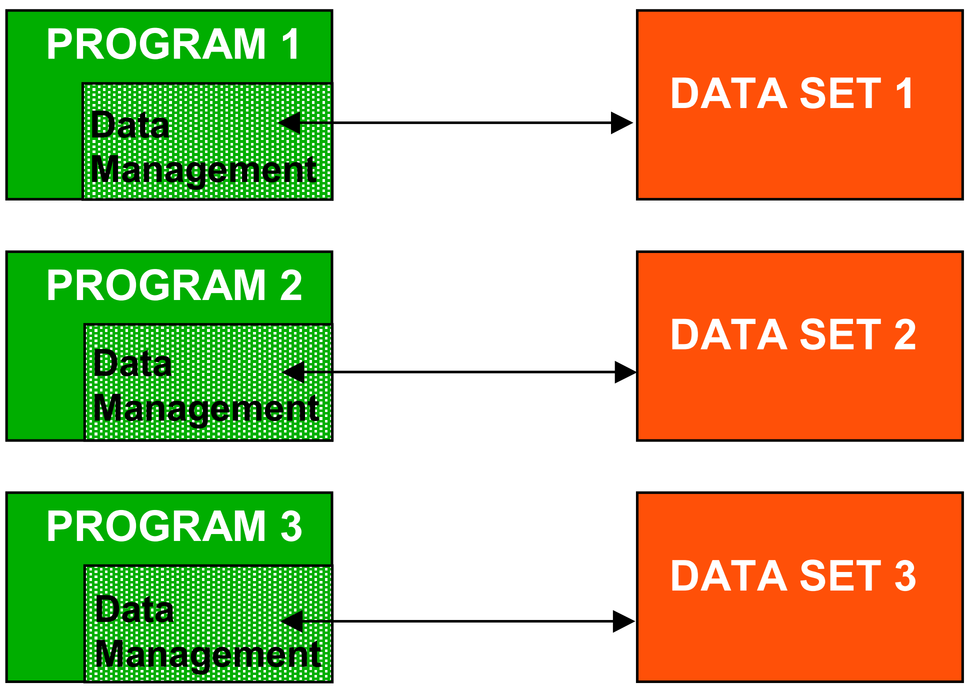
# What is "Data"?

- ANSI definition:
  - Data
    - ❶ A representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or by automatic means.
    - ❷ Any representation such as characters or analog quantities to which meaning is or might be assigned. Generally, we perform operations on data or data items to supply some information about an entity.
- Volatile vs. persistent data
  - Our concern is primarily with persistent data

# Early Data Management - Ancient History

- Data are not stored on disk
- Programmer defines both logical data structure and physical structure (storage structure, access methods, I/O modes, etc)
- One data set per program. High data redundancy.

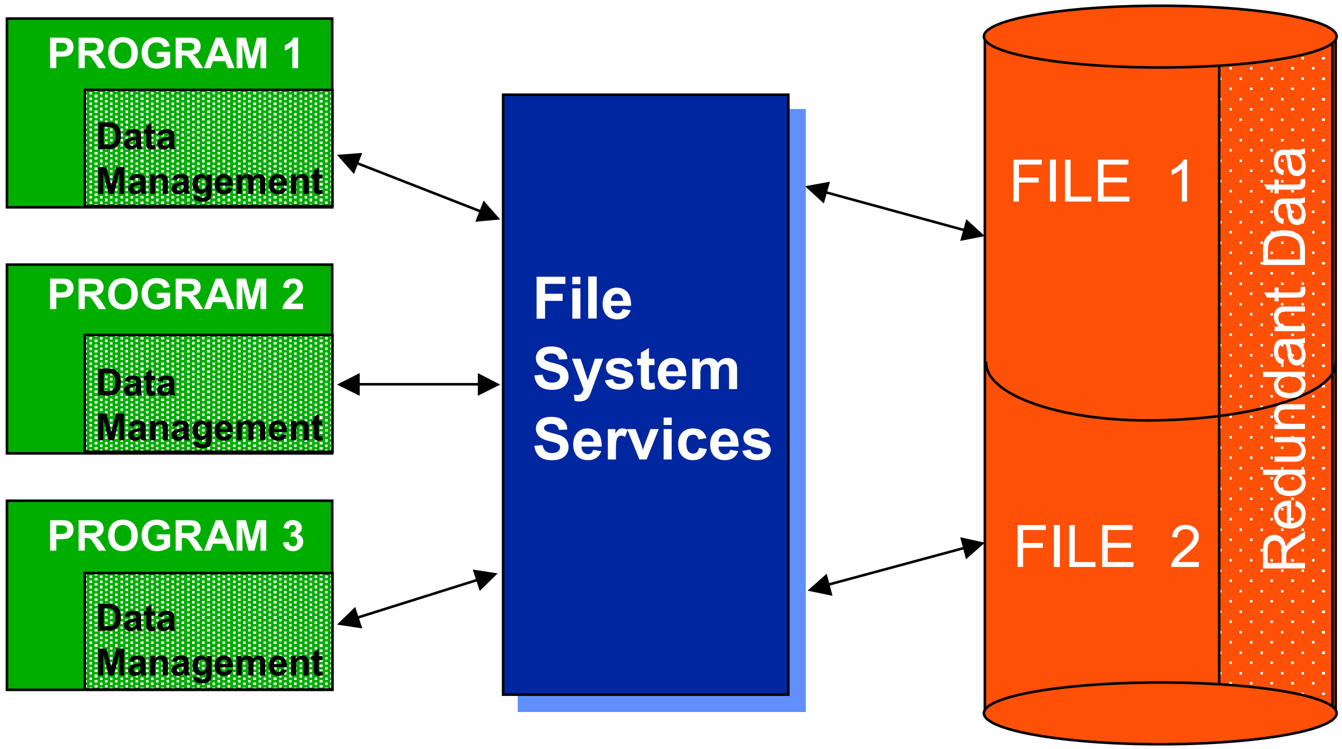


# Problems

- There is no persistence.
  - All data is transient and disappears when the program terminates.
- Random access memory (RAM) is expensive and limited
  - All data may not fit available memory
- Programmer productivity low
  - The programmer has to do a lot of tedious work.

# File Processing - Recent (and Current) History

- Data are stored in files with interface between programs and files.
- Various access methods exist (e.g., sequential, indexed, random)
- One file corresponds to one or several programs.



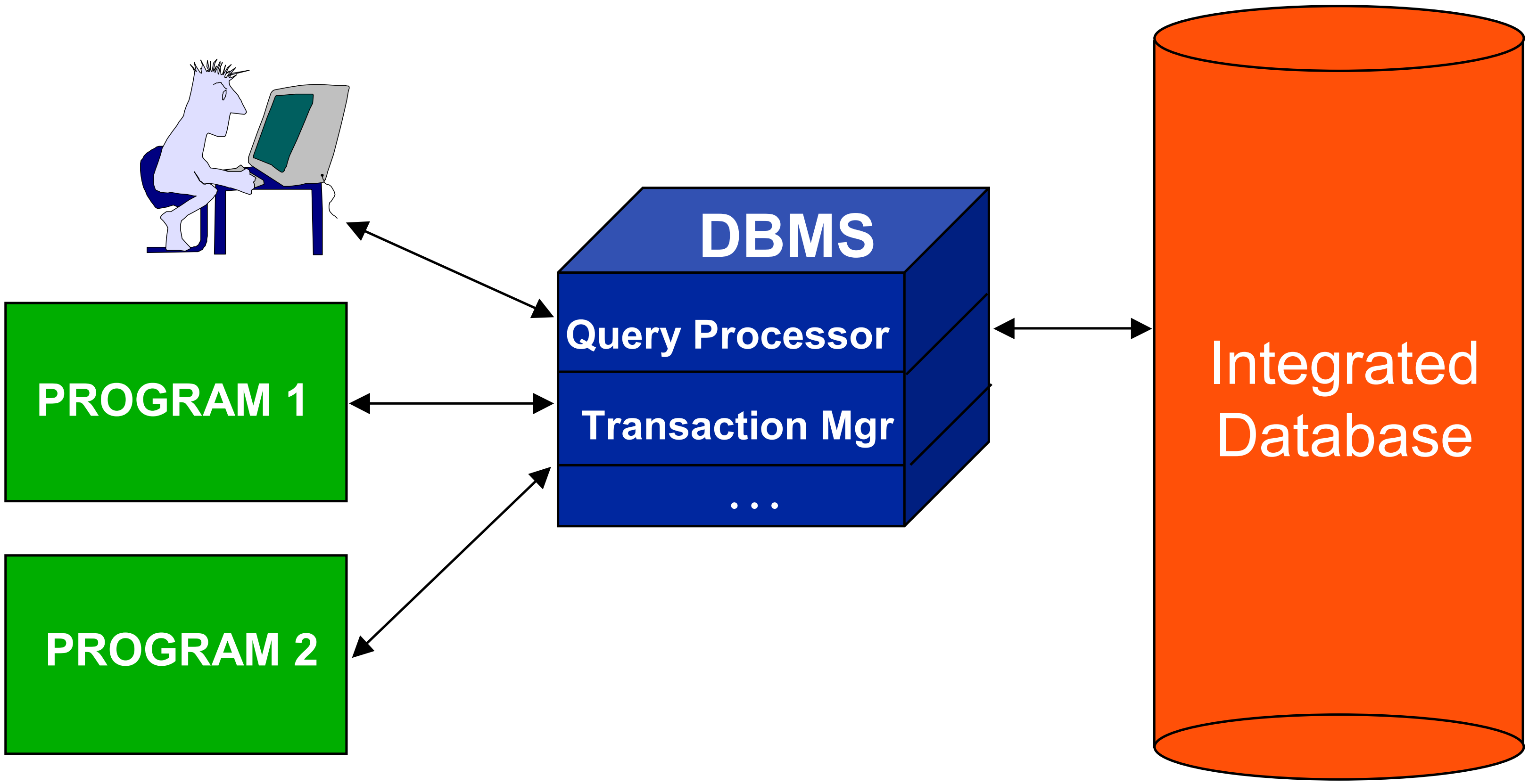
# File System Functions

- Mapping between logical files and physical files
  - Logical files: a file viewed by users and programs.
    - Data may be viewed as a collection of bytes or as a collection of records (collection of bytes with a particular structure)
    - Programs manipulate logical files
  - Physical files: a file as it actually exists on a storage device.
    - Data usually viewed as a collection of bytes located at a physical address on the device
    - Operating systems manipulate physical files.
- A set of services and an interface (usually called application independent interface – API)

# Problems With File Systems

- Data are highly redundant
  - sharing limited and at the file level
- Data is unstructured
  - "flat" files
- High maintenance costs
  - data dependence; accessing data is difficult
  - ensuring data consistency and controlling access to data
- Sharing granularity is very coarse
- Difficulties in developing new applications

# Database Approach



# What is a Database?

- A database is an integrated and structured collection of stored operational data used (shared) by application systems of an enterprise

| | |
|---|---|
| Manufacturing | Product data |
| University | Student data, courses |
| Hospital | Patient data, facilities |
| Bank | Account data |

# What is a Database?

- A database (DB) is a structured collection of data about the entities that exist in the environment that is being modeled.
- The structure of the database is determined by the abstract data model that is used.
- A database management system (DBMS) is the generalized tool that facilitates the management of and access to the database.

# Data Model

- Formalism that defines what the structure of the data are
  - within a file
  - between files
- File systems can at best specify data organization within one file
- Alternatives for business data
  - Hierarchical; network
  - Relational
  - Object-oriented
- This structure is usually called the schema
  - A schema can have many instances

# Example Relation Instances

EMP

| ENO | ENAME | TITLE |
|---|---|---|
| E1 | J. Doe | Elect. Eng. |
| E2 | M. Smith | Syst. Anal. |
| E3 | A. Lee | Mech. Eng. |
| E4 | J. Miller | Programmer |
| E5 | B. Casey | Syst. Anal. |
| E6 | L. Chu | Elect. Eng. |
| E7 | R. Davis | Mech. Eng. |
| E8 | J. Jones | Syst. Anal. |

WORKS

| ENO | PNO | RESP | DUR |
|---|---|---|---|
| E1 | P1 | Manager | 12 |
| E2 | P1 | Analyst | 24 |
| E2 | P2 | Analyst | 6 |
| E3 | P3 | Consultant | 10 |
| E3 | P4 | Engineer | 48 |
| E4 | P2 | Programmer | 18 |
| E5 | P2 | Manager | 24 |
| E6 | P4 | Manager | 48 |
| E7 | P3 | Engineer | 36 |
| E7 | P5 | Engineer | 23 |
| E8 | P3 | Manager | 40 |

PROJ

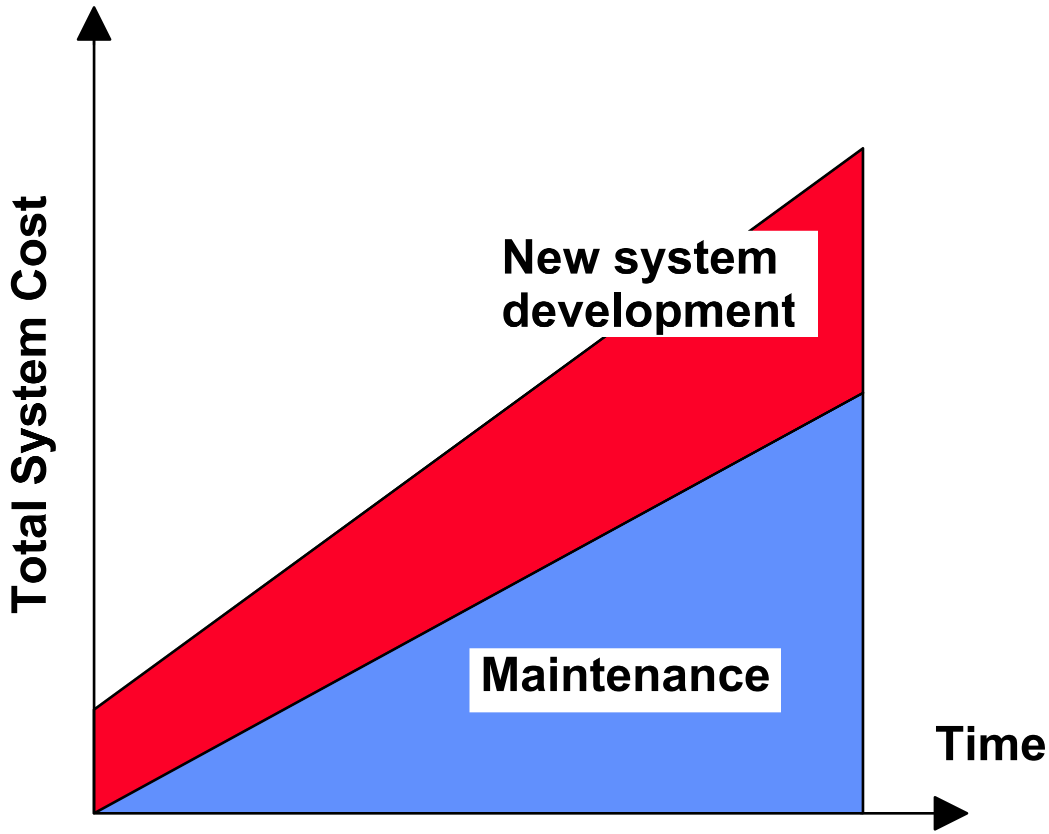
| PNO | PNAME | BUDGET |
|---|---|---|
| P1 | Instrumentation | 150000 |
| P2 | Database Develop. | 135000 |
| P3 | CAD/CAM | 250000 |
| P4 | Maintenance | 310000 |
| P5 | CAD/CAM | 500000 |

# Why Database Technology

- Data constitute an organizational asset ⇨ Integrated control
  - Reduction of redundancy
  - Avoidance of inconsistency
  - Sharability
  - Standards
  - Improved security
  - Data integrity
- Programmer productivity ⇨ Data Independence

# Why Database Technology

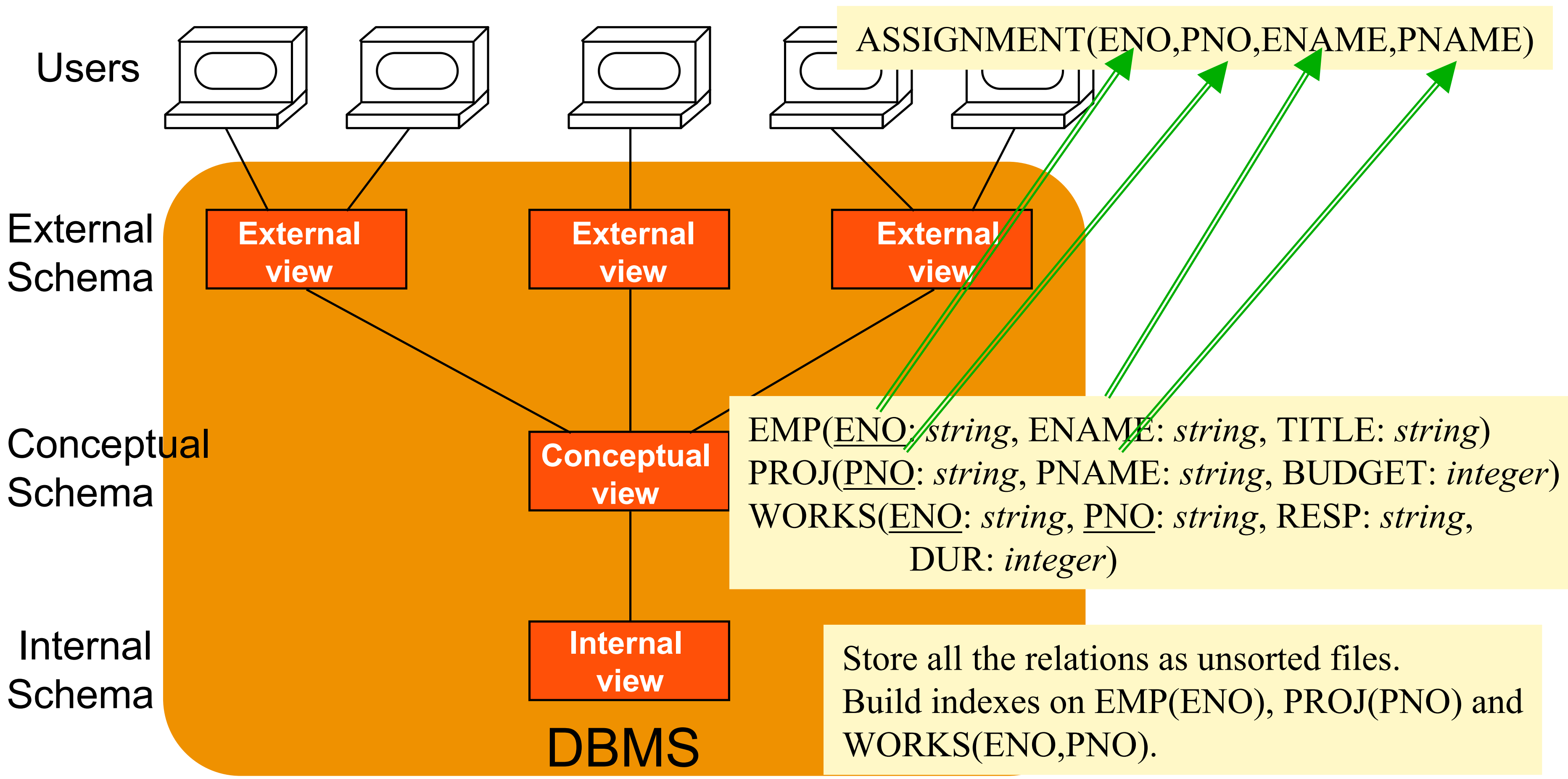
- Programmer productivity
  - High data independence



# Data Independence

- Invisibility (transparency) of the details of conceptual organization, storage structure and access strategy to the users
  - Logical
    - transparency of the conceptual organization
    - transparency of logical access strategy
  - Physical
    - transparency of the physical storage organization
    - transparency of physical access paths

# ANSI/SPARC Architecture



# Database Functionality

- Integrated schema
  - Users have uniform view of data
  - They see things only as relations (tables) in the relational model
- Declarative integrity and consistency enforcement
  - $24000 \leq \text{Salary} \leq 250000$
  - No employee can have a salary greater than his/her manager.
  - User specifies and system enforces.
- Individualized views
  - Restrictions to certain relations
  - Reorganization of relations for certain classes of users

# Database Functionality (cont'd)

- Declarative access
  - Query language - SQL
    - Find the names of all electrical engineers.

```
SELECT   ENAME
FROM     EMP
WHERE    TITLE = "Elect. Eng."
```

    - Find the names of all employees who have worked on a project as a manager for more than 12 months.

```
SELECT   EMP.ENAME
FROM     EMP,ASG
WHERE    RESP = "Manager"
AND      DUR > 12
AND      EMP.ENO = ASG.ENO
```

- System determined execution
  - Query processor & optimizer

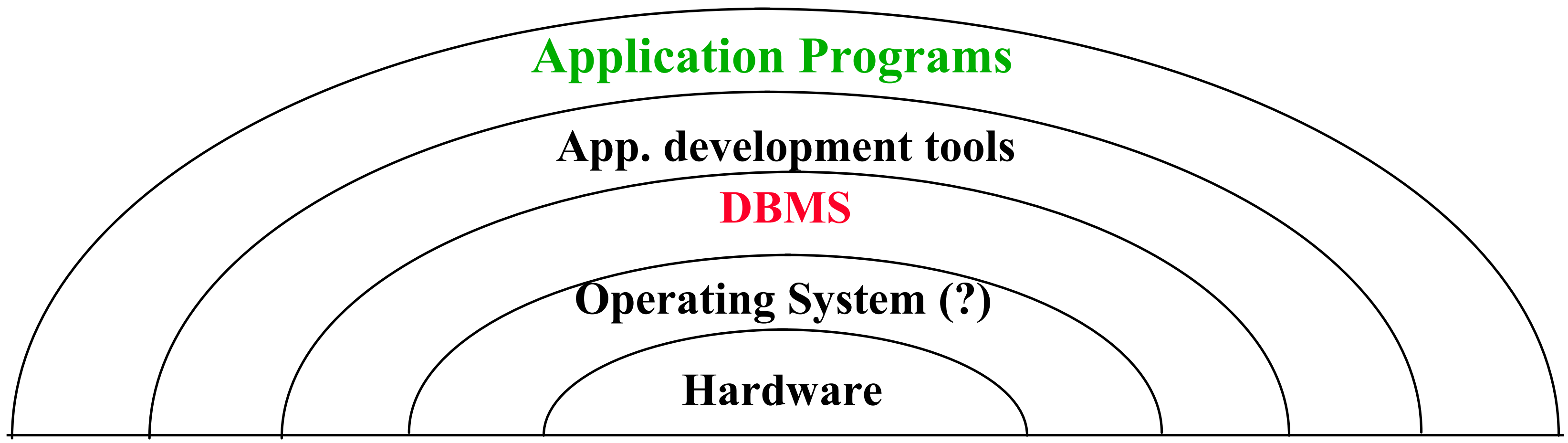
# Database Functionality (cont'd)

- Transactions
  - Execute user requests as atomic units
  - May contain one query or multiple queries
  - Provide
    - Concurrency transparency
      - Multiple users may access the database, but they each see the database as their own personal data
      - Concurrency control
    - Failure transparency
      - Even when system failure occurs, database consistency is not violated
      - Logging and recovery

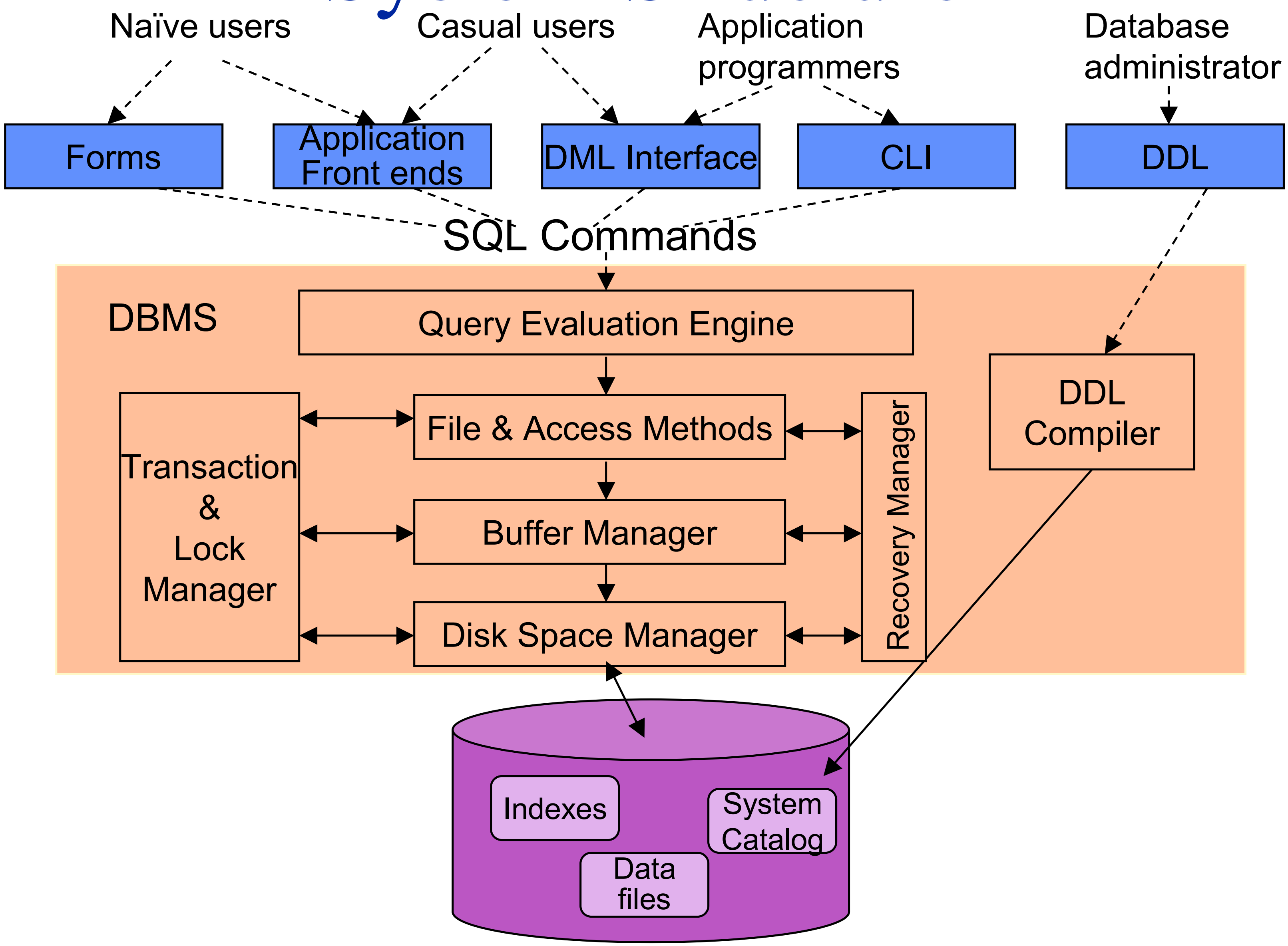
# Database Functionality (cont'd)

- Transaction Properties
  - Atomicity
    - All-or-nothing property
  - Consistency
    - Each transaction is correct and does not violate database consistency
  - Isolation
    - Concurrent transactions do not interfere with each other
  - Durability
    - Once the transaction completes its work (commits), its effects are guaranteed to be reflected in the database regardless of what may occur

# Place in a Computer System



# System Structure



# Database Users

- End user
  - Naïve or casual user
  - Accesses database either through forms or through application front-ends
  - More sophisticated ones generate ad hoc queries using DML
- Application programmer/developer
  - Designs and implements applications that access the database (some may be used by end users)
- Database administrator (DBA)
  - Defines and manages the conceptual schema
  - Defines application and user views
  - Monitors and tunes DBMS performance (defines/modifies internal schema)
  - Loads and reformats the database
  - Responsible for security and reliability

# DBMS Languages

- Data Definition Language (DDL)
  - Defines conceptual schema, external schema, and internal schema, as well as mappings between them
  - Language used for each level may be different
  - The definitions and the generated information is stored in system catalog
- Data Manipulation Language (DML)
  - Can be
    - embedded query language in a host language
    - “stand-alone” query language
  - Can be
    - Procedural: specify where and how (navigational)
    - Declarative: specify what

# Brief History of Databases

- 1960s:
  - Early 1960s: Charles Bachmann developed first DBMS at Honeywell (IDS)
    - Network model where data relationships are represented as a graph.
  - Late 1960s: First commercially successful DBMS developed at IBM (IMS)
    - Hierarchical model where data relationships are represented as a tree
    - Still in use today (SABRE reservations; Travelocity)
  - Late 1960s: Conference On DAta Systems Languages (CODASYL) model defined. This is the network model, but more standardized.

# Brief History of Databases

- 1970s:
  - 1970: Ted Codd defined the relational data model at IBM San Jose Laboratory (now IBM Almaden)
  - Two major projects start (both were operational in late 1970s)
    - INGRES at University of California, Berkeley
      - Became commercial INGRES, followed-up by POSTGRES which was incorporated into Informix
    - System R at IBM San Jose Laboratory
      - Became DB2
  - 1976: Peter Chen defined the Entity-Relationship (ER) model

# Brief History of Databases

- 1980s
  - Maturation of relational database technology
  - SQL standardization (mid-to-late 1980s) through ISO
  - The real growth period
- 1990s
  - Continued expansion of relational technology and improvement of performance
  - Distribution becomes a reality
  - New data models: object-oriented, deductive
  - Late 1990s: incorporation of object-orientation in relational DBMSs → Object-Relational DBMSs
  - New application areas: Data warehousing and OLAP, Web and Internet, interest in text and multimedia