTID must be unique
- Database state is set of instances of entities specified in the schema
- As data loaded into databse, DBMS ensures valid states by ensuring data instances conform to schema and meet constraints
- Sometimes schema called intension, state called extension

Three-Schema Architecure

Three layers of abstraction:

- External level: external schemas, a.k.a. "views"
  - An external schema also representational, but tailored to particular (class of) user(s)
- Conceptual level: conceptual schema
  - Conceptual schema corresponds to representational (implementation) model, not conceptual model
- Internal level: internal schema – physical storage structures

Transformations of data between levels is called mapping; may be computationally expensive
Note: be careful not to confuse categories of data models with levels of abstraction in the three-schema architecture.

Three Schema Diagram

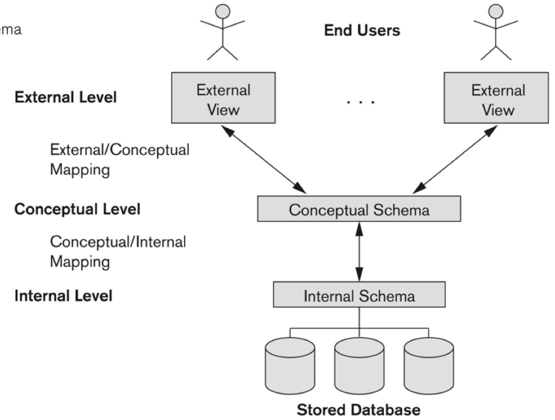


Figure 2.2
The three-schema architecture.

Data Independence

- Goal of Three-Schema Architecure is to separate user applications from physical database. We call this data independence: isolation of changes at one level from levels above
  - Logical data independence: changes to the conceptual schema don't requre changes to external schemas
    - Mappings, e.g., view definitions, may need to change
  - Physical data independence: changes to internal schema don't require changes to conceptual schema

Database Languages

- Data definition language (DDL) specifies conceptual and internal schemas
  - Some systems have a seaprate storage definition language (SDL) to specify internal schemas
- View definition language (VDL) specifies user views (external schema)
- Data manipulation language (DML) used to insert, retrieve, update, and delete data from database

Modern DBMS systems don't have distinct languages.

- SQL combines DDL, VDL, and DML

Database System Architectures

- Centralized
- Client/Server
- Three-tier and n-tier

Centralized Database Architecture

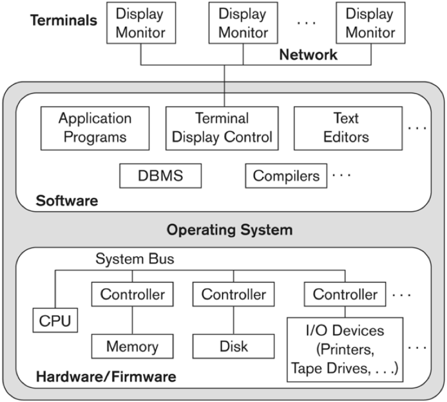
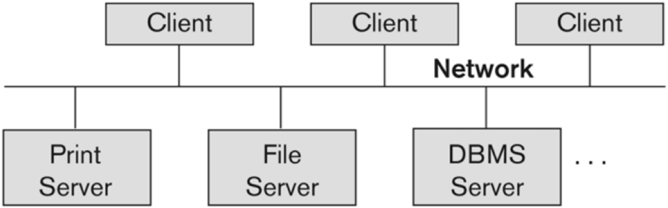


Figure 2.4
A physical centralized architecture.

Client/Server Database Architecture

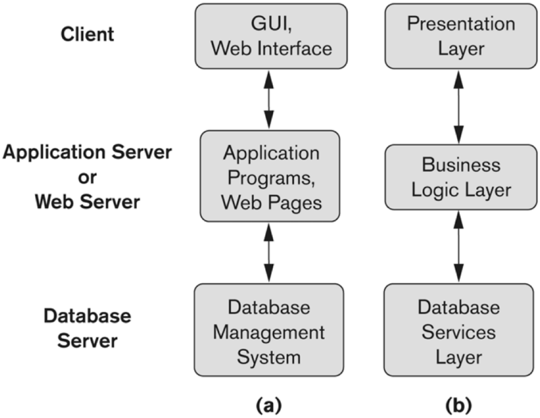
Also known as "two-tier."

Figure 2.5
Logical two-tier client/server architecture.



Three-tier and n-tier Database Architecture

Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.



DBMS Classification Criteria

- Type of data model supported
  - relational, key-value, document-based, graph-based
- Number of users supported – single user vs. multi-user
- Number of sites
  - Centralized vs. distributed
  - Homogeneous, heterogeneous
  - middleware
  - federated multi-database systems
- Cost