Database Systems

Lecture #02

Sang-Wook Kim
Hanyang University

Objectives

- To learn the concepts and architecture of a database system
  - Data models
  - Schemas and instances
  - Data independency
  - Database languages and interfaces

Outline

- Data Models
- Categories of Data Models
- Schemas and Instances
- Three-Schema Architecture
- Data Independency
- DBMS Interfaces
- DBMS Component Modules
- Classification of DBMSs

Data Models

- Collection of concepts that *describe the structure* of a database
  - Data types, relations, constraints, …
- Provide a means to achieve data abstraction
- Classified by the levels of concepts provided
  - Physical models
  - Conceptual models
  - Representational models

Physical Data Models

- Describe the details of how data is organized and stored on computer storage media
  - Close to the way that data stored physically
  - Low-level data model
- Provide record-related information
  - Type
  - Index
  - Access path

Conceptual Data Models

- Describe how data is represented conceptually for human beings
  - Close to the way many users perceive data
  - High-level data model
- Ex. entity-relationship model

Representational Data Models

- Positioned between physical and conceptual data models
  - Easily understood by users
  - Also similar to models that represents how data is organized in computer storage media
  - Also called *implementation data models*
- Used in most commercial DBMSs
- Ex. relational model, network model, hierarchical model

Schemas and Instances

- Database schema (or meta-data)
  - Descriptions of a database structure (via models)
  - Also called the *intension*
  - Defined when designing a database
  - Rarely changed during the database life cycle
- Schema diagram
  - Displays selected aspects of schema via a diagram

Example of a Schema Diagram

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

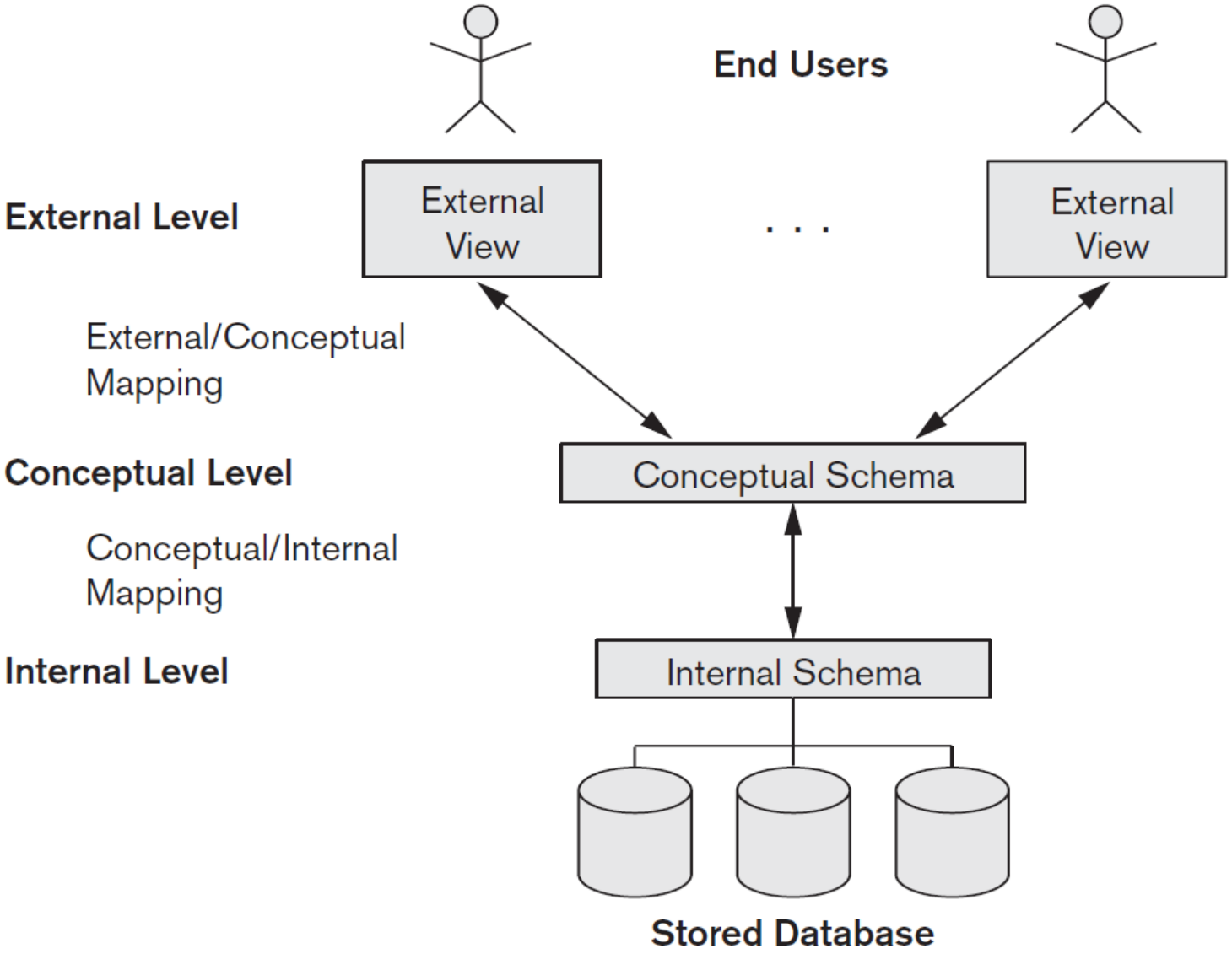
Schemas and Instances

- Database instance
  - Data and its structure in database *at a particular moment* in time
  - Also called *database state, snapshot, occurrence, extension*
  - Often changed by insertions, updates, and deletions of data in a database

Three-Schema Architecture

- Motivation
  - *To prevent an application program from being dependent on its database structure*
  - For independence
- Database designer
  - Selects data to be stored in a database
  - Defines the structure and characteristics of the database

Three-Schema Architecture



Internal Level

- Describes the *physical storage structure* of the whole database
- Represented by a physical data model

Conceptual Level

- Describes the *logical structure* of the whole database for users
  - Entities, data types, relations, user operations, constraints
- Conceals the details of its physical storage structure
- Represented by a higher level data model
  - Conceptual data models
  - Representational data models

External or view level

- Describes a *part of the logical database structure* that a particular user group is interested in
  - Conceals the rest part in the whole database
- Represented by a higher level data model
  - Conceptual data models
  - Representational data models

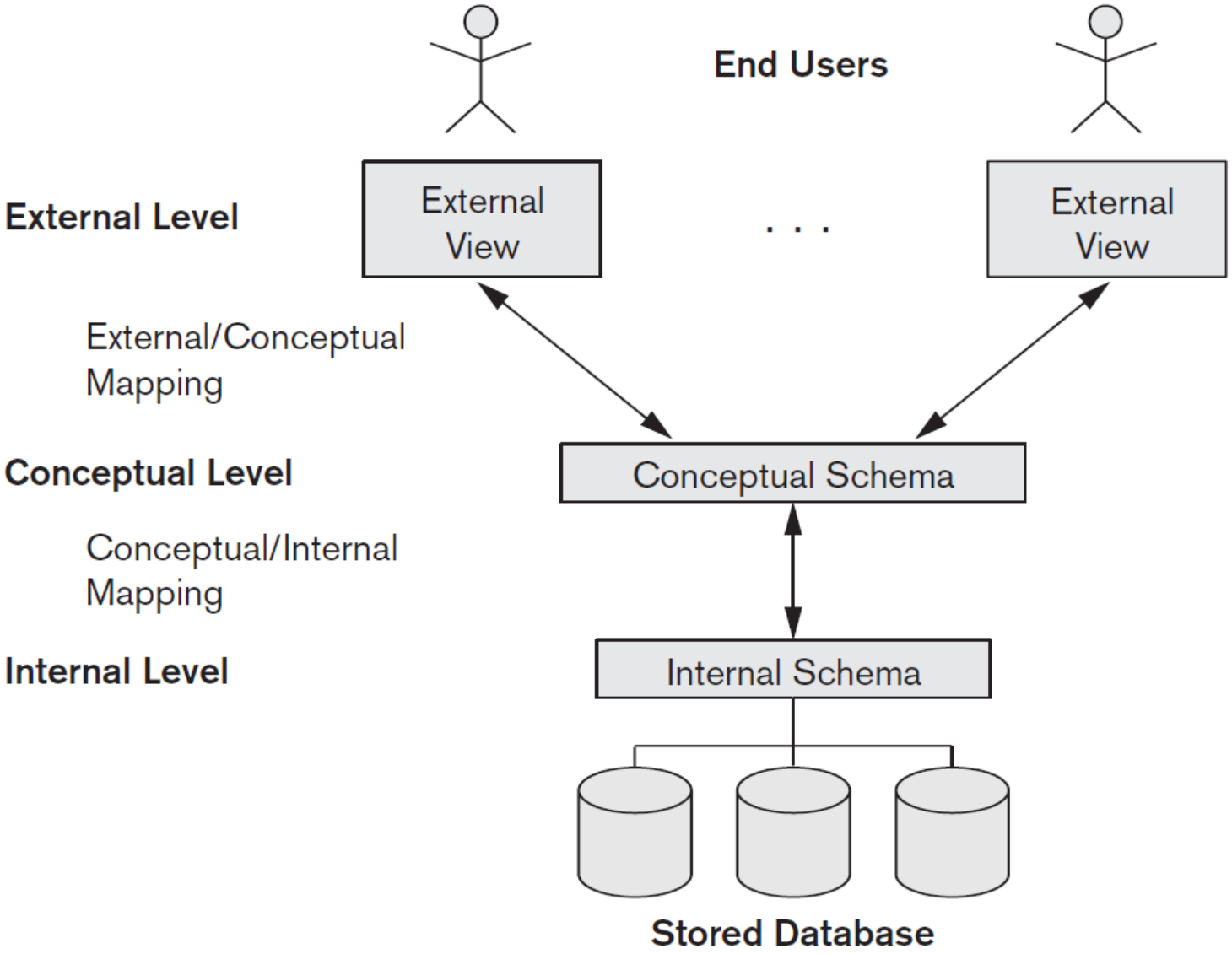
Three-Schema Architecture

- External and internal schema describes the database in a different way
  - A physical database is stored in storage in the way as described in the *internal schema*
  - A user refers to a logical or conceptual database in the way as described in the *external schema*

Mapping

- Mapping mechanism between schema levels
  1. User queries are for *external level schema*
  2. Map *external level* queries into *conceptual level schema*
  3. Map *conceptual level* queries into *internal level schema*
  4. Get the results from the *stored database*
  5. Map the results to *external view*

Three-Schema Architecture



Data Independence

- Definition
  - Able to *change a schema at one level* of a database system
  - Without requiring to change the schema at the *next higher level*
- Types
  - Logical data independence
  - Physical data independence

Logical Data Independence

- When changing the *conceptual schema*
  - No need to change existing *external schema*
  - Result: no need to rewrite existing *application programs* referring to the external schema

Physical Data Independence

- When changing the *internal schema*
  - No need to change existing *conceptual schema*
  - No need to change existing *external schema*
  - No need to rewrite existing *application programs* referring to the external schema

Data Independence

- One of the most important features of DBMSs
- Significantly reduces the maintenance overhead of existing application programs

DBMS Languages

- Provided by DBMSs
- Languages for controlling/managing databases
- Data definition language (DDL)
  - Defines external/conceptual/internal schemas for a target database
- Data manipulation language (DML)
  - Allows users to retrieve, insert, delete, and modify data

DBMS Languages

- Storage definition language (SDL)
  - Specifies the internal schema for a target database
- View definition language (VDL)
  - Specifies user views/mappings to conceptual schema for a target database

DBMS Languages

- Procedural DML
  - Define user queries *procedurally*
  - Focus on *how to retrieve data*
    - Specify the detailed steps
- Non-procedural DML
  - Define user queries *declaratively*
  - Focus on *what (which data) to retrieve*

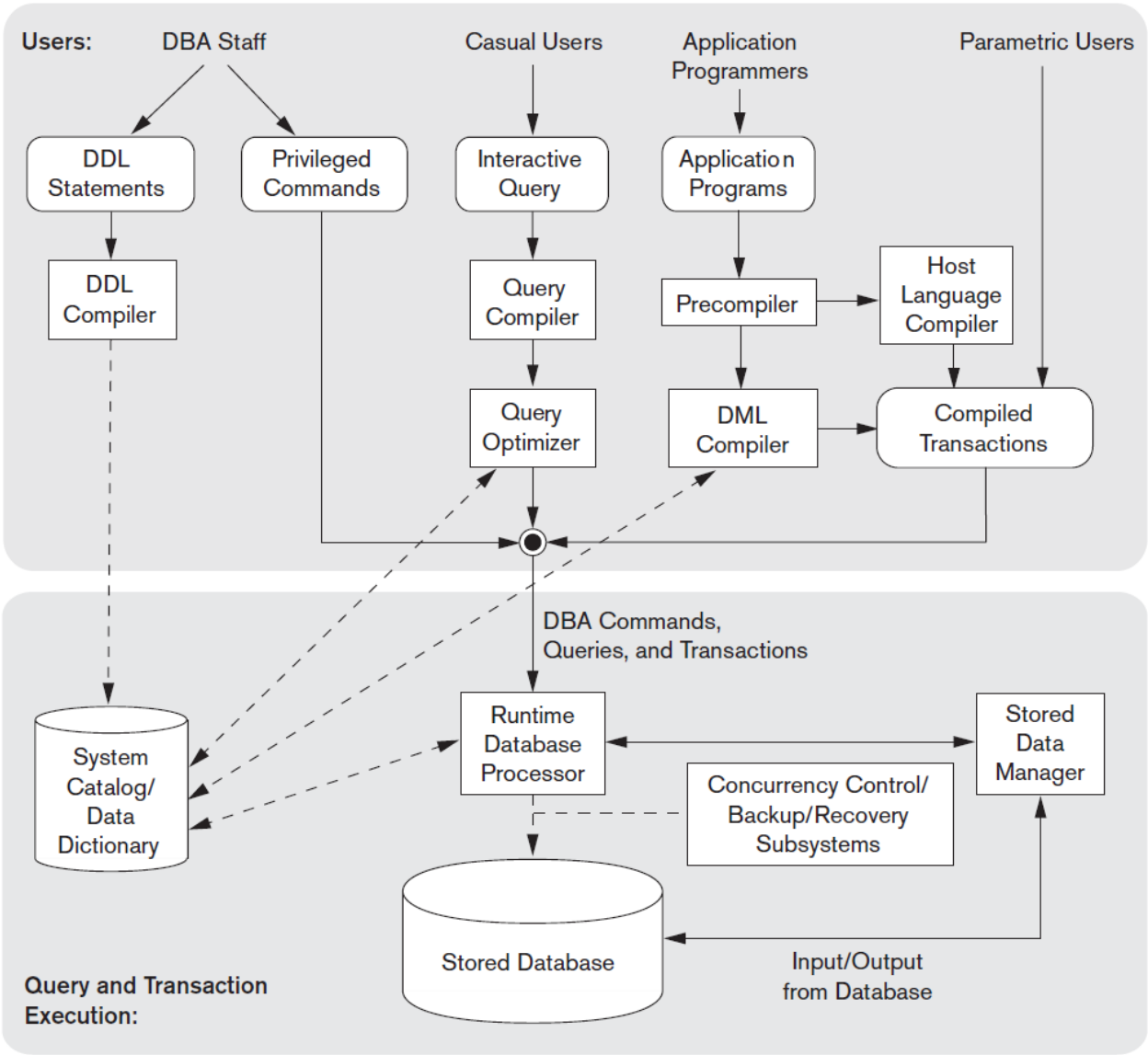
DBMS Interfaces

- Menu-based interfaces
  - Access a database by selecting queries from the list of items in menu
  - Usually for web applications
- Graphical user interfaces
  - Access a database by manipulating graphical representation of database schemas
  - Applications for naïve users

DBMS Interfaces

- Forms-based interfaces
  - Access a database in a way of filling up a predefined form
- Natural language interfaces
  - Access a database with an interface of a natural language style
- Interfaces for DBAs
  - Provide for DBMS management environment

DBMS Component Modules



DBMS Component Modules

- Stored data manager
  - Controls accesses to all the data stored on storage
    - User data
    - Meta data in a system catalog
- DDL compiler
  - Processes schema definitions in DDL
  - Stores schema-related meta data in the DBMS system catalog

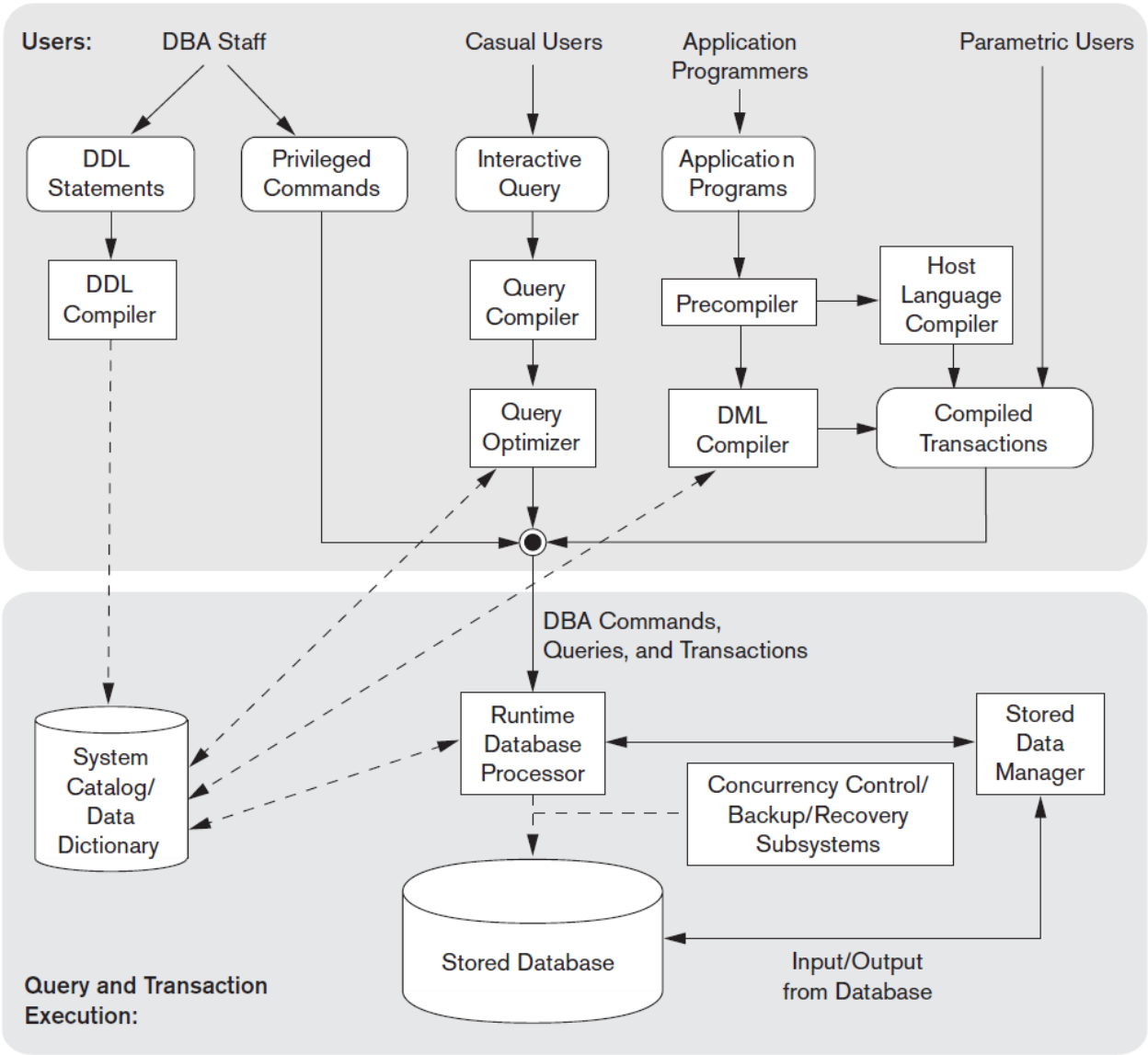
DBMS Component Modules

- Runtime database processor
  - Accesses a database to process user queries
- Query compiler (interactive SQL)
  - Parses high-level DBMS queries
    - Compiles into an internal form
  - Make calls on the runtime database processor

DBMS Component Modules

- Precompiler
  - Extracts DML commands from an application program written in a host programming language
- DML compiler
  - Compiles DML commands into object code for database accesses

DBMS Component Modules



Database System Utilities

- Database utilities
  - Help DBA manage easily database systems
- Loading
  - Loads existing data files outside of DBMS into a database
    - Automatically converts the files into DBMS database format
  - Useful for building a new database

Database System Utilities

- Backup
  - Creates a backup copy of a database
  - Backup copies can be used to restore a database after a variety of failures
- Database storage reorganization
  - Reorganizes database files into different file organizations
  - Helps to improve the DBMS performance

Database System Utilities

- Performance monitoring
  - Monitors a database usage and provides statistics to the DBA
  - Helps make decisions such as whether or not to reorganize files in a different format

Database System Utilities

- Data dictionary system
  - Stores additional information
    - Design decisions
    - Usage standards
    - Application program descriptions
    - User information
  - Similar to the DBMS system catalog

Classification of DBMS

- By data models supported by DBMS
  - Relational model
    - A database is represented as a collection of *tables*
    - A table contains a set of *records*
    - Most popular these days
  - Network model
    - Represents data as a *record type* and a *1:N set type*

Classification of DBMS

- By data models (cont'd)
  - Hierarchical model
    - Represents data as a *hierarchical tree structure*
  - Object model
    - A database is represented as a collection of *classes*
    - Data is represented as *objects* belonging to a *class*
    - Provides rich modeling features
      - Inheritance, complex objects, …

Classification of DBMS

- By number of users
  - Single user DBMS
  - Multi-user DBMS

Classification of DBMS

- By number of sites
  - Centralized DBMS
  - Distributed DBMS
    - Homogeneous DBMS
    - Heterogeneous DBMS

Classification of DBMS

- By purpose
  - General purpose DBMS
  - Special purpose DBMS

Summary

- Concepts used in database systems
- Main categories of data models
  - Physical data models
  - Conceptual data models
  - Representational data models

Summary

- Types of languages supported by DMBSs
- Interfaces provided by the DBMS
- DBMS classification criteria:
  - Data model, number of users, number of sites, purpose

References

1. E. F. Codd, "A relational model for large shared data banks," *Comm. ACM* **13**:6, pp. 377-387, 1970.
2. J.-M. Nicolas, "Logic for improving integrity checking in relational databases," *Acta Informatica* **18**:3, pp. 227-253, 1982.
3. Y. Papakonstantinou, H. Garcia-Molina, and J. Widom, "Object exchange across heterogeneous information sources," *IEEE Intl. Conf. on Data Engineering*, pp. 251-260, March 1995.
4. World-Wide-Web Consortium, http://www.w3.org/XML/

Have a nice day!