DBMS

Definition

- A database management system consists of a collection of ***interrelated data*** and a set of programs to access those data. The collection of data, normally referred as the ***database***, contains information about one particular organization.
- The primary goal of a DBMS is to provide an environment that is both convenient and efficient to use in retrieving and storing database information.

What are the limitations of traditional File Processing System?

1. Data Duplication and Inconsistency.
2. Data Access is very Difficult.
3. Data Isolation.
4. Integrity Problems.
5. Transaction Atomicity Problem.
6. Concurrency Problem.
7. Problem Of Security.
8. Data Dependence.

Instances and Schema

Databases change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an **instance** of the database. The overall design of the database is called the database **schema**. Schemas are changed infrequently, if at all.

Database systems have several schemas, partitioned according to the levels of abstraction. The **physical schema** describes the database design at the physical level, while the **logical schema** describes the database design at the logical level. A database may also have several schemas at the view level, sometimes called **subschemas**, that describe different views of the database.

Database Languages

- A database system provides a ***data-definition language*** to specify the database schema and a ***data-manipulation language*** to express database queries and updates.

Data Definition Language (DDL)

- A database schema is specified by a set of definitions expressed by a special language called a data definition language. The result of compilation of DDL statements is a set of tables that is stored in a special file called ***data dictionary*** or ***data directory***.
- A data dictionary is a file that contains ***metadata***-that is data about data. This file is consulted before actual data are read or modified in the database system

- **Metadata** is “data that provides information about other data". Three distinct types of metadata exist: **descriptive metadata**, **structural metadata**, and **administrative metadata**.
- **Descriptive metadata** describes a resource for purposes such as discovery and identification. It can include elements **such as** title, abstract, author, and keywords.
- **Structural metadata** is metadata about containers of metadata and indicates how compound objects are put together, **for example**, how pages are ordered to form chapters.
- **Administrative metadata** provides information to help manage a resource, **such as** when and how it was created, file type and other technical information, and who can access it.

Data Manipulation Language (DML)

- A data manipulation language DML is a language that enables user to access or manipulate data as organized by the appropriate data model. The types of access are:

✓ Retrieval of information stored in the database.
✓ Insertion of new information into the database.
✓ Deletion of information from the database.
✓ Modification of information stored in the database.

There are basically two types:

- **Procedural DMLs** require a user to specify ***what*** data are needed and ***how*** to get those data.
- **Non Procedural DMLs** require a used to specify ***what*** data are needed ***without*** specifying how to get those data.

Non procedural DMLs are normally easier to learn and use than are procedural DMLs. However, since a user does not have to specify how to get the data, these languages may generate code that is not as efficient as that produced by procedural languages.

A query is a statement requesting the retrieval of information. The portion of a DML that involves information retrieval is called a query language.

View Of Data

- A DBMS is a collection of interrelated files and a set of programs that allow users to access and modify these files. A major purpose of a database system is to provide users with an *abstract view* of Data. That is, the system hides certain details of how the data are stored and maintained.

The Three Level Architecture of DBMS (Data Abstraction)

For the system to be usable, it must retrieve data efficiently. The need for efficiency has led designers to use complex data structures to represent data in the database. Since many database-system users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify users' interactions with the system:

- **Physical Level**
- **Logical Level**
- **View Level**

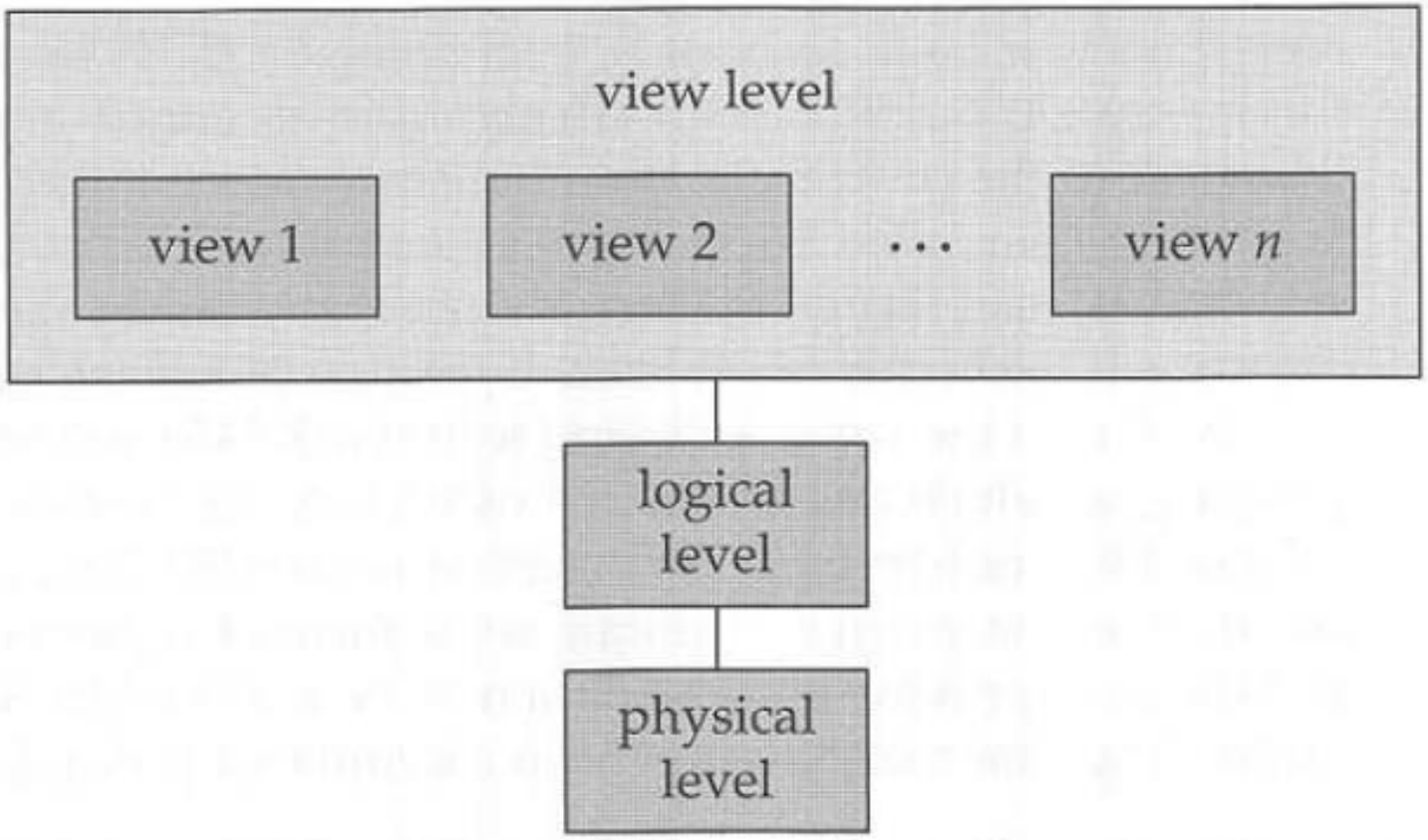


Figure 1.1 The three levels of data abstraction.

Physical Level

- **Physical level.** The lowest level of abstraction describes *how* the data are actually stored. The physical level describes complex low-level data structures in detail.

Physical Level (Internal Level)

- ✓ This is the lowest level of abstraction which describes how the data is actually stored.
- ✓ This also describes data structures and access methods used by database.
- ✓ At this level certain physical components like buffer and control structures are adjusted to provide optimal performance.
- ✓ The internal view is expressed by an internal schema.

Logical Level

- **Logical level.** The next-higher level of abstraction describes *what* data are stored in the database, and what relationships exist among those data. The logical level thus describes the entire database in terms of a small number of relatively simple structures. Although implementation of the simple structures at the logical level may involve complex physical-level structures, the user of the logical level does not need to be aware of this complexity. Database administrators, who must decide what information to keep in the database, use the logical level of abstraction.

Conceptual Level (Global level or logical level)

- ✓ This level of abstraction describes what data is stored in the database and their relationship.
- ✓ The conceptual view is described by the conceptual schema.
- ✓ One conceptual view represents the entire database, that is, there is one conceptual schema per database.
- ✓ This level of abstraction is used by database administrations (DBA's) who decide what information should be kept in the database.
- ✓ This level also includes features that help to maintain database consistency and integrity.
- ✓ Various objects like tables, views, procedure are described at this level.
- ✓ If the underlying hardware is changed then the effects are limited only between physical and the conceptual level.

View Level

- **View level.** The highest level of abstraction describes only part of the entire database. Even though the logical level uses simpler structures, complexity remains because of the variety of information stored in a large database. Many users of the database system do not need all this information; instead, they need to access only a part of the database. The view level of abstraction exists to simplify their interaction with the system. The system may provide many views for the same database.

External Level (User Level or View Level)

- ✓ This is the highest level of abstraction in which user access is provided as per the user requirements.
- ✓ There can be many external views for a given conceptual view.
- ✓ Each external view is represented by an external schema.
- ✓ The external schema contains the method of deriving objects to the external view from the objects in the conceptual view.
- ✓ The external level is closest to the user.
- ✓ The external view is concerned with the way the data is viewed by the individual users.

Mapping

- We know that three view-levels are described by means of three schemas. These schemas are stored in the data dictionary. In DBMS, each user refers only to its own external schema. Hence, the DBMS must transform a request on. A specified external schema into a request against conceptual schema, and then into a request against internal schema to store and retrieve data to and from the database. The process to convert a request (from external level) and the result between view levels is called mapping.
- The mapping defines the correspondence between three view levels. The mapping description is also stored in data dictionary. The DBMS is responsible for mapping between these three types of schemas.

Mapping Between The Views

1. External&ConceptualMapping:- This mapping exists between External view and the conceptual view. This mapping provides ***logical data Independence***.
2. Conceptual&InternalMapping:- This Mapping exists between the conceptual view and the internal view. This mapping provides ***physical data Independence.***

Data Independence

- The very important advantage of DBMS is that it provides **DATA INDEPENDENCE**, that is application programs are independent from the change in the way the data is structured and stored.

Data Independence is the ability to modify a schema definition in one level without affecting a schema definition in the next higher level.

- Three levels of abstraction along with mapping from internal to conceptual and from conceptual to external, provides two levels of data Independence

Logical Data Independence

- It is the ability to change conceptual schema without affecting external schemas. The changes would be absorbed by the mapping between the external and the conceptual levels.
- Modifications at the logical level are needed whenever logical structure of the database altered.
- Logical data independence is provided by the external view of the database and external/conceptual mapping.
- Logical data independence is more difficult to achieve than physical data independence because programs are strongly dependent on logical structure of data.

Physical Data Independence

- It is the ability to change physical schema without affecting existing conceptual schema. Here, physical storage structures or devices could be changed without affecting the conceptual schema.
- The changes would be absorbed by the mapping between the conceptual and the physical level.
- Modifications at the physical level are sometimes necessary to improve performance.
- Physical data Independence is provided by the physical level and the mapping between the conceptual and the internal level.

Data Models

- Underlying the structure of a database is the Data Model:

A collection of conceptual tools for describing data, data relationships and consistency constraints.

- A data model provides a way to describe the design of a database at the physical, logical and view level.

The data models can be classified in 4 different categories:

1. Relational Model
2. The Entity-Relationship Model
3. Object-Based Data Model
4. Semi-structured Data Model

Relational Model

Relational Model. The relational model uses a collection of tables to represent both data and the relationships among those data. Each table has multiple columns, and each column has a unique name. The relational model is an example of a record-based model. Record-based models are so named because the database is structured in fixed-format records of several types. Each table contains records of a particular type. Each record type defines a fixed number of fields, or attributes. The columns of the table correspond to the attributes of the record type. The relational data model is the most widely used data model, and a vast majority of current database systems are based on the relational model.

The Entity-Relationship Model

The Entity-Relationship Model. The entity-relationship (E-R) data model is based on a perception of a real world that consists of a collection of basic objects, called *entities*, and of *relationships* among these objects. An entity is a "thing" or "object" in the real world that is distinguishable from other objects. The entity-relationship model is widely used in database design,

Object-Based Data Model

Object-Based Data Model. The object-oriented data model is another data model that has seen increasing attention. The object-oriented model can be seen as extending the E-R model with notions of encapsulation, methods (functions), and object identity. The object-relational data model combines features of the object-oriented data model and relational data model.

In object-oriented programming (OOP), objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process. In between, each object is made into a generic class of object and even more generic classes are defined so that objects can share models and reuse the class definitions in their code. Each object is an instance of a particular class or subclass with the class's own methods or procedures and data variables. An object is what actually runs in the computer.

Like the ER Model, the object-oriented model is based on a collection of Objects. An object contains values stored in *instance variables* within the objects. An object also contains bodies of code that operate on the object. These bodies of code are called *methods*.

Objects that contains the same types of values and the same methods are grouped together into *classes*. A class may be viewed as a type definition for objects.

Semi-structured Data Model

Semistructured Data Model. The semistructured data model permits the specification of data where individual data items of the same type may have different sets of attributes. This is in contrast to the data models mentioned earlier, where every data item of a particular type must have the same set of attributes. The **Extensible Markup Language (XML)** is widely used to represent semistructured data.

The XML language was initially designed as a way of adding markup information to text documents, but has become important because of its applications in data exchange. XML provides a way to represent data that have nested structure, and furthermore allows a great deal of flexibility in structuring of data, which is important for certain kinds of nontraditional data.

Database Users

- There are four different types of database system users:

- ❑ Naive Users.
- ❑ Application Programmers.
- ❑ Sophisticated Users.
- ❑ Specialized Users.

Naïve Users

Naive users are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously. For example, a bank teller who needs to transfer $50 from account *A* to account *B* invokes a program called *transfer*. This program asks the teller for the amount of money to be transferred, the account from which the money is to be transferred, and the account to which the money is to be transferred.

Application Programmers

Application programmers are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces. **Rapid application development (RAD)** tools are tools that enable an application programmer to construct forms and reports with minimal programming effort.

Sophisticated Users

Sophisticated users interact with the system without writing programs. Instead, they form their requests in a database query language. They submit each such query to a **query processor**, whose function is to break down DML statements into instructions that the storage manager understands. Analysts who submit queries to explore data in the database fall in this category.

Specialized Users

Specialized users are sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework. Among these applications are computer-aided design systems, knowledge-base and expert systems, systems that store data with complex data types (for example, graphics data and audio data), and environment-modeling systems.

DBA
(DataBase Administrator)

- One of the main reasons for using DBMSs is to have central control of both the data and the programs that access those data.
- ***The person who has such central control over the system is called the database administrator (DBA).***
- The functions of the DBA include the following:

Functions Of DBA

- ✓ Schema Definition.
- ✓ Storage and Access-Methods definition.
- ✓ Schema and Physical-organization modification.
- ✓ Granting of authorization for Data Access.
- ✓ Integrity-Constraint Specification.
- ✓ Routine Maintenance.

Schema Definition.

- The DBA creates the original database schema by writing a set of definitions that is translated by the DDL compiler to a set of tables that is stored permanently in the *data dictionary.*

Storage and Access-Methods definition

- The DBA creates appropriate storage structures and access methods by writing a set of definitions, which is translated by the data-storage and data-definition-language compiler.

Schema and Physical-organization modification

- Programmers accomplish the relatively rare modifications either to the database schema or to the description of the physical storage organization by writing a set of definitions that is used by either the DDL compiler or the data-storage and data-definition-language compiler to generate modifications to the appropriate internal system tables.

Granting of authorization for Data Access

- The granting of different types of authorization allows the database administrator to regulate which parts of the database various users can access.
- The authorization information informed is kept in a special system structure that is consulted by the database system whenever access to the data is attempted in the system.

Integrity-Constraint Specification

- The data value stored in the database must satisfy certain consistency constraints.
- For e.g., the number of hours an employee may work in a week may not exceed a specified limit (say, 80 hours).
- Such a constraint must be specified by the database administrator. The integrity constraints are kept in a special system structure that is consulted by the database system whenever an update takes place in the system

Routine Maintenance

- Examples of the database administrator's routine maintenance activities are:

- Periodically backing up the database, either onto discs/tapes or onto remote servers, to prevent loss of data in case of disasters.
- Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
- Monitoring jobs running on the database and ensuring that performance in not degraded by very expensive tasks submitted by some users.

DBMS Architecture
(DBMS System Structure)

- A database system is partitioned into modules that deal with each of the responsibilities of the overall system.
- The functional components of a database system can be broadly divided into ***Query Processor component*** and ***Storage Manager component***.

Query Processor Components Include

- **DML Compiler:** Which translates DML statements in a query language into low-level instructions that the query evaluation engine understands.
- **DDL Interpreter:** Which interprets DDL statements and records them in a set of tables containing *metadata.*
- **Query Evaluation Engine:** Which executes low-level instructions generated by the DML compiler.

Storage manager components

- It provides interface between the low-level data stored in the database and the application programs and queries submitted to the system. This Component includes:

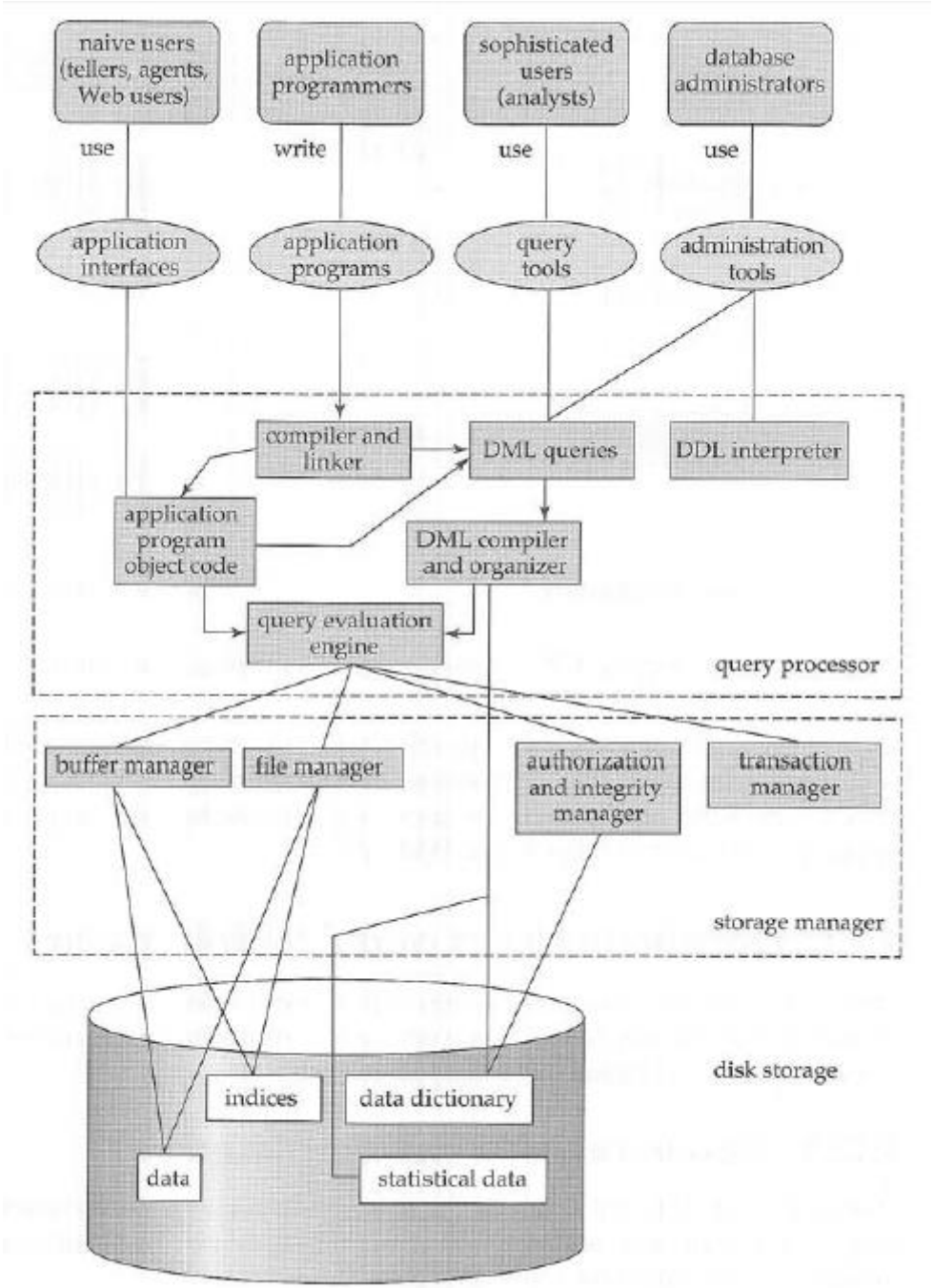
1. **Transaction Manager.**
2. **Buffer Manager.**
3. **File Manager.**

The storage manager components include.

- **Authorization and integrity manager:** Which tests for the satisfaction of integrity constraints and checks the authority of users to access data.
- **Transaction manager:** Which ensures that the database remains in a consistent (correct) state despite system failures and that concurrent transaction executions proceed without conflicting.
- **File manager:** Which manages the allocation of space on disk storage and the data structures used to represent information stored on disk.
- **Buffer manager:** Which is responsible for fetching data from disk storage into main memory, and deciding what data to cache in memory.

In addition, several data structures are required as part of the physical system implementation

- **Data Files:** Which store the database itself.
- **Data Dictionary:** Which stores metadata about the structure of the database.
- **Indices:** Which provide fast access to data items that hold particular values.
- **Statistical Data:** Which stores statistical information about the data in the database. This information is used by the query processor to select efficient ways to execute a query.


naive users (tellers, agents, Web users)
application programmers
sophisticated users (analysts)
database administrators
use
write
use
use
application interfaces
application programs
query tools
administration tools
compiler and linker
DML queries
DDL interpreter
application program object code
DML compiler and organizer
query evaluation engine
query processor
buffer manager
file manager
authorization and integrity manager
transaction manager
storage manager
disk storage
indices
data dictionary
data
statistical data