CELEBRATING 25 YEARS
CMRIT
CMR INSTITUTE OF TECHNOLOGY, BENGALURU.
ACCREDITED WITH A+ GRADE BY NAAC

Internal Assessment Test 1 – November 2021

Sub:	Data Base Management Systems				Sub Code:	18CS53	Branch :	ISE
Date:	12/11/2021	Duration:	90 min's	Max Marks: 50	Sem/Sec :	V / A,B and C		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1a)	Define Data Base and DBMS. Explain the architecture of Data Base Management System in detail with a neat diagram	6	CO1	L2
1b)	Define i) Meta Data ii) Entity Set iii) Relationship Set iv) Weak Entity Vs Strong Entity v) Types of attributes.	4	CO1	L2
2a)	Define Abstraction. Explain different types of Abstraction Levels with a neat Sketch.	4	CO1	L2
2b)	Explain the following data models with examples. i) Relational Data Model ii) E-R Data Model iii) Hierarchical Data Model iv) Network Data Model	6	CO1	L2
3	Write down the Real time Applications of DBMS	10	CO1	L2
4	List and Explain the advantages and disadvantages of DBMS. Discuss any Five advantages by comparing with File Systems.	10	CO1	L2

5 (a)	Consider the following schema for a Library Database:
	BOOK(Book_id, Title, Publisher_Name, Pub_Year)
	BOOK_AUTHORS(Book_id, Author_Name)
	PUBLISHER(Name, Address, Phone)
	BOOK_COPIES(Book_id, Programme_id, No-of_Copies)
	BOOK_LENDING(Book_id, Programme_id, Card_No, Date_Out, Due_Date)
	LIBRARY_PROGRAMME(Programme_id, Programme_Name, Address) and Design an E-R Diagram for Library Data Base by Specifying Primary keys and foreign keys if any.
		6	CO1	L3

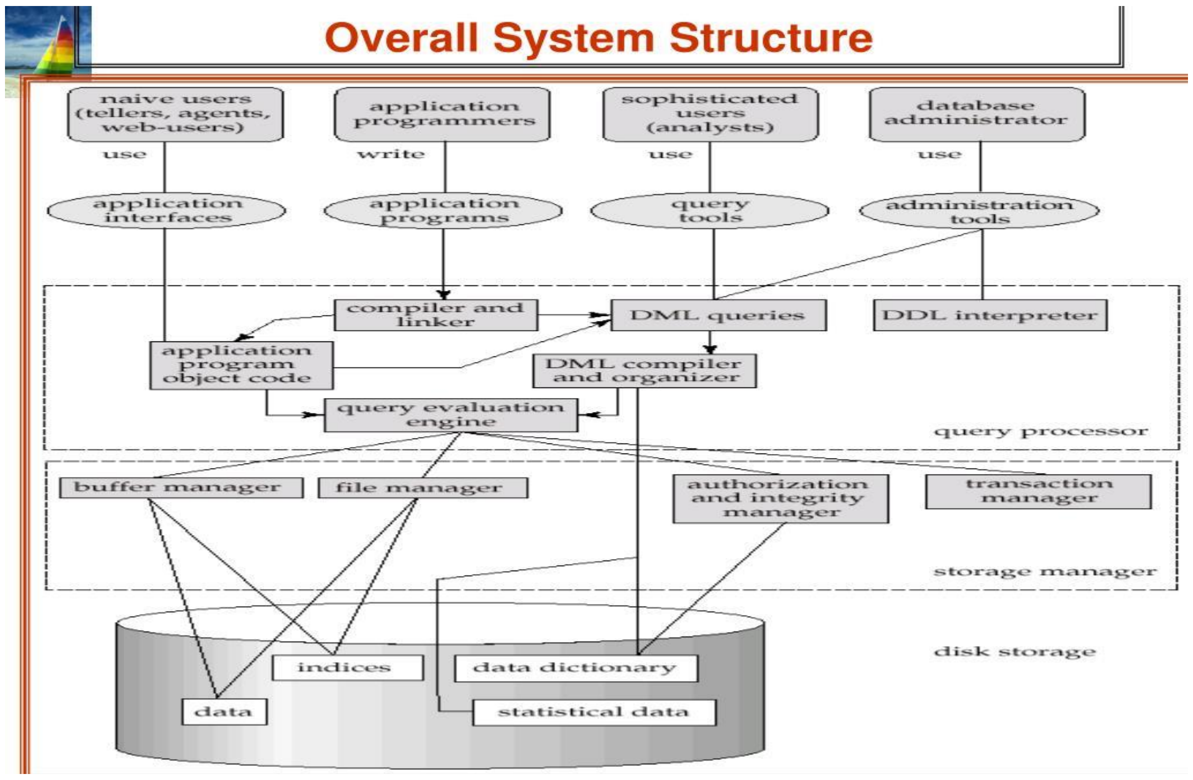
5 (b)	What is Data Independence? Explain Different types of Data Independence.	4	CO1	L1
6 (a)	Explain Different DDL and DML Commands	4	CO1	L2
6(b)	Explain the main Characteristics of DBMS Approach versus the File Processing Approach.	6	CO1	L2

1. Define Data Base and DBMS. Explain the architecture of Data Base Management System in detail with a neat diagram.

   Ans: A **database** is a collection of related data. By **data**, we mean known facts that can be recorded and that have implicit meaning. For example, consider the names, telephone numbers, and addresses of the people you know. Nowadays, this data is typically stored in mobile phones, which have their own simple database software.

   A database has the following implicit properties:
   ■ A database represents some aspect of the real world, sometimes called the **miniworld** or the **universe of discourse (UoD)**. Changes to the miniworld are reflected in the database.
   ■ A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
   ■ A database is designed, built, and populated with data for a specific purpose.
   It has an intended group of users and some preconceived applications in which these users are interested.

   A **database management system (DBMS)** is a computerized system that enables users to create and maintain a database. The DBMS is a *general-purpose software system* that facilitates the processes of *defining, constructing, manipulating,* and *sharing* databases among various users and applications. **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; it is 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.



1. Query Processor :
It interprets the requests (queries) received from end user via an application program into instructions. It also executes the user request which is received from the DML compiler.
Query Processor contains the following components –

- **DML Compiler –** It processes the DML statements into low level instruction (machine language), so that they can be executed.

- **DDL Interpreter –** It processes the DDL statements into a set of table containing meta data (data about data).

- **Embedded DML Pre-compiler –** It processes DML statements embedded in an application program into procedural calls.

- **Query Optimizer –** It executes the instruction generated by DML Compiler.

2. Storage Manager :
Storage Manager is a program that provides an interface between the data stored in the database and the queries received. It is also known as Database Control System.

- **Authorization Manager –** It ensures role-based access control, i.e,. checks whether the particular person is privileged to perform the requested operation or not.
- **Transaction Manager –** It controls concurrent access by performing the operations in a scheduled way that it receives the transaction. Thus, it ensures that the database remains in the consistent state before and after the execution of a transaction.

- **File Manager –** It manages the file space and the data structure used to represent information in the database.

- **Buffer Manager –** It is responsible for cache memory and the transfer of data between the secondary storage and main memory.

3. Disk Storage :
It contains the following components –

- **Data Files –** It stores the data.

- **Data Dictionary –** It contains the information about the structure of any database object. It is the repository of information that governs the metadata.

- **Indices –** It provides faster retrieval of data item.

1b). Define

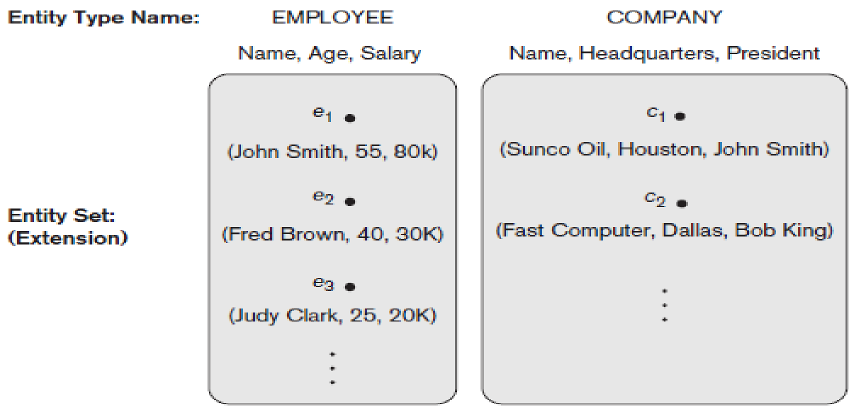
a. Meta Data
b. Entity Set
c. Relationship Set
d. Weak Entity Vs Strong Entity
e. Types of attributes.

Ans:
Meta Data
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

Entity Set

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. Figure shows two entity types: EMPLOYEE and COMPANY, and a list of some of the attributes for each.



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**

Relationship Set

A **relationship type** *R* among *n* entity types *E*1, *E*2, . . . , *En* defines a set of associations—or a **relationship set**—among entities from these entity types. Similar to the case of entity types and entity sets, a relationship type and its corresponding relationship set are customarily referred to by the *same name, R*.

Mathematically, the relationship set *R* is a set of **relationship instances** *ri*, where each *ri* associates *n* individual entities (*e*1, *e*2, . . . , *en*), and each entity *ej* in *ri* is a member of entity set *Ej*, $1 \leq j \leq n$.

Hence, a relationship set is a mathematical relation on *E*1, *E*2, . . . , *En*; alternatively, it can be defined as a subset of the Cartesian product of the entity sets *E*1 × *E*2 × . . . × *En*. Each of the entity types *E*1, *E*2, . . . , *En* is said to **participate** in the relationship type *R*; similarly, each of the individual entities *e*1, *e*2, . . . , *en* is said to **participate** in the relationship instance *ri* = (*e*1, *e*2, . . . , *en*).

Weak Entity Vs Strong Entity

Entity types that do not have key attributes of their own are called **weak entity types**. In contrast, **regular entity types** that do have a key attribute are called **strong entity types**. Entities belonging to a weak entity type are identified by being related to specific entities from another entity type in combination with one of their attribute values.

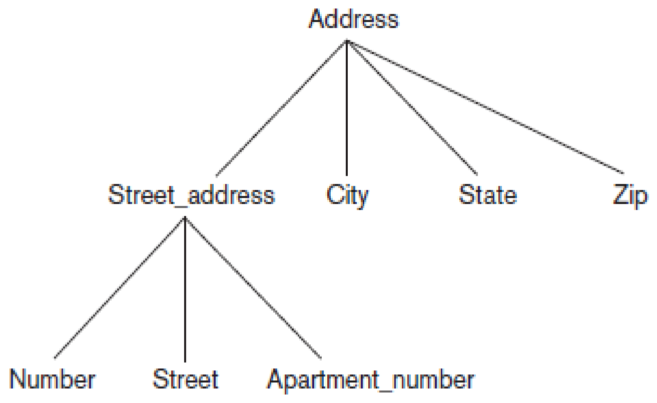
Types of attributes.
The basic concept that the ER model represents is an **entity**, which is a *thing* or *object* in the real world with an independent existence. An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course).

Each entity has **attributes**—the particular properties that describe it. For example, an EMPLOYEE entity may be described by the employee's name, age, address, salary, and job.

Several types of attributes occur in the ER model:

- *simple* versus *composite*,
- *single valued* versus *multi valued*, and
- *Stored* versus *derived*.

Composite versus Simple (Atomic) Attributes.
Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. Attributes that are not divisible are called **simple** or **atomic attributes**. Composite attributes can form a hierarchy; for example, Street_address can be further subdivided into three simple component attributes: Number, Street, and Apartment_number.



may have lower and upper bounds to constrain the *number of values* allowed for each individual entity. For example, the Colors attribute of a car may be restricted to have between one and two values, if we assume that a car can have two colors at most. **Stored versus Derived Attributes.** In some cases, two (or more) attribute values are related—for example, the Age and Birth_date attributes of a person. For a
Single-Valued versus Multivalued Attributes. Most attributes have a singlevalue for a particular entity; such attributes are called **single-valued**. For example, Age is a single-valued attribute of a person.

In some cases an attribute can have a set of values for the same entity—for instance, a Colors attribute for a car, or a College_degrees attribute for a person. Cars with one color have a single value, whereas two-tone cars have two color values. Similarly, one person may not have any college degrees, another person may have one, and a third person may have two or more degrees;

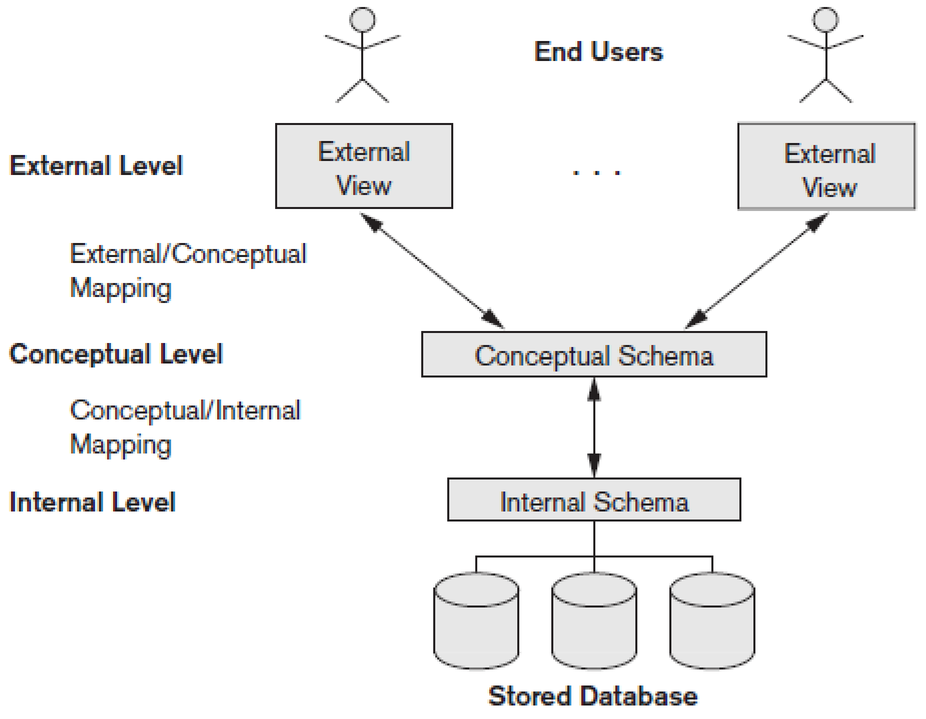
therefore, different people can have different *numbers* of *values* for the College_degrees attribute. Such attributes are called **multivalued**. A multivalued attribute may have lower and upper bounds to constrain the *number of values* allowed for each individual entity. For example, the Colors attribute of a car may be restricted to have between one and two values, if we assume that a car can have two colors at most.

Stored versus Derived Attributes.

In some cases, two (or more) attribute values are related—for example, the Age and Birth_date attributes of a person. For a particular person entity, the value of Age can be determined from the current (today's) date and the value of that person's Birth_date. The Age attribute is hence called a **derived attribute** and is said to be **derivable from** the Birth_date attribute, which is called a **stored attribute**. Some attribute values can be derived from *related entities*; for example, an attribute Number_of_employees of a DEPARTMENT entity can be derived by counting the number of employees related to (working for) that department.

2 a) Define Abstraction. Explain different types of Abstraction Levels with a neat Sketch

Abstraction :
The characteristic that allows program-data independence and program-operation independence is called **data abstraction**. 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. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This *implementation conceptual schema* is often based on a *conceptual schema design* in a high-level data model.

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. As in the previous level, 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.

2 b) Explain the following data models with examples.

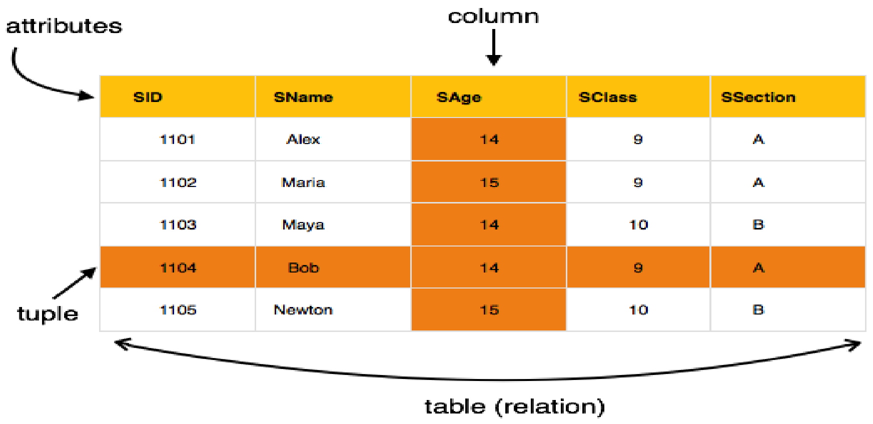
i) Relational Data Model
ii) E-R Data Model
iii) Hierarchical Data Model
iv) Network Data Model

A **data model**—a collection of concepts that
can be used to describe the structure of a database—provides the necessary means
to achieve this abstraction.2 By *structure of a database* we mean the data types, relationships,
and constraints that apply to the data. Most data models also include a
set of **basic operations** for specifying retrievals and updates on the database.

Categories of Data Models

Relational Model:

- The most popular data model in DBMS is the Relational Model. It is more scientific a model than others. This model is based on first-order predicate logic and defines a table as an **n-ary relation**.

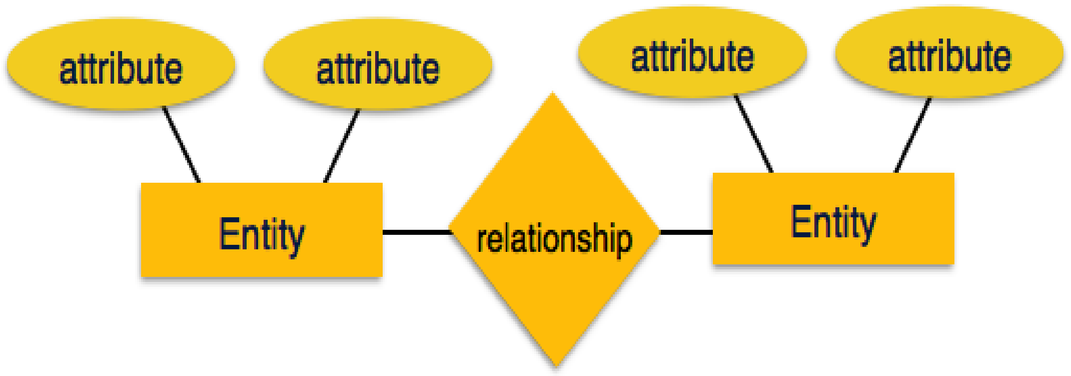


SID	SName	SAge	SClass	SSection
1101	Alex	14	9	A
1102	Maria	15	9	A
1103	Maya	14	10	B
1104	Bob	14	9	A
1105	Newton	15	10	B

- The main highlights of this model are −
- Data is stored in tables called **relations**.
- Relations can be normalized.
- In normalized relations, values saved are atomic values.
- Each row in a relation contains a unique value.
- Each column in a relation contains values from a same domain.

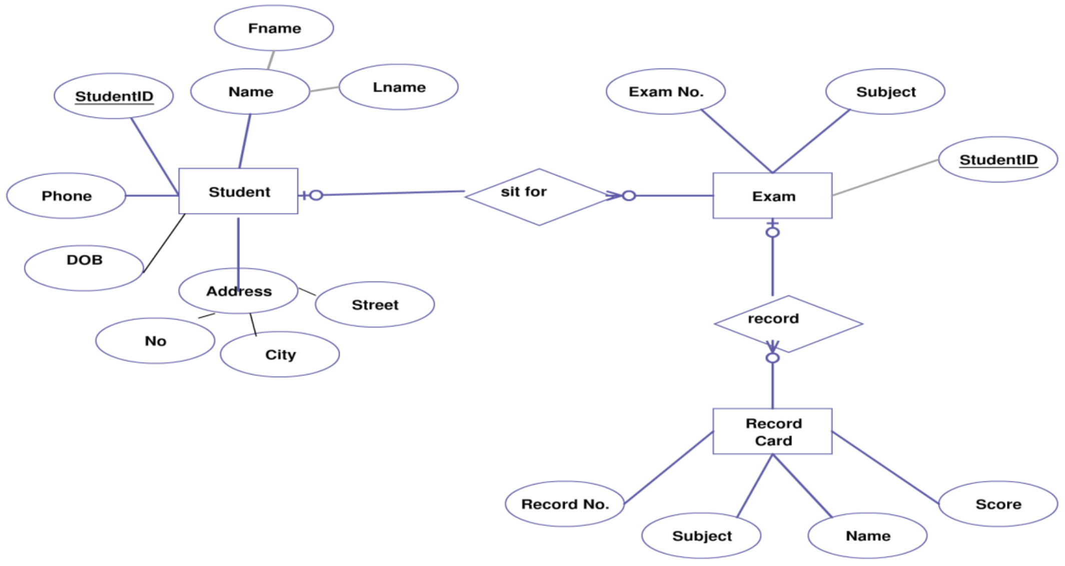
ii) Entity-Relationship Model

- Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them. While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes and constraints.
- ER Model is best used for the conceptual design of a database.
- ER Model is based on −
  - **Entities** and their *attributes.*
  - **Relationships** among entities.



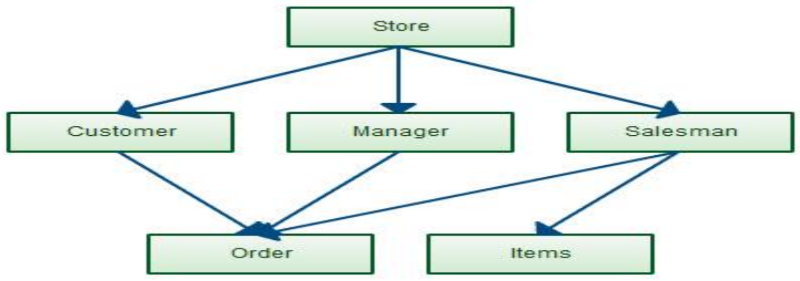
- **Entity** − An entity in an ER Model is a real-world entity having properties called **attributes**. Every **attribute** is defined by its set of values called **domain**. For example, in a school database, a student is considered as an entity. Student has various attributes like name, age, class, etc.
- **Relationship** − The logical association among entities is called ***relationship***. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of association between two entities.
- Mapping cardinalities −
  - one to one
  - one to many
  - many to one
  - many to many

E-R DIAGRAM FOR STUDENT DATABASE



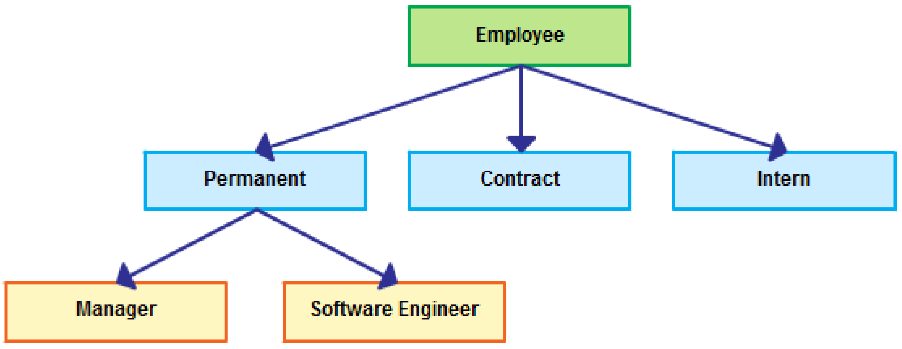
Network Data Model

- Network model has the entities which are organized in a graphical representation and some entities in the graph can be accessed through several paths.



Hierarchical Data Model

- Hierarchical model has one parent entity with several children entity but at the top we should have only one entity called root. For example, department is the parent entity called root and it has several children entities like students, professors and many more.



3. Write down the Real time Applications of DBMS

i) **Railway Reservation System –**
In the rail route reservation framework, the information base is needed to store the record or information of ticket appointments, status about train's appearance, and flight. Additionally, if trains get late, individuals become acquainted with it through the information base update.

ii) **Library Management System –**
There are loads of books in the library so; it is difficult to store the record of the relative multitude of books in a register or duplicate. Along these lines, the data set administration framework (DBMS) is utilized to keep up all the data identified with the name of the book, issue date, accessibility of the book, and its writer.

iii) **Banking –**
Database the executive's framework is utilized to store the exchange data of the client in the information base.

iv) **Education Sector –**
Presently, assessments are led online by numerous schools and colleges. They deal with all assessment information through the data set administration framework (DBMS). In spite of that understudy's enlistments subtleties, grades, courses, expense, participation, results, and so forth all the data is put away in the information base.

v) **Credit card exchanges –**
The database Management framework is utilized for buying on charge cards and age of month to month proclamations.

vi) **Social Media Sites –**
We all utilization of online media sites to associate with companions and to impart our perspectives to the world. Every day, many people group pursue these online media accounts like Pinterest, Facebook, Twitter, and Google in addition to. By the utilization of the data set administration framework, all the data of clients are put away in the information base and, we become ready to interface with others.

vii) **Broadcast communications –**
Without DBMS any media transmission organization can't think. The Database the executive's framework is fundamental for these organizations to store the call subtleties and month to month postpaid bills in the information base.

viii) **Account –**
The information base administration framework is utilized for putting away data about deals, holding and acquisition of monetary instruments, for example, stocks and bonds in a data set.

ix) **Online Shopping –**
These days, web-based shopping has become a major pattern. Nobody needs to visit the shop and burn through their time. Everybody needs to shop through web based shopping sites, (for example, Amazon, Flipkart, Snapdeal) from home. So all the items are sold and added uniquely with the assistance of the information base administration framework (DBMS). Receipt charges, installments, buy data these are finished with the assistance of DBMS.

x) **Human Resource Management –**
Big firms or organizations have numerous specialists or representatives working under them. They store data about worker's compensation, assessment, and work with the assistance of an information base administration framework (DBMS).

xi) **Manufacturing –**
Manufacturing organizations make various kinds of items and deal them consistently. To keep the data about their items like bills, acquisition of the item, amount, inventory network the executives, information base administration framework (DBMS) is utilized.

xii) **Airline Reservation System –**
This framework is equivalent to the railroad reservation framework. This framework additionally utilizes an information base administration framework to store the records of flight takeoff, appearance, and defer status.

4. List and explain the advantages and disadvantages of DBMS. Discuss any five advantages by comparing with File Systems.

Advantages of DBMS over File system

What is File System?

A File Management system is a DBMS that allows acces to single files or tables at a time. In a File System, data is directly stored in set of files. It contains flat files that have no relation to other files (when only one table is stored in single file, then this file is known as flat file).

What is DBMS?

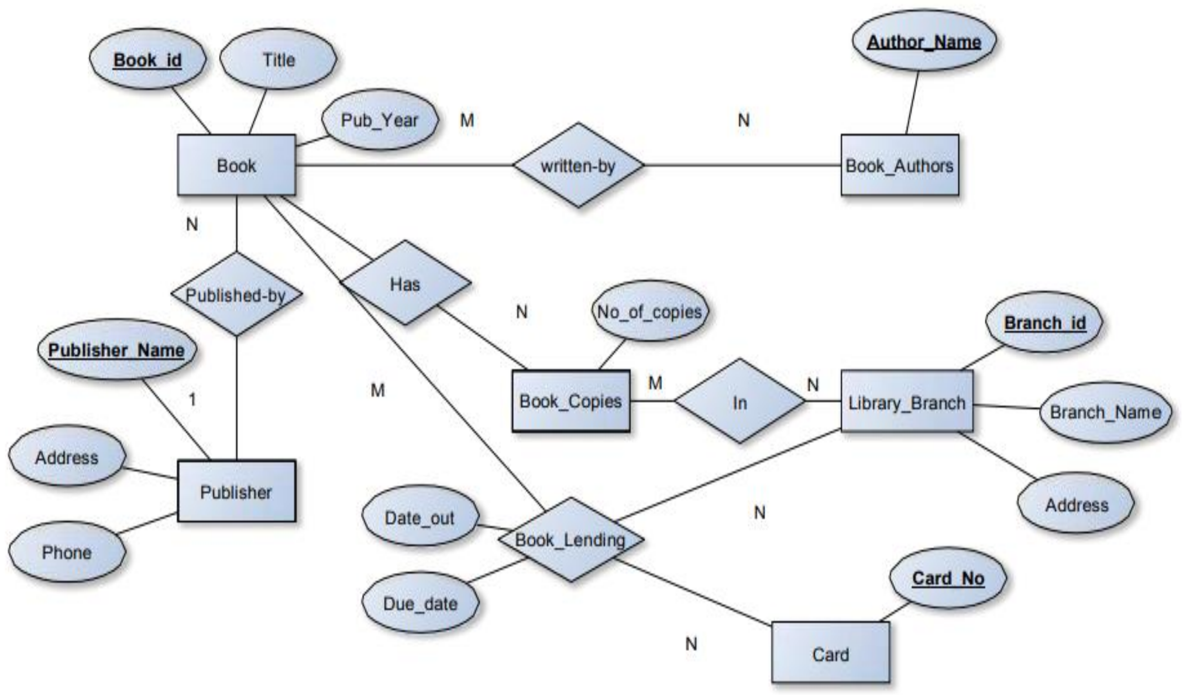
A Database Management System (DBMS) is a application software that allows users to efficiently define, create, maintain and share databases. Defining a database involves specifying the data types, structures and constraints of the data to be stored in the database. Creating a database involves storing the data on some storage medium that is controlled by DBMS. Maintaining a database involves updating the database whenever required to evolve and reflect changes in the mini world and also generating reports for each change. Sharing a database involves allowing multiple users to access the database. DBMS also serves as an interface between the database and end users or application programs. It provides control access to the data and ensures that data is consistent and correct by defining rules on them. An application program accesses the database by sending queries or requests for data to the DBMS. A query causes some data to be retrieved from database.

- **Data redundancy and inconsistency –**
  Redundancy is the concept of repetition of data i.e. each data may have more than a single copy. The file system cannot control redundancy of data as each user defines and maintains the needed files for a specific application to run. There may be a possibility that two users are maintaining same files data for different applications. Hence changes made by one user does not reflect in files used by second users, which leads to inconsistency of data. Whereas DBMS controls redundancy by maintaining a single repository of data that is defined once and is accessed by many users. As there is no or less redundancy, data remains consistent.
- **Data sharing –**
  File system does not allow sharing of data or sharing is too complex. Whereas in DBMS, data can be shared easily due to centralized system.
- **Data concurrency –**
  Concurrent access to data means more than one user is accessing the same data at the same time. Anomalies occur when changes made by one user gets lost because of changes made by other user. File system does not provide any procedure to stop anomalies. Whereas DBMS provides a locking system to stop anomalies to occur.
- **Data searching –**
  For every search operation performed on file system, a different application program has to be written. While DBMS provides inbuilt searching operations. User only have to write a small query to retrieve data from database.
- **Data integrity –**
  There may be cases when some constraints need to be applied on the data before inserting it in database. The file system does not provide any procedure to check these constraints automatically. Whereas DBMS maintains data integrity by enforcing user defined constraints on data by itself.
- **System crashing –**
  In some cases,systems might have crashes due to various reasons. It is a bane in case of file systems because once the system crashes, there will be no recovery of the data that's been lost. A DBMS will have the recovery manager which retrieves the data making it another advantage over file systems.

- **Data security –**
  A file system provides a password mechanism to protect the database but how longer can the password be protected?No one can guarantee that. This doesn't happen in the case of DBMS. DBMS has specialized features that help provide shielding to its data.

DBMS is continuously evolving from time to time. It is power tool of data storage and protection. In the coming years, we will get to witness an AI based DBMS to retrieve database of ancient eras.

5a. Consider the following schema for a Library Database:
BOOK(Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS(Book_id, Author_Name)
PUBLISHER(Name, Address, Phone)
BOOK_COPIES(Book_id, Programme_id, No-of_Copies)
BOOK_LENDING(Book_id, Programme_id, Card_No, Date_Out, Due_Date)
LIBRARY_PROGRAMME(Programme_id, Programme_Name, Address) and Design an E-R Diagram for Library Data Base by Specifying Primary keys and foreign keys if any.



Primary key: This constraint defines a column or combination of columns which uniquely identifies each row in the table.

Syntax to define a Primary key at column level:

Column_namedatatype [CONSTRAINT constraint_name] PRIMARY KEY

Syntax to define a Primary key at table level:

[CONSTRAINT constraint_name] PRIMARY KEY(column_name1, column_name2,..)

Foreign key or Referential Integrity: This constraint identifies any column referencing the PRIMARY KEY in another table. It establishes a relationship between two columns in the same table or between different tables. For a column to be defined as a Foreign Key, it should be a defined as a Primary Key in the table which it is referring. One or more columns can be defined as foreign key.

Syntax to define a Foreign key at column level:

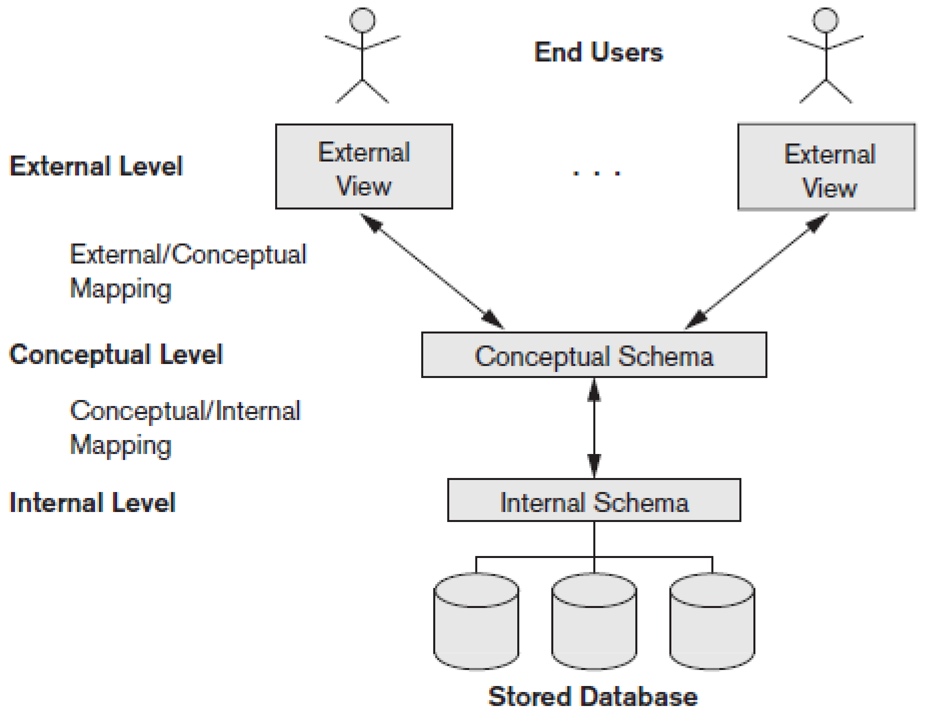
[CONSTRAINT constraint_name] REFERENCES referenced_table_name(column_name)

5b. What is Data Independence? Explain Different types of Data Independence.

The three-schema architecture can be used to further explain the concept of **data independence**, which can be 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. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item). In the last case, external schemas that refer only to the remaining data should not be affected. For example, the external schema of Figure 1.5(a) should not be affected by changing the GRADE_REPORT file (or record type) shown in Figure 1.2 into the one shown in Figure 1.6(a). 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—for example, by creating additional access structures—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. For example, providing an access path to improve retrieval speed of SECTION records (Figure 1.2) by semester and year should not require a query such as *list all sections offered in fall 2008* to be changed, although the query would be executed more efficiently by the DBMS by utilizing the new access path.



6a) Explain Different DDL and DML Commands

DDL:
DDL is Data Definition Language which is used to define data structures. For example: create table, alter table are instructions in SQL.
DML:
DML is Data Manipulation Language which is used to manipulate data itself. For example: insert, update, delete are instructions in SQL.

1. Data Definition Language (DDL)

- DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc.
- All the command of DDL are auto-committed that means it permanently save all the changes in the database.

Here are some commands that come under DDL:

- CREATE
- ALTER
- DROP
- TRUNCATE

a. CREATE It is used to create a new table in the database.
Syntax:

1. CREATE TABLE TABLE_NAME (COLUMN_NAME DATATYPES [,....]);

Example:

CREATE TABLE EMPLOYEE(Name VARCHAR2(20), Email VARCHAR2(100), DOB DAT
E);

2. **DROP:** It is used to delete both the structure and record stored in the table.

Syntax

DROP TABLE table_name;
Example

DROP TABLE EMPLOYEE;

3. **ALTER:** It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.

Syntax:
To add a new column in the table

ALTER TABLE table_name ADD column_name COLUMN-definition;
To modify existing column in the table:

ALTER TABLE table_name MODIFY(column_definitions....);
EXAMPLE

1. ALTER TABLE STU_DETAILS ADD(ADDRESS VARCHAR2(20));
2. ALTER TABLE STU_DETAILS MODIFY (NAME VARCHAR2(20));

d. TRUNCATE: It is used to delete all the rows from the table and free the space containing the table.
Syntax:

TRUNCATE TABLE table_name;
Example:

TRUNCATE TABLE EMPLOYEE;

2. Data Manipulation Language

- DML commands are used to modify the database. It is responsible for all form of changes in the database.

- The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

Here are some commands that come under DML:

- INSERT
- UPDATE
- DELETE

a. INSERT: The INSERT statement is a SQL query. It is used to insert data into the row of a table.

Syntax:

```
INSERT INTO TABLE_NAME
(col1, col2, col3,.... col N)
VALUES (value1, value2, value3, .... valueN);
```

Or

```
INSERT INTO TABLE_NAME
VALUES (value1, value2, value3, .... valueN);
```

For example:

```
INSERT INTO javatpoint (Author, Subject) VALUES ("Sonoo", "DBMS");
```

b. UPDATE: This command is used to update or modify the value of a column in the table.

Syntax:

```
UPDATE table_name SET [column_name1= value1,...column_nameN = valueN] [WHERE CO
NDITION]
```

For example:

```
UPDATE students SET User_Name = 'Sonoo'  WHERE Student_Id = '3'
```

c. DELETE: It is used to remove one or more row from a table.

Syntax:

DELETE FROM table_name [WHERE condition]; **For example:**

```
DELETE FROM javatpoint
WHERE Author="Sonoo";
```

6b) Explain the main Characteristics of DBMS Approach versus the File Processing Approach.

In the database approach, a single repository maintains data that is defined once and then accessed by various users repeatedly through queries, transactions, and application programs. The main characteristics of the database approach versus the file-processing approach are the following:

- **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**

Self-Describing Nature of a Database System

- One of the most fundamental characteristics of the database approach is that the database system contains not only the database itself but also an entire definition or description of the database structure and constraints also known as metadata of the database.
- This definition is stored within the DBMS catalog, which contains information like the structure of every file, the sort and storage format of every data item, and various constraints/rules on the information.

- The knowledge stored within the catalog is named meta-data, and it describes the structure of the first database The catalog is employed by the DBMS software and also by database users such as database administrators who required to know the information about the database structure.
- A general-purpose DBMS software package is not written for a selected database application. Therefore, it must ask the catalog to understand the structure of the files during a specific database, like the sort and format of knowledge it will access.
- The DBMS software must work equally well with any number of database applications, For example, a university database, a banking database, or a corporation database as long as because the database definition is stored within the catalog In traditional file processing, data definition is usually a part of the files. File processing software can access only specific databases, Database Management software can access various databases by extracting the database definitions or schemas from the catalog and using thesedefinitions.

Isolation between Programs and Data, and Data Abstraction :

- In a traditional file processing system, the structure of database knowledge files is embedded within the application programs, so any changes to the structure of a file may require changing all programs that access that file.
- Against this, DBMS access programs don't require such changes in most cases, so independence is achieved between them.
- The structure of knowledge files is stored within the DBMS catalog separately from the programs that access them. We call this property program-data independence.
- The characteristic that allows program-data independence and program-operation independence is known as data abstraction.
- A DBMS provides users with a conceptual representation of knowledge that doesn't include much of the small print of how the information is stored or how the operations are implemented internally. Informally, a knowledge model may be a sort of data abstraction that won't provide this conceptual representation.
- The information model uses logical concepts, like objects, their properties, and their relationships between them, which will be easier for many users to know than memory concepts or storage concepts. Hence, the information model hides storage and implementation details that are not of interest to most database users, so unnecessary complications are hidden from them.

Support for Multiple Views of the Data :

- A database sometimes has many users, each of whom may require a special perspective or view of the database.
- A view could also be a subset of the database, or it's going to contain virtual data that is derived from the database files but isn't explicitly stored.
- Some users might not get to remember whether the information they ask for is stored or derived.
- A multi-user DBMS whose users have a spread of distinct applications must provide facilities for outlining multiple views. This provides many benefits for large databases.

Sharing of knowledge and Multi-user Transaction Processing :

- A multi-user DBMS, as its name implies, must allow multiple users to access the database at an equivalent time or concurrently.
- This is often essential if data for multiple applications is to be integrated and maintained during a single database such as the latest feature of WhatsApp integration with Facebook.
- The DBMS must implement concurrency control in the software to make sure that several users trying to update equivalent data do so in a controlled manner in order that the results of the updates are correct.

- For instance, when several reservation agents attempt to assign a seat on an airline flight, the DBMS should make sure that each seat is often accessed by just one user agent at a single time for an assignment to a passenger.
- These sorts of applications are generally called online transaction processing (OLTP) applications. A fundamental role of multi-user DBMS software is to make sure that concurrent transactions operate correctly and efficiently with no inconsistency.
- The concept of a transaction has become central to several database applications. A transaction is an executing program or process that has one or more database accesses, like reading or updating of database records or inserting new records.
- The isolation property ensures that every transaction appears to execute in isolation from other transactions, many transactions could also be executed concurrently without affecting each other.
- The atomicity property ensures that either all the database operations during a transaction are executed or none are, these all ACID properties we know.