# Introduction to Database Management System

**1.1 Database Management System**

**Data:**

Data is commonly defined as raw facts or observation, typically about physical phenomena or business transactions.

Example of data would be the marks obtained by students in different subjects.

Data can be in any form:

- numerical (mathematically transformable)
- textual (more correctly, *alphanumerical*)
- graphical (components are known entities; mathematically describable)
- image: fixed (reflection, photograph)
- image: moving (video)
- sound

**Information:**

Information is defined as refined or processed data that has been transformed into meaningful and useful form for specific users. For example, after processing the marks obtained by student it transformed into information, which is meaningful and from which we can decide which student stood first, second and so forth.

Information comes from data and takes the form of table, graphs, diagrams etc.

**Database**:-

A database is a collection of related data. By data, we mean known facts that can be recorded and that have implicit meaning. For e.g. consider the name, telephone numbers and addresses of people you know. You may have recorded this data in an indexed address book, or you may have stored it on hard disk drives, using Microsoft Access or excel. This is a collection of related data with an implicit meaning and hence is a database.

- ✓ A database is designed, built, and populated with data for specific purpose.

**Database management system (DBMS)**

It is a collection of interrelated data and a set of programs to access the data

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is hence a general purpose software system that facilitates the process of defining, constructing, manipulating, and sharing databases among various users and applications.

- **Defining** a database involves specifying the data types, structures, and constraints for the data to be stored in the database.
- **Constructing** the database is the process of storing the data itself on some storage medium that is controlled by the DBMS.
- **Manipulating** a database includes such functions as querying the database to retrieve specific data, updating the database and generating reports from the data.
- **Sharing** a database allows multiple users and programs to access the database concurrently.

Examples of DBMS are MS Access, Oracle, and MYSQL etc.

**Database system**

A database system consists of database (DB) + database Management system (DBMS) + application program.

OR

A database system is just a computerized record keeping system. Database is a repository for a collection of computerized data files. Users of database system can perform a variety of operation on such file

**Database System involve four major components**

- Data
- Hardware
- Software
- User

**Objective of DBMS:**

- ✓ To provide large space or storage for relevant data.
- ✓ To provide easy access to the data for the users.
- ✓ To provide quick response to user request for any data.
- ✓ To remove duplicate (redundant) data.
- ✓ To update the database latest modification immediately.
- ✓ To allow the multiple users to be active at one time.
- ✓ As the organization grows, DBMS allows the growth of the database system.
- ✓ To provide maximum protection to data from any physical damage and unauthorized access.

**Database Applications:**

- **Banking:** all transactions
- **Airlines**: reservations, schedules
- **Universities:** registration, grades
- **Sales**: customers, products, purchases
- **Online retailers**: order tracking, customized recommendations
- **Manufacturing**: production, inventory, orders, supply chain
- **Human resources**: employee records, salaries, tax deductions

***Databases touch all aspects of our lives***

### Some of the Database software applications are:

1. Oracle
2. Sybase
3. Microsoft SQL Server
4. DB2 (IBM)
5. MySQL
6. Postgres
7. DBASE
8. Ms-Access

**1.2 Purpose of Database system**

The early information system (File Processing System) is supported by conventional operating system. Permanents records are stored in various files, and different application programs are written to extract record from, and to add records to, the appropriate files. Before advent of DBMS, organizations typically stored information using such system.

**Drawbacks of using file systems to store data:**

**Data redundancy and inconsistency**

- Multiple file formats, duplication of information in different files

**Difficulty in accessing data**

- Need to write a new program to carry out each new task

**Data isolation** — multiple files and formats

**Integrity problems**

- Integrity constraints (e.g. account balance > 0) become "buried" in program code rather than being stated explicitly
- Hard to add new constraints or change existing ones

**Atomicity of updates**

- Failures may leave database in an inconsistent state with partial updates carried out
- Example: Transfer of funds from one account to another should either complete or not happen at all

**Concurrent access by multiple users**

- Concurrent accessed needed for performance
- Uncontrolled concurrent accesses can lead to inconsistencies
  - Example: Two people reading a balance and updating it at the same time

**Security problems**

- Hard to provide user access to some, but not all, data

***OR***

***Details….***

1. **Data Redundancy & Inconsistency:-**
   Different programmers work on a single project, so various files is created by different programmers at some interval of time. So various files are created in different formats & different programs are written in different programming language.
   Same information is repeated. For ex name & address may appear in saving account file as well as in checking account. This redundancy results in higher storage space & access cost. It also leads to data inconsistency which means that if we change some record in one place the change will not be reflected in all the places. For ex. a changed customer address may be reflected in saving record but not anywhere else.
2. **Difficulty in Accessing data**
   Accessing data from a list is also a difficulty in file system. Suppose we want to see the records of all customers who has a balance less than $10,000, we can either check the list & find the names manually or write an application program .If we write an application program & at some later time,

we need to see the records of customer who have a balance of less than $20,000, then again a new program has to be written.

It means that file processing system do not allow data to be accessed in a convenient manner.

**3. Data Isolation**

As the data is stored in various files, & various files may be stored in different format, writing application program to retrieve the data is difficult.

**3. Integrity Problems**

Sometimes, we need that data stored should satisfy certain constraints as in a bank a minimum deposit should be of $100. Developers enforce these constraints by writing appropriate programs but if later on some new constraint has to be added then it is difficult to change the programs to enforce them.

**4. Atomicity Problems**

Any mechanical or electrical device is subject to failure, and so is the computer system. In this case we have to ensure that data should be restored to a consistent state. For example an amount of $50 has to be transferred from Account A to Account B. Let the amount has been debited from account A but have not been credited to Account B and in the meantime, some failure occurred. So, it will lead to an inconsistent state.

So, we have to adopt a mechanism which ensures that either full transaction should be executed or no transaction should be executed i.e. the fund transfer should be atomic.

**6. Concurrent access Problems**

Many systems allow multiple users to update the data simultaneously. It can also lead the data in an inconsistent state. Suppose a bank account contains a balance of $ 500 & two customers want to withdraw $100 & $50 simultaneously. Both the transaction reads the old balance & withdraw from that old balance which will result in $450 & &400 which is incorrect.

**7. Security Problems**

All the user of database should not be able to access all the data. For example a payroll

Personnel needs to access only that part of data which has information about various employees & are not needed to access information about customer accounts.

***Database systems offer solutions to all the above problems***

**Disadvantages of DBMS**

- ✓ Expensive to buy.
- ✓ Expensive to maintain (need an administrator).
- ✓ High initial investment for hardware ,software and training
- ✓ Complex to understand and implement.
- ✓ Chance of losing the data.
- ✓ Too many rules.
- ✓ Unavailability of trained manpower.
- ✓ Fast changing technology.

**1.3 View of Data**

A database contains a no. of files & certain programs to access & modify these files. But the actual data is not shown to the user, the system hides actual details of how data is stored & maintained.

**Data Abstraction**

It is mechanism to hide complexity of database in database system. It allows database system to provide abstract view of database user. It hides how data are actually stored and maintain in database. Data abstraction simplifies users' interactions with the system.

There are three level of data abstraction

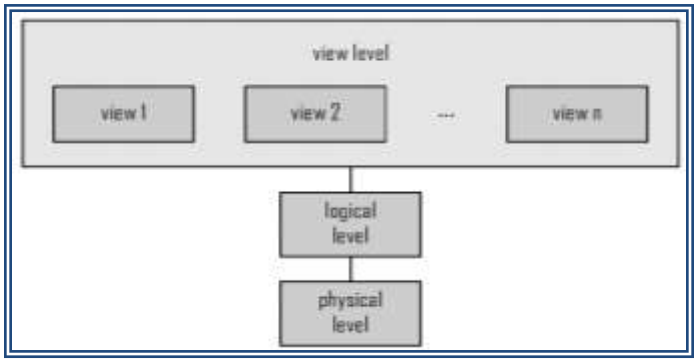



Fig: Three level of data abstraction

**Physical level:** It is the lowest level of abstraction describes how a record (e.g., customer) is stored. It describes complex low level data structures in detail.

**Logical level:** The next higher level of abstraction describes what data are stored in database, and the relationships exist among the data. It describes entire database relatively in a simple structure. The user in logical level needs not to aware the complexity of physical level structure. This level is used by database administrators, who must decide what information is to be kept in the database

e.g. **type** *customer* = **record**

*customer_id*: string;
*customer_name*: string;
*customer street*: string;
*customer_city* : integer;

**end**;

**View level:** The highest level of abstraction describes only part of entire database. It simplifies interaction with the system. It allows database system to provide many views for the same database. Application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

**1.4 Data Models**

A collection of tools for describing Data, Data relationships, Data semantics and Data constraints. It describes the underlying structure of database. There are several data models which can be group into three categories

**1.4.1 Object-based logical Models:**

Object based logical model describe data at the logical and view levels. It has flexible structuring capabilities .It allows to specify data constraints explicitly. Some widely used models are:

- **Entity-Relationship data model (mainly for database design)**

  The E-R data model is based on a perception of real world that consists of a collection of basic objects, called entities and of relationships among these objects.

  Components of E-R diagram are as follows.
  **Rectangle**: Which represent entity sets.
  **Ellipses**: Which represent attributes.
  **Diamonds:** Which represent relationship among entity sets.
  **Lines**: Which link attributes to entity sets and entity sets to relationships.

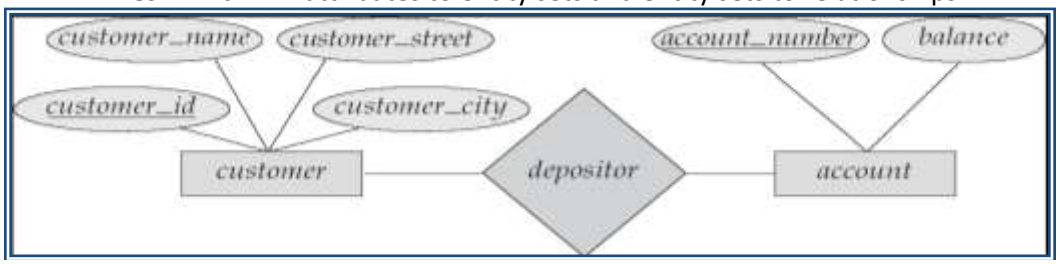



**Fig: Sample E-R diagram**

**Advantages**

- ✓ Exceptional conceptual simplicity
- ✓ Visual representation
- ✓ Effective communication tool
- ✓ Integrated with the relational database model

**Disadvantages**

- ✓ Limited constraint representation
- ✓ Limited relationship representation
- ✓ No data manipulation language
- ✓ Loss of information content

- **Object-oriented model (Object-oriented and Object-relational)**

  This model is based on a collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies

of code are called methods. Objects that contain the same types of values and the same methods are grouped together into classes.

**Advantages**

- ✓ Adds semantic content
- ✓ Visual presentation includes semantic content
- ✓ Database integrity
- ✓ Both structural and data independence

**Disadvantages**

- ✓ Complex navigational data access
- ✓ High system overhead slows transactions

### 1.4.2 Record -based logical Models:

It also describes data at the logical and view levels. But it describes logical structure of database in more detail for implementation point of view. It describes database structure in terms of fixed-format records of different types. Each table contains records of particular type and each record type defines fixed number of fields (attributes).Each field is usually of fixed length.

Some widely used models are:

- **Relational Model**

In a relational data model, the data elements are organized in the form of multiple tables with rows and columns. Each table of the database is stored as a separate file. Each table column represents a data field and each row represents a data record. A record is also known as a tuple. The data in one table is related to a data in another table with a common field.

| customer_id | customer_name | customer_street | customer_city |
|---|---|---|---|
| 192-83-7465 | Johnson | 12 Alma St. | Palo Alto |
| 677-89-9011 | Hayes | 3 Main St. | Harrison |
| 182-73-6091 | Turner | 123 Putnam Ave. | Stamford |
| 321-12-3123 | Jones | 100 Main St. | Harrison |
| 336-66-9999 | Lindsay | 175 Park Ave. | Pittsfield |
| 019-28-3746 | Smith | 72 North St. | Rye |

(a) The *customer* table

| account_number | balance |
|---|---|
| A-101 | 500 |
| A-215 | 700 |
| A-102 | 400 |
| A-305 | 350 |
| A-201 | 900 |
| A-217 | 750 |
| A-222 | 700 |

(b) The *account* table

| customer_id | account_number |
|---|---|
| 192-83-7465 | A-101 |
| 192-83-7465 | A-201 |
| 019-28-3746 | A-215 |
| 677-89-9011 | A-102 |
| 182-73-6091 | A-305 |
| 321-12-3123 | A-217 |
| 336-66-9999 | A-222 |
| 019-28-3746 | A-201 |

(c) The *depositor* table

Fig. An example of relational database.

The relational database model provides greater flexibility of data organization and future enhancements in the database as compared to the hierarchical and network database models. If a new data is to be added to an existing relational database, it is not necessary to redesign the database. Rather new table containing the new data can be added to the database and then these tables can be related to the existing tables with common key fields.

Advantages:-

- ✓ Very less redundancy.
- ✓ Normalization of database is possible
- ✓ Quick database processing is possible
- ✓ Since one table will link with another with common key field, the rule implemented in one table can easily be implemented in another.

Disadvantages:-

- ✓ It is more complex than other models.
- ✓ Too many rules makes database non - user friendly.

**Hierarchical Data Model:** In a hierarchical data model, the data elements are linked in the form of an inverted tree structure with the root at the top and the branches formed below. Below the single root data element are subordinate elements, each of which, in turn, has one or more other elements. There is a parent child relationship among the data elements of a hierarchical database. There may be many child elements under each parent element, but there can be only one parent element for any child element. The branches in the tree are not connected.

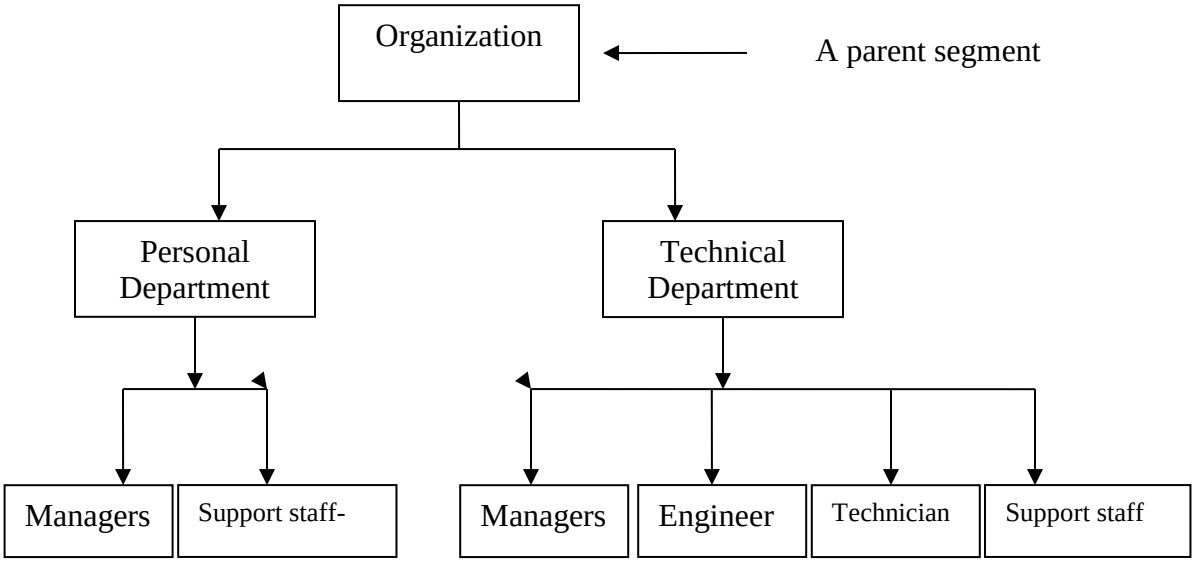



Fig. An example of hierarchical

Hierarchical data model is used in several database applications because the data elements of many applications can be neatly organized in the form of hierarchical tree structure. The main limitation of this structure is that it does not support flexible data access, because data can be accessed only by following the path down the tree structure.

Advantages:

- ➢ It is the easiest model of database.
- ➢ It is secure model as nobody can modify the child without consulting to its parent.
- ➢ Searching is fast and easy is parent is known.
- ➢ Very efficient in handling one to many relationship.

Disadvantages:-

- ➢ It is old fashion, outdated database model
- ➢ Modification and addition of child without consulting its parent is impossible.
- ➢ Cannot handle many to many relationships.
- ➢ Increase redundancy.
- ➢ It does not support flexible data access, because data can be accessed only by following the path down the tree structure.

**Network Data Model:** A network data model is an extension of the hierarchical database structure. In this model also, the data elements of a database are organized in the form of parent-child relationships and all types of relationships among the data elements must be determined when the database is first designed. In a network database, a child data element can have more than one parent element or no parent at all. Moreover, in this type of database, the database management system permits the extraction of the needed information from any data element in the database structure, instead of starting from the root data element.

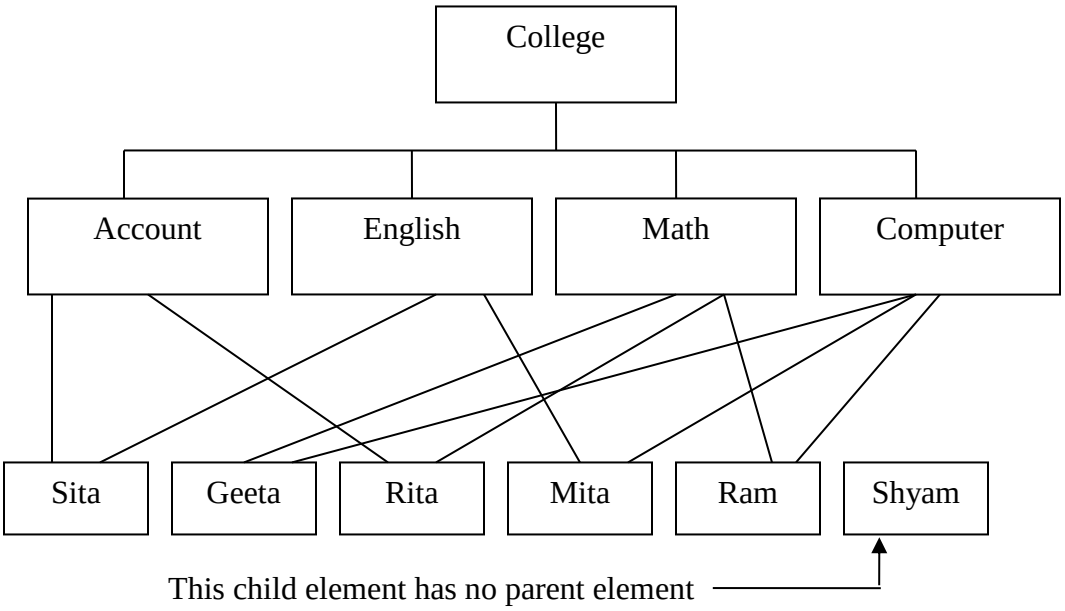



Fig. An example of a network database

Advantages:-

- More flexible than hierarchical database because it accept many to many relationship.
- Searching is faster because of multidirectional pointers.
- Promotes database integrity
- Data independence

Disadvantages:-

- Complex type of database model.
- Less secure than hierarchical as it is open to all.
- Need long program to handle the relationship.

### 1.4.3 Physical data Models:

Physical data models are used to describe data at the lowest level. Two widely used physical models are: Unifying model and frame-memory model.

### 1.5 Instances and Schemas

**Instance** – Database change over time as information is inserted, deleted and updated. The collection of information stored in the database at particular moment is called an instance of the database.

**Schema** – The overall design (logical structure) of the database is called Schema

Example: The database consists of information about a set of customers and accounts and the relationship between them)

Database system has several schemas, partitioned according to the levels of abstraction.

**Physical schema**: database design at the physical level

**Logical schema**: database design at the logical level

**Subschema:** at highest (View) level

**1.6 Data Independence**

The ability to modify a scheme definition in one level without affecting a scheme definition in a higher level is called **data independence**. There are two level of data independence.

**Physical Data Independence:**

- It is the ability to modify the physical scheme without causing application programs to be rewritten
- Modifications at this level necessary to improve the performance

**Logical Data Independence**

- It is the ability to modify the conceptual /Logical schema without causing application programs to be rewritten or any change to external schema.
- Usually done when logical structure of database is altered such as adding or removing new entities, attributes or relationship.

**1.7 Database Administrator**

Centralized control of the database is exerted by a person or group of persons. This person or group of persons is called database administrator (DBA). They are the users who are most familiar with the database and are responsible for creating, modifying its three levels. The DBA specifies the external view of various users and applications. They coordinate all the activities of the database system; the database administrator has a good understanding of the enterprise's information resources and needs.

**Database administrator's duties include:**

- **Schema definition**:-The Scheme Definition creates the original database scheme by executing a set of data definition statements in the DDL.
- **Storage structure and access method definition**:- To write a set of definitions translated by the data storage and definition language complier.
- **Schema and physical organization modification**:- To carry out changes to the scheme as changes needed in an organization. He alters the physical organization to improve performance.
- **Granting user authority to access the database**:- To grant different types of authorization for various users to access data
- **Routine maintenance**

- ✓ Monitoring performance and responding to changes in requirements
- ✓ Periodically backing up the database so that if a data entry error, program bug or hardware failure occurs, the DBA can bring the database back in time before damage was done. (Recoverability)
- ✓ Availability:- Ensuring maximum uptime. Ensuring that enough free space is available for normal operation.

**1.8 Database Users**

A primary goal of a database system is to provide an environment for retrieving information from and storing new information into the database. There are four different types of database-system users, differentiated by the way they expect to interact with the system.

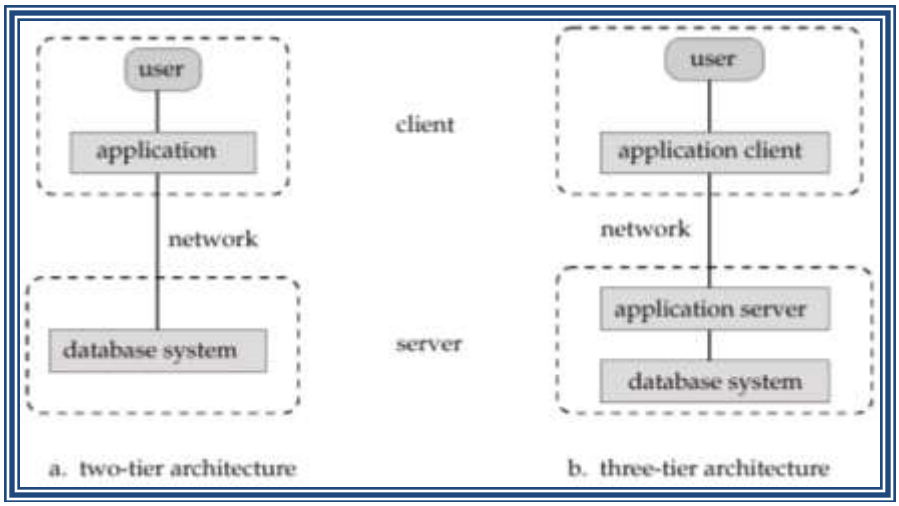
**Application programmers** are computer professionals who interact with the system through DML calls, which are embedded in a program written in a host language (for example, COBOL, PL/I, Pascal, C). These programs are commonly referred to as application programs. Examples in a banking system include programs that generate payroll checks that debit accounts, that credit accounts, or that transfers funds between accounts. They write application programs usually using 4GL (fourth generation language). Even they use of RAD (Rapid Application Development) tools:- to generate forms and reports easily

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

**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 (CAD), knowledge base and expert systems, systems that store data with complex data types (for example, graphics data and audio data) and environment-modeling systems.

**Naive users** are unsophisticated users who interact with the system by invoking one of the permanent application programs (ready – made programs) that have been written previously. For example, a bank teller who needs to transfer Rs 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. Other examples are people accessing database over the web, bank tellers, clerical staff, and Automatic teller machine. They interact with the system through the interface that are given to them.

**1.9 Application Architecture**




**1.9.1 Two-tier architecture**:

The *2-tier model* is more simple, but more limited, than a 3-tier model, and often includes

- ✓ The database management system (DBMS)
- ✓ The main application software, including GUI

Here, the entire application is generally run on the client machine

In some contexts, the 2-tier model is also known as the *client-server model*, where the server can be something other than a database

E.g. client programs using ODBC/JDBC to communicate with a database

**Advantages of Two-Tier Approach**

- ✓ Clients do not have to be as powerful
- ✓ Greatly reduces data traffic on the network
- ✓ Improved data integrity since it is all processed centrally

**Limitations**

- ✓ Performance deteriorates if number of users is greater than 100
- ✓ Restricted flexibility and choice of DBMS, since data language used in server is proprietary to each vendor
- ✓ Limited functionality in moving program functionality across servers

**1.9.2 Three-tier architecture**:

A ***three-tier*** *architecture* is one which has a *client tier*, a *middle tier*, and a *database tier*.

- The *database tier* manages the database
- The *middle tier* contains most of the logic and communicates between the other tiers
- The *client tier* is the interface between the user and the system

E.g. web-based applications and applications built using "middleware"

**Advantages of Three-Tier Architectures**

- ✓ Scalability
- ✓ Technological flexibility
- ✓ Long-term cost reduction
- ✓ Better match of systems to business needs
- ✓ Improved customer service
- ✓ Reduced risk

**Challenges of Three-tier Architectures**

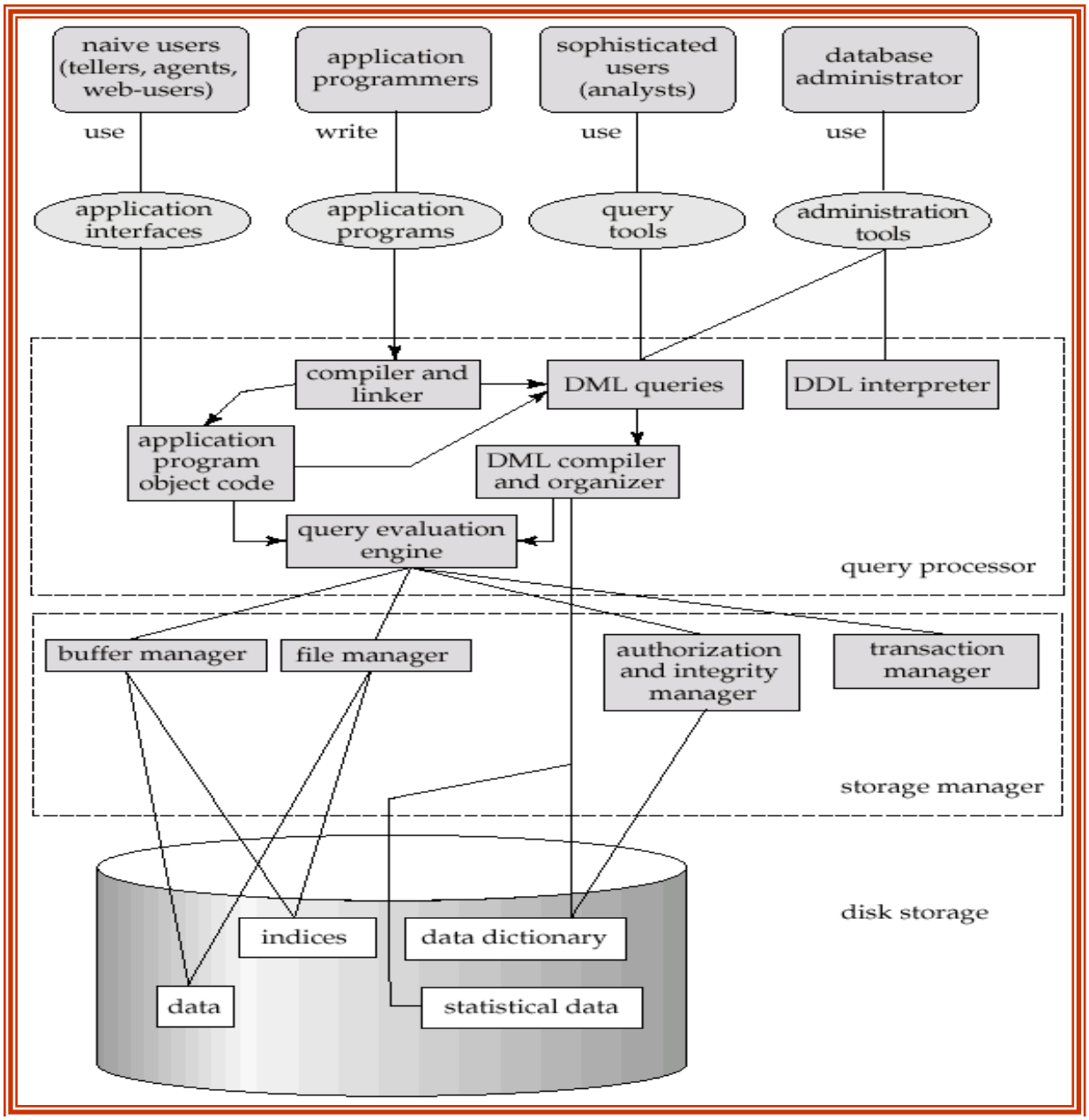
- ✓ High short-term costs
- ✓ Tools and training
- ✓ Experience
- ✓ Incompatible standards
- ✓ Lack of compatible end-user tools

**1.9.3 N-tier architecture**:

- ✓ **An *n-tier architecture*** is one which has *n* tiers, usually including a *database tier*, a *client tier*, and *n-2* tiers in between.
- ✓ In general an *n-tier model* will have
  - The database management system (DBMS)
  - (*n*-2) application layers
  - A GUI (thin or thick)

**1.10 Overall Database System Structure and Components**

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 the storage manager and the query processor components.




**Storage Management**

- Storage manager is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- Storage management is important because database typically requires lots of storage space (gigabytes, terabytes). This is to be stored in dicks and hence data are moved between disk storage and main memory as needed. This is ensure by storage management.
- The storage manager is responsible to the following tasks:
    - interaction with the file manager
    - Efficient storing, retrieving and updating of data and hence improve the performance.

**The storage management component includes:**

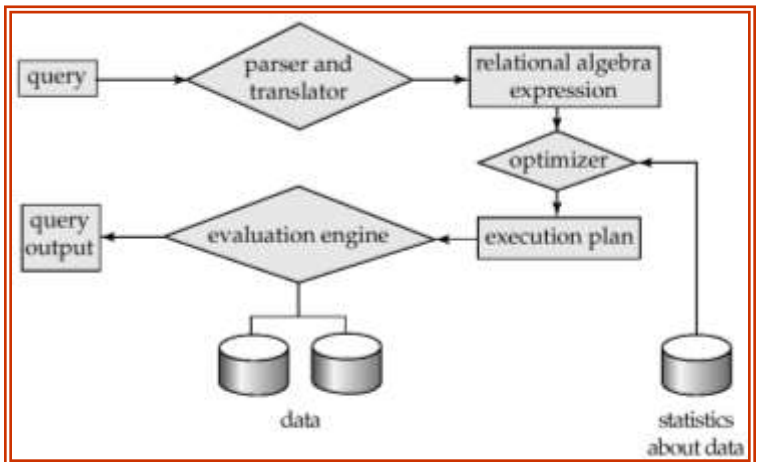
- **Authorization and integrity manager**:- Which check that updates do not violate integrity constraints and the authority of users to access data.
- **Transaction manager**:- which ensures that the database remains in a consistent (correct) state despite system failures, and that concurrent transaction executions proceed without confliction.
- **File manager**:- which manages the allocation of space on dick storage and 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 main memory.

**Storage management work with**

- **Data files**:- which store the database itself
- **Data dictionary**:- which store metadata
- **Indices**:- which provide fast access to data items that hold particular values

**Query Processing**

1. Parsing and translation
2. Optimization
3. Evaluation




- Alternative ways of evaluating a given query
  - ✓ Equivalent expressions
  - ✓ Different algorithms for each operation
- Cost difference between a good and a bad way of evaluating a query can be enormous
- Need to estimate the cost of operations
  - ✓ Depends critically on statistical information about relations which the database must maintain
  - ✓ Need to estimate statistics for intermediate results to compute cost of complex expressions

**Query Processor**:- translates updates and queries written in a non-procedural languages into an efficient sequence of operations at the physical level. Its main responsibility is to facilitate accessing the data
The query processor components include:

1. **DDL Interpreter**:- interpreters DDL statements and records the definitions in data dictionary.
2. **DML Compiler**:- Translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.

A query can usually be translated into any of a number of alternative evaluations plans that all give the same result. The DML compiler also performs query optimization, that is, it picks the lowest cost evaluation plan from among the alternatives.

3. **Query Evaluation engine**:- Executes low-level instructions generated by the DML compiler

### 1.11 Database Language

Database provides two languages, one to specify the database scheme, other to express queries and updates.

**DDL (Data Definition Language)**

- A data base schema is specifies by a set of definitions expressed by a special language called DDL.

- Used to specify a database scheme at logical level to describe details of data
  - DDL statements refer to create, drop and alter
  - *Create:-* To make a new database, table, index, or stored query

```
Create table student
(
    Rollno number (9) primary key,
    Name varchar (20),
    Section varchar (3),
    Faculty varchar (5)
);
```

  - *Drop:-* To destroy an existing database, table, index, or view.

```
Drop table student;
```

  - *Alter:-* To modify an existing database object.

```
Alter table student ass (address varchar (30));
```

- ✓ DDL compiler generates a set of tables stored in a *data dictionary*
- ✓ Data dictionary contains metadata (i.e., data about data)
  - ➢ Database schema
  - ➢ Data *storage and definition* language
    - ▸ Specifies the storage structure and access methods used
  - ➢ Integrity constraints
    - ▸ Domain constraints
    - ▸ Referential integrity (**references** constraint in SQL)
    - ▸ Assertions
  - ➢ Authorization

**DML (Data Manipulation Language)**

- ✓ Language for accessing and manipulating the data organized by the appropriate data model.
- ✓ Data manipulation means:
    - ✓ The retrieval of information stored in the database.
    - ✓ The insertion of new information into the database.
    - ✓ The deletion of information from the database
    - ✓ The modification of information stored in the database.
- ✓ DML also known as query language
- ✓ Two classes of languages
    - ✓ **Procedural** – user specifies what data is required and how to get those data
    - ✓ **Declarative (nonprocedural)** – user specifies what data is required without specifying how to get those data
- ✓ SQL is the most widely used query language

**1.12 Transaction Concept**

- ✓ A ***transaction*** is a *unit* of program execution that accesses and possibly updates various data items.
- ✓ A transaction must see a consistent database.
- ✓ During transaction execution the database may be inconsistent.
- ✓ When the transaction is committed, the database must be consistent.
- ✓ Two main issues to deal with:
    - ★ Failures of various kinds, such as hardware failures and system crashes
    - ★ Concurrent execution of multiple transactions

**ACID Properties**

To preserve integrity of data, the database system must ensure:

- **Atomicity.** Either all operations of the transaction are properly reflected in the database or none are.
- **Consistency.** Execution of a transaction in isolation preserves the consistency of the database.
- **Isolation.** Although multiple transactions may execute concurrently, each transaction must be unaware of other concurrently executing transactions. Intermediate transaction results must be hidden from other concurrently executed transactions.
    - ★ That is, for every pair of transactions $T_i$ and $T_j$, it appears to $T_i$ that either $T_j$, finished execution before $T_i$ started, or $T_j$ started execution after $T_i$ finished.
- **Durability.** After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures.

**Example of Fund Transfer**

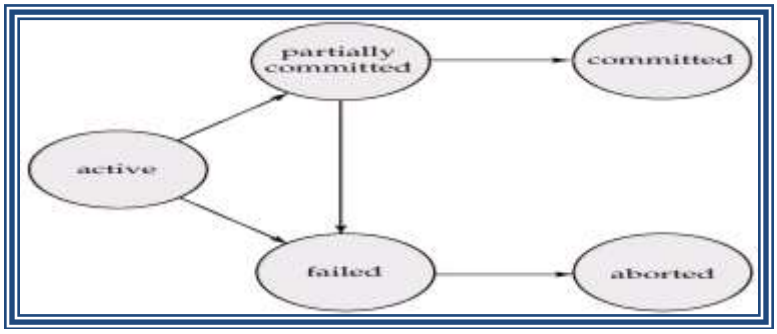
***Transaction to transfer $50 from account A to account B:***

1. **read**(*A*)
2. *A* := *A* – 50
3. **write**(*A*)

4. **read**(*B*)
5. *B* := *B* + 50
6. **write**(*B)*

- **Consistency requirement** – the sum of *A* and *B* is unchanged by the execution of the transaction.
- **Atomicity requirement** — if the transaction fails after step 3 and before step 6, the system should ensure that its updates are not reflected in the database, else an inconsistency will result.
- **Durability requirement** — once the user has been notified that the transaction has completed (i.e., the transfer of the $50 has taken place), the updates to the database by the transaction must persist despite failures.
- **Isolation requirement** — if between steps 3 and 6, another transaction is allowed to access the partially updated database, it will see an inconsistent database (the sum *A + B* will be less than it should be). Can be ensured trivially by running transactions ***serially,*** that is one after the other. However, executing multiple transactions concurrently has significant benefits, as we will see.

**Transaction State**




- **Active,** the initial state; the transaction stays in this state while it is executing
- **Partially committed,** after the final statement has been executed.
- **Failed,** after the discovery that normal execution can no longer proceed.
- **Aborted,** after the transaction has been rolled back and the database restored to its state prior to the start of the transaction. Two options after it has been aborted:
  - ★ restart the transaction – only if no internal logical error
  - ★ kill the transaction
- **Committed,** after *successful completion*.

Short question and answer

1. What is database?

A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

2. What is **DBMS**?
It is a collection of programs that enables user to create and maintain a database. In other words it is general-purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications.

3. What is a Database system?
The database and **DBMS** software together is called as Database system.

4. Advantages of **DBMS**?

- ✓ Redundancy is controlled.
- ✓ Unauthorized access is restricted.
- ✓ Providing multiple user interfaces.
- ✓ Enforcing integrity constraints.
- ✓ Providing backup and recovery.

5. Disadvantage in File Processing System?

- ✓ Data redundancy & inconsistency.
- ✓ Difficult in accessing data.
- ✓ Data isolation.
- ✓ Data integrity.
- ✓ Concurrent access is not possible.
- ✓ Security Problems.

6. Describe the three levels of data abstraction?
There are three levels of abstraction:
**Physical level**: The lowest level of abstraction describes how data are stored.
**Logical level**: The next higher level of abstraction, describes what data are stored in database and what relationship among those data.
**View level:** The highest level of abstraction describes only part of entire database.

7. What is Data Model?
A collection of conceptual tools for describing data, data relationships data semantics and constraints.

8. What is E-R model?
This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

9. What is Object Oriented model?
This model is based on collection of objects. An object contains values stored in instance variables with in the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

10. What is DDL (Data Definition Language)?
A data base schema is specifies by a set of definitions expressed by a special language called DDL.

11. What is DML (Data Manipulation Language)?
This language that enable user to access or manipulate data as organized by appropriate data model.
**Procedural DML** or Low level: DML requires a user to specify what data are needed and how to get those data.
**Non-Procedural DML** or High level: DML requires a user to specify what data are needed without specifying how to get those data.

12. What is DML Compiler?
It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

13. What is Query evaluation engine?
It executes low-level instruction generated by compiler.

14. What is DDL Interpreter?
It interprets DDL statements and records them in tables containing metadata.

*15. What is E-R model?*

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

*16. What is Object Oriented model?*

This model is based on collection of objects. An object contains values stored in instance variables with in the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

*17. What is an Entity?*

It is a 'thing' in the real world with an independent existence.

*18. What is an Entity type?*

It is a collection (set) of entities that have same attributes.

*19. What is an Entity set?*

It is a collection of all entities of particular entity type in the database.