Database Management Systems
UNIT-I

Two marks questions:

1. What is database? Define DBMS

A **database** is a collection of inter-related data which contains the information of an enterprise. It is obtained by collecting the data from all sources of the organization. Database can be shared, integrated and managed with other computer applications.

A **Database Management System (DBMS)** is a collection of interrelated data and set of programs to access those data. The goal of database is to facilitate an interface which is suitable for the users to access and store data.

2. What do you mean by defining, constructing, manipulating and sharing a database?

Defining a database involves specifying the data types, structures, and constraints of the data to be stored in the database. The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary called meta-data.
Constructing the database is the process of storing the data on some storage medium that is controlled by the DBMS.
Manipulating a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
Sharing a database allows multiple users and programs to access the database simultaneously.

3. Define data abstraction. Give any two levels of abstraction

The characteristic that allows program-data independence and program-operation independence is called **data abstraction**.

4. Define query and transaction

A **query** typically causes some data to be retrieved.
A **transaction** may cause some data to be read and some data to be written into the database.
A transaction is an *executing program* or *process* that includes one or more database accesses, such as reading or updating of database records. Each transaction is supposed to execute a logically correct database access if executed in its entirety without interference from other transactions.

5. List the phases of database application

1. **requirements specification and analysis:** In this phase, requirements are documented in detail .
2. **conceptual design :** Requirements are represented and manipulated using some computerized tools so that it can be easily maintained, modified, and transformed into a database implementation.
3. **logical design :** Conceptual design is then translated in this phase so that it can be expressed in a data model and implemented in a commercial DBMS.
4. **physical design:** In this phase further specifications are provided for storing and accessing the database. The database design is implemented, populated with actual data, and continuously maintained to reflect the state of the miniworld.

6. List the characteristics of Database approach

- Self-describing nature of a database system
- Insulation between programs and data, and data abstraction
- Support of multiple views of the data
- Sharing of data and multiuser transaction processing

7. Give the disadvantages of file processing system.

In traditional file processing, the structure of data files is embedded in the application programs, so any changes to the structure of a file may require changing all programs that access that file.

8. Define Metadata. Give an example

Meta-data is a database catalog or dictionary in which the database definition or descriptive information is stored by the DBMS.

Eg: Descriptions of schema constructs and constraints.

9. What do you mean by program data independence and program-operation independence?

DBMS access programs do not require such changes in most cases. The structure of data files is stored in the DBMS catalog separately from the access programs. This property is called **program-data independence.**

User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented. This property is called **program-operation independence**.

10. Define Data Model. Give the categories of data model

A **data model** is a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve this abstraction.

Structure of a database refers to the data types, relationships, and constraints that apply to the data. Data models also include a set of basic operations for specifying retrievals and updates on the database.

The categories of data model are:

1. High-level or conceptual data models
2. Representational (or implementation) data models
3. low-level or physical data models

11. Mention the following

12 actors on the scene

Actors on the scene are people whose jobs involve the day-to-day use of a large database.

13. workers behind the scene

Workers behind the scene are people who work to maintain the database system environment but who are not actively interested in the database contents as part of their daily job.

14. What is redundancy? Which problems may arise while storing redundant data?

Problems due to redundancy are:

1. there is the need to perform a single logical update multiple time. This leads to duplication of effort.

2. storage space is wasted when the same data is stored repeatedly, and this problem may be serious for large databases.
3. files that represent the same data may become inconsistent. This may happen because an update is applied to some of the files but not to others

15. List the limitations/disadvantages of DBMS

1. High initial investment in hardware, software, and training
2. The generality that a DBMS provides for defining and processing data
3. Overhead for providing security, concurrency control, recovery, and integrity functions

16. Who is a DBA? What are the responsibilities of a DBA?

Database administrator (DBA) is a chief administrator who oversees and manages database, DBMS and related software. The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed. The DBA is accountable for problems such as security breaches and poor system response time.

17. Who are database designers? What are their responsibilities?

Database designers are responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data.

Their responsibilities are:

1. to communicate with all prospective database users in order to understand their requirements and to create a design that meets these requirements.
2. interact with each potential group of users and develop views of the database that meet the data and processing requirements of these groups.
3. analyze each view and integrate with the views of other user groups.

18. What is the responsibility of backup and recovery subsystem?

The backup and recovery subsystem of the DBMS is responsible for recovery.
For example, if the computer system fails in the middle of a complex update transaction, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the transaction started executing.
Disk backup is also necessary in case of a catastrophic disk failure.

19. What are the advantages of using a DBMS?

1. Controlling Redundancy
2. Restricting Unauthorized Access
3. Providing Persistent Storage for Program Objects
4. Providing Storage Structures and Search Techniques for Efficient Query Processing
5. Providing Backup and Recovery
6. Providing Multiple User Interfaces
7. Representing Complex Relationships among Data
8. Enforcing Integrity Constraints
9. Permitting Inferencing and Actions Using Rules and Triggers

20. What does the isolation and atomicity property ensure?

The **isolation property** ensures that each transaction appears to execute in isolation from other transactions, even though hundreds of transactions may be executing concurrently.

The **atomicity property** ensures that either all the database operations in a transaction are executed or none are.

21. Define Entity, Attribute and relationship

An **entity** represents a real-world object or concept, such as an employee or a project from the miniworld that is described in the database.

An **attribute** represents some property of interest that further describes an entity, such as the employee's name or salary.

A **relationship** among two or more entities represents an association among the entities, for example, a works-on relationship between an employee and a project.

22. What is database schema, Schema diagram?

The description of a database is called **the database schema,** which is specified during database design and is not expected to change frequently.

A displayed schema is called a **schema diagram**.

23. What do you mean by database state? When do you say the database state is valid?

The data in the database at a particular moment in time is called a **database state or snapshot**. It is also called the current set of occurrences or instances in the database.

A database state is **valid** if it satisfies the structure and constraints specified in the schema.

24. Define the terms Internal, external and conceptual schema (any twos)

Internal schema describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

Conceptual schema describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints.

External schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

25. What do you mean by Logical Data Independence?

Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs.

26. What do you mean by physical data Independence?

Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well.

27. What is the purpose of storage manager?

Stored data manager is one of the main components of DBMS responsible for the below tasks.

1. It utilizes operation system services in order to perform the low-level / input operations between disk and main memory.

2. Maintains consistency and integrity using constraints. It does not allow entering any duplicate row in the database and inserting/updating if data has child entry.
3. Converts requests from query optimizer to machine language code and makes actual calls to the database.

28. List the functions of runtime database processor

1. It performs various functions like executing privileged commands, query plans, and canned transactions with runtime parameters.
2. Handles data transfer and management of buffer in the main memory.
3. It also works with the system catalog and updates the statistics if required.
4. Also communicates with stored data manager when necessary.
5. It consists of a submodule called integrity checker where it checks for all the keys like primary, unique, and foreign key.

29. Discuss about Data Definition language?

Data Definition language (DDL), is used by the DBA and by database designers to define both schemas. The DBMS will have a DDL compiler whose function is to process DDL statements in order to identify descriptions of the schema constructs and to store the schema description in the DBMS catalog.

30. Discuss about Data Manipulation language?

The DBMS provides a set of operations or a language called the data manipulation language (DML) for manipulating the database.
Typical manipulations include retrieval, insertion, deletion, and modification of the data.

31. List the functions of Database System Utilities

a. Loading.
b. Backup.
c. Database storage reorganization
d. Performance monitoring

32. Define a) meta-data b) data dictionary

The information stored in the catalog is called **meta-data**, and it describes the structure of the primary database.

Data dictionary (or data repository) is a tool that stores catalog information about schemas and constraints, and other information, such as design decisions, usage standards, application program descriptions, and user information.

33. List various types of attributes?

1. Composite versus Simple (Atomic) Attributes.
2. Single-Valued versus Multivalued Attributes
3. Stored versus Derived Attributes.
4. NULL Values
5. Complex Attributes

34. Explain data model and list the types of data model used?

A **data model** is a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve this abstraction.

Structure of a database refers to the data types, relationships, and constraints that apply to the data. Data models also include a set of basic operations for specifying retrievals and updates on the database.

The categories of data model are:

1. **High-level or conceptual data models** : They provide concepts that are close to the way many users perceive data.
2. **Representational (or implementation) data models:** They provide concepts that may be easily understood by end users but that are not too far removed from the way data is organized in computer storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly.
3. **Low-level or physical data models**: They provide concepts that describe the details of how data is stored on the computer storage media, typically magnetic disks. Concepts provided by physical data models are generally meant for computer specialists, not for end users.

35. Define Single-Valued versus Multivalued Attributes, give example

Attributes which have a single value for a particular entity are called **single-valued attributes**. **For example**, Age is a single-valued attribute of a person.

Attributes which have multiple values for a particular entity are called **multivalued-valued**. A multivalued attribute may have lower and upper bounds to constrain the number of values allowed for each individual entity. They are described using double oval symbol.
For Example, color of car is a multivalued attribute

36. Define Stored versus Derived Attributes, give example

Stored attribute is an attribute whose value cannot be derived from the values of the attributes.

Eg: Date of Birth of a person

Derived attributes do not exist physically in the database. Their values are derived from other attributes present in the system. Derived attributes are represented using dotted oval.

Eg: Age can be derived from Date of Birth attribute.

37. Define Composite versus Simple (Atomic) Attributes, give example

Simple values are atomic values which cannot be divided further.

Eg: Weight, salary attributes of employee entity cannot be divided further.

Composite Attributes can be divided into further parts.

Eg: Employee name may be further divided into first name and last name.

38. Composite versus Simple (Atomic) Attributes (Repeat)

39. Define the terms i) Entity set ii) Relationship set

The collection of all entities of a particular entity type in the database at any point in time is called an **entity set** or entity collection. It is also referred to as extension of entity type.

An **entity type** defines a collection (or set) of entities that have the same attributes. Each entity type in the database is described by its name and attributes.

40. Define the terms i) Key attribute ii) Value set(Domain)

An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity set. Such an attribute is called a **key attribute**, and its values can be used to identify each entity uniquely.

Each simple attribute of an entity type is associated with **a value set** (or domain of values). A value set specifies the set of values that may be assigned to that attribute for each individual entity.

41. What do you mean by relationship type and relationship instances?

A relationship type R among n entity types $E_1, E_2, \ldots, E_n$ defines a set of associations.

The relationship set R is a set of **relationship instances** r_i, where each r_i associates n individual entities $(e_1, e_2, \ldots, e_n)$, and each entity e_j in r_i is a member of entity set Ej, $1 \leq j \leq n$.

42. Define "degree of relationship type"

The degree of a relationship type is the number of participating entity types.
A relationship type of degree two is called **binary**, and one of degree three is called **ternary.**

43. What do you mean by role names in the relationship? Give example

Each entity type that participates in a relationship type plays a particular role in the relationship. The **role name** signifies the role that a participating entity from the entity type plays in each relationship instance, and it helps to explain what the relationship means.

For example, in the WORKS_FOR relationship type, EMPLOYEE plays the role of *employee* or *worker* and DEPARTMENT plays the role of *department* or *employer*.

44. What does the cardinality ratio specify? Which are the possible cardinality ratios for binary relationship types

The **cardinality ratio** for a binary relationship specifies the *maximum* number of relationship instances that an entity can participate in.

The possible cardinality ratios for binary relationship types are 1:1, 1:N, N:1,and M:N.

45. Define the terms i) Total participation ii) Partial Participation

When each entity in an entity set participates in a relation, it is called Total Participation.

When all entities in the given entity set do not participate in a relation, it is called Partial Participation

46. Define weak and strong entity types?

Entity types that do not have key attributes of their own are called **weak entity types.**

Entity types that have key attributes of their own are called **strong entity types**

47. What is partial key? Give example

a **partial key** is the attribute that can uniquely identify weak entities that are *related to the same owner entity*.

EG; Consider the entity type DEPENDENT, related to EMPLOYEE, which is used to keep track of the dependents of each employee. if we assume that no two dependents of the same employee ever have the same first name, the attribute Name of DEPENDENT is the partial key.

48. Who is Database Administrator? (Repeat – Q:16)

49. Definition of system designer, tool developer, operator & maintenance personnel.

DBMS system designers and implementers design and implement the DBMS modules and interfaces as a software package.

Tool developers design and implement **tools**, the software packages that facilitate database modelling and design, database system design, and improved performance.

Operators and maintenance personnel (system administration personnel) are responsible for the actual running and maintenance of the hardware and software environment for the database system.

50. What is meant by restricting unauthorized access?

When multiple users share a large database, it is likely that most users will not be authorized to access all information in the database. For example, financial data such as salaries and bonuses is often considered confidential, and only authorized persons are allowed to access such data these restrictions automatically. Unauthorised access violates integrity constraints and hence it should be restricted.

51. What is deductive database system?

Deductive database systems provide capabilities for defining *deduction rules* for *inferencing* new information from the stored database facts.
For example, there may be complex rules in the miniworld application for determining when a student is on probation. These can be specified *declaratively* as **rules**, which when compiled and maintained by the DBMS can determine all students on probation.

52. Why storage definition language (SDL) is used?

The Storage Definition Language is used to specify the internal schema. The mapping between the two schemas may be specified in either Data Definition Language or Storage Definition Language.

53. Why interfaces for DBA are used?

Interfaces for DBA are used for creating accounts, setting system parameters, granting account authorization, changing a schema, reorganizing the storage structures of a databases.

3 OR MORE MARKS

1. What are the characteristics of DBMS? Explain

1. **Self-describing nature of a database system**
   A fundamental characteristic of the database approach is that the database system contains not only the database itself but also a complete definition or description of the database structure and constraints. This definition is stored in the DBMS catalog,

which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data. The information stored in the catalog is called meta-data, and it describes the structure of the primary database.

The catalog is used by the DBMS software and also by database users who need information about the database structure.

2. **Insulation between programs and data, and data abstraction**

DBMS access programs do not require such changes in most cases. The structure of data files is stored in the DBMS catalog separately from the access programs. We call this property **program-data independence.**

In object - oriented and object-relational systems , users can define operations on data as part of the database definitions. An operation (also called a function or method) is specified in two

parts. The interface (or signature) of an operation includes the operation name and the data types of its arguments (or parameters). The implementation (or method) of the operation is specified separately and can be changed without affecting the interface. User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented. This may be termed **program-operation independence**.

The characteristic that allows program-data independence and program-operation independence is called **data abstraction**.

3. **Support of multiple views of the data**

A database typically has many types of users, each of whom may require a different perspective or **view** of the database. A view may be a subset of the database or it may contain **virtual data** that is derived from the database files but is not explicitly stored. Some users may not need to be aware of whether the data they refer to is stored or derived. A multiuser DBMS whose users have a variety of distinct applications must provide facilities for defining multiple views.

4. **Sharing of data and multiuser transaction processing**

Multiuser DBMS allows multiple users to access the database at the same time. This is essential if data for multiple applications is to be integrated and maintained in a single database.

The DBMS must include concurrency control software to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct. A fundamental role of multiuser DBMS software is to ensure that concurrent transactions operate correctly and efficiently.

Transaction is an executing program or process that includes one or more database accesses, such as reading or updating of database records.

The **isolation** property ensures that each transaction appears to execute in isolation from other transactions, even though hundreds of transactions may be executing concurrently.

The **atomicity** property ensures that either all the database operations in a transaction are executed or none are.

2. Explain the different categories of End Users in DBMS?

1. **End users** are the people whose jobs require access to the database for querying, updating, and generating reports; the database primarily exists for their use. There are several categories of end users:

2. **Casual end users** occasionally access the database, but they may need different information each time. They use a sophisticated database query interface to specify their requests and are typically middle- or high-level managers or other occasional browsers.

3. **Naive** or **parametric end users** make up a sizable portion of database end users. Their main job function revolves around constantly querying and updating the database, using standard types of queries and updates that have been carefully programmed and tested.
   Eg: Bank customers , Reservation agents or customers for airlines, hotels, and car rental companies, Social media users

4. **Sophisticated end users** include engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirements.

5. **Standalone users** maintain personal databases by using ready-made program packages that provide easy-to-use menu-based or graphics-based interfaces.
   An example is the user of a financial software package that stores a variety of personal financial data.

3. Write a short note on
4. DBA ii. Database Designers

Database administrator (DBA) is a chief administrator who oversees and manages database, DBMS and related software. The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed. The DBA is accountable for problems such as security breaches and poor system response time.

Database designers are responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data.

Their responsibilities are:

1. to communicate with all prospective database users in order to understand their requirements and to create a design that meets these requirements.
2. interact with each potential group of users and develop views of the database that meet the data and processing requirements of these groups.
3. analyze each view and integrate with the views of other user groups.

5. Explain the different categories of workers those are associated with the design, development and operation of the DBMS

Workers behind the Scene are those who design, use, and administer a database, others are associated with the design, development, and operation of the DBMS software and system environment.

a) **DBMS system designers and implementers** design and implement the DBMS modules and interfaces as a software package. A DBMS is a very complex software system that consists of many components, or **modules**, including modules for implementing the catalog, query language processing, interface processing, accessing and buffering data, controlling concurrency, and handling data recovery and security. The DBMS must interface with other system software, such as the operating system and compilers for various programming languages.

b) **Tool developers** design and implement **tools**, the software packages that facilitate database modelling and design, database system design, and improved performance. Tools are optional packages that are often purchased separately. They include packages for database design, performance monitoring, natural language or graphical interfaces, prototyping, simulation, and test data generation. Independent software vendors develop and market these tools.

c) **Operators and maintenance personnel** (system administration personnel) are responsible for the actual running and maintenance of the hardware and software environment for the database system

6. Explain any four advantages of DBMS

1. **Controlling Redundancy**

   Redundancy refers to duplication of data(storing same data multiple times). In the database approach, the views of different user groups are integrated during database design. It stores each logical data item, such as a student's name or birth date, in only one place in the database. This is known as data normalization, and it ensures consistency and saves storage space. Controlling redundancy also prevents inconsistencies among the files.

2. **Restricting Unauthorised Access**

   When multiple users share a large database, it is likely that most users will not be authorized to access all information in the database. For example, financial data such as salaries and bonuses are often considered confidential, and only authorized persons are allowed to access such data. In addition, some users may only be permitted to retrieve data, whereas others are allowed to retrieve and update. Hence, the type of access operation, such as retrieval or update, must also be controlled. Typically, users or user groups are given account numbers protected by passwords, which they can use to gain access to the database. A DBMS should provide a security and authorization subsystem, which the DBA uses to create accounts and to specify account restrictions.

3. **Providing Multiple User Interface**

   Many types of users with varying levels of technical knowledge use a database and DBMS provides a variety of user interfaces. These include apps for mobile users, query languages for casual users, programming language interfaces for application programmers, forms and command codes for parametric users, and menu-driven interfaces and natural language interfaces for standalone users.

4. **Providing Backup and Recovery**

   A DBMS provides facilities for recovering from hardware or software failures. The backup and recovery subsystem of the DBMS is responsible for recovery. For example, if the computer system fails in the middle of a complex update transaction, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the transaction started executing. Disk backup is also necessary in case of a catastrophic disk failure.

7. Explain the different categories of data models

A **data model** is a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve this abstraction.

Structure of a database refers to the data types, relationships, and constraints that apply to the data. Data models also include a set of basic operations for specifying retrievals and updates on the database.

The categories of data model are:

1. **High-level or conceptual data models** : They provide concepts that are close to the way many users perceive data. Conceptual data models use concepts such as entities, attributes, and relationships.
   An **entity** represents a real-world object or concept, such as an employee or a project from the miniworld that is described in the database. An **attribute** represents some property of interest that further describes an entity, such as the employee's name or salary. A **relationship** among two or more entities represents an association among the entities
2. **Representational (or implementation) data models:** They provide concepts that may be easily understood by end users but that are not too far removed from the way data is organized in computer storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly.
   Representational or implementation data models are the models used most frequently in traditional commercial DBMSs.
   Representational data models represent data by using record structures and hence are sometimes called record-based data models.
3. **Low-level or physical data models:** They provide concepts that describe the details of how data is stored on the computer storage media, typically magnetic disks. Concepts provided by physical data models are generally meant for computer specialists, not for end users. Physical data models describe how data is stored as files in the computer by representing information such as record formats, record orderings, and access paths.

8. Explain three schema architecture of database with a neat diagram

The goal of the three-schema architecture is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels:

1. The **internal level** has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.
2. The **conceptual level** has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. A representational data model is used to describe the conceptual schema when a database system is implemented.
3. The **external or view level** includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group. Each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level conceptual data model.

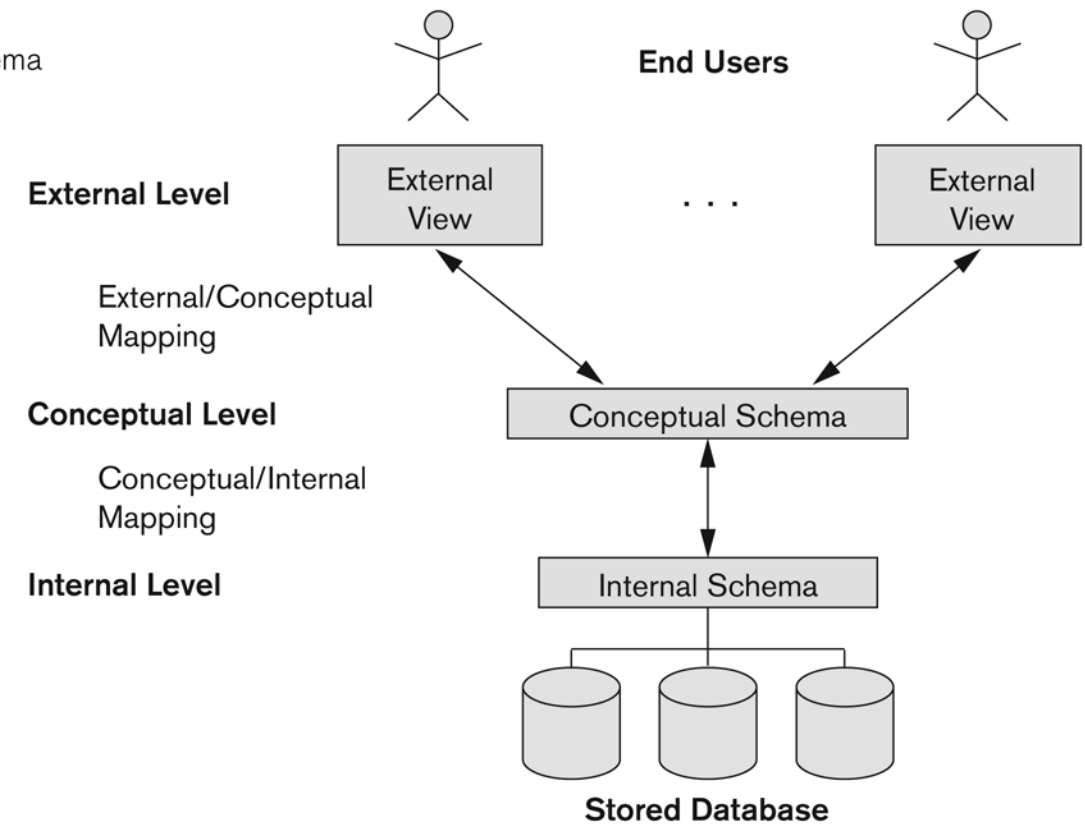


Figure 2.2
The three-schema architecture.

9. What is data Independence? Explain their types

Data independence is defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. We can define two types of data independence:

1. **Logical data independence** is the capacity to change the conceptual schema without having to change external schemas or application programs. Conceptual schema can be changed to expand the database, to change constraints, or to reduce the database. External schemas that refer only to the remaining data should not be affected. Only the view definition and the mappings need to be changed in a DBMS that supports logical data independence.
   After the conceptual schema undergoes a logical reorganization, application programs that reference the external schema constructs must work as before.
   Changes to constraints can be applied to the conceptual schema without affecting the external schemas or application programs.
2. **Physical data independence** is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized to improve the performance of retrieval or update. If the same data as before remains in the database, we should not have to change the conceptual schema.

10. What are the different database languages? Explain with example

1. **Data Definition Language(DDL):**
   It is used by the DBA and by database designers to define both schemas. The DBMS will have a DDL compiler whose function is to process DDL statements in order to identify descriptions of the schema constructs and to store the schema description in the

DBMS catalog. In DBMSs where a clear separation is maintained between the conceptual and internal levels, the DDL is used to specify the conceptual schema only.

a) **storage definition language (SDL),** is used to specify the internal schema.
b) **view definition language (VDL),** to specify user views and their mappings to the conceptual schema.

2. Data Manipulation Language (DML):

It is used to manipulate the database after the database schemas are compiled and the database is populated with data. Manipulations include retrieval, insertion, deletion, and modification of the data.

There are two main types of DMLs.

a) **High-level or nonprocedural DML:** It can be used on its own to specify complex database operations concisely. High-level DML statements can either to be entered interactively from a display monitor or terminal or to be embedded in a general-purpose programming language.
High-level DMLs, such as SQL, can specify and retrieve many records in a single DML statement; therefore, they are called **set-at-a-time or set-oriented DMLs**. A query in a high-level DML often specifies which data to retrieve rather than how to retrieve it; therefore, such languages are also called **declarative.**
b) **Low-level or procedural DML** must be embedded in a general-purpose programming language. This type of DML typically retrieves individual records or objects from the database and processes each separately.
Low-level DMLs are also called **record-at-a-time DMLs** because it needs to use programming language constructs, such as looping, to retrieve and process each record from a set of records.

11. What are the User-friendly interfaces provided by DBMS? Explain any four

User-friendly interfaces provided by a DBMS may include the following:

a) **Menu-based Interfaces for Web Clients or Browsing:** These interfaces present the user with lists of options (called menus) that lead the user through the formulation of a request. Pull-down menus are a very popular technique in Web-based user interfaces. They are also often used in browsing interfaces, which allow a user to look through the contents of a database in an exploratory and unstructured manner.
b) **Apps for Mobile Devices:** These interfaces present mobile users with access to their data. For example, banking, reservations, and insurance companies, among many others, provide apps that allow users to access their data through a mobile phone or mobile device. The apps have built-in programmed interfaces that typically allow users to login using their account name and password; the apps then provide a limited menu of options for mobile access to the user data, as well as options such as paying bills or making.
c) **Forms-based Interfaces:** A forms-based interface displays a form to each user. Users can fill out all of the form entries to insert new data, or they can fill out only certain entries, in which case the DBMS will retrieve matching data for the remaining entries. Forms are usually designed and programmed for naive users as interfaces to canned transactions. Many DBMSs have forms specification languages, which are special languages that help programmers specify such forms. Examples of form-based interfaces are SQL*Forms , Oracle Forms Etc
d) **Graphical User Interfaces:** A GUI typically displays a schema to the user in diagrammatic form. The user then can specify a query by manipulating the diagram. In many cases, GUIs utilize both menus and forms.

12. Write a note on Database system utilities

Database utilities help the DBA to manage the database system. Common utilities have the following types of functions:

1. **Loading:** A loading utility is used to load existing data files into the database. The source format of the data file and the target database file structure are specified to the utility, which then automatically reformats the data and stores it in the database. Vendors offer conversion tools that generate the appropriate loading programs, given the existing source and target database storage descriptions (internal schemas).
2. **Backup:** A backup utility creates a backup copy of the database, usually by dumping the entire database onto tape or other mass storage medium. The backup copy can be used to restore the database in case of catastrophic disk failure. Incremental backups are also often used, where only changes since the previous backup are recorded. Incremental backup is more complex, but saves storage space.
3. **Database storage reorganization:** This utility can be used to reorganize a set of database files into different file organizations and create new access paths to improve performance.
4. **Performance monitoring:** Such a utility monitors database usage and provides statistics to the DBA. The DBA uses the statistics in making decisions such as whether or not to reorganize files or whether to add or drop indexes to improve performance.

13. Write a short note on Centralized DBMSs Architecture

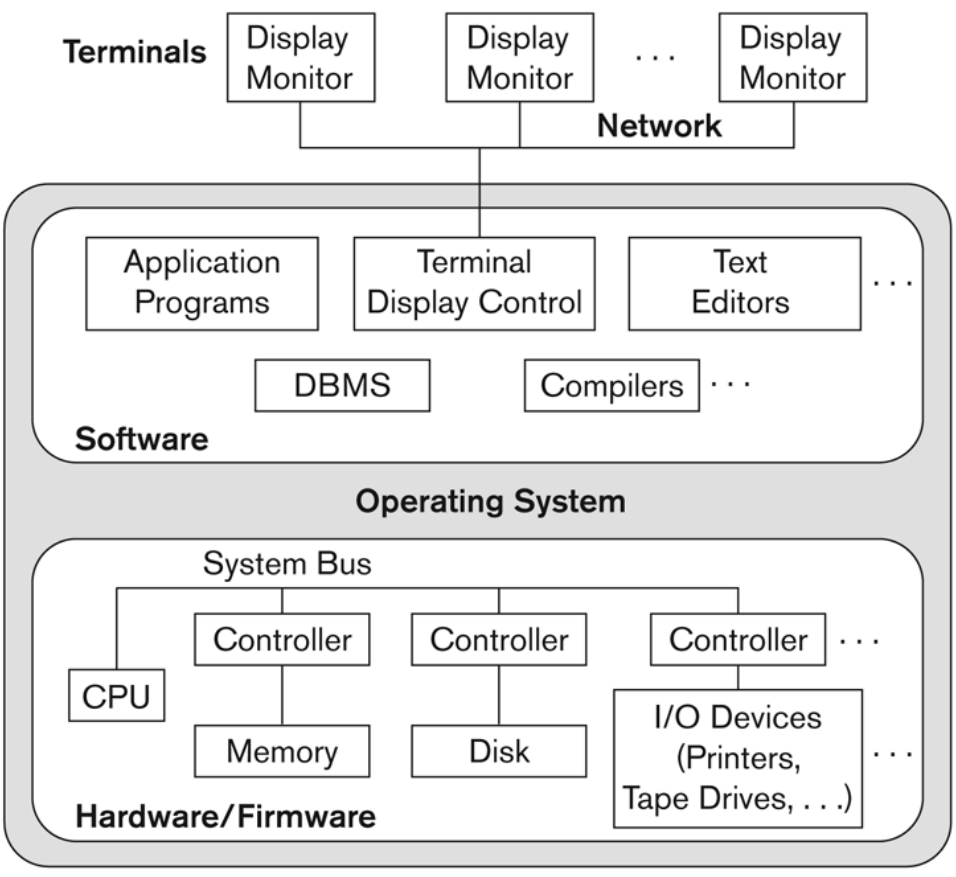


Figure 2.4
A physical centralized architecture.

It 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 centralized database is stored at a single location such as a mainframe computer. It is maintained and modified from that location only and usually accessed using an internet connection such as a LAN or WAN. All the information for the organisation is stored in a single database.

14. Write a short note on Client/Server Architectures

A **client** is typically a user machine that provides user interface capabilities and local processing. When a client requires access to additional functionality, such as database access, that does not exist at the client, it connects to a server that provides the needed functionality.
A **server** is a system containing both hardware and software that can provide services to the client machines, such as file access, printing, archiving, or database access.

The **client/server architecture** was developed to deal with computing environments in which a large number of PCs, workstations, file servers, printers, database servers, Web servers, e-mail servers, and other software and equipment are connected via a network. Specialized servers were defined with specific functionalities.
For example,

a. **file server** maintains the files of the client machines.
b. **printer server** is connected to various printers and all print requests by the clients are forwarded to this machine.
c. **Web servers or e-mail servers** also fall into the specialized server category.

The resources provided by specialized servers can be accessed by many client machines.
The **client** machines provide the user with the appropriate interfaces to utilize these servers, as well as with local processing power to run local applications.

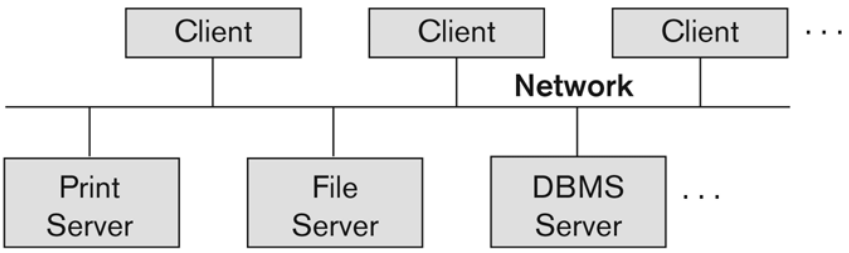


Figure 2.5
Logical two-tier client/server architecture.

15. Explain any four types of attributes with example

1. **Simple attribute :**
   An attribute that cannot be further subdivided into components is a simple attribute.
   Example: The roll number of a student, the id number of an employee.
2. **Composite attribute :**
   An attribute that can be split into components is a composite attribute.
   Example: The address can be further split into house number, street number, city, state, country, and pin code, the name can also be split into first name middle name, and last name.
3. **Single-valued attribute :**
   The attribute which takes up only a single value for each entity instance is a single-valued attribute.

Example: The age of a student.

4. **Multi-valued attribute :**
   The attribute which takes up more than a single value for each entity instance is a multi-valued attribute.
   Example: Phone number of a student: Landline and mobile.
5. **Derived attribute :**
   An attribute that can be derived from other attributes is derived attributes.
   Example: Total and average marks of a student.
6. **Stored attribute:**
   The stored attribute are those attribute which doesn't require any type of further update since they are stored in the database.
   Example: DOB(Date of birth) is the stored attribute.
7. **Complex attributes :** They are the nesting of two or more composite and multi-valued attributes. Therefore, these multi-valued and composite attributes are called 'Components' of complex attributes.
   These components are grouped between parentheses '( )' and multi-valued attributes between curly braces '{ }', Components are separated by commas ', '.
   For example: let us consider a person having multiple phone numbers, emails, and an address.
   Here, phone number and email are examples of multi-valued attributes and address is an example of the composite attribute, because it can be divided into house number, street, city, and state.

16. Write a short note on Entity types and Entity sets

A database usually contains groups of entities that are similar. For example, a company employing hundreds of employees may want to store similar information concerning each of the employees. These employee entities share the same attributes, but each entity has its *own value*(*s*) for each attribute.
An **entity type** defines a *collection* (or *set*) of entities that have the same attributes.
Each entity type in the database is described by its name and attributes. It is also called **schema** or **intension.**

The collection of all entities of a particular entity type in the database at any point in time is called an **entity set** or **entity collection**; the entity set is usually referred to using the same name as the entity type, even though they are two separate concepts. It is also called **extension** of the entity type.

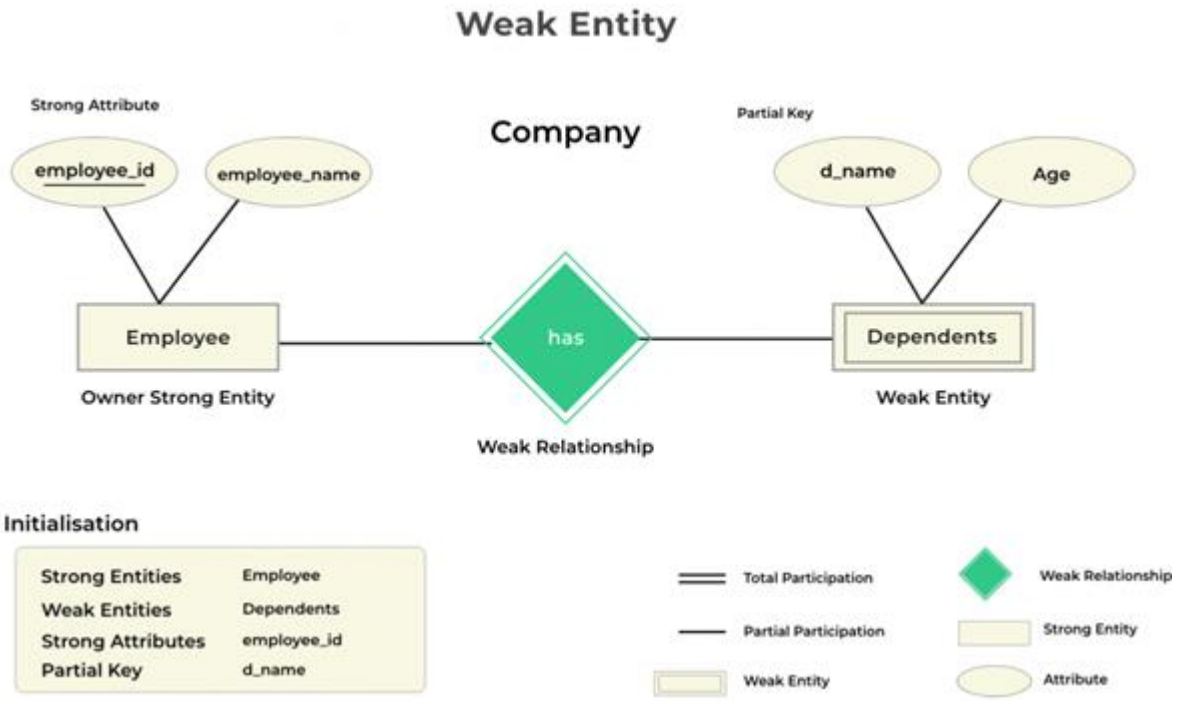
For example, EMPLOYEE refers to both a *type of entity* as well as the current collection *of all employee entities* in the database.

17. Explain Strong and weak entity types in DBMS with example

Entity types that do not have key attributes of their own are called **weak entity types.** Weak entity is dependent on strong entity and cannot exists without a corresponding strong. It has a foreign key which relates it to a strong entity. A weak entity is represented by double rectangle. Relationship between a strong entity and a weak entity is represented by double diamond.

Entity types that do have key attributes of their own are called are called **strong entity types.** Strong entity always have a primary key. In ER diagram, a strong entity is represented by

rectangle. Relationship between two strong entities is represented by a diamond. A set of strong entities is known as strong entity set.



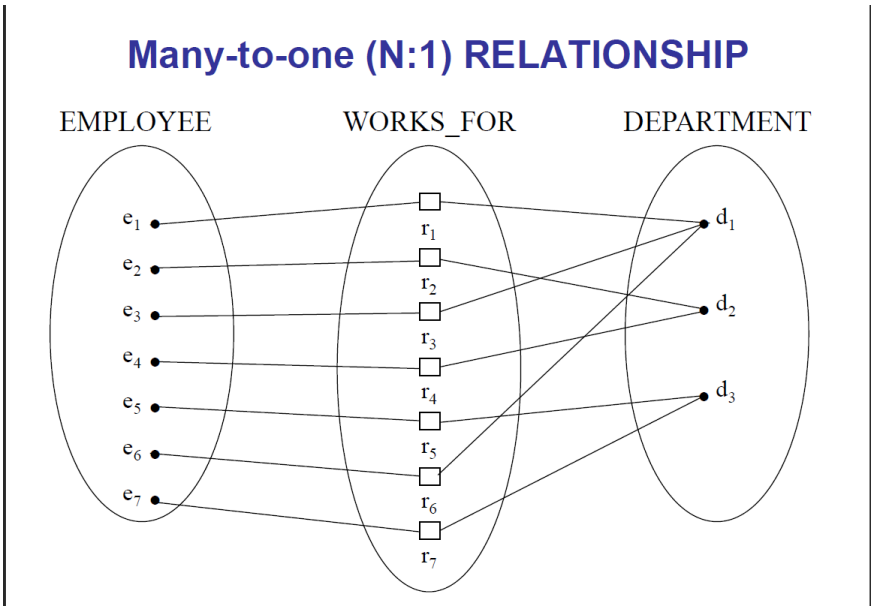
18. Why constraints are used in Relationship types ?

Relationship types usually have certain constraints that limit the possible combinations of entities that may participate in the corresponding relationship set. These constraints are determined from the miniworld situation that the relationships represent.

19. Explain Relationship Type and cardinality ratio with an example

A **relationship type** R among n entity types $E_1, E_2, \ldots, E_n$ defines a set of associations,or a **relationship set**, among entities from these entity types.

The **cardinality ratio** for a binary relationship specifies the maximum number of relationship instances that an entity can participate in.
For example, in the WORKS_FOR binary relationship type, DEPARTMENT:EMPLOYEE is of cardinality ratio **1:N**, meaning that each department can be related to (that is, employs) any number of employees (N), but an employee can be related to (work for) at most one department (1).
This means that for this particular relationship type WORKS_FOR, a particular department entity can be related to any number of employees (N indicates there is no maximum number).
On the other hand, an employee can be related to a maximum of one department.
The possible cardinality ratios for binary relationship types are **1:1, 1:N, N:1,and M:N.**

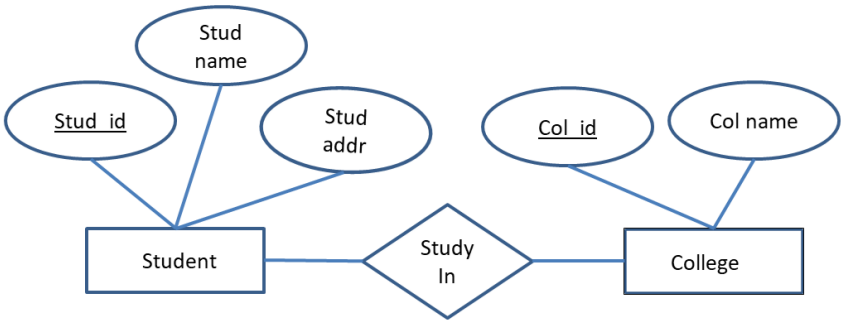


20. Illustrating with an example explain the term entity, attributes and relationships

An **entity** a thing or object in the real world with an independent existence.
An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course).
It is represented as a rectangle in ER diagram.

Attributes are properties that describe the entity.
For example, an EMPLOYEE entity may be described by the employee's name, age, address, salary, and job.
A particular entity will have a value for each of its attributes.
It is represented as an oval in ER diagram.

When an attribute of one entity type refers to another entity type , then there is a **relationship** between the two entities. Relationship represent references as relationships not attributes. It is represented a diamond shape in ER diagram.



21. Explain the Database System Environment

The top part refers to the various users of the database environment and their interfaces. The lower part refers to the internal modules of the DBMS responsible for storage of data and processing of transactions.

1. **DBMS component modules**
    a. **Buffer management**
    The database and the DBMS catalog are usually stored on disk. Access to the disk is controlled primarily by the **operating system (OS)**, which schedules disk read/write. DBMSs have their own **buffer management module** to schedule disk read/write, because management of buffer storage has a considerable effect on performance.

    b. **Stored data manager**
    A higher-level stored data manager module of the DBMS controls access to DBMS information that is stored on disk, whether it is part of the database or the catalog.

    c. **DDL compiler**
    The DDL compiler processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog. The catalog includes information such as the names and sizes of files, names and data types of data items, storage details of each file, mapping information among schemas, and constraints.

    d. **Interactive query interface :**
    Casual users and persons with occasional need for information from the database interact using the interactive query interface.
        i. **Query compiler**
        Queries are parsed and validated for correctness of the query syntax, the names of files and data elements, and so on by a query compiler that compiles them into an internal form.
        ii. **Query optimizer**
        The query optimizer is concerned with the rearrangement and possible reordering of operations, elimination of redundancies, and use of efficient search algorithms during execution

    e. **Precompiler**
    The precompiler extracts DML commands from an application program written in a host programming language. These commands are sent to the DML compiler for compilation into object code for database access.

2. **Runtime database processor**
    the runtime database processor executes
    (1) the privileged commands,
    (2) the executable query plans, and
    (3) the canned transactions with runtime parameters.
3. **System catalog :** Works with runtime database processor to update statistics.
4. **Concurrency control system and Backup and recovery system:**
    They are integrated into the working of the runtime database processor for purposes of transaction management.

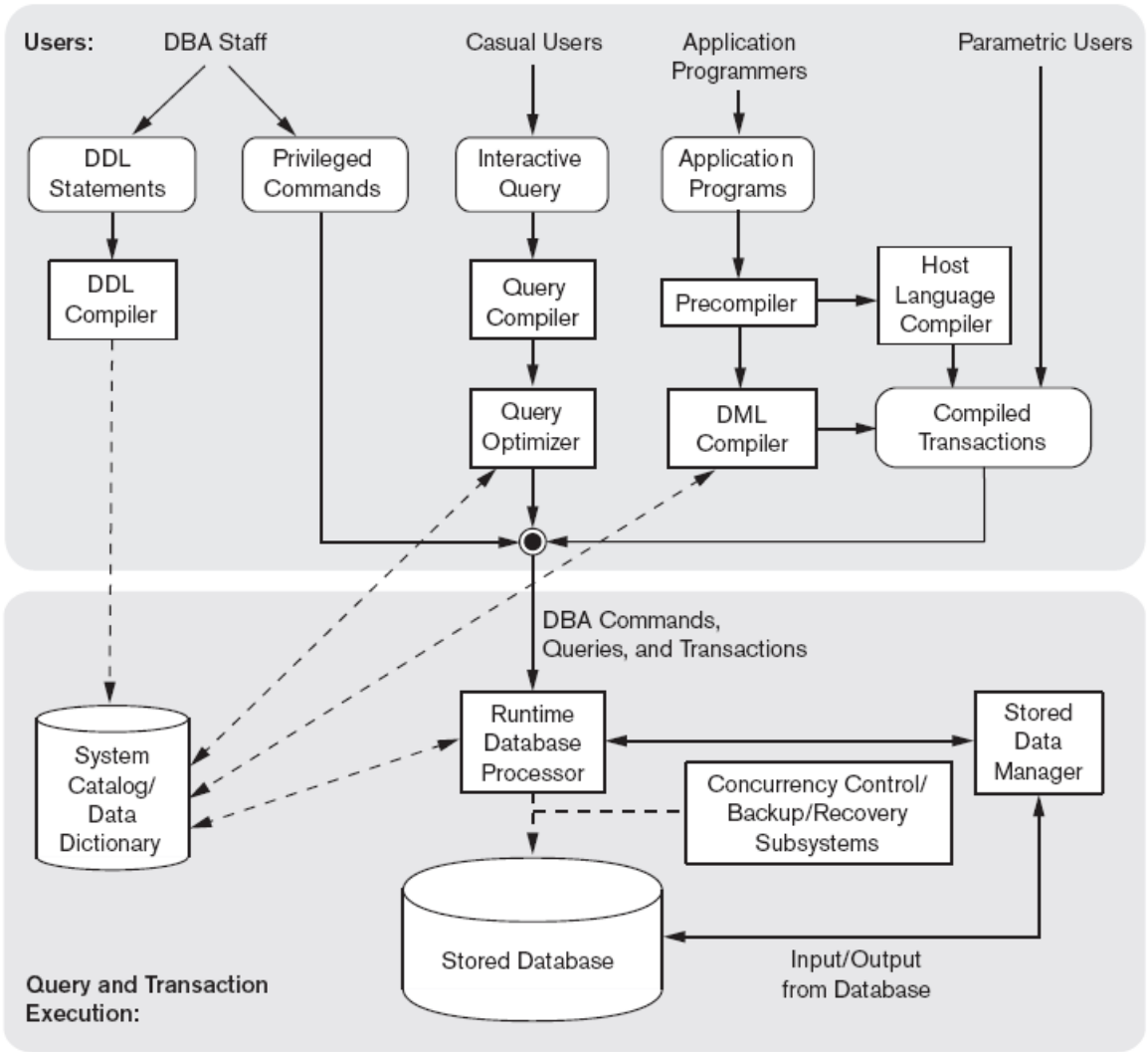


Figure 2.3
Component modules of a DBMS and their interactions.

22. Explain the different types of Database Management System

Database Management System are classified based on the following criteria:

1. **the data model on which the DBMS is based:**
   i. **relational data model:** The **relational data model** represents a database as a collection of tables, where each table can be stored as a separate file.
   ii. **object data model:** The **object data model** defines a database in terms of objects, their properties, and their operations. Objects with the same structure and behaviour belong to a **class**, and classes are organized into **hierarchies** (or **acyclic graphs**).
   iii. **big data systems:** In Big data systems the **key-value data model** associates a unique key with each and provides very fast access to a value given its key.
   iv. **hierarchical** models: The **hierarchical model** represents data as hierarchical tree structures. Each hierarchy represents a number of related records. There is no standard language for the hierarchical model.
   v. **network data models**: The **network model** represents data as record types and also represents a limited type of 1:N relationship, called a **set type**. A 1:N, or one-to-many, relationship relates one instance of a record to many record instances using some pointer linking mechanism in these models.

vi. **object-relational DBMS :** Relational DBMSs have been extending their models to incorporate object database concepts and other capabilities are referred to as **object-relational** or **extended relational systems**.
vii. **XML model** : It emerged as a standard for exchanging data over the Web and has been used as a basis for implementing several prototype native XML systems.XML uses hierarchical tree structures. It combines database concepts with concepts from document representation models.

2. **the number of users supported by the system :**
   i. **Single-user systems** support only one user at a time and are mostly used with PCs.
   ii. **Multiuser systems**, which include the majority of DBMSs, support concurrent multiple users.

3. **the number of sites over which the database is distributed:**
   i. **Centralized DBMS**: A DBMS is **centralized** if the data is stored at a single computer site. A centralized
   ii. DBMS can support multiple users, but the DBMS and the database reside totally at a single computer site.
   iii. **Distributed** DBMS: A **distributed** DBMS (DDBMS) can have the actual database and DBMS software distributed over many sites connected by a computer network.
   iv. **Big data systems** are often massively distributed, with hundreds of sites. The data is often replicated on multiple sites so that failure of a site will not make some data unavailable.

4. **On the basis of Cost:**
   i. **open source (free) DBMS:** open source (free) DBMS products like MySQL and PostgreSQL are supported by third-party vendors with additional services.
   ii. **Paid DBMS:** Standalone single-user versions of some systems like Microsoft Access are sold per copy or included in the overall configuration of a desktop or laptop.

5. **on the basis of the types of access path options for storing files:**
   i. **inverted file structures**
   ii. **general purpose** or **special purpose :** special-purpose DBMS can be designed and built for a specific application; such a system cannot be used for other applications without major changes.

23. Describe the various symbols used in an ER diagram using example.

Summary of ER-Diagram Notation for ER Schemas

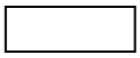
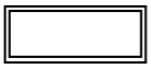
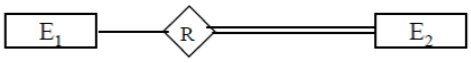
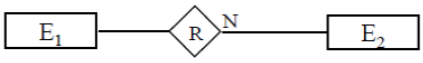
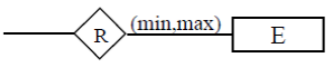
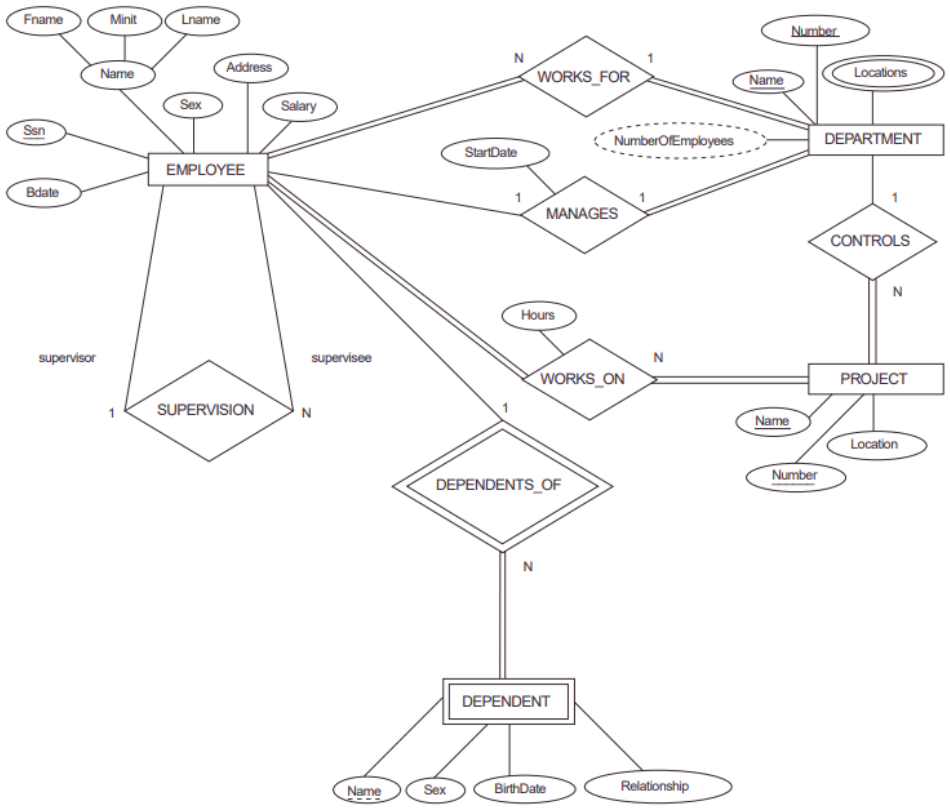
Symbol	Meaning
	ENTITY TYPE
	WEAK ENTITY TYPE
	RELATIONSHIP TYPE
	IDENTIFYING RELATIONSHIP TYPE
	ATTRIBUTE
	KEY ATTRIBUTE
	MULTIVALUED ATTRIBUTE
	COMPOSITE ATTRIBUTE
	DERIVED ATTRIBUTE
	TOTAL PARTICIPATION OF E_2 IN R
	CARDINALITY RATIO 1:N FOR E_1:E_2 IN R
	STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R

Figure 3.2 ER schema diagram for the company database.



a) In a company, an employee works on many projects which are controlled by one department.
b) One employee supervises many employees.
c) An employee has one or more dependents.
d) One employee manages one department.

24. Explain the various types of attributes used in ER modelling. (Repeat)

25. What is participation role? When is it necessary to use role names in the description of relationship types?

The participation role is part of a connection in which each entity participates. When the same entity type participates in a relationship type in many roles, the role name must be used in the description of the relationship type.

Each entity type that participates in a relationship type plays a particular role in the relationship. The role name signifies the role that a participating entity from the entity type plays in each

relationship instance, and it helps to explain what the relationship means. For example, in the WORKS_FOR relationship type, EMPLOYEE plays the role of employee or worker and DEPARTMENT plays the role of department or employer.

Role names are not technically necessary in relationship types where all the participating entity types are distinct, since each participating entity type name can be used as the role name. However, in some cases the same entity type participates more than once in a relationship type in different roles. In such cases the role name becomes essential for distinguishing the meaning of the role that each participating entity plays. Such relationship types are called recursive relationships or self-referencing relationships.

Example : In given figure ,The SUPERVISION relationship type relates an employee to a supervisor, where both employee and supervisor entities are members of the same EMPLOYEE entity set.
Hence, the EMPLOYEE entity type participates twice in SUPERVISION: once in the role of supervisor (or boss), and once in the role of supervisee (or subordinate).
Each relationship instance r_i in SUPERVISION associates two different employee entities ej and ek, one of which plays the role of supervisor and the other the role of supervisee.
The lines marked '1' represent the supervisor role, and those marked '2' represent the supervisee role; hence, e1 supervises e2 and e3, e4 supervises e6 and e7, and e5 supervises e1 and e4. In this example, each relationship instance must be connected with two lines, one marked with '1' (supervisor) and the other with '2' (supervisee).

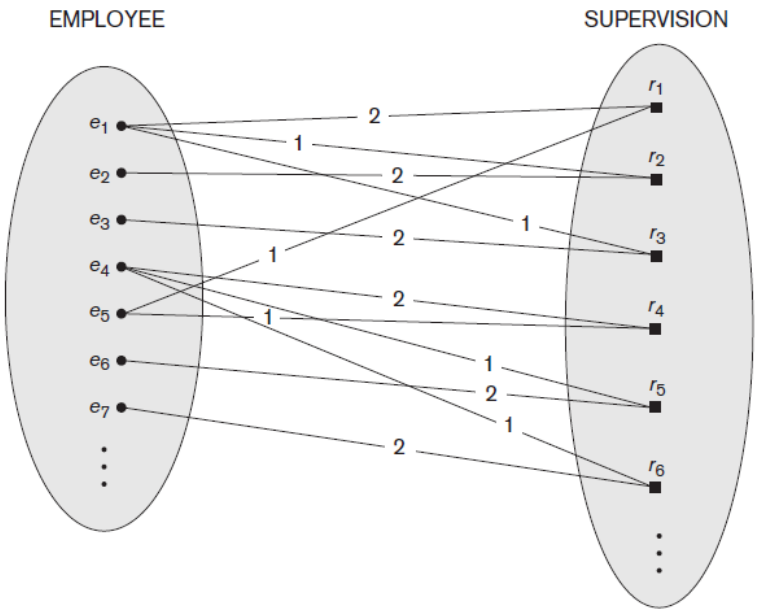


Figure 3.11
A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).