Information Management II

2. Database Architecture

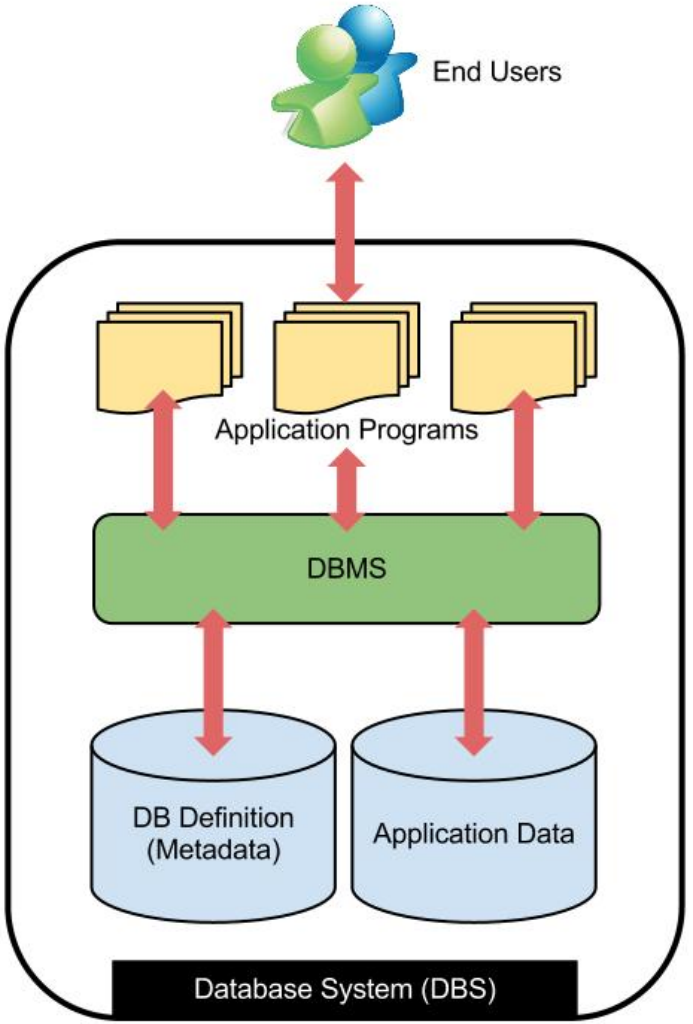
CS4D2a – 4CSLL1 – CS3041

Vincent Wade

Vincent.wade@scss.tcd.ie

Database Systems

- Database System (DBS)
  - DBMS
  - DB
    - application data
    - associated metadata
  - Application programs
- Metadata and data are stored separately



DBMS Architecture

- Database users are provided an abstract view of the data by hiding certain details of how it is physically stored
- DBMS describe Databases at three levels:
  - Internal (Physical) Level
  - Conceptual (Logical) Level
  - External (View) Level
- This is commonly referred to as the “three-level DBMS architecture”

Schemas

- Each level of the architecture consists of one or more views of the underlying data
- Views are described by *schemas* (meta-data)
- A DB consists of:
  - physical data
  - an internal or physical schema
  - a conceptual or logical schema
  - several external schemas
- Schemas are stored in the system catalogue

DBMS Architecture

- Internal or Physical level
  - The lowest level of data abstraction
  - Internal Schema describes how the data is physically stored and organized on the storage medium
  - Various aspects are considered to achieve optimal runtime performance and storage space utilization, including:
    - storage space allocation techniques
    - access paths such as indexes
    - data compression and encryption techniques

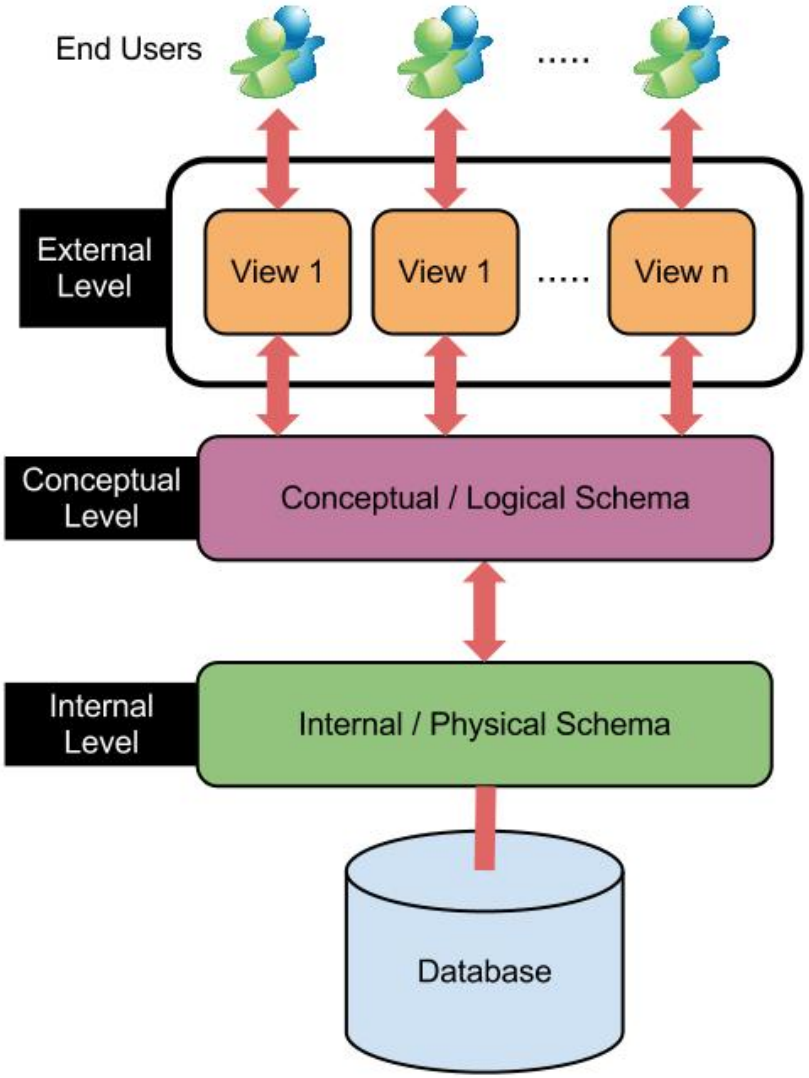
DBMS Architecture

- Conceptual or Logical level
  - Deals with the logical structure of the entire database
  - Conceptual Schema describes what data is stored in the database and the relationships among the data without any concern for the physical implementation
  - This is the overall view of the database and includes all the information that is going to be represented in the database

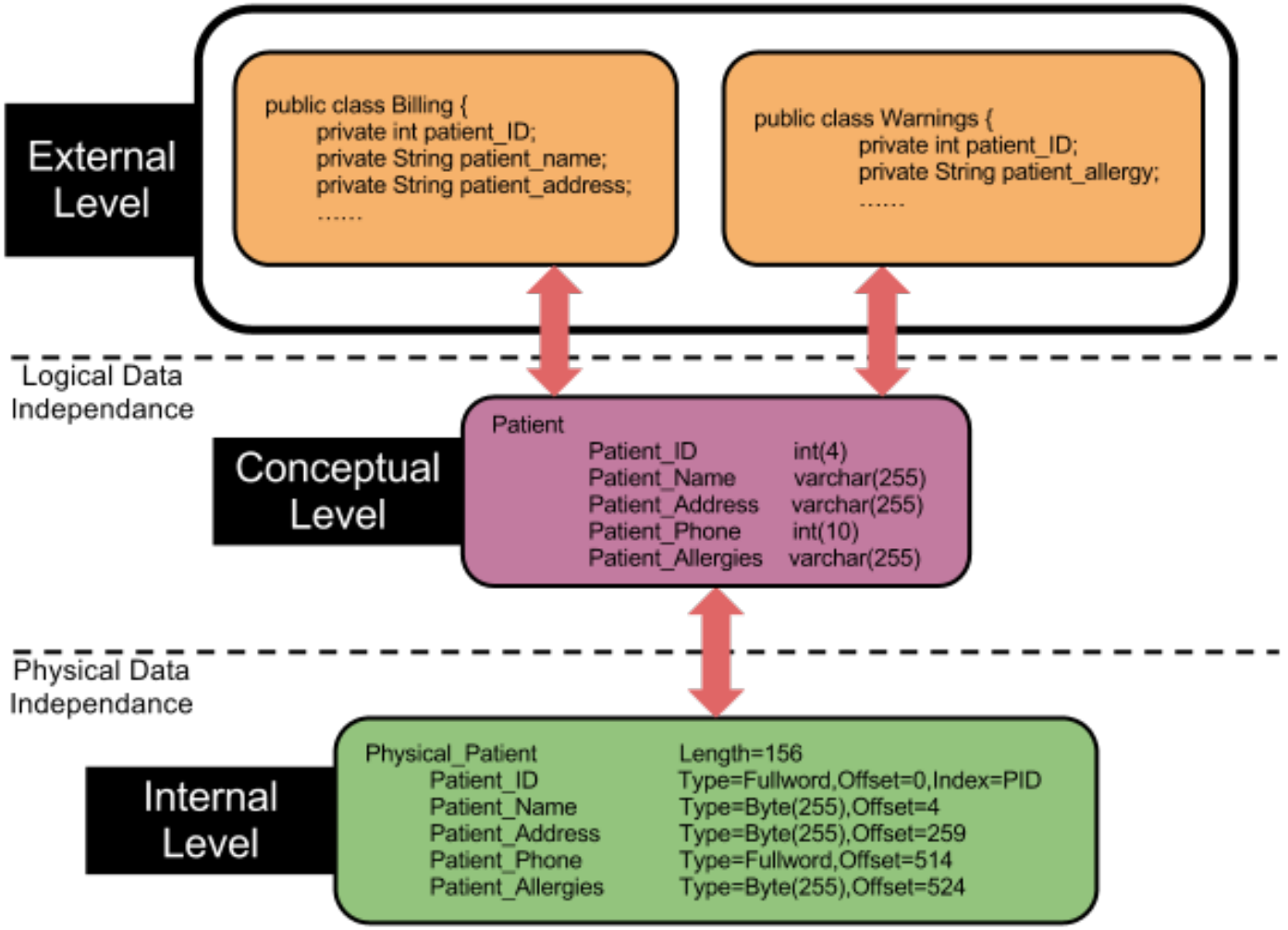
DBMS Architecture

- External or View level
  - The highest level of abstraction that deals with the user's view of the database
  - Most users and applications do not require access to the entire data stored in the database.
  - External Schemas (or User Views) describe a part of the database for a particular group of users or applications.
  - This is a powerful and flexible security mechanism, as parts of the database are hidden from certain users
    - the user is not aware of the existence of any attributes that are missing from the view.

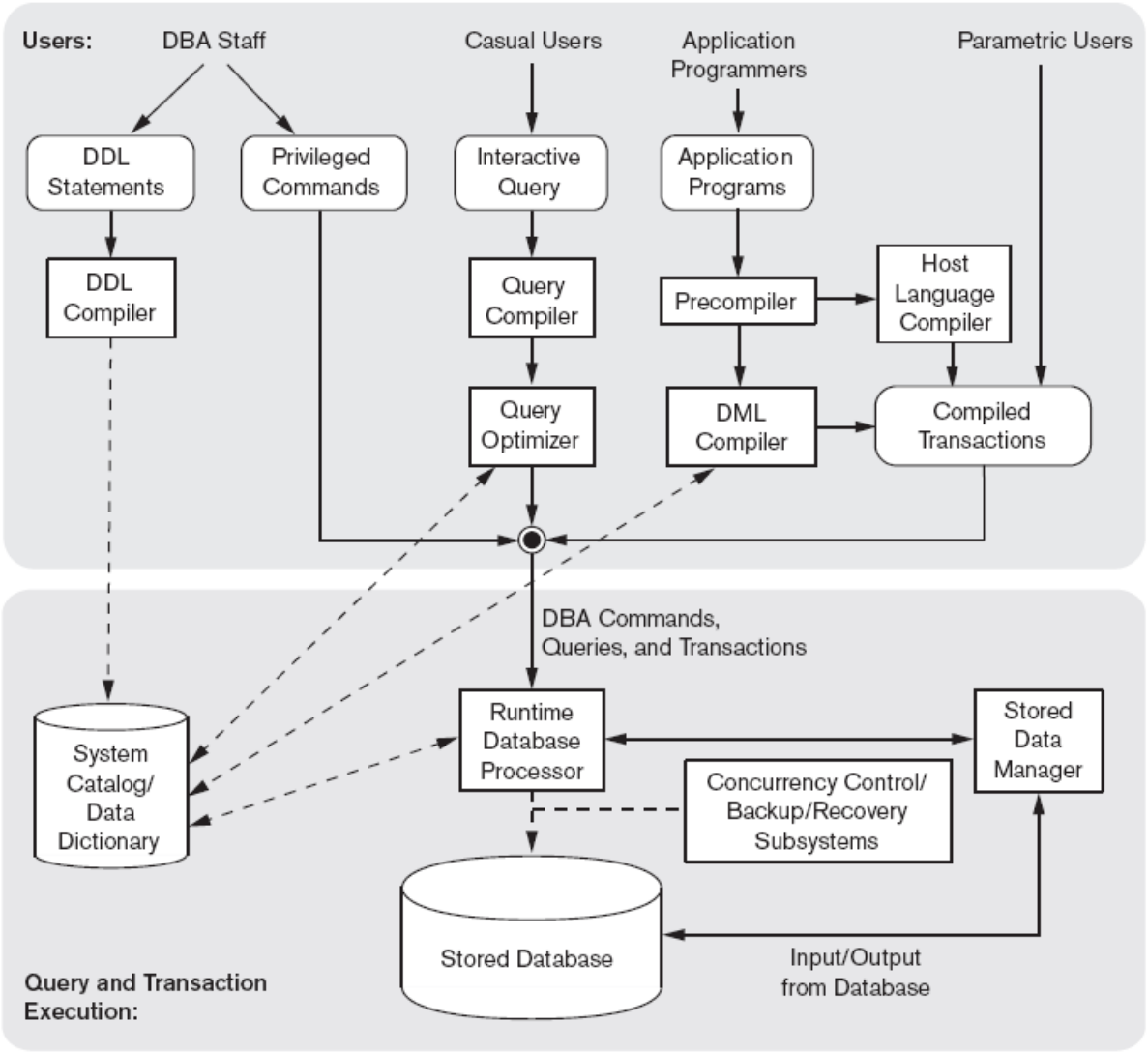
DBMS Architecture



Schemas


External Level
public class Billing {
private int patient_ID;
private String patient_name;
private String patient_address;
.....
public class Warnings {
private int patient_ID;
private String patient_allergy;
.....
Logical Data Independance
Conceptual Level
Patient
Patient_ID int(4)
Patient_Name varchar(255)
Patient_Address varchar(255)
Patient_Phone int(10)
Patient_Allergies varchar(255)
Physical Data Independance
Internal Level
Physical_Patient Length=156
Patient_ID Type=Fullword,Offset=0,Index=PID
Patient_Name Type=Byte(255),Offset=4
Patient_Address Type=Byte(255),Offset=259
Patient_Phone Type=Fullword,Offset=514
Patient_Allergies Type=Byte(255),Offset=524

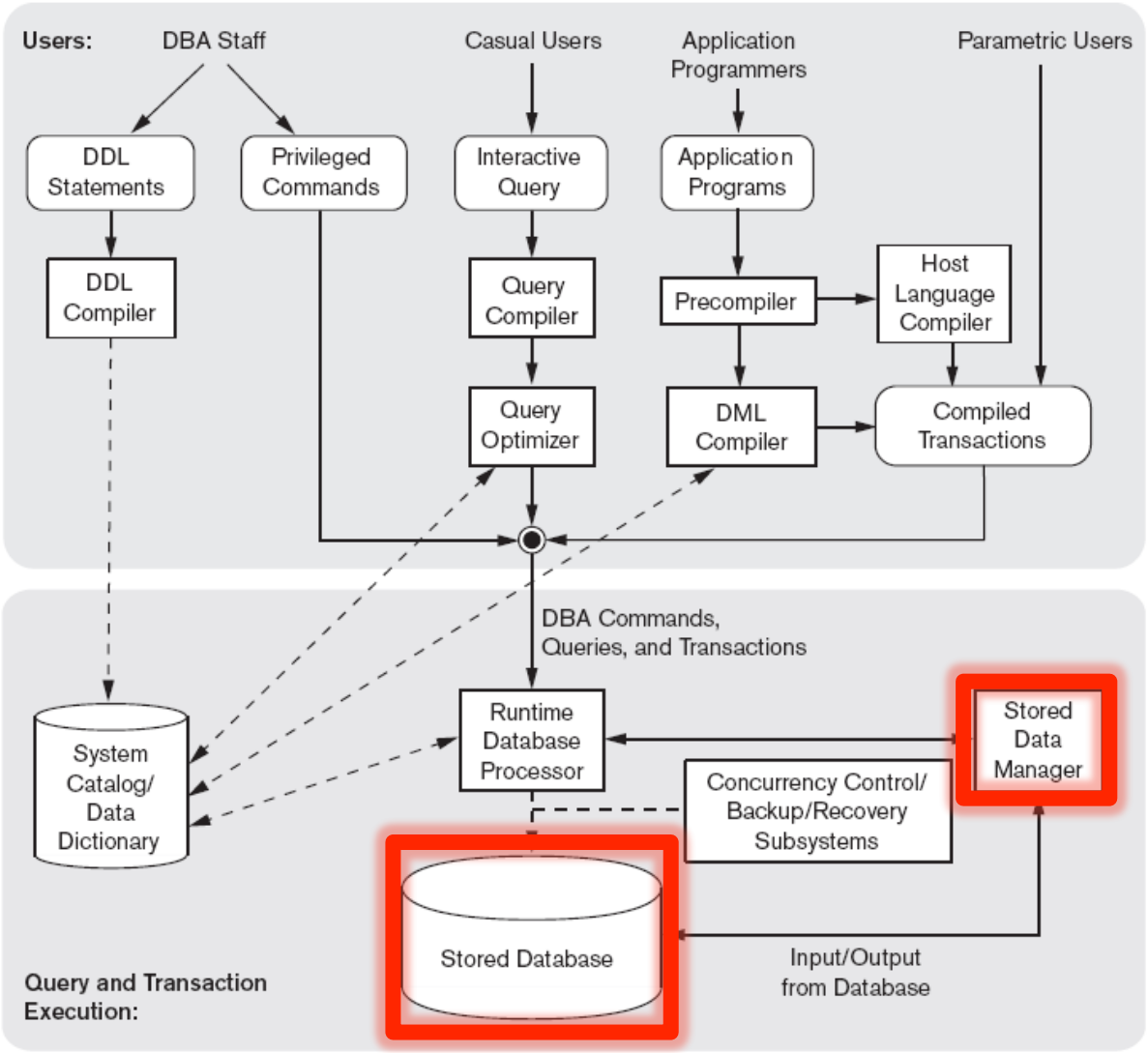
DBMS Components



DBMS Components

- The physical Database is usually stored on Hard Disk
  - The OS controls disk access
- The **Stored Data Manager (SDM)** controls access to DBMS information on disk
  - including buffer management

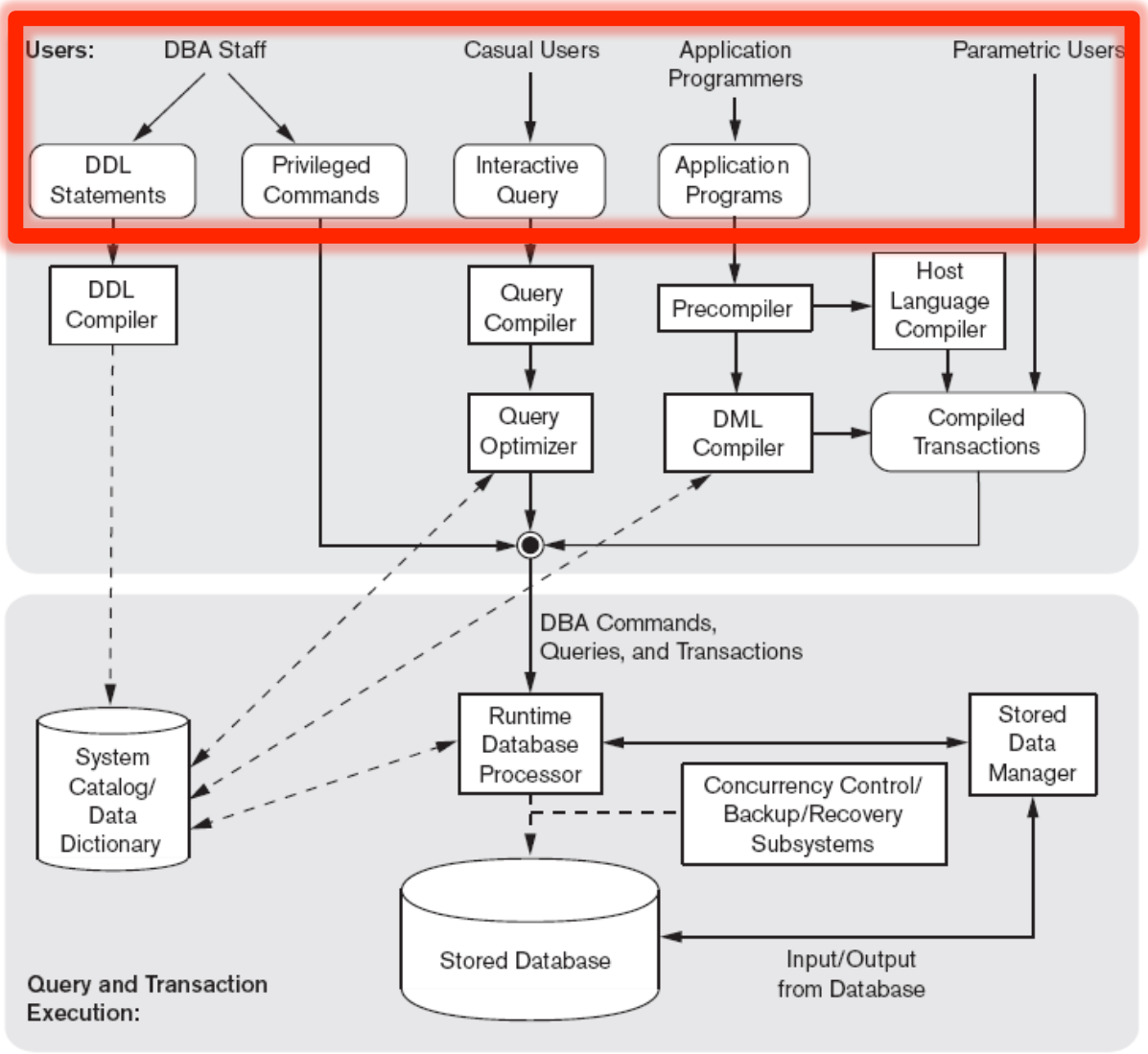
DBMS Components



DBMS Components

- DBMS Users
  - Casual Users
  - Application Programmers
  - Parametric Users
  - DBA Staff
- Different Interfaces are used by each type of user

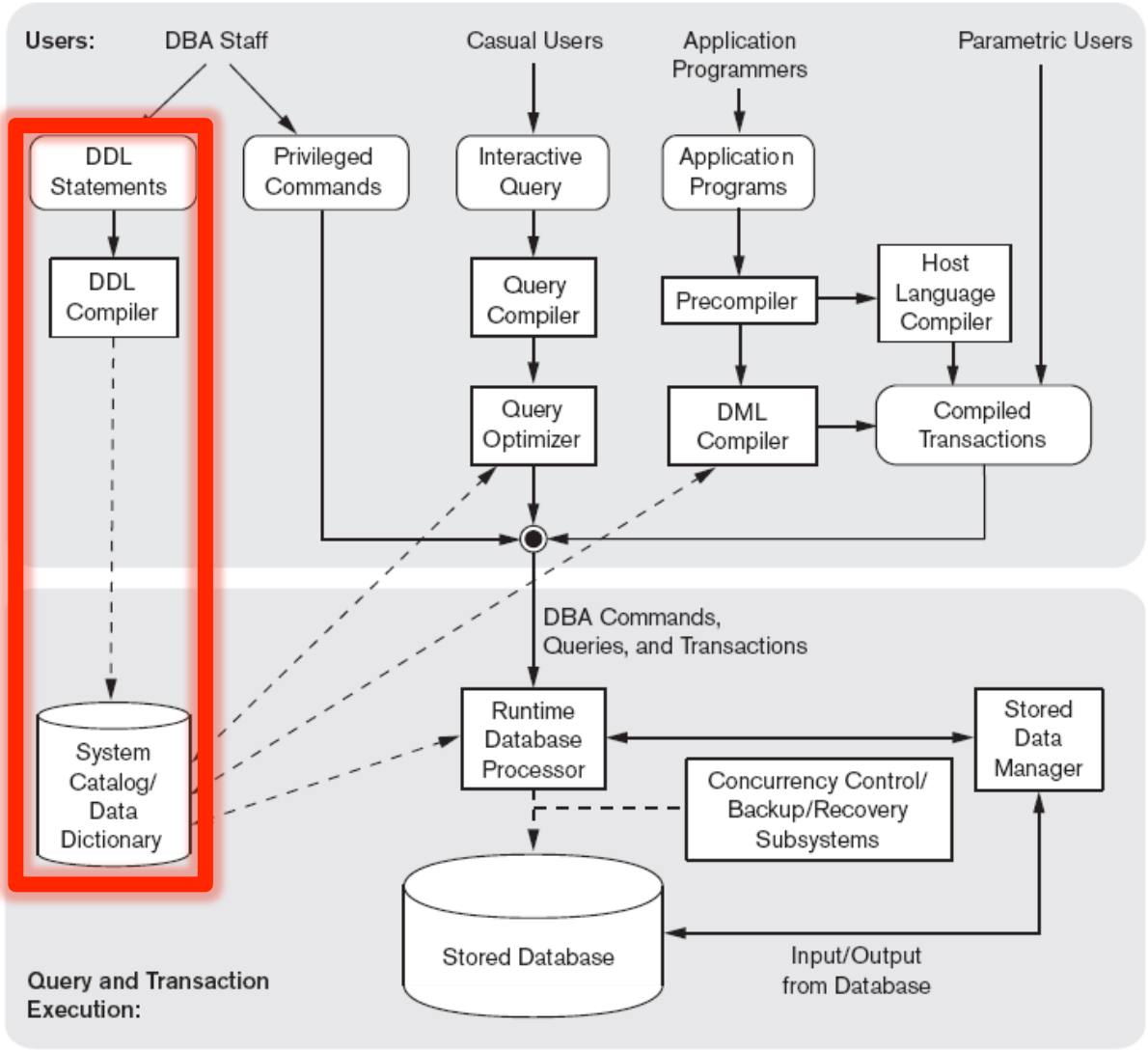
DBMS Components



DBMS Components

- **DDL compiler** processes schema definitions and stores them in catalogue
- **Catalogue** contains information such as:
  - Names and Sizes of Files
  - Names and Data Type of Data Items
  - Storage details
  - Mapping information among Schemas
  - Constraints
  - ….

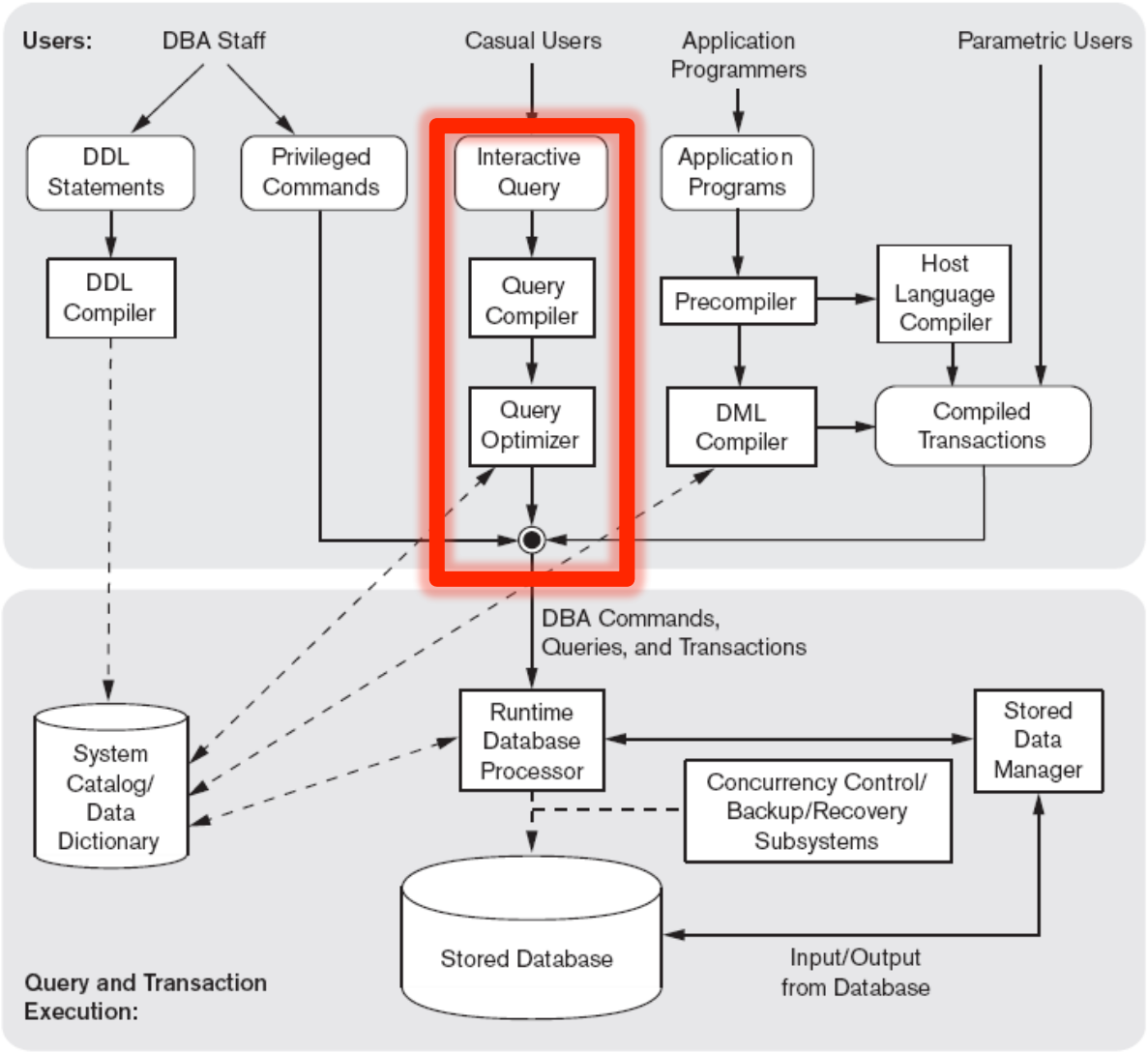
DBMS Components



DBMS Components

- "Casual Users" use an **Interactive Query Interface**
- The **Query Compiler** parses and validates the submitted query
- The internal query is then processed for **Query Optimization**
  - Consults the DBMS Catalogue
  - Generates Executable Code

DBMS Components



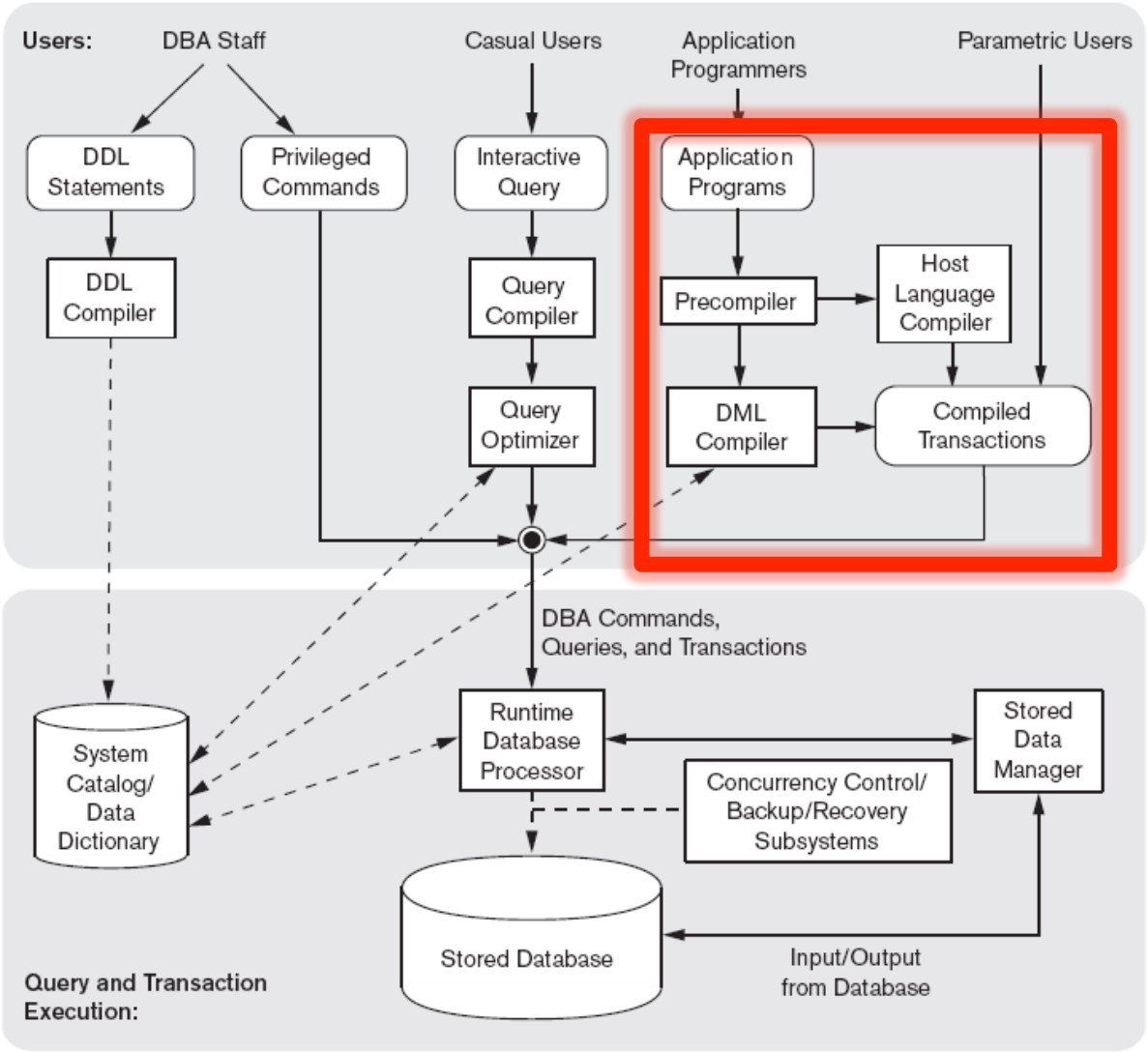
DBMS Components

- Application Programmers write programs (Java, C++ etc.) which need to access a DB
- The **Precompiler** extracts DML commands from the host language program
- The extracted commands are sent to the **DML Compiler**
- The rest of the program is sent to the **Host Language Compiler**

DBMS Components

- Object code for DML commands and the rest of the program are linked forming a **canned transaction**
  - The executable code of a canned transaction calls the run-time processor
  - Canned transactions are used by parametric users

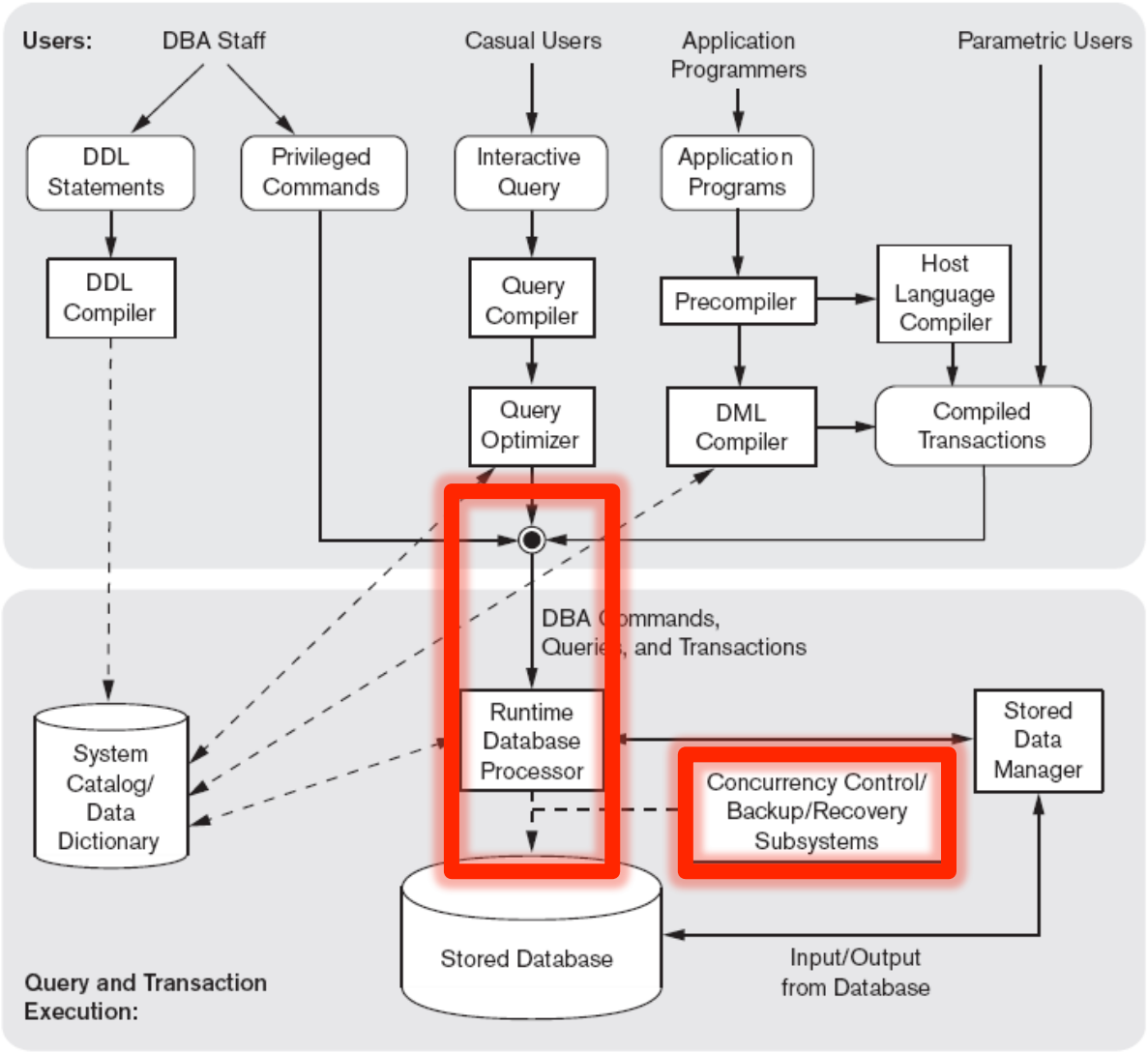
DBMS Components



DBMS Components

- **Run-time Database Processor** handles all Database access at run-time:
  - Privileged Commands
  - Executable Queries
  - Canned Transactions
- Utilises and Updates the Catalogue
- May be responsible for Buffer Management
- Manages **Concurrency Control** and **Backup and Recovery** as part of Transaction Management

DBMS Components



System Catalogue and Data Dictionary

- The DDL, and hence the system catalogue, are primarily concerned with *syntactic* definition of the data
- Data Dictionaries augment the internal DBMS catalogue with *semantic* support
  - Accessed directly by users (i.e. DBA)
  - Catalogue accessed by the DBMS

System Catalogue and Data Dictionary

- There are two main ways of coupling Data Dictionaries and System Catalogues:
  - Integrated Data Dictionary
  - Independent Data Dictionary

Integrated

- The majority of DBMS have an integrated Data Dictionary
- Data Dictionary is an integral part of DBMS
  - Documents the computerised data that is managed by the DBMS
- It is generally fully active
  - accessed at run-time by DBMS software

Independent

- Independent, free-standing system performing its own data management functions
- Normally *passive*
  - No run-time link between the Data Dictionary and the DBMS
  - Hence DBMS has to have its own System Catalogue
- Often generates metadata automatically for a variety of DBMS in the form of DDL
  - Helps to ensure consistency of metadata between the Data Dictionary and the System catalogue

Data Dictionary Systems

A fully functional DDS should store and manage:

a) Descriptions of the database schemas

b) Detailed information on physical database design
   - Storage structures
   - Access paths
   - File and record sizes.

c) Descriptions of the types of database users, their responsibilities and their access rights.

d) High-level descriptions of transactions, applications and the relationships of users to transactions.

Data Dictionary Systems (con't)

e) The relationship between database transactions and the data items referenced by them.

f) Usage statistics such as frequencies of queries and transactions and access counts to different portions of the database.

g) The history of any changes made to the database and applications, and documentation that describes the reasons for these changes.