CMSC 127

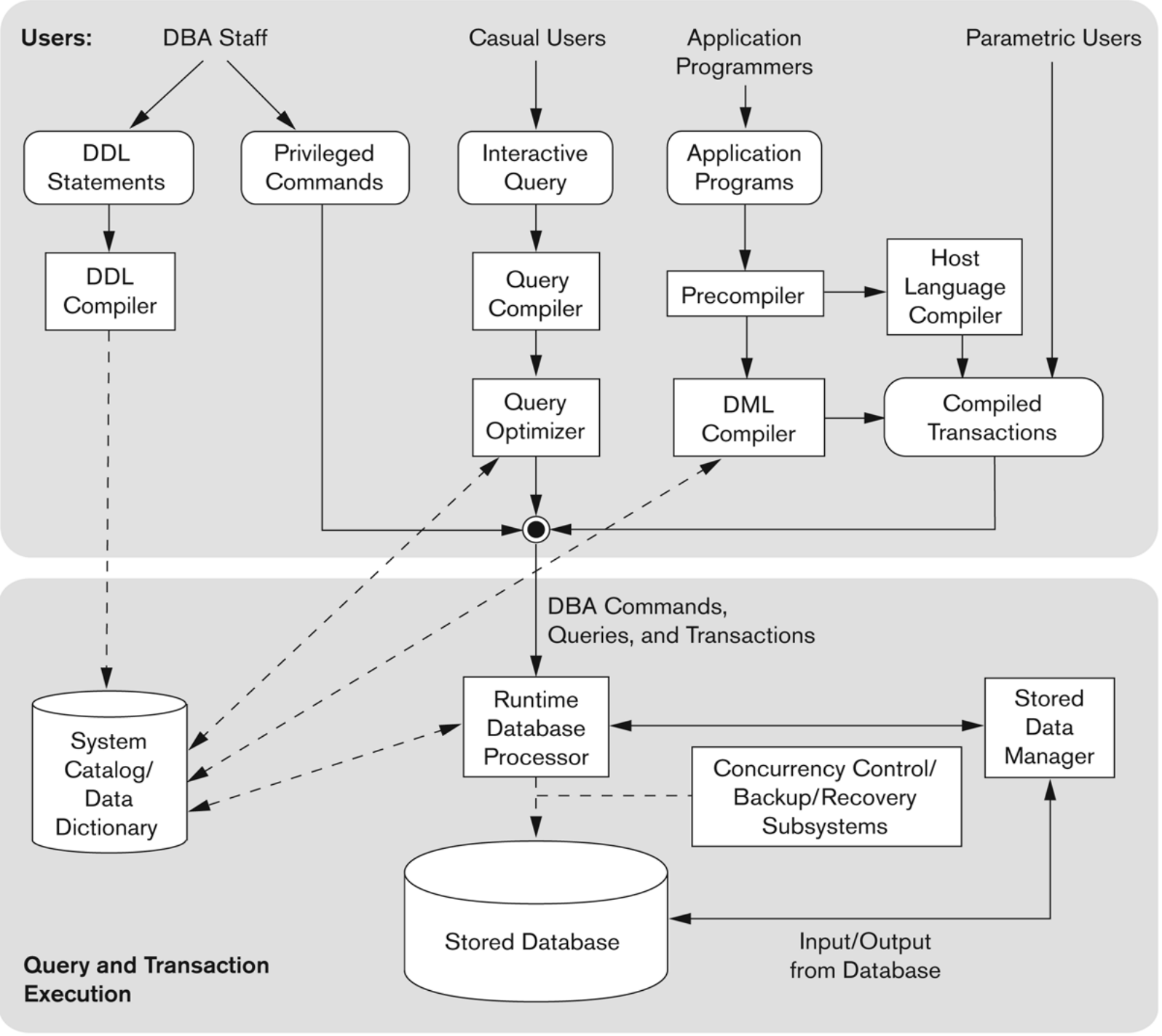
Database System Concepts and Architecture

Reginald Neil C. Recario
Institute of Computer Science
University of the Philippines Los Baños

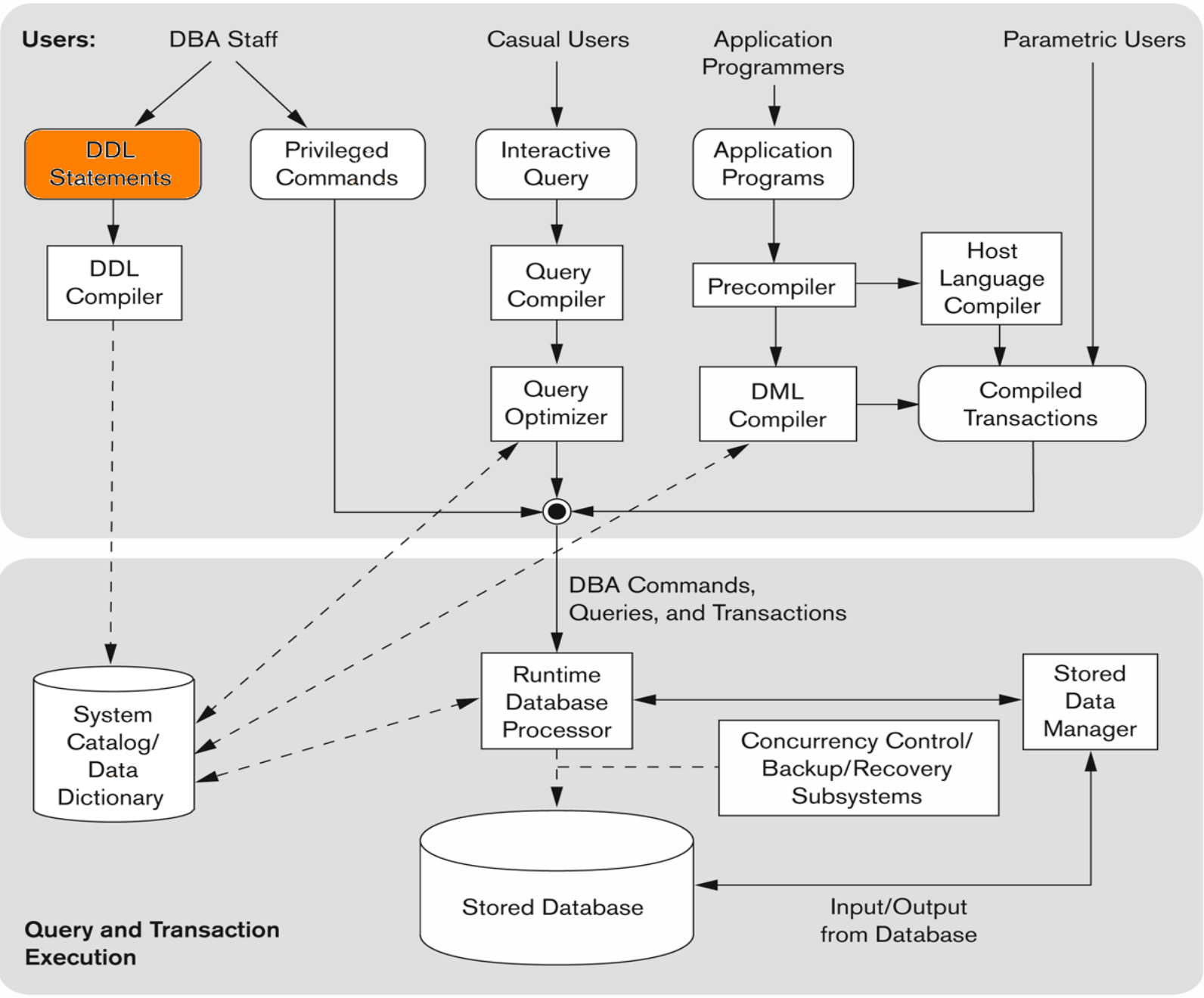
Overview

- Typical DBMS Component Modules
- DBMS Architectures
- Data Models

Typical DBMS Component Modules



Typical DBMS Component Modules



DBMS Languages

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)

DBMS Languages

- ***Data Definition Language (DDL):***
  - Used by the DBA and database designers to specify the description of the database.
  - Example:

```
CREATE TABLE student(
name VARCHAR2(50),
student_no NUMBER(9),
classification VARCHAR2(9)
);
```

DBMS Languages

- ***Data Manipulation Language (DML):***
  - Used to specify database retrievals and updates (insertion, deletion and modification of data)
  - Examples:

    SELECT * from student;

    INSERT INTO student values ('Mark Santos',200712345,'JUNIOR');

    UPDATE student set classification = 'SENIOR' where name = 'Mark Santos';

DBMS Interfaces

- ***Stand-alone query language interfaces***
  - Example: Entering SQL queries at the DBMS interactive SQL interface (e.g. SQL*Plus in ORACLE)
- ***Programmer interfaces for embedding DML in programming languages***
- ***User-friendly interfaces***
  - Menu-based, forms-based, graphics-based, etc.

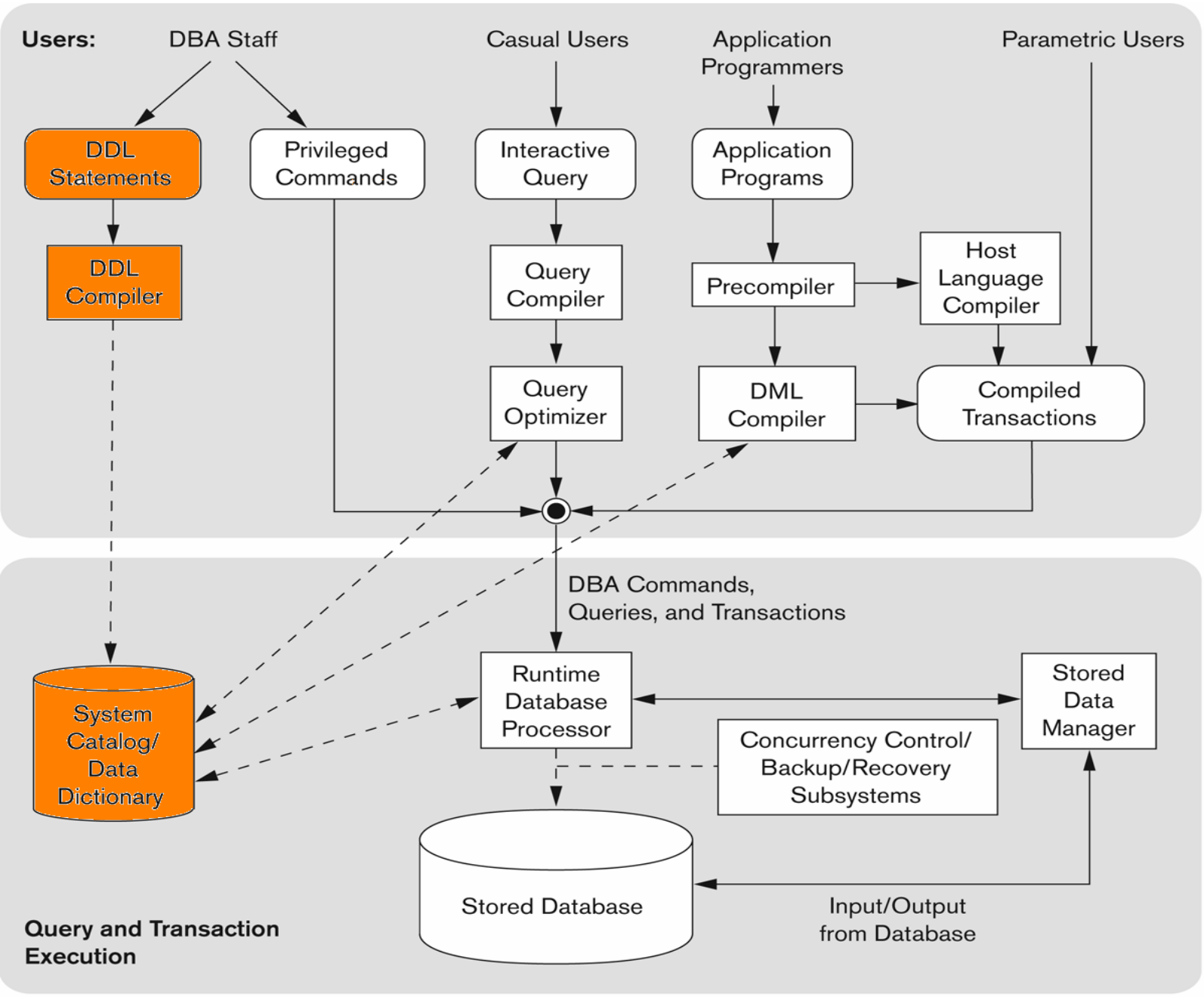
User-Friendly DBMS Interfaces

- ***Menu-based, popular for browsing on the web***
- ***Forms-based, designed for naïve users***
- ***Graphics-based***
  - (Point and Click, Drag and Drop, etc.)
- ***Natural language: requests in written English***
- ***Combinations of the above:***
  - For example, both menus and forms used extensively in Web database interfaces

Other DBMS Interfaces

- ***Speech as Input and Output***
- ***Parametric interfaces, e.g., bank tellers using function keys.***
- ***Interface for the DBA***

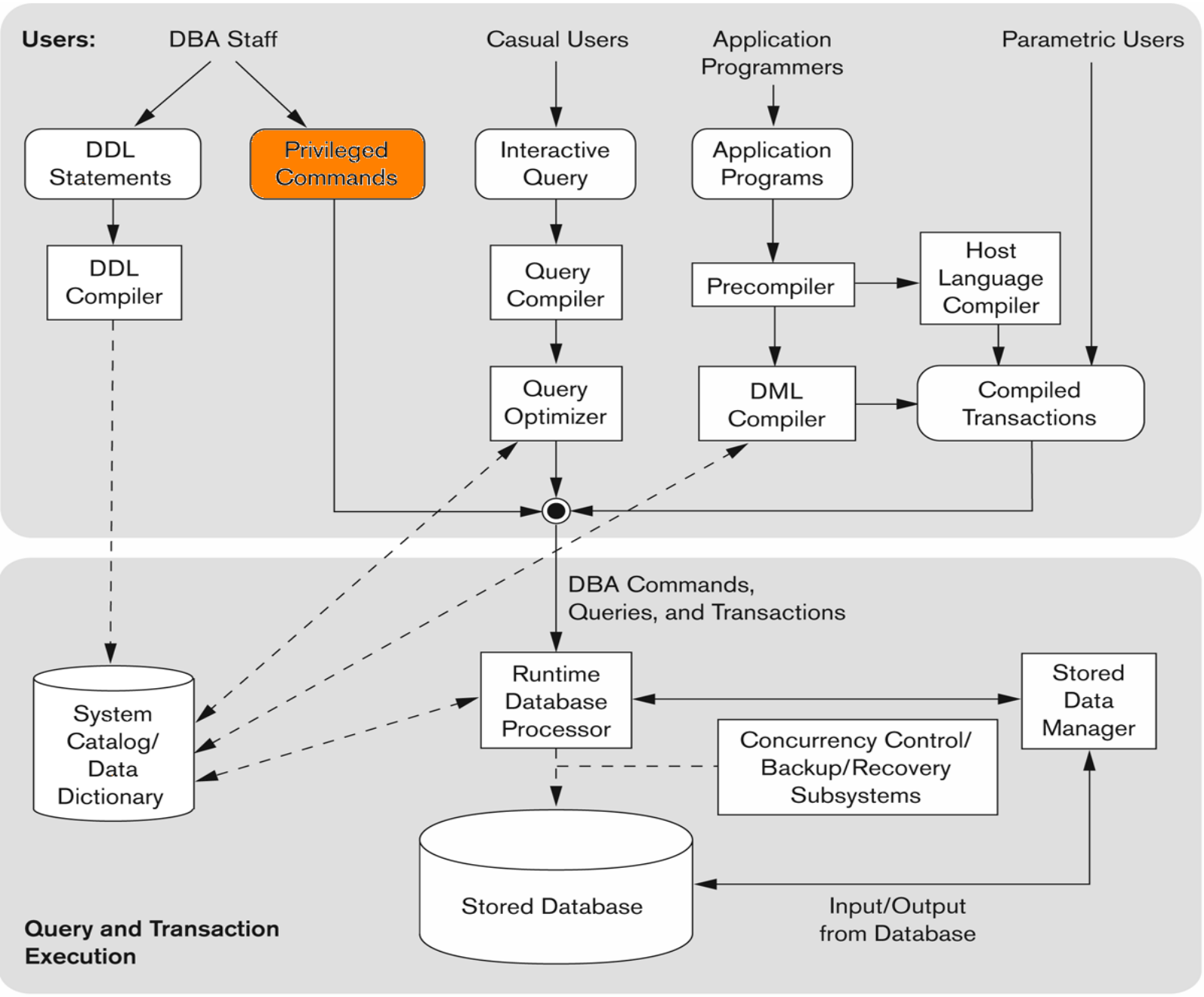
Typical DBMS Component Modules



DBMS Component Modules

- ***DDL Compiler***
  - Process Data Definition Language (DDL) statements and stores database description in the DBMS catalog

Typical DBMS Component Modules



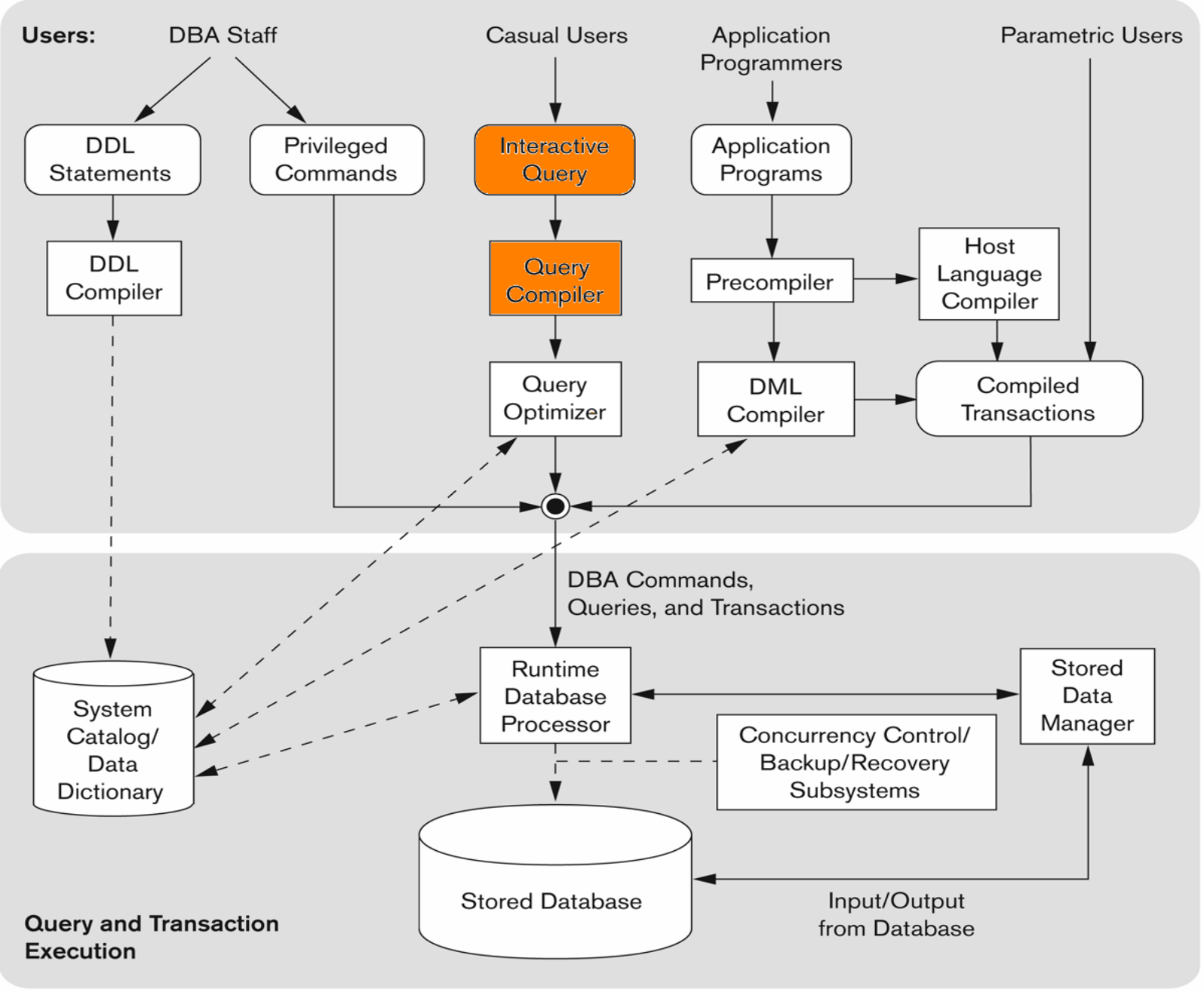
DBMS Component Modules

- ***Privileged Commands***
  - Commands used only by the DBA or the DBA staff
  - These includes commands for creating accounts, granting account authorization, changing database description, etc.

DBMS Component Modules

- ***Privileged Commands***
  - Commands used only by the DBA or the DBA staff
  - These includes commands for creating accounts, granting account authorization, changing database description, etc.

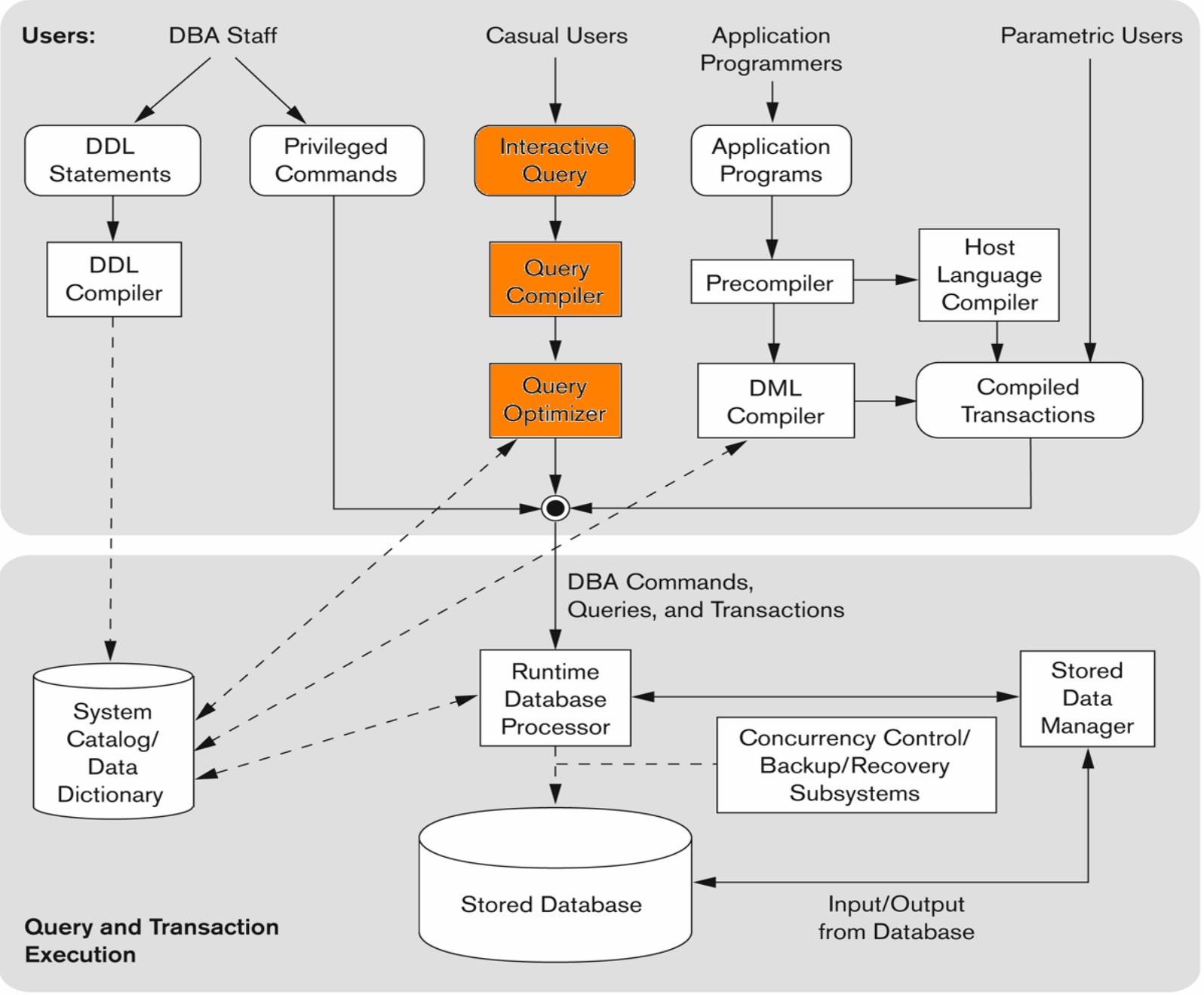
Typical DBMS Component Modules



DBMS Component Modules

- ***Query Compiler***
  - Parses queries, analyzes queries for correctness of operations and names of data elements, and compiles them into an internal form

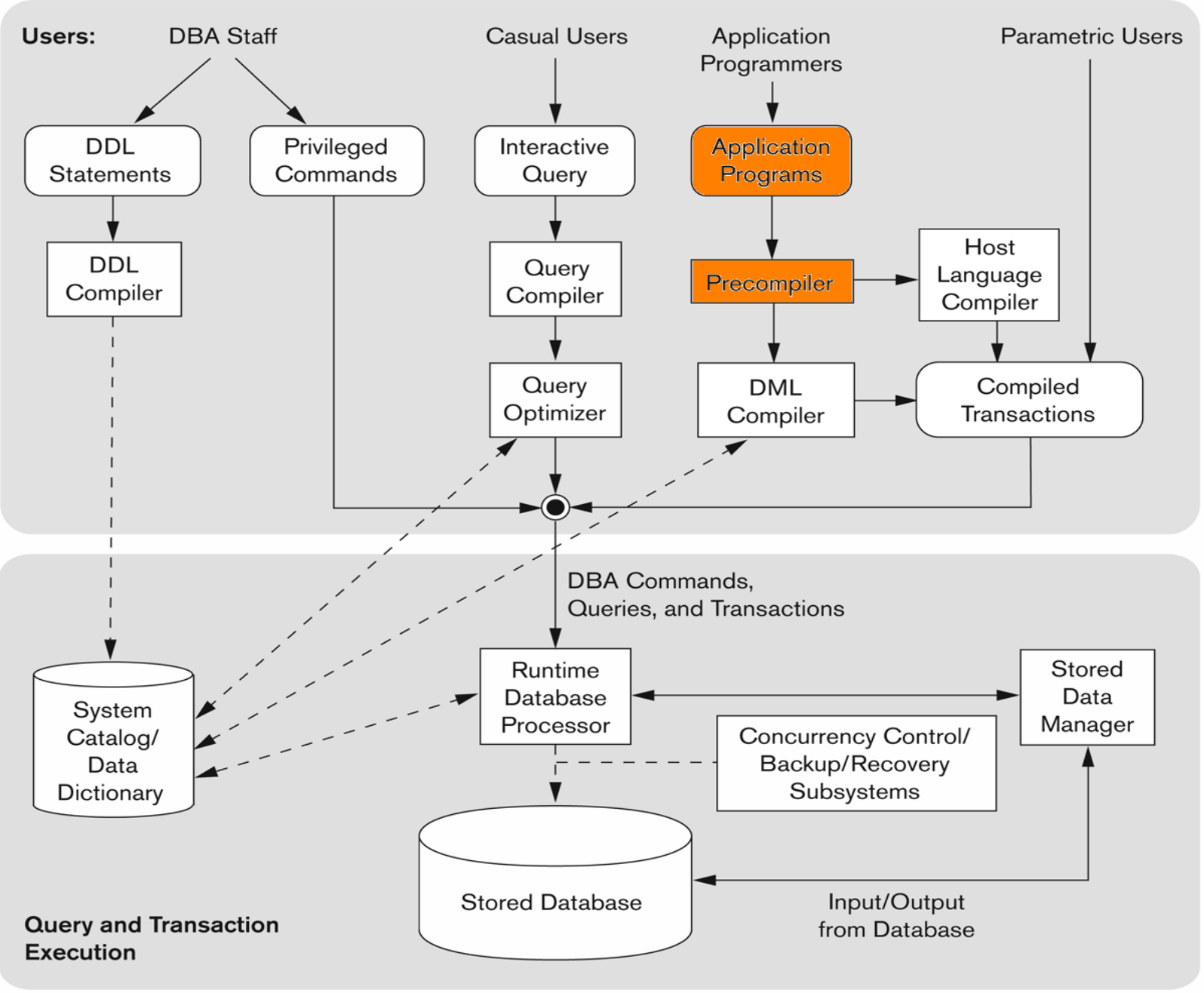
Typical DBMS Component Modules



DBMS Component Modules

- ***Query Optimizer***
  - Concerned with the rearrangement and possible reordering of operations, elimination of redundancies and use of correct algorithms during execution.
  - Generates executable code and makes calls on the runtime processor

Typical DBMS Component Modules



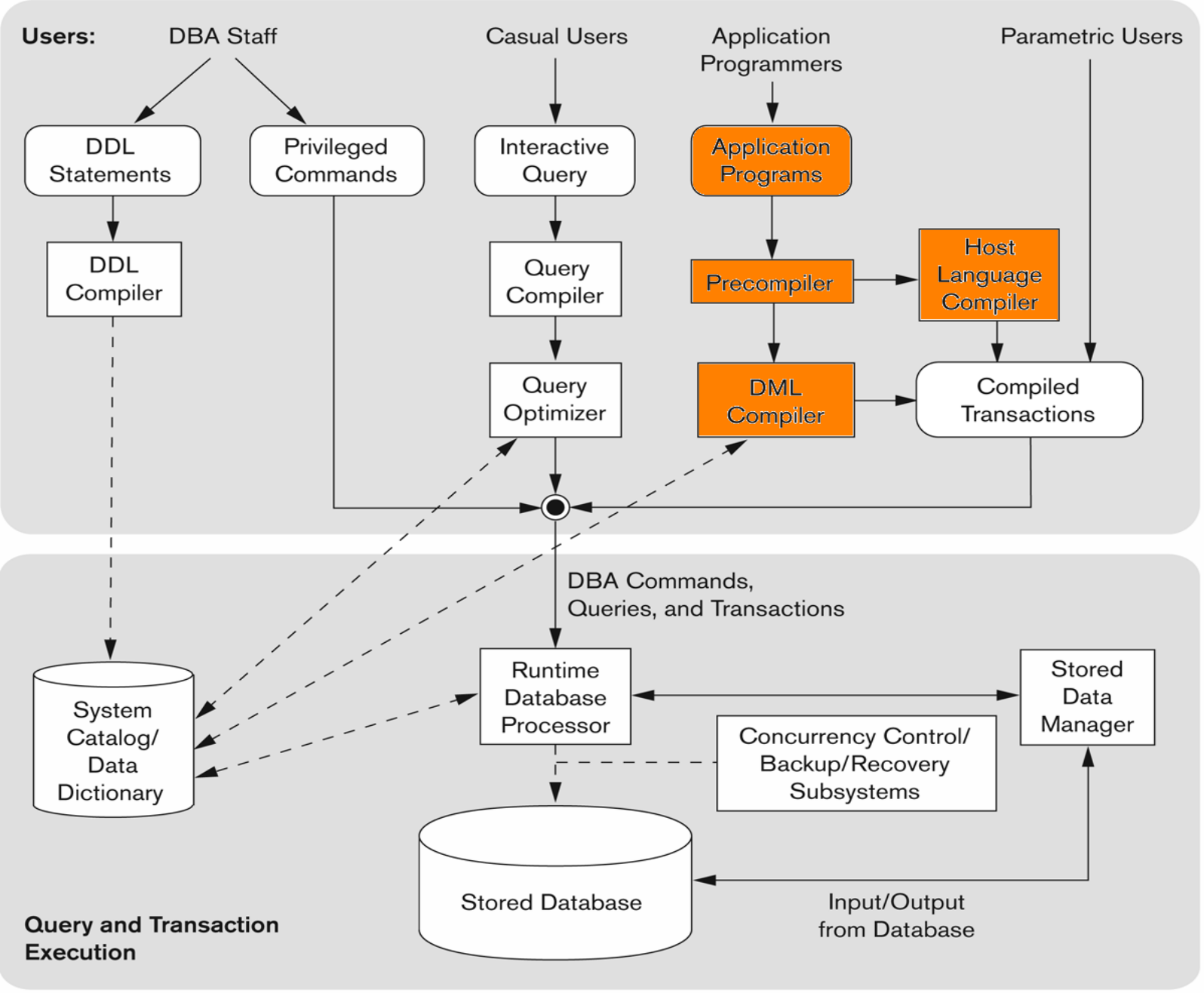
DBMS Component Modules

- ***Precompiler***
  - Extracts DML commands from an application program

Sample Embedded SQL in C

```
loop = 1;
while (loop) {
   prompt ("Enter SSN: ", ssn);
   EXEC SQL
   select FNAME, LNAME, ADDRESS, SALARY
   into :fname, :lname, :address, :salary
   from EMPLOYEE where SSN == :ssn;
   if (SQLCODE == 0) printf(fname, …);
   else printf("SSN does not exist: ", ssn);
   prompt("More SSN? (1=yes, 0=no): ", loop);
   END-EXEC
}
```

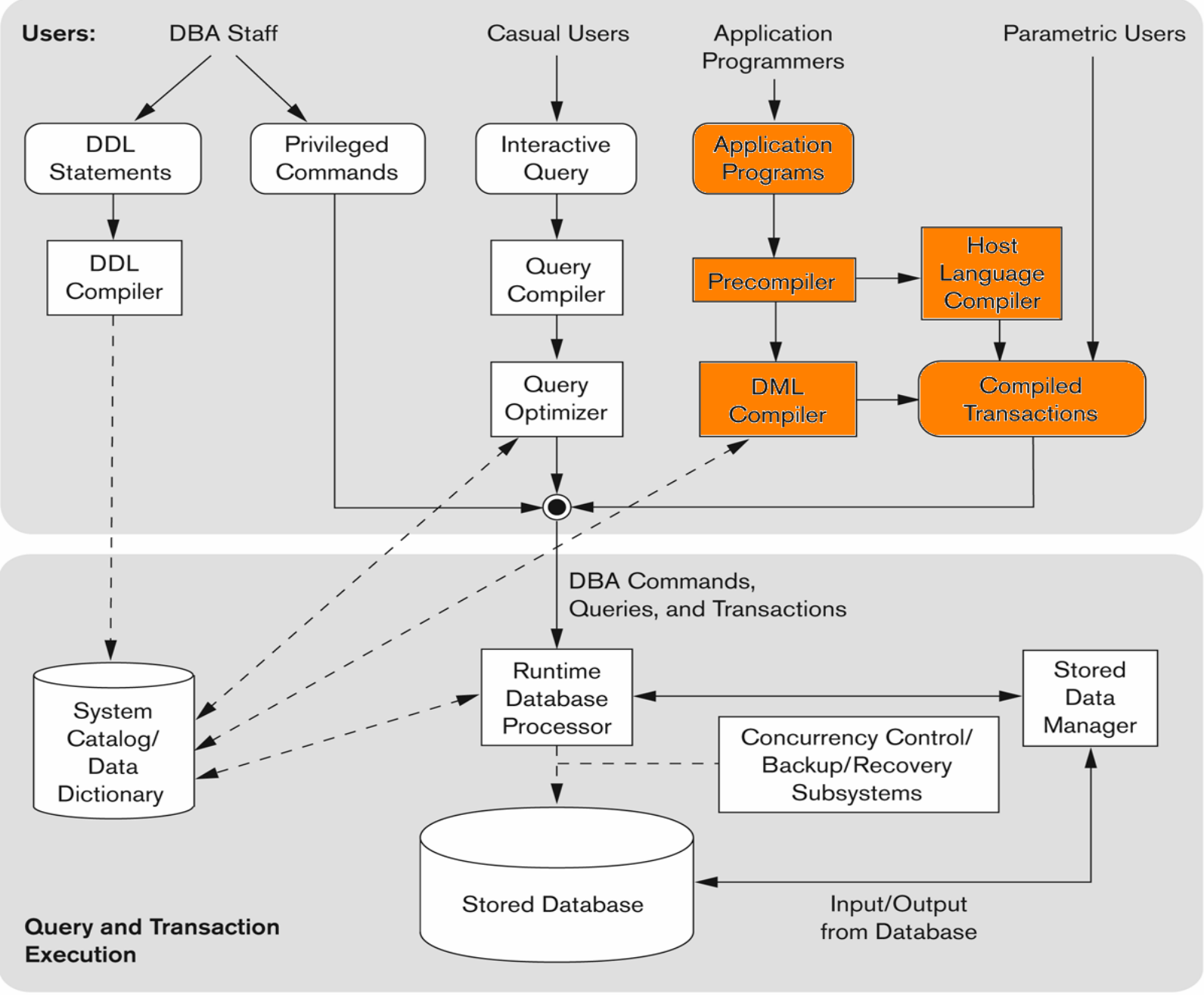
Typical DBMS Component Modules



DBMS Component Modules

- ***DML Compiler***
  - Parses DML statements and translates them into an object code
- ***Host Language Compiler***
  - Parses and compiles host language statements

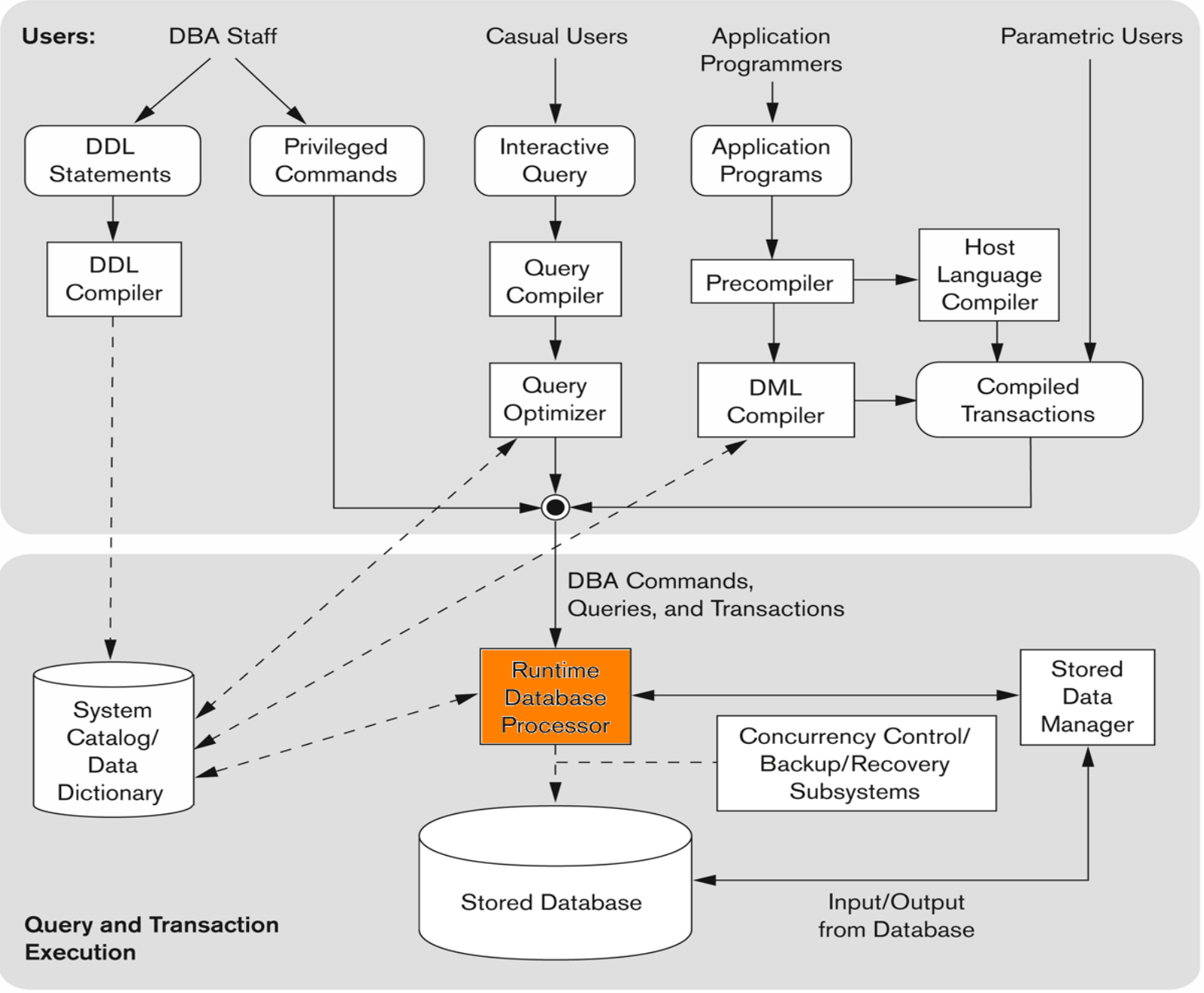
Typical DBMS Component Modules



DBMS Component Modules

- ***Compiled Transaction***
  - Canned transactions that is based on the object codes produced by the DML and host language compiler

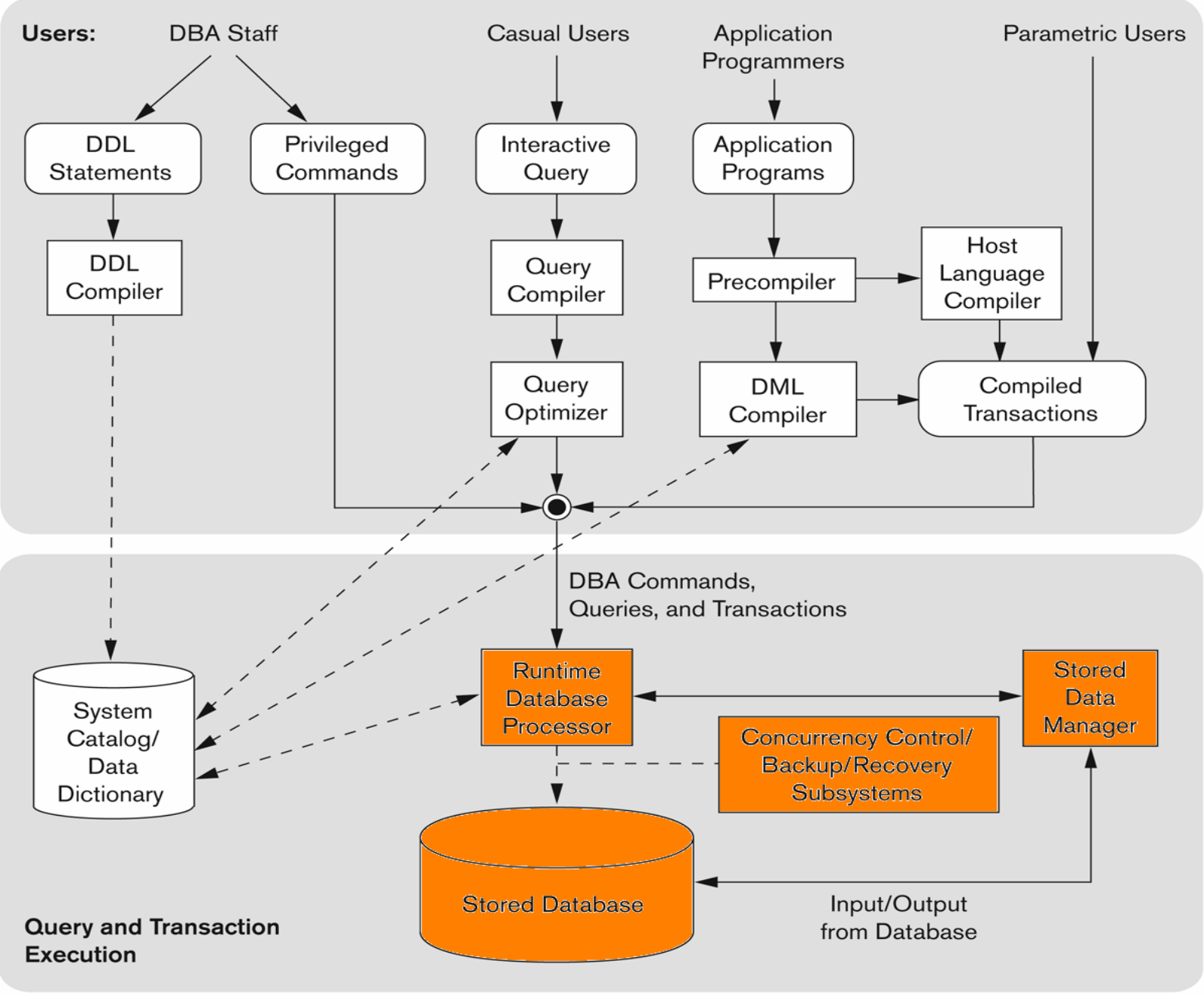
Typical DBMS Component Modules



DBMS Component Modules

- ***Runtime Database Processor***
  - Executes privileged commands, executable query plans and canned transactions with runtime parameters

Typical DBMS Component Modules



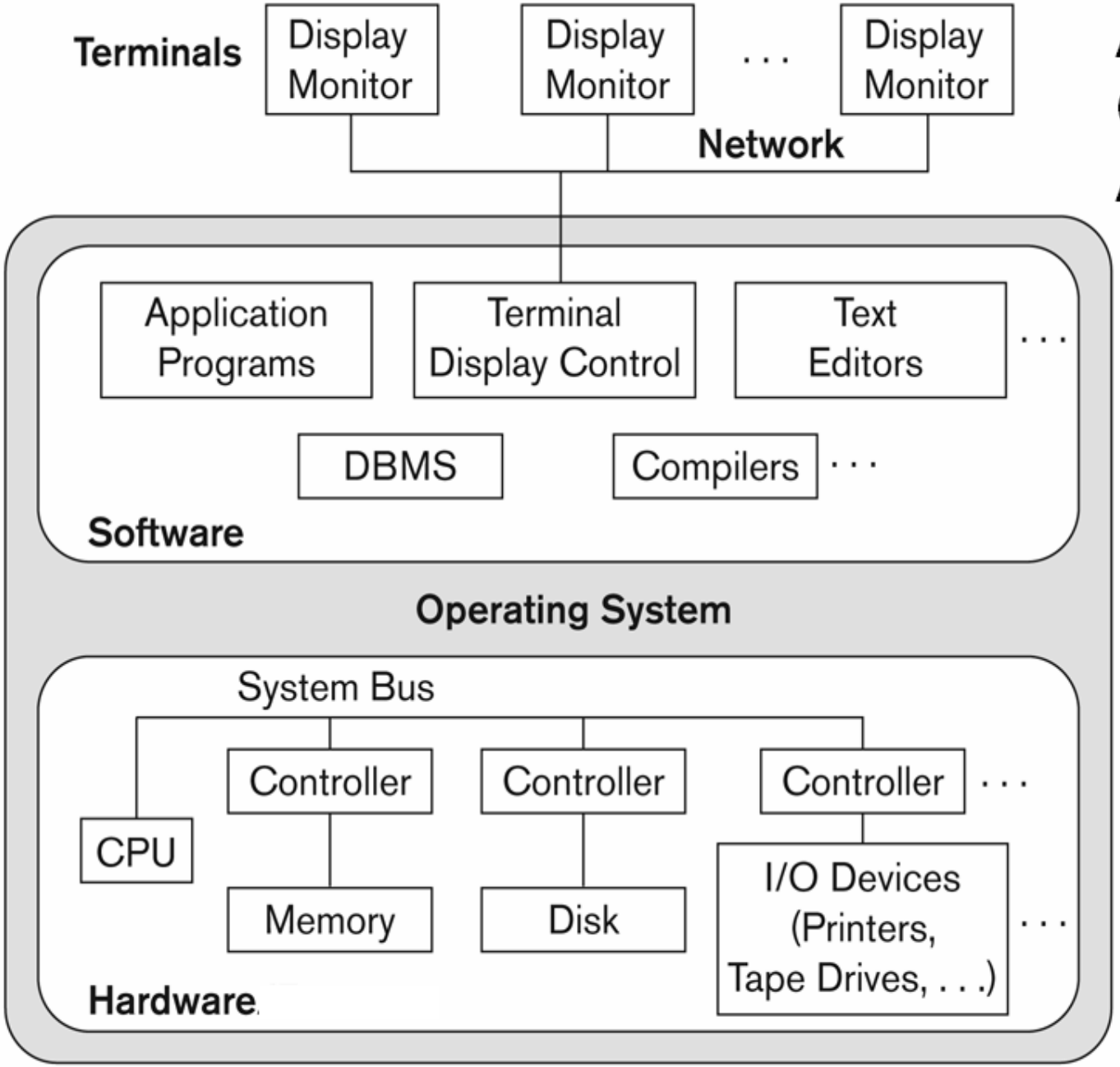
DBMS Component Modules

- ***Stored Data Manager***
  - Controls access to the information stored on the disk
  - Interacts with the Operating System to carry put low-level input/output operations between the disk and main memory

Centralized and Client-Server DBMS Architectures

- ***Centralized DBMS:***
  - Combines everything into single system including- DBMS software, hardware, application programs, and user interface processing software.
  - User can still connect through a remote terminal – however, all processing is done at centralized site.

A Physical Centralized Architecture



Client

- A user machine that provides user interface capabilities and local processing
- May be diskless machines or PCs or Workstations with disks with only the client software installed.
- Connected to the servers via some form of a network.

Server

- A system containing both hardware and software that can provide services to the client machines such as file access, printing, archiving or database access

Client

- A user machine that provides user interface capabilities and local processing
- May be diskless machines or PCs or Workstations with disks with only the client software installed.
- Connected to the servers via some form of a network.

Client

- ***Thin client***
  - Dependent on another machine (e.g., server) to complete a process
- ***Thick client***
  - Can provide functionality independent of the server

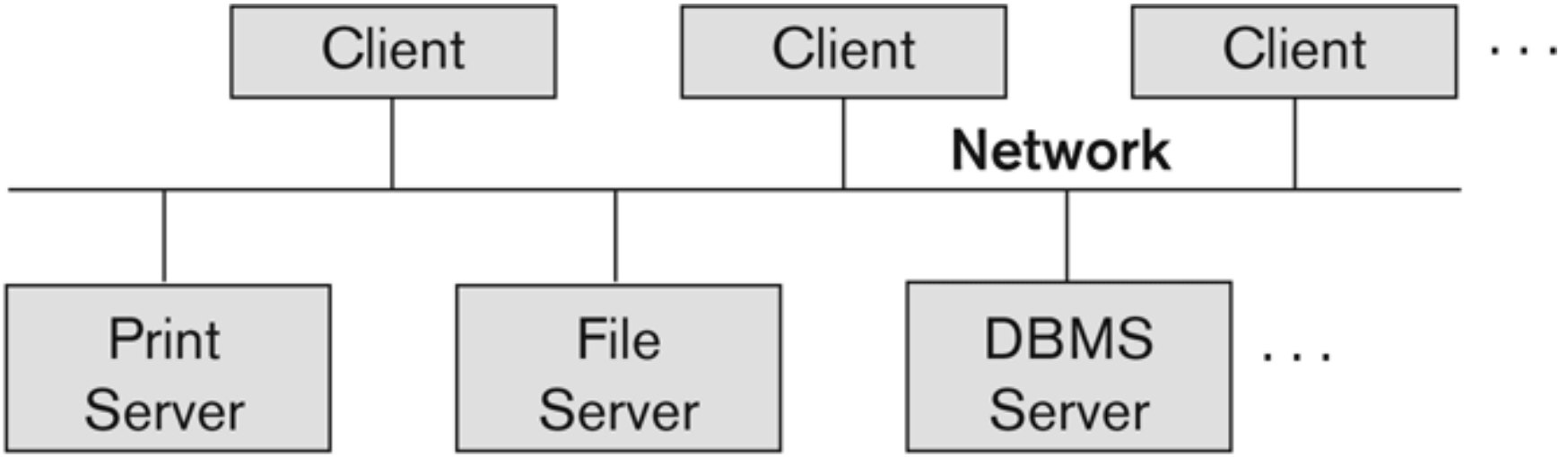
Server

- A system containing both hardware and software that can provide services to the client machines such as file access, printing, archiving or database access

Basic Client-Server Architecture

- Specialized Servers with Specialized functions
  - Print server
  - File server
  - Web server
  - Email server
  - DBMS server
- Clients can access the specialized servers as needed

Client server architecture



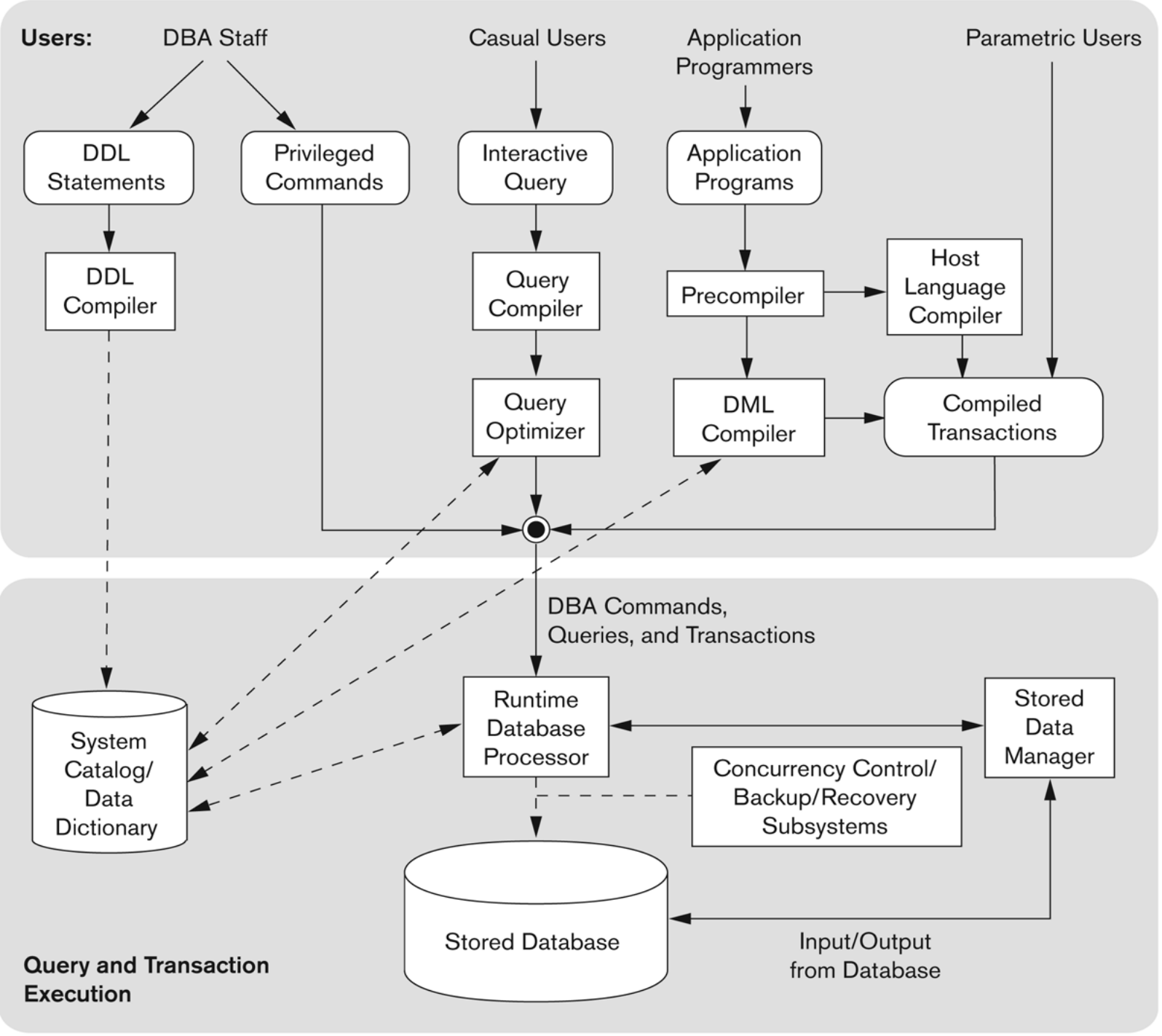
Two Tier Client-Server Architecture

- ***1st Approach***
  - Client contains the user interface and application programs
  - Server contains the DBMS

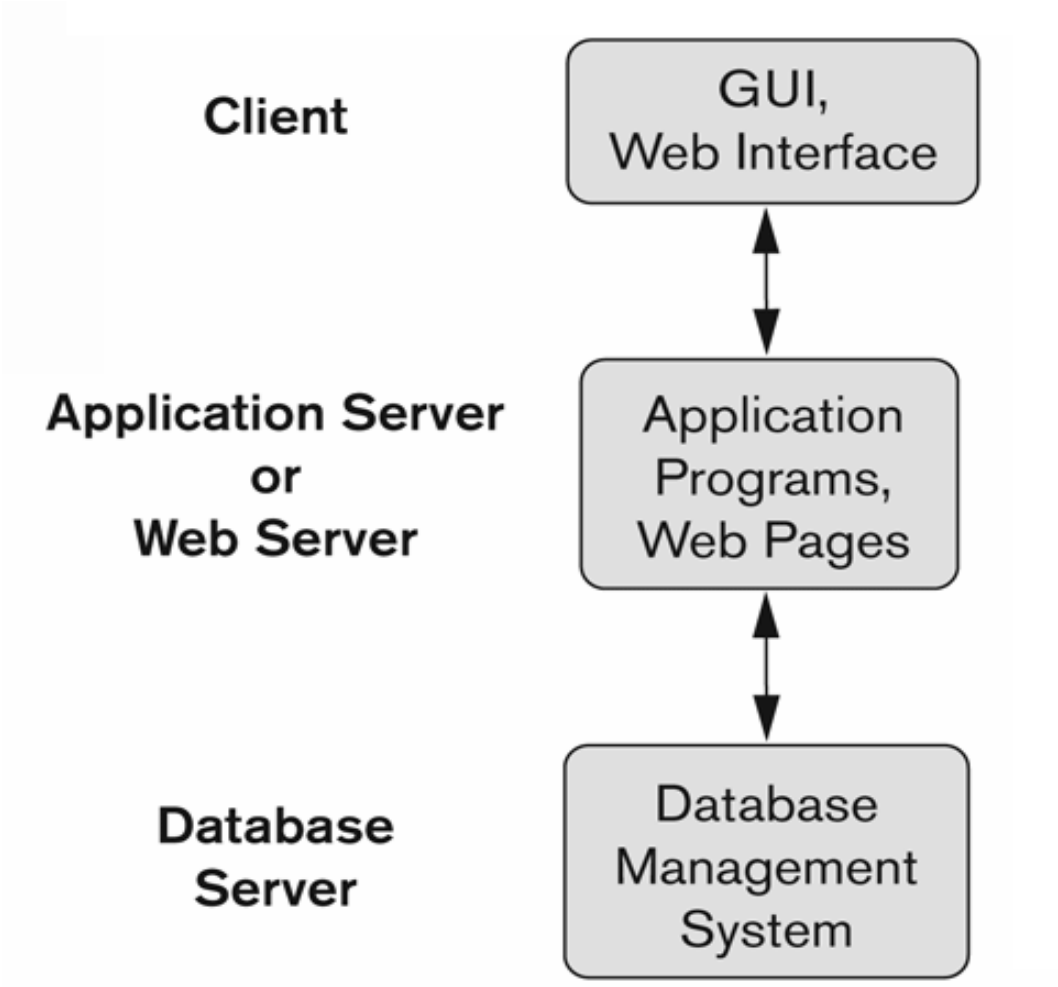
Two Tier Client-Server Architecture

- ***2nd Approach***
  - The DBMS was divided into client and server
  - Client may handle user interface, application programs, data dictionary functions, DBMS interactions with the compilers, query optimizer, etc.
  - Server may include softwares responsible for handling data storage on disks, buffering of disk pages, etc. (also known as data server)

Typical DBMS Component Modules



Three-tier client-server architecture



Two Tier Client-Server Architecture

- The client communicates with the application server usually through a form's interface.
- The application server stores the business rules (procedures) that are used to access data from the database.
- The database server will provide the data.

Two Tier Client-Server Architecture

- Three-tier Architecture Can Enhance Security:
  - Database server is only accessible via middle tier
  - Clients cannot directly access database server

Schemas versus States

- ***Database Schema***:
  - The description of a database.
  - Includes descriptions of the database structure, data types, and the constraints on the database.
- ***Schema Diagram***:
  - An illustrative display of (most aspects of) a database schema.
- ***Schema Construct:***
  - A component of the schema or an object within the schema, e.g., STUDENT, COURSE.

Database Schema

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 versus States

- ***Database State*:**
  - The actual data stored in a database at a *particular moment in time*. This includes the collection of all the data in the database.
  - Also called ***database instance*** (or ***occurrence*** or ***snapshot***).

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

Database State

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Database State

- ***Empty State*:**
  - Refers to the database state after defining a new database.
- ***Initial State*:**
  - Refers to the database state when the database is first populated or loaded with initial data.
- ***Valid State (Current State)*:**
  - A state that satisfies the structure and constraints of the database at any point in time.

Database Schema vs. Database State

- ***Distinction***
  - The ***database schema*** changes very infrequently.
  - The ***database state*** changes every time the database is updated.

- ***Schema*** is also called ***intension***.
- ***State*** is also called ***extension***.

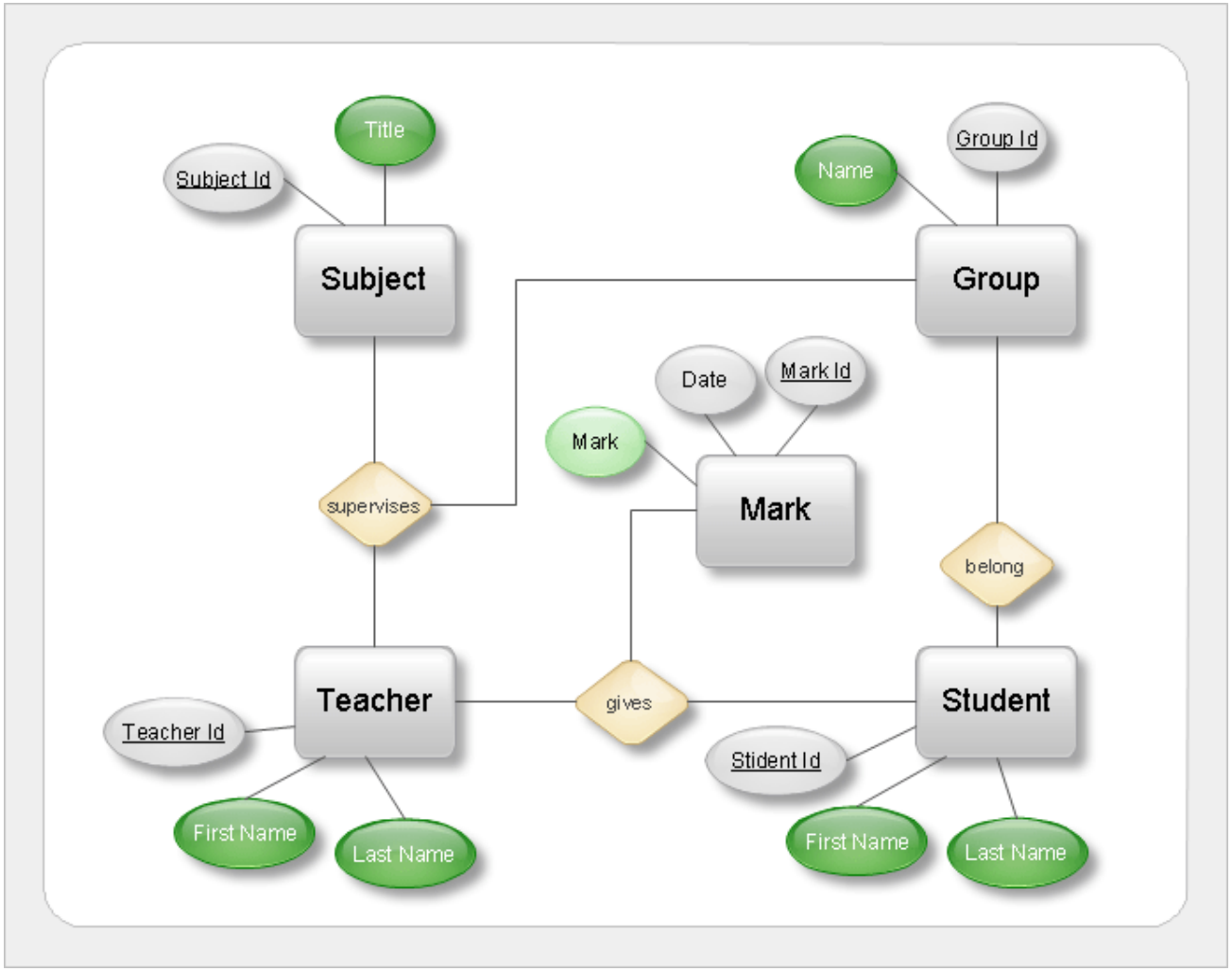
Data Models

- ***Data Model***:
  - A collection of concepts used to describe the structure of a database and the operations for manipulating these structures
- ***Database Structure and Constraints***:
  - Structure of a database includes datatypes, relationships, and constraints
  - Constraints specify some restrictions on valid data; these constraints must be enforced at all times

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.)

Reference(s):2

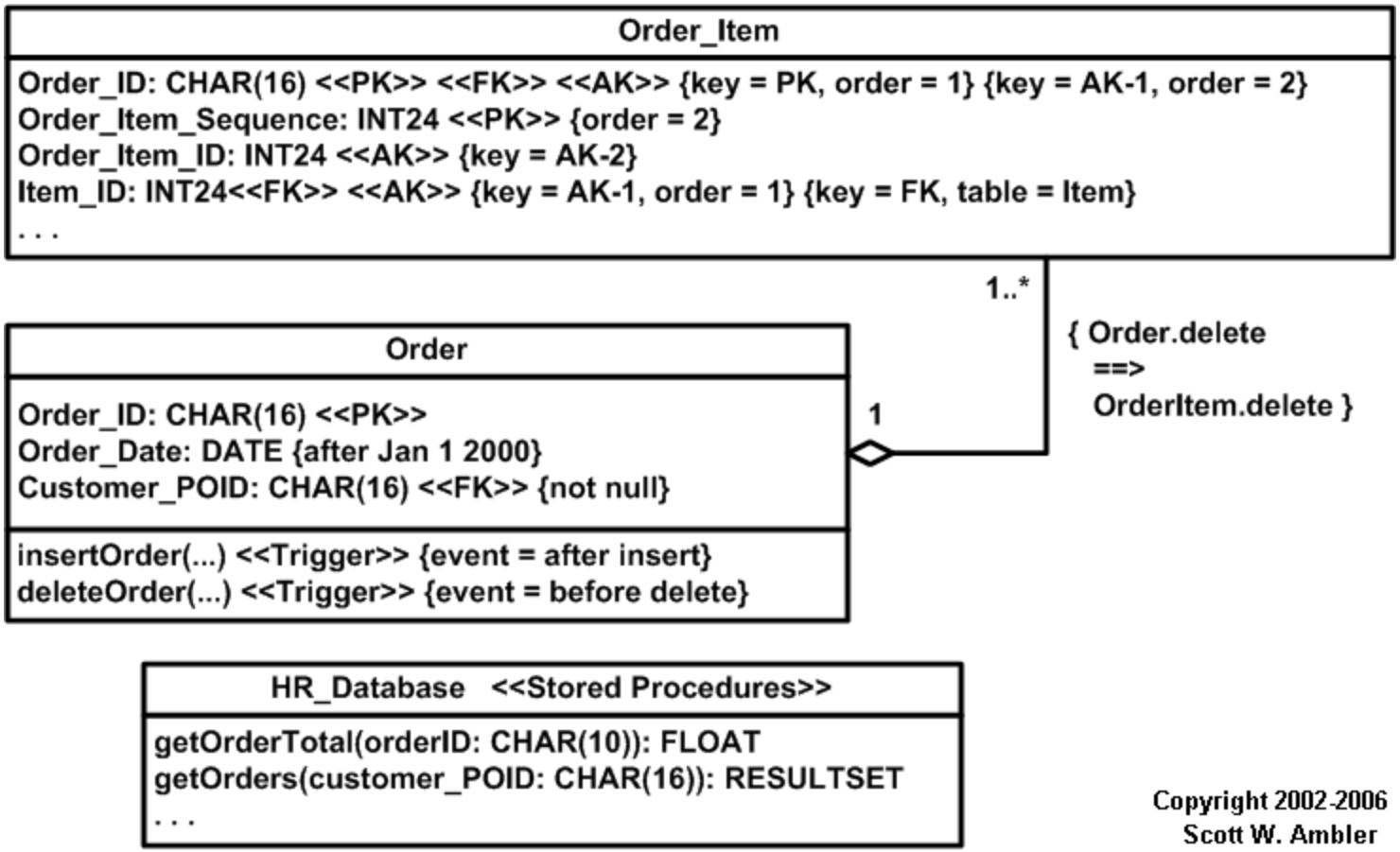


From http://www.conceptdraw.com/products/img/ScreenShots/cd5/software/Chen_ERD.gif

Categories of Data Models

- ***Physical (low-level, internal) data models***:
  - Provide concepts that describe details of how data is stored in the computer.
  - Generally meant for computer specialists and not for typical end-users

Sample Physical Data Model



From http://www.agiledata.org/essays/umlDataModelingProfile.html

Categories of Data Models (continued)

- ***Implementation (representational) data models***:
  - Provide concepts that fall between the above two, used by many commercial DBMS implementations (e.g. relational data models used in many commercial systems)

Relational Data Model

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Three-Schema Architecture

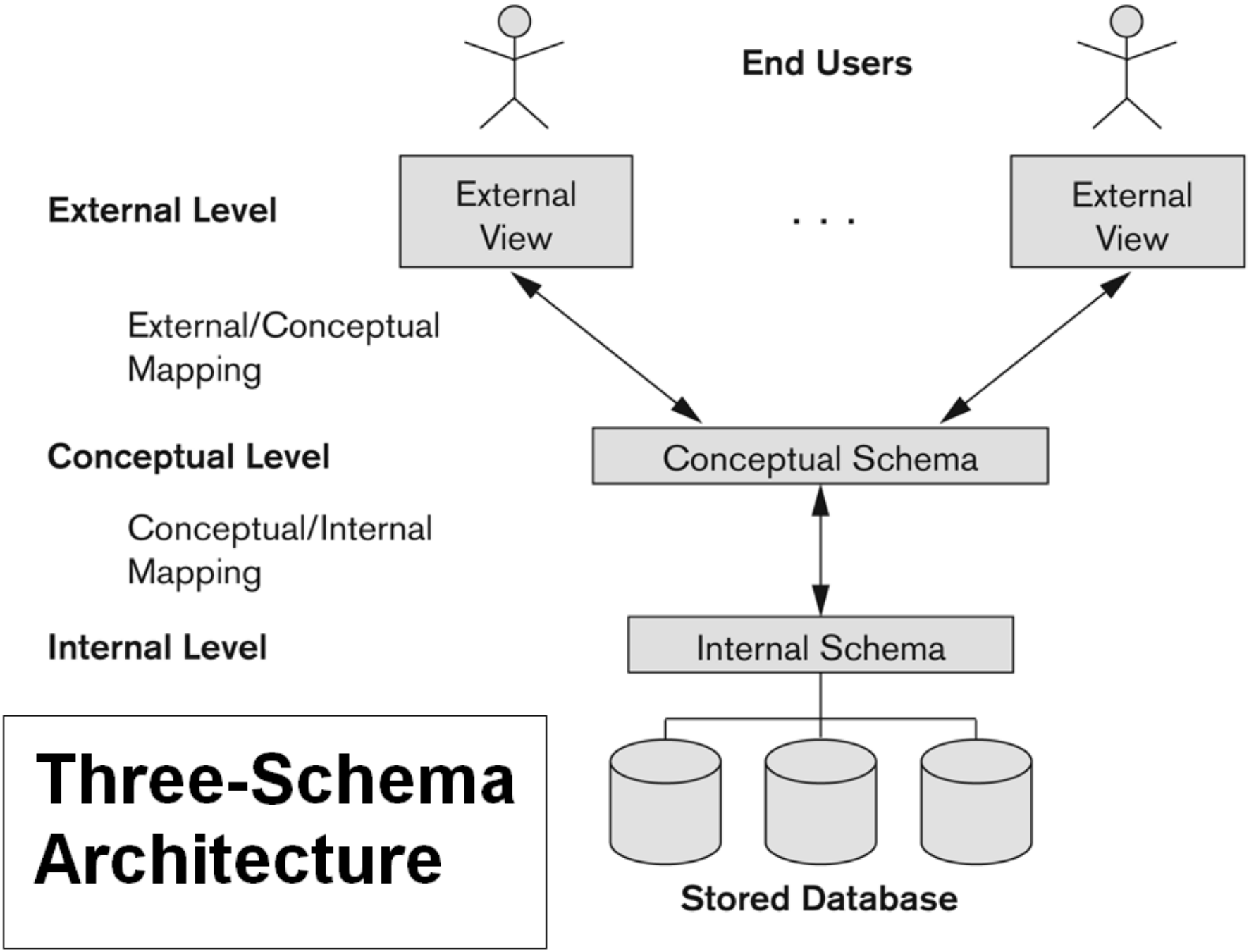
- Proposed to support DBMS characteristics of:
  - Support of ***multiple views*** of the data
  - ***Program-data independence***
- Not explicitly used in commercial DBMS products, but has been useful in explaining database system organization

Three-Schema Architecture

- Defines DBMS schemas ***at three*** levels:
  - ***Internal schema*** at the internal level to describe physical storage structures and access paths (e.g indexes).
    - Typically uses a ***physical*** data model.
- ***Conceptual schema*** at the conceptual level to describe the structure of the whole database for a community of users.
  - Uses a ***conceptual*** or an ***implementation*** data model.

Three-Schema Architecture (continued)

- ***External schemas*** at the external level to describe various user views.
  - Usually uses the same data model as the conceptual schema.



Three-Schema Architecture

Conceptual level

```
struct info{
    int id;
    char name[100];
    char address[200];
}
```

Three-Schema Architecture

- Mappings among schema levels are needed to ***transform requests*** and ***data***.
  - Programs refer to an external schema, and are mapped by the DBMS to the internal schema for execution.
  - Data extracted from the internal DBMS level is reformatted to match the user's external view (e.g. formatting the results of an SQL query for display in a Web page)

Data Independence

- ***Logical Data Independence*:**
  - The capacity to change the conceptual schema without having to change the external schemas and their associated application programs
- ***Physical Data Independence*:**
  - The capacity to change the internal schema without having to change the conceptual schema

Data Independence (continued)

- When a schema at a lower level is changed, only the ***mappings*** between this schema and higher-level schemas need to be changed in a DBMS that fully supports data independence.
- The higher-level schemas themselves are ***unchanged***.
  - Hence, the application programs do not need to be changed since they refer to the external schemas.

Classification of DBMSs

- Based on the data model used
  - ***Traditional***: Relational, Network, Hierarchical
  - ***Emerging***: Object-oriented, Object-relational

Relational Model

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

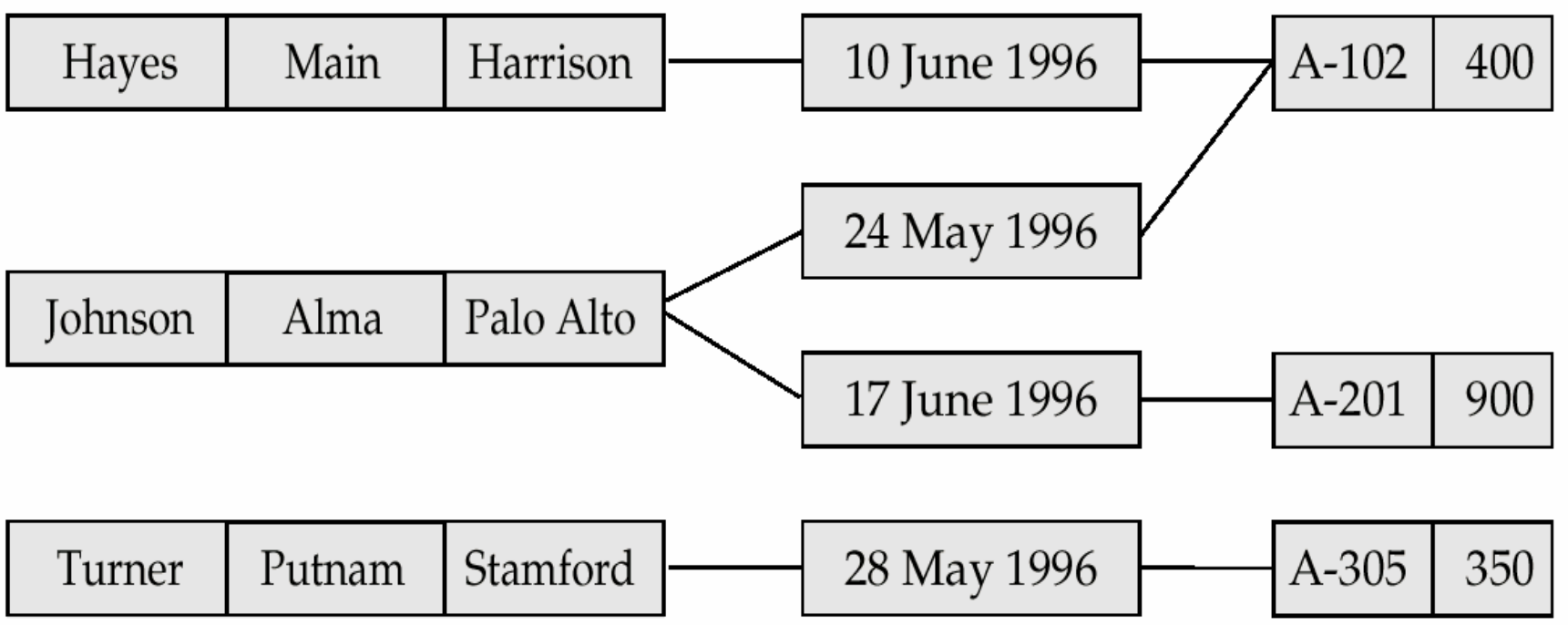
GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

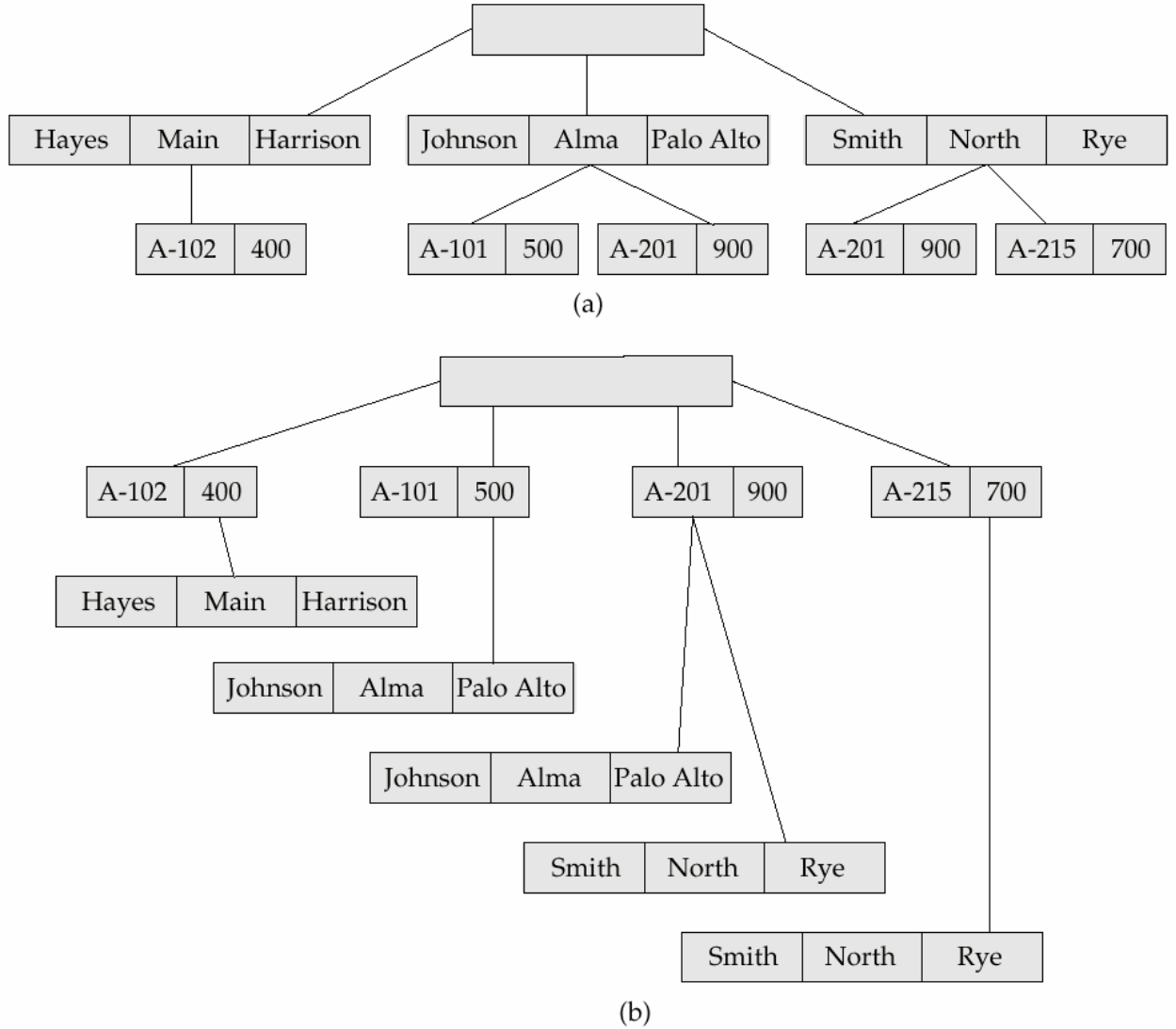
PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Network Model



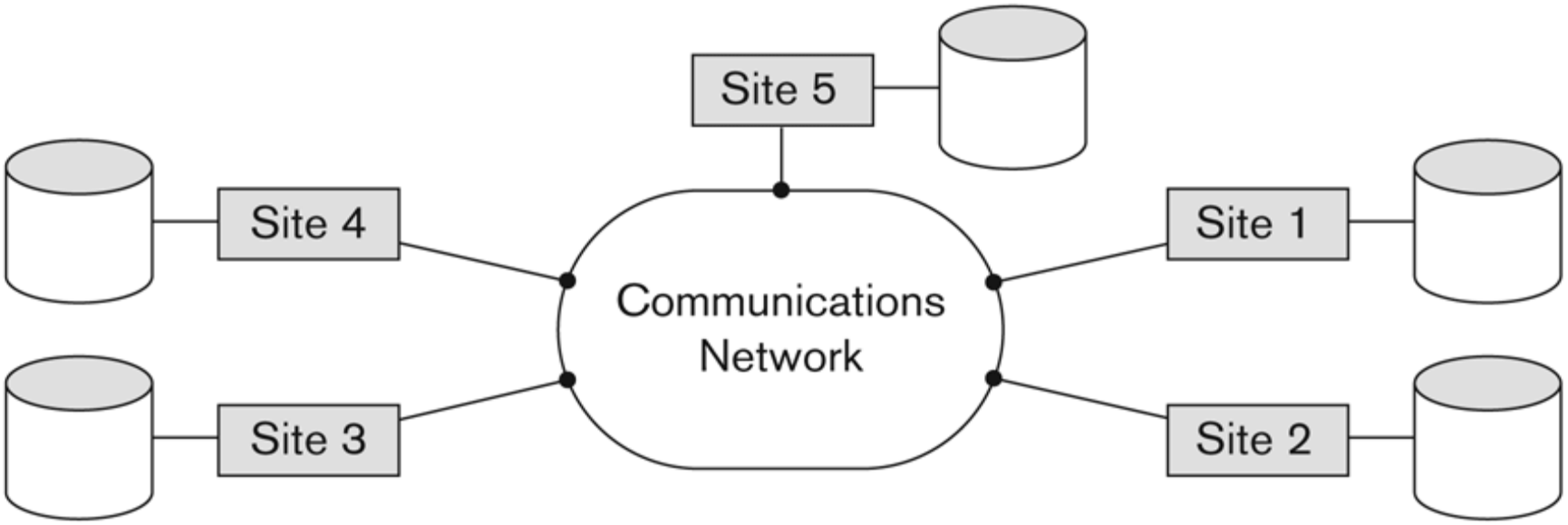
Hierarchical Model



Classification of DBMSs

- Other classifications
  - Single-user vs. multi-user
  - Centralized vs. distributed
  - Free vs. commercialized

Distributed Database System



Reference(s):

- **Elmasri, R. and S.B. Navathe**. 2010. Fundamentals of Database Systems. 6th Edition. Addition Wesley. ISBN-13: 978-0-136-08620-8
- **Elmasri, R. and S.B. Navathe**. 2007. Fundamentals of Database Systems. 5th Edition. Addition Wesley. ISBN: 981-06-9800-3