MAA OMWATI DEGREE COLLEGE HASSANPUR (PALWAL)

Notes

BCA 3rd Sem

Introduction to Database System

Q2.(b) What is database? What are the objectives of database? Also discuss the advantages and disadvantages of database. MDU BCA 2016

Ans. Database

Database is a collection of related information that acts as a single centralized data bank for various data processing applications.

A database is a collection of interrelated data stored together with controlled redundancy to serve one or more applications in a best possible way. The data are stored in such a way that they are independent of the programs used by the people for accessing the data.

In other words, when the related information is placed in an organized form it makes a database. This organization is necessary, as unorganized information has no meaning. For example, telephone directory is organized information where the names are arranged in an alphabetic order.

A database may be generated and maintained manually or it may be computerized. For example, library card catalogue is a good example of this, which can be automated as well. In a manual system a database is stored on paper whereas in a computer based system it is stored on secondary storage devices.

Database has following features:

- ***Items of data occur only once, i.e., redundancy is eliminated.***
- ***The data is independent of any one program.***
- ***The data is usable by many users.***
- ***The data is related and well organized.***

- *The data is accessible in different orders without great difficulty.*

Thus, a database is a well organized collection of data that are related in a meaningful way which can be accessed in different logical orders but are stored only once. The data in the database is therefore integrated, structured and shared.

Objectives of database

The database aims at handling organizational data as an integrated whole. A database is a general collection of interrelated data stored with minimum redundancy and used by a variety of users. The general objective of database is to make data access easy, quick, inexpensive and flexible for the users. Thus; following are the objectives of database:

1. Minimization of redundant data
2. Data independence
3. Consistency
4. Sharing of Data
5. Flexibility
6. To restrict unauthorized access
7. Accuracy
8. Recovery from failure
9. Privacy and Security
10. To enforce integrity

1. Minimization of redundant data

Data redundancy is the duplication of same data at more than one storage place. Redundant data often causes system to give conflicting information, as versions of the same data are in different phases of updating. This redundancy has to be eliminated by integrating the data

at one place. This will control redundancy and improve system performance.

2. Data independence

One of the prime objectives of database is to eliminate the need to rewrite application programs in case hardware or storage procedures are altered or new data is added. The database should adapt to the new setup without rewriting programs.

3. Consistency

Data duplication creates multiple levels of updation. At some occasions, updation of duplicated data entries may supply incorrect or conflicting information. At such times, the database is said to be inconsistent. In other words, inconsistency means when we query for the same data, that is placed on two different places and updation is done at only one place, then result will not same. Consistency of data has to be achieved through redundancy control.

4. Sharing of Data

This means various users can use the same data in the database. Moreover new applications can be developed according to the needs to operate against the same stored data. Hence the objective of database is to satisfy the data requirement of various new applications without the need of having separate data for each application.

5. Flexibility

Another objective of database should be flexible enough. This is needed to meet the future specifications. In future, requirements can be changed so database should be able to adapt them. Thus, it should make the application developments cheaper, faster and flexible.

6. To restrict unauthorized access

Data in database must be secured. Hence an important objective of database is to restrict unauthorized access. To assure this database system must provide:

- ***Identification of users of the database before they can use the database.***
- ***Monitoring user's actions so that if they do something wrong they are likely to be found.***
- ***All contents should be proper and not easy to crack.***

7. Accuracy

The accuracy of a database ensures that the quality and content of data remains constant. Integrity procedure detects and controls data inaccuracies as they take place.

8. Recovery from failure

As many users have an access to a database, the system must recover quickly in case of failure and that too with no loss of transaction data. This objective also facilitates data accuracy and integrity.

9. Privacy and Security

Privacy means when, how and to what extent data should be given to users. Data bases are costly products and hence their security is very important. Security of data is needed from accidental as well as intentional disclosures. Thus to achieve privacy and security is also an objective of database.

10. To enforce integrity

Integrity is data accuracy. It also implies that incorrect information cannot be stored in the database. In order to achieve the objective of integrity, some integrity constraints are enforced on the database.

Advantages of database

A database is a collection of interrelated data of an organization or enterprise. It is a way to consolidate and control the operational data centrally. The advantages of having a centralized control of data are:

1. Redundancy can be reduced
2. Inconsistency can be avoided
3. Data can be shared
4. Security can be enforced
5. Integrity can be maintained
6. Standards can be enforced
7. Faster development of new application

1. Redundancy can be reduced

Data redundancy is the duplication of same data at more than one storage place. By having a centralized database most of this can be avoided.

2. Inconsistency can be avoided

Data duplication may leads to inconsistency. If a fact is stored at two locations, then it is possible that user may forget to update other location. Hence at an instance of time, system returns two values. So, if the redundancy is removed, chances of having inconsistent data are also removed.

3. Data can be shared

This means various users can use the same data in the database. Moreover new applications can be developed according to the needs to operate against the same stored data.

4. Security can be enforced

Security of data is needed from accidental as well as intentional disclosures. DBA can define the access paths for accessing the data stored in database and he can define authorization checks whenever access to sensitive data is attempted.

5. Integrity can be maintained

Integrity means that the data in the database is accurate. It also implies that incorrect information cannot be stored in the database. Centralized control of the data helps in permitting the administrator to define some integrity constraints to the data in the database.

6. Standards can be enforced

With central control of the database, the DBA can ensure that all applicable standards are observed in the representation of the data.

7. Faster development of new application

When a new application is proposed, it is likely that the data required for it is already stored in the database. Thus, development time is reduced for the new application.

Disadvantages of database

The disadvantages of having a centralized control of data are:

1. Problems associated with centralization
2. Complexity of backup and recovery
3. Privacy and security
4. Enterprise Vulnerability
5. Database damage
6. Complexity increases

1. Problems associated with centralization

Centralization of data in database increases the security problem and disruption of operations of the organization because of the down times and failures.

2. Complexity of backup and recovery

Due to centralization of data in database, there is no duplication of data. It means data must be adequately backed up so that in case of failure data can be recovered.

3. Privacy and security

As data in database is centralized and is made available to users from remote locations, the possibilities of misuse are often more than in a conventional data processing system.

4. Enterprise Vulnerability

Centralized all data of an enterprise in one database means that the database becomes an indispensable resource. The survival of the enterprise may depend on reliable information being available from its database. The enterprise therefore becomes vulnerable to the destruction or to unauthorized modification of database.

5. Database damage

All data is integrated in a single database. If database is damaged due to database corruption or electric failure, valuable data is lost forever.

6. Complexity increases

The data structure may become more complex because of the centralized database supporting many applications in an organization. This may lead to difficulties in its management.

Q2.(c) What is data base system? Discuss the main objectives of any data base system also.

MDU BCA 2010

Ans. Data base system

It is a computer based record keeping system. It is a system that records and maintains large amount of information.

For example, a college keeps all the information of the students through a database system.

Thus, the main purpose of database system is to store complete information and help in retrieving it whenever required. It uses an integrated approach that all data is stored at one place. It is shared by all. Any user can query the data as per his requirement.

Main objectives of any data base system

The main objectives of data base system are:

1. Minimization of redundancy
2. Consistency
3. Sharing of Data
4. Simplicity
5. Flexibility
6. Data migration
7. To restrict unauthorized access
8. Privacy and Security
9. Maintaining standards
10. To enforce integrity

1. Minimization of redundancy

Data redundancy is the duplication of same data at more than one storage place. The duplication of data leads to

Q3.(a) How database approach differs from traditional file system? MDU BCA 2018

OR

Explain the difference between database approach and traditional file based system. IGU BCA 2018

OR

Differentiate between file system and database system.

Ans. Following are the differences between file system and database system:

File System	Database System
1. Redundant data Redundant data is the duplication of same data at more than one storage place. Since all the files are designed and developed independently, same fields are stored in more than one file.	**1. Redundancy control** Being the central control over data, DBMS removes duplication of data. Thus DBMS controls redundancy by ensuring that the data are not repeated many times.
2. No security Security restrictions can not be applied on the data.	**2. Rigorous security** As data is stored in a common place, thus, DBMS assures a sound security system.
3. Simple structure It has simple structure.	**3. Complex structure** It has complex structure.

File System	Database System
4. Often single user It has often single user.	**4. Multiple users** It has multiple users.
5. Isolated data Data is separated and scattered at various locations. It may be in different formats. To make a decision, the programmer may need different files. Thus, to extract data from different files and their coordinating is a different process.	**5. Data can be shared** Various users can use the same data in the database. Data can be shared not only be existing applications but new applications can also be developed to use the same stored data. Thus, this means that data requirements of various new applications may also be satisfied without creating new files.
6. Relatively cheap File management systems are relatively cheap as compared to database systems.	**6. Relatively expensive** Database systems are expensive. The cost can be of two types: (i) Software cost which includes purchase of DBMS and development of DBMS. (ii) Hardware cost includes upgradation needed to allow the extra overheads put up by DBMS.

File System	Database System
7. Small systems File management systems are small systems.	**7. Large systems** Database management systems are large systems.
8. Distributed control It has distributed control	**8. Central Control** It has central control.
9. Simple backup & recovery There are simple procedures for backup and recovery process.	**9. Complex backup & recovery** Data recovery in case of hardware or software failure is difficult due to the complexity of the database system.
10. No search capability There is no search capability in file system to give response to user queries.	**10. Search capability** There is search capability in database system to give response to user queries.
11. Chances of inconsistent output In file processing system inconsistent output is often produced.	**11. Consistent output** In database system consistent output is produced.

Q4.(a) What do you mean by DBMS? Also discuss the advantages and disadvantages of DBMS.

MDU BCA 2018

Ans. Database Management System

Database management system (DBMS) is a software that enables users to interact with the database. It enables users to store, modify and extract information from a database as per the requirements.

It is an intermediate layer between user programs and the data. User programs access the DBMS which then accesses the data. It controls the creation, maintenance and utilization of the databases of an organization.

Thus a DBMS is a collection of

- *Database which acts as a single centralized databank for various data processing applications.*
- *Programs for the management of database.*

Advantages of DBMS

The advantages of using a DBMS are:

1. Redundancy Control
2. Access Control
3. Proyides multiple User Interfaces
4. Easy and Fast access of data
5. Simplicity
6. Sharing of data
7. Reducing application development time
8. Persistent storage of data
9. Backup and Recovery

10. Flexible
11. Reduced programming and Maintenance Cost
12. Integrity, Security and Privacy
13. Data Independence

1. Redundancy Control

Data redundancy is the duplication of same data at more than one storage place. Data redundancy has the following disadvantages:

- ***Wastage of storage space.***
- ***Wastage of time to enter the data more than once.***
- ***Cost of generating data when same data already exists.***
- ***Inconsistency of results.***
- ***Multiple level of updation.***

DBMS controls redundancy by ensuring that the data are not repeated many times. DBMS as a central control over data removes:

- ***Duplication of data.***
- ***Saving of storage space and***
- ***Inconsistency of the data***

2. Access Control

As multiple users share a database, it is likely that some users will not be authorized to access all information in the database. Only authorized user make changes in the existing database, whereas other user category is allowed only to retrieve the database. Database Administrator (DBA) uses security and authorization subsystem in order to specify access restrictions. It

database. To enforce this integrity constraints are imposed easily on the database instead of file system where these constraints are imposed on all the applications separately.

2. A student database has the information regarding marks obtained in an examination. A student cannot obtain marks more than100 – the maximum marks in one subject. Integrity constraints can be specified so that database will accept marks only in the range of 0 to 100 per subject.

DBMS assures privacy and security for data by providing:

- ***Identification of users of the database before they can use the database.***
- ***Reconstruction facility for damaged data.***
- ***Monitoring user's actions so that if they do something wrong they are likely to be found.***
- ***Auditing subsystems.***
- ***Tamperproof and authorization subsystems.***

13. Data Independence

DBMS allows any change in the layout of data without affecting all applications manipulating the database. Some common approach is used to store, retrieve and manipulate data in database.

Disadvantages of DBMS

Disadvantages of database system are:

1. Expensive
2. Complex
3. Higher Programming Cost
4. Integration

5. Recovery difficult
6. High Conversion Cost
7. Slow processing in some applications

1. Expensive

Cost is the main disadvantage of DBMS. The cost can be of two types:

- ***Software cost which includes purchase of DBMS and development of DBMS.***
- ***Hardware cost includes upgradation needed to allow the extra overheads put up by DBMS.***

2. Complex

Database processing tends to be complex due to complex structure. To handle a large amount of data, sophisticated programming is needed.

3. Higher Programming Cost

Due to complex structure, it requires experienced programmers with complete knowledge. The cost of hiring of system analyst, database designers, database administrators, data processing personnels and cost of training will also be considered. Thus the programming cost is high.

4. Integration

The centralization of resources increases the vulnerability of the system. A failure in any component of the integrated system can stop the complete system. This situation is very critical.

5. Recovery difficult

Data recovery in case of hardware or software failure is difficult due to the complexity of the database system.

Q4.(b) Discuss the capabilities that should be provided by a DBMS. MDU BCA 2017

OR

What are the main / common facilities that every DBMS should provide? Explain. MDU BCA 2010

Ans. DBMS should provide the following capabilities / facilities:

1. Minimal Redundancy
2. Flexibility
3. Sharing of Data
4. Multiple User Interface
5. Consistency
6. Restrict Unauthorized Access
7. Privacy and Security
8. Simplicity
9. Enforce Integrity

1. Minimal Redundancy

Data redundancy is the duplication of same data at more than one storage place. This duplication of data leads to wastage of storage space & time and affects cost also. This redundancy has to be eliminated by integrating the data at one place and DBMS should facilitate this.

2. Flexibility

DBMS should allow changes to the structure of the database without affecting the stored data and the existing application. It should make the application developments cheaper, faster and flexible.

Q4.(c) What is DBMS? Explain DBMS functions with suitable examples. **MDU BCA 2014**

Ans. ***Database management system (DBMS) is a software that enables users to interact with the database. It enables users to store, modify and extract information from a database as per the requirements.***

It is an intermediate layer between user programs and the data. User programs access the DBMS which then accesses the data. It controls the creation, maintenance and utilization of the databases of an organization.

Thus a DBMS is a collection of

- ***Database which acts as a single centralized databank for various data processing applications.***
- ***Programs for the management of database.***

Example: A University Management System has records of employees. The DBMS adds newly appointed employees, deletes records when an employee has left the University or accesses each employee record once every month to compute the salary. Query such as "How many employees have salary greater than 15000", the DBMS processes the query and reports accordingly on the VDU or on a printer.

There are different types of DBMS available in the market. Popular used DBMS are dbase, foxpro and oracle.

To sum up, DBMS is a set of programs designed to:

- ***Set up storage structures.***
- ***Load the data.***

16

- *Accept data requests from programs or users in a form the program or user expects.*
- *Hide data from an unauthorized access.*
- *Allow concurrent use of data without requiring user's interference.*
- *Perform backup and recovery automatically.*

Functions of DBMS

The various functions performed by DBMS are:

1. Database Creation
2. Database Maintenance
3. Database processing
4. Redundancy Control
5. Access Control
6. Provides multiple User Interfaces
7. Backup and Recovery
8. Reducing application development time
9. Easy and Fast access of data
10. Integrity, Security and Privacy

1. Database Creation

The DBMS helps the users by providing methods and procedures to specify the data, their types and conditions for the data to be stored in the database. It organizes the data and creates the necessary database.

2. Database Maintenance

Various database maintenance tasks like addition, deletion and updation of data are performed by the

Q5.(a) Explain the role of DBA. MDU BCA 2016, 2015

OR

Explain database administrator briefly with suitable examples. MDU BCA 2014

OR

Discuss duties of database administrator with examples. MDU BCA 2013

OR

What do you mean by Database Administration (DBA)? Explain the responsibilities of DBA? MDU BCA 2012

OR

What are the responsibilities / functions of Database Administrator (DBA)? MDU BCA 2010

Ans. DBA

DBA is a person or group of persons who have central control over data and the programs comprising the database system.

DBA is one who has central control over DBMS. He has the central control on both data and programs to access those data.

DBA's job requires a high degree of technical expertise and the ability to understand management requirements. They are most familiar with the database and are responsible for:

- *Creating*
- *Manipulating and*
- *Authorizing access to the database.*

Main responsibilities of DBA are:

1. Schema definition
2. Deciding the information Contents of Database
3. Physically organize the data
4. Storage structure and access methods
5. Intermediator
6. Maintenance of database
7. Ensures the security of the system
8. Maintaining data dictionary
9. Ensuring safe modification of schema
10. Backup and recovery
11. Granting of authorization for data access
12. Ensuring Integrity
13. Works with database team
14. Provides support to users
15. Data implementation
16. Tuning the database

1. Schema Definition

The DBA specifies a database schema by writing a set of definitions.

2. Deciding the information Contents of Database

It is the duty of DBA to decide what type of information is held in the database. DBA has to identify:

- ***The entities of interest to the organization and***
- ***The information to be recorded about these entities.***

Q5.(b) Explain various types of DBMS languages in detail. **IGU BCA 2018**

OR

Discuss different languages used in DBMS for storage, manipulation and querying of data. **MDU BCA 2017**

OR

What do you mean by database language? Explain. **MDU BCA 2015**

OR

Explain DBMS languages briefly with suitable examples. **MDU BCA 2014**

Ans. DBMS languages

Database management system allows users to perform a number of operations on the database such as insert, delete and retrieve data without knowing about the physical representations of data. To help the users have these facilities, DBMS provides one or more specialized programming languages called Database or DBMS languages.

The DBMS provides the following database languages:

1. Data Definition Language

It is a language that allows the users to define data and their relationship to other types of data. It is mainly used to create files, databases, data dictionary and tables within databases.

Data Definition Language (DDL) is used by the DBA and database designers to specify the conceptual schema of a database. In many DBMS, the DDL is also used to

define internal and external schemas (views). In some DBMS, separate Storage Definition Language (SDL) and View Definition Language (VDL) are used to define internal and external schemas.

Storage definition language (SDL):- SDL is used to specify the internal schema. The specified set of SDL statements specifies the storage structure and access methods used by the database system. The implementation details of the database schemas which are generally hidden from the users are implemented by the specified SDL statements.

View Definition Language (VDL):- VDL is used to define the external schema. There are two views of data: the **logical view** as perceived by the programmer and **physical view** that is data stored on storage devices. VDL is used to specify user's views and their mappings to the conceptual schema.

2. Data Manipulation Language (DML)

It is a language that provides a set of operations to support the basic data manipulation operations on the data stored in the databases. It allows the users to insert, update, delete and retrieve data from the database.

The statement which is used to retrieve the information is called a query. The part of the DML used to retrieve the information is called a query language.

Types of DML

Procedural DML: It requires a user to specify what data are needed and how to get those data.

Declarative DML: It requires a user to specify what data are needed without specifying how to get those data. It is also referred to as non-procedural DML.

3. Data Control Language (DCL)

DCL is used to create user roles, grant permissions and control access to database by securing it. DCL statements control access to data and the database using statements like GRANT and REVOKE.

Q5.(c) What do you understand by a Query Language? Distinguish between DDL, DML and DCL. Explain any two DDL & DML commands using suitable examples.

MDU BCA 2016

Ans. Query Language

The DBMS provides data manipulation language (DML) that enables users to retrieve and manipulate the data. The statement that is used to retrieve the information is called a query. The part of the DML used to retrieve the information is called a query language.

Distinguish between DDL, DML and DCL

The database language which is used to define database objects; to drop database objects; to alter (change) database objects, such as tables, views, is known as Data Definition Language (DDL). Data Definition Language (DDL) is used by the DBA and database designers to specify the conceptual schema of a database.

The database language, which is used to insert data, manipulate data, delete data or retrieve data in tables or views, is known as Data Manipulation Language (DML). Thus, this is a language that provides a set of operations to support the basic data manipulation operations on the data stored in the databases.

The database language which is used to control data access is known as Data Control Language (DCL). DCL statements control access to data and the database using statements like GRANT and REVOKE.

DDL commands

DDL commands are: CREATE, ALTER, DROP, TRUNCATE, GRANT, REVOKE etc.

CREATE is the most commonly used DDL command. It may be used to create a database, a schema, a new table, a new index or a new view. To create an empty table student which stores student's data – Roll number, name and date of birth, the CREATE TABLE command is:

```
CREATE TABLE student
(roll_no number(5)
name char(20)
birth_date date);
```

ALTER command can be used to make the following changes to any table:

ADD - for adding new columns into the table.
MODIFY - for modifying the structure of columns
DROP - for removing a column in the table (8i)
RENAME - for renaming the column name (9i)

Syntax is:

ALTER TABLE <tablename> (ADD | MODIFY | DROP | RENAME] (column(s));

To add two new columns sex and fname to student table, the command is:

ALTER TABLE student ADD (sex VARCHAR(6), fname VARCHAR(20));

To change the datatype of column sex from varchar to char, the command is:

ALTER TABLE student MODIFY (sex CHAR(1));

DML commands

DML commands are: INSERT, UPDATE, DELETE, SELECT etc.

To add data to tuples (or rows) of table student, the syntax used is:

INSERT INTO <table-name> (columns to insert) VALUES (values to insert)

Another most commonly used DML command is SELECT. It's syntax is:

SELECT <column_name>
FROM <table_name>

or

SELECT <column_name>
FROM <table_name>
WHERE <condition>;

For example, consider the following table employee:

employee

Emp_no	Name	Design	Basic	Address
101	RAJIV	CLERK	5850	ROHTAK
102	MOHAN	SALESMAN	6400	ROHTAK
103	RAVINDER	MANAGER	9000	KARNAL
105	BALVINDER	CLERK	5850	PANIPAT
106	RAJBIR	CLERK	5850	ROHTAK
108	ASHU	SALESMAN	6400	ROHTAK

(i) Selecting all columns

Query:- What data are in the employee table?

```
SQL > SELECT *
      FROM employee;
```

Once the above command is obeyed, the table will be:

employee

Emp_no	Name	Design	Basic	Address
101	RAJIV	CLERK	5850	ROHTAK
102	MOHAN	SALESMAN	6400	ROHTAK
103	RAVINDER	MANAGER	9000	KARNAL
105	BALVINDER	CLERK	5850	PANIPAT
106	RAJBIR	CLERK	5850	ROHTAK
108	ASHU	SALESMAN	6400	ROHTAK

The asterisk (*) is used to select all columns in the table. This is very useful when the columns names are not known. Thus, a wild card (*) character indicates to view data from every column in the table.

(ii) Selecting a specific column

```
SQL > SELECT Name
      FROM employee;
```

Once the above command is obeyed, the table will be:

Name
RAJIV
MOHAN
RAVINDER
BALVINDER
RAJBIR
ASHU

(iii) Selecting multiple columns

```
SQL > SELECT Emp_no, Name, Design
      FROM employee;
```


Once the above command is obeyed, the table will be:

Emp_no	Name	Design
101	RAJIV	CLERK
102	MOHAN	SALESMAN
103	RAVINDER	MANAGER
105	BALVINDER	CLERK
106	RAJBIR	CLERK
108	ASHU	SALESMAN

(iv) Select from where

It is one of the most important clause available in SQL. By specifying a WHERE clause, specific rows can be chosen. When a WHERE clause is present, the database program goes through the entire table one row at a time and checks each row to determine if the condition is true with respect to that row.

```
SQL > SELECT  Name
            FROM  employee
                WHERE Basic = 6400 ;
```

Once the above command is obeyed, the table will be:

Name
MOHAN
ASHU

Q6.(a) Explain the various roles in database environment. **IGU BCA 2018**

Ans. A database system refers to an organisation of components that define and regulate the collection, storage, management and use of data within a database environment.

A database is designed and built to serve a specific purpose as needed by a group of users. Roles are designed to ease the administration of the end user system. Each of the following has to perform specified set of roles or responsibilities:

1. Data Administrator

The data administrator or DA refers to a person or a group of persons who understand and manage the data items.

The Data Administrator is responsible for defining data elements, data names and their relationship with the database.

His basic role/responsibilities include:

- ***To install and configure the RDBMS applications***
- ***To know the requirements of the software application in terms of functions and assure the data integrity.***
- ***To validate the data and files.***
- ***To decide what data to be stored in database.***
- ***To establish policies for data maintenance.***

Data Administrator is also known as data analyst.

2. Application developer or Application programmer

Applications developers translate software requirements into workable programming code and maintain and develop programs for use in business.

Their job titles and specific duties may vary in different organizations but the role mainly involves writing specifications and designing, building, testing and implementing applications using programming languages and development tools. In simple words, the main function of an application developer is to make computers perform specific tasks based on the client's specifications.

Application developer:

- ***Program new applications for regular running.***
- ***Test new applications and run into existing ones.***
- ***Restart the unsuccessful jobs.***

3. Database Designers

Database designers are the database architects.

- ***They develop strategies for design, modeling and implementation stages of the database lifecycle.***
- ***They often join with software architects and system analysts to understand and best address the particular needs of certain businesses or industries.***
- ***They draw the blueprints of a database which technicians and other specialists then use to put the technology together.***
- ***They also develop data models, metadata tables and related database structures for applications.***

- *They are responsible for designing persistent data storage, identifying the data to be stored, choosing the structure to represent and store this data.*

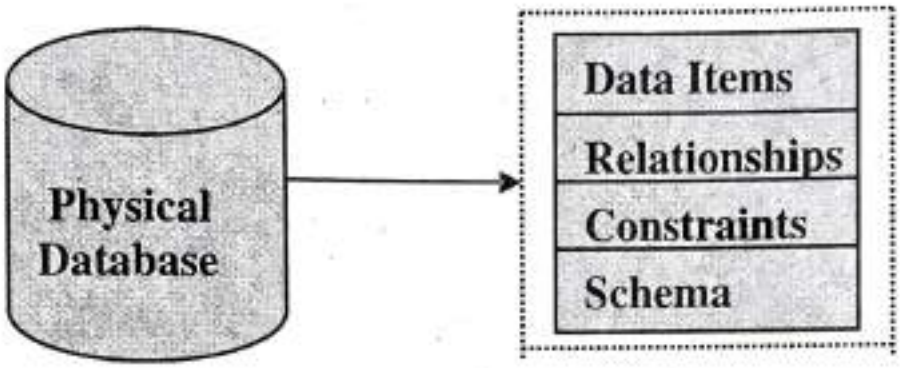
4. End Users

End users are the persons who interact with the applications. They insert, delete and update data in the database. They use the database through applications programs. They perform all operations by using commands provided in the user interface.

Q6.(b) Explain Components of Database.

Ans. A database has the following four components:

1. Data items
2. Relationships
3. Constraints
4. Schema



1. Data Items

Data items refer to a distinct piece of information.

2. Relationships

It represents a correspondence between various data elements.

3. Constraints

These are the predicates that define correct database states.

4. Schema

The overall design of the database is called schema. It describes the organization of data and relationships within the database. The schema consists of definitions of the various types of record in the database, the data

Q6.(c) Explain the various functional components of DBMS with the help of suitable diagram. **MDU BCA 2018**

OR

Components of DBMS **MDU BCA 2015**

Ans. Components / Elements of DBMS

DBMS is a complex software which provides interface for each category of the user. DBMS interprets user commands so that computer system can operate on that command to perform the desired task. The important components of DBMS are:

1. Data Definition Language (DDL) Compiler
2. Data Manager
3. File Manager
4. Disk Manager
5. Query Processor
6. Telecommunication system
7. Data files
8. Data Dictionary

1. Data Definition Language (DDL) Compiler

DDL is used by the database designers or application programmers to specify the contents and structure of the database. DDL compiler converts the data definition statements that are in the source form to object form.

2. Data Manager

Data manager is the central component of the DBMS. Data manager performs the following:

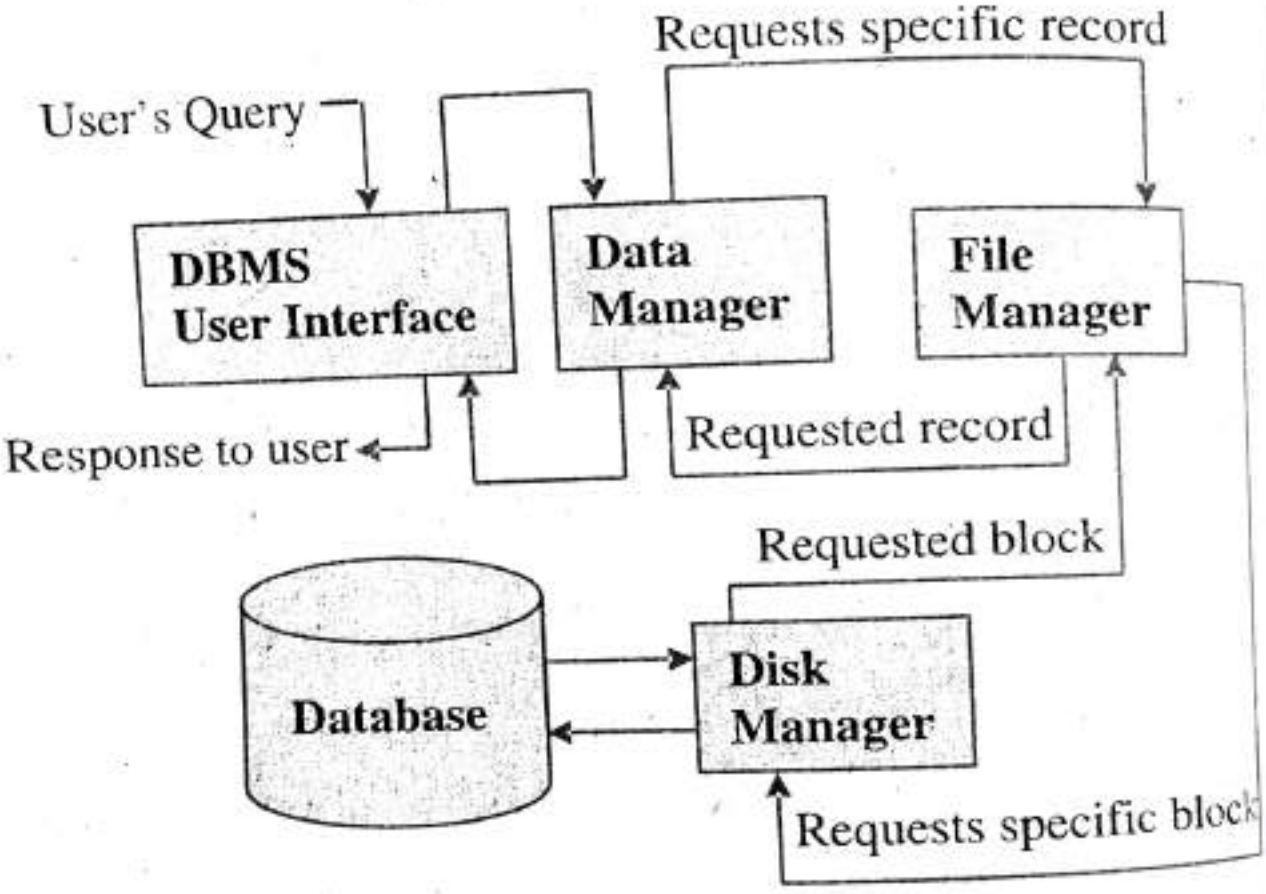
(a) Maintains the backup and recovery operations.

(b) *Converts the user queries from the user logical view to a physical file system.*

(c) *Maintains the data consistency, integrity and security.*

(d) *Organizes to provide the synchronization in the simultaneous operations performed by concurrent users.*

(e) *Interfaces with the file manager.*



3. File Manager

File manager:

- *Manages the allocation of file space on disk storage.*
- *Manages data structure to represent information stored on disk.*

- ***Receives the request from the data manager for a specific physical record, locates the block containing the required record and then demands for block of data from the disk manager. After receiving the required block transmits the required record to the data manager.***

4. Disk Manager

Disk manager:

- ***Performs all the physical input and output operations.***
- ***Transfers the block requested by the file manager so that the file manager need not be aware of the physical characteristics of the storage media.***

5. Query Processor

Query processor is the component of the DBMS responsible for generating the best plan or strategy to process the query.

Query processor interprets the online user query and converts it into a series of efficient operations that are carried out one by one by the data manager for execution. The query processor has access to the following information stored in the data dictionary:

- ***Number of tuples in the relation.***
- ***Size of the record in bytes.***
- ***Number of blocks used to store relation.***
- ***Number of distinct values that appear in the relation.***

These statistics are used to estimate the cost of different access plans and thus help in selecting the best strategy to process the query.

6. Telecommunication system

Users remote or local, communicate with the computer system by sending and receiving message over communication line. These messages are routed by telecommunication system.

7. Data files

Data files store all the data portion of the database.

8. Data Dictionary

Data dictionary stores the information about the data in database. It contains the information about the entities, attributes, checks and validation. Data dictionary is an integral component of the DBMS and is used to control the database operations, data integrity and accuracy.

Q6.(d) Explain Components of DBMS environment.

Ans. DBMS environment has the following five major components:

1. Hardware
2. Software
3. Data
4. Database Users
5. Procedures

1. Hardware

The hardware is the actual computer system used for keeping and accessing the database. This can range from a single PC to a single mainframe or to a network of computers. Basic hardware components are:

- ***Hardware processor and main memory for the support of execution of the database system.***
- ***Magnetic discs, drums etc. on which the database physically resides.***
- ***On-line terminals for accessing and updating database files.***
- ***Database machines specifically designed to support a database system.***

2. Software

The software used for databases is known as DBMS. All requests made to database by users are handled by this software.

Q7.(a) Explain three level architecture of database system with suitable diagram. IGU BCA 2018

OR

Explain the three-level architecture of DBMS with the help of an example.

MDU BCA 2016, 2015

OR

Discuss architecture of DBMS with an example.

MDU BCA 2014

OR

What do you mean by Database System Architecture? Explain in detail.

MDU BCA 2012

OR

Describe the architecture of a database system. Why a database is described to be an integrated one?

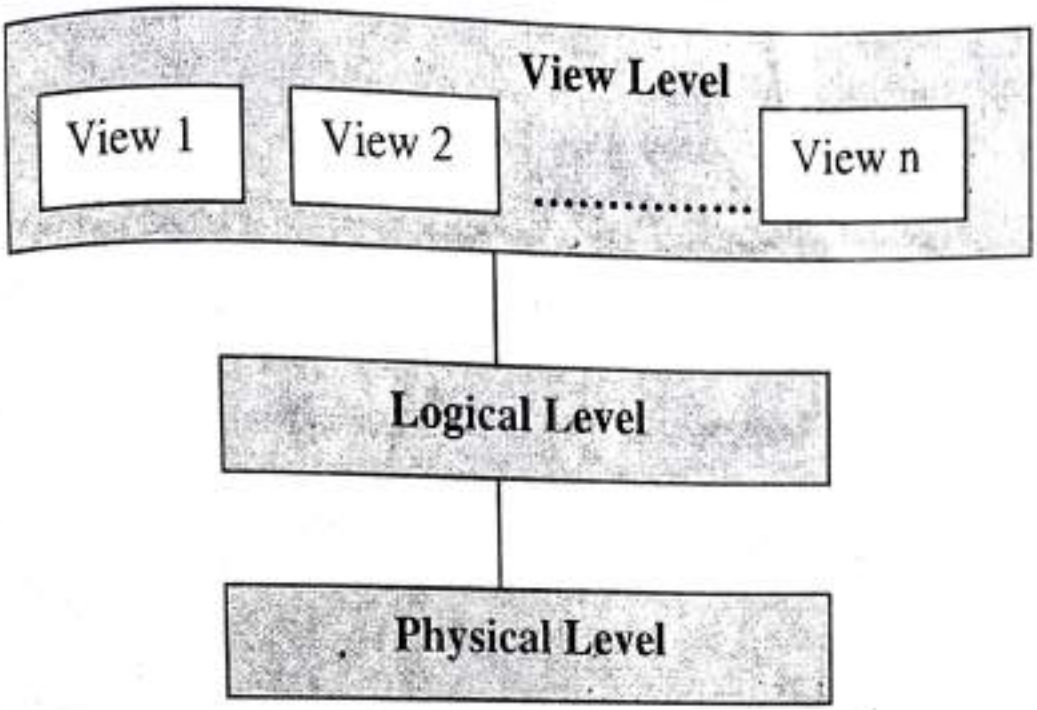
Ans. Three-level architecture of a DBMS

A DBMS provides three levels of data is said to follow three-level architecture. The goal of the three-schema architecture is to separate the user applications and the physical database. The view at each of these levels is described by a schema. The processes of transforming requests and results between levels are called mappings.

Thus, database system provides its users with an abstract view of the data, i.e., the system hides certain details of how data are actually stored in the databases so that the user interaction with the database is independent of physical data organization. So the main objective of this three-level architecture is to separate each user's view of the database from the way the database is physically represented.

The three levels or layers of DBMS architecture are:

1. Physical Level
2. Logical Level
3. View Level



1. Physical Level

This is the lowest level close to hardware. This level is also known as internal level or storage level.

It describes the physical layout of the data on the storage devices i.e. it describes how data is actually stored on the storage devices.

Since it contains the details of how the data are actually stored, it hides minimum information. It includes:

- ***Data structure.***
- ***File organizations used to store data on storage devices.***
- ***Database Definition Tables.***
- ***Definition of Access paths to files containing the data of the database.***

It interfaces with the operating system access methods to place the data on the storage devices, build the indexes, and retrieve the data and so on.

The security of data is also defined at this level. This view is seen by the system programmer and the system designers.

2. Logical Level

The middle level in the three level architecture is the logical level. It is the next higher level of abstraction.

It is the overall design of the database. It is a complete view of the data requirements of the organization.

Logical level is also called conceptual level. It hides the details of physical storage structures. At the logical level, following are defined:

- ***All record types.***
- ***Relationship existing among these records.***
- ***The constraints on the data.***
- ***Security and integrity information.***

It describes what data is stored without taking physical storage details. For example, the description of an entity should contain only data types of attributes like integer, real, character; and their length like maximum number of digits or characters; but not any storage considerations like the number of bytes occupied. Thus, it describes:

- ***Entities***
- ***Data types***
- ***Relationships***
- ***Constraints***

This view is seen by database administrator (DBA) who decides what information is to be kept in the database.

3. View Level

This is the highest level close to user. View level is also known as external view.

It describes only that part of the database, which is actually needed by the user.

User Interface is placed at this level. This level deals with the way in which individual users view data. Individual users are given different views according to their requirements. A number of users of a database system are not concerned with all the information contained in the database. Instead, they need a part of database. So they will usually access a subset of total data. At view level, parts of the database are described. Thus, view level provides many views of the same database.

View level increases the security of the database. It does not allow the user to access the unnecessary parts of the data. For example, in an examination branch of the University, a clerk can access student information but not staff information. In addition different views may have different representations of the same data. For example, one user may view data in the form (day, month, year) while another may view date as (month, day, year).

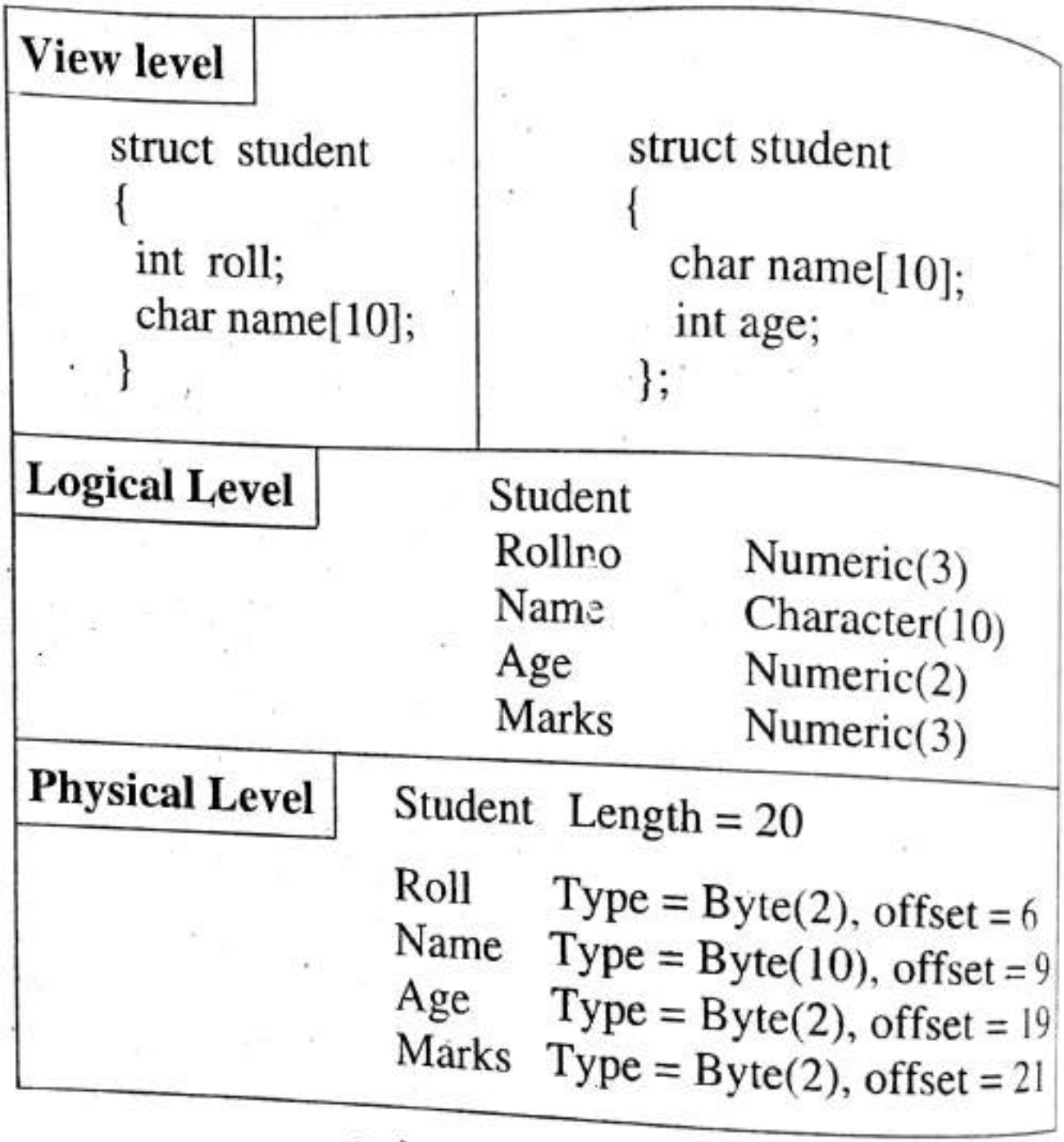
Why a database is described to be an integrated one?

A database is desired to be an integrated one because of the following:

- ***To avoid duplicacy of data.***

- *To make database a shareable resource.*
- *The database is completely controlled by a software package called DBMS.*

For example, the three levels of architecture can be given as:



Q7.(b) What do you mean by Schema and Instance? Explain difference between external, conceptual and internal schemas.

Ans. Schema

The overall design of the database is called schema, which is specified during database design and changes very rarely.

Database system has several schemas corresponding to each level of abstraction. At the physical level one physical schema, at logical level one logical schema and at the view level, several sub-schemas. In general, database system supports:

- ***One physical / internal schema.***
- ***One logical / conceptual schema***
- ***Several subschema / external schema***

Internal schema: Internal schema is a complete description of the internal model, containing definition of stored records, the methods of representation, the data fields and the indexes used.

Conceptual schema: Conceptual schema describes all the entities, attributes and relationship together with integrity constraints.

External schema: External schema corresponds to different views of the data. 42

Instance

Data in the database is of dynamic nature. It keeps changing from time to time. The collection of data stored in the database at a particular moment of time is called the instance of a database. It is also called database state or snapshot.

Difference between external, conceptual and internal schemas:

External Schema	Conceptual Schema	Internal Schema
1. External schema describes the external view of three level architecture of DBMS.	1. Conceptual schema describes the conceptual view of three level architecture of DBMS.	1. Internal schema describes the internal view of three level architecture of DBMS.
2. The external schema defines a view or views of the database for particular users.	2. The conceptual schema defines the stored data structures in terms of the database model used.	2. The internal schema defines how and where data are organized in physical data storage.
3. It includes different views of data.	3. It describes all the entities, attributes and relationship together with integrity constraints.	3. It is a complete description of the internal model and contains definition of stored records.
4. It contains the method of describing the objects in the external view from the objects in the conceptual view.	4. It contains the method of deriving the objects in the conceptual view from the objects in the internal view.	4. It contains the method of representing the data fields and the indexes used.

Q8.(a) What do you mean by data independence? Why it is important to maintain data independence in a database? MDU BCA 2018

OR

What is data independence and why is it important? MDU BCA 2016

OR

Describe data independence with examples. MDU BCA 2014

OR

What do you mean by Data Independence? Also explain Logical and Physical data independence. MDU BCA 2012

Ans. Data Independence

Data Independence is the capacity to change the schema at one level of a database system without having to change the schema at the next higher level of abstraction.

A major objective for three-level architecture is to provide data independence which means that upper levels are unaffected by changes in lower levels.

The two levels of data independence are:

1. Physical Data Independence
2. Logical Data Independence

1. Physical Data Independence

It is the capacity/capability to change the physical schema without having to change to logical schema.

Changes to internal schema may be needed because some physical files have to be reorganized. The change would be absorbed by the mapping between the logical

and physical levels. Physical data independence is achieved by the presence of the internal level of the database and the mapping or transformation from the conceptual level of the database to the internal level.

2. Logical Data Independence

It is the capacity to change the logical schema without having to change view level.

The conceptual schema may be changed 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). Only the view definition and the mappings need be changed in a DBMS that supports logical data independence. Changes to constraints can be applied to the conceptual schema without affecting the external schemas or application programs.

Thus, changes at this level are necessary whenever logical structure is changed. The change would be absorbed by the mapping between the view and logical levels. Logical data independence is more difficult to obtain than physical data independence, as it requires the flexibility in the design of database.

Importance of data independence

Data independence is an important characteristic of DBMS as it allows changing the structure of the database without making any changes in the application programs that use the database. Thus, application programs do not depend on any particular physical representation or access technique of the database. Thus, it is important to maintain data independence in a database

Q8.(b) What is the difference between physical and logical data independence?

MDU BCA 2016, 2015

Ans. Following are the differences between physical and logical data independence:

Physical data independence	Logical data independence
1. It is the capacity or capability to change the physical schema without having to change to logical schema.	1. It is the capacity to change the logical schema without having to change view level.
2. The change would be absorbed by the mapping between the logical and physical levels.	2. The change would be absorbed by the mapping between the view and logical levels.
3. It is not so complex.	3. It is complex.
4. It can change internal schema without changing conceptual schema.	4. It can change conceptual schema without having to change external schema.
5. Modifications in physical level occur occasionally whenevereas there is a need to improve performance.	5. Modifications in logical level occur frequently more than that in physical level, whenever there is an alteration in the logical structure of the database.

Q8.(c) What is mapping? Also explain Conceptual / Internal mapping and External / Conceptual mapping.

OR

What types of mapping are used in database systems?

Ans. Mapping

Mapping is the process of transforming requests and results between any of the two levels of logical architecture of a DBMS.

Programs refer to an external schema and are mapped by the DBMS to the internal schema for execution. Data extracted from the internal DBMS level is reformatted to match the user's external view.

Types of mapping

Two types of mapping are used in database system with three different views:

1. Conceptual / Internal mapping
2. External / Conceptual mapping

1. Conceptual / Internal mapping

It exists between conceptual level and internal level. It defines the correspondence between the records and the fields of the conceptual view and the files and data structures of the internal view.

It specifies the method of deriving the conceptual record from the physical database. Many differences may exist between the conceptual and internal views. These differences can be:

- ***Representation of numeric values could be different in the two views. One view could consider a field to be numeric and the other could consider it as binary.***
- ***Representation of string data can be considered by the two views to be coded differently. One can perceive string data to be in ASCII code and the other can consider it to be in EBCDIC code.***
- ***The value for a field in one view could be computed from the values in one or more fields of the other view.***

These differences are indicated and resolved in the mapping.

Thus, to make the conceptual view stable, when the structure of the stored database is changed, the conceptual / internal mapping is also changed accordingly. So, conceptual /internal mapping provides physical data independence for the database.

2. External / Conceptual mapping

It exists between the external level and the conceptual level. It gives correspondence among the records and the relationships of the external and conceptual views.

An external/conceptual mapping defines a particular external view and conceptual view correspondence. It describes the contents of the database as perceived by the user or application program of that view.

The external view is an abstraction of the conceptual view, which in its turn is an abstraction of the internal view.

To make the external level stable, when the structure of the database is modified, the external / conceptual mapping also changes accordingly.

Q9.(a) What are the different ways of classifying a DBMS? MDU BCA 2017

OR

How DBMS can be classified into different categories on the basis of different criteria? Explain.

Ans. Classification of DBMSs

The DBMS can be classified into different categories on the basis of several criteria such as the data model they are using, number of users they support, number of sites over which the database is distributed and the purpose they serve.

1. Based on Data Models

Depending on the data model the DBMSs use, the DBMSs can be categorized as hierarchical, network, relational and object-oriented etc. The most popular data model in use today is the relational data model. Well known DBMSs like Oracle, MS SQL Server, DB2, MySQL support this model. Many current commercial DBMSs are based on relational data model.

Other traditional models are hierarchical data model, or network data model. Some of the old applications still run on the database systems based on these models.

In the recent years, the object-oriented data models have come up though these models do not have widespread use. Some examples of Object-oriented DBMSs are O2, ObjectStore or Jasmine. The relational databases being the most popular, the object oriented concept have been introduced in these databases. This led to the development of a new class of DBMSs called object-relational DBMSs.

2. Based on the number of users

Depending on the number of users supported by the system, DBMSs are divided into two classes:

- Single user system
- Multi-user system

A single user database system supports one user at a time. In this system the database resides on one computer and is only accessed by one user at a time. Here the user performs all the user roles such as designing, maintaining and manipulating the database as per the requirements.

Multi-user systems support multiple users concurrently. In this system multiple users can access the database simultaneously. In this DBMS the data is both integrated and shared.

3. Based on the ways database is distributed:-

Depending on the ways over which the database is distributed, DBMS is divided into two types:

- Centralized database Systems
- Distributed database Systems

With centralized database systems, the system is stored at a single site. Centralized database systems run on a single computer system. Both the database and DBMS software reside at the same computer site. The user interacts with the centralized system through a dummy terminal connected to it.

In distributed database systems the database and DBMS software are distributed in various sites connected by a computer network. Several computers located at different sites communicate with each other through

various communication media such as high speed network or telephone lines. Distributed database systems are of two types namely homogeneous and heterogeneous.

In homogeneous distributed database systems all sites use the same DBMS software. Data is exchanged between various sites and can be handled easily.

In heterogeneous distributed Database Systems different sites use different DBMS software. There is additional software to support data exchange between sites.

4. Based on purpose

Depending on the purpose the DBMSs serve, they can be categorized in two classes:

- General purpose
- Specific purpose

By definition, DBMS is a general purpose software system. However it can be designed for specific purposes. For example, airline or railway reservations. Specific purpose database systems come in the category of online transaction processing systems. Such systems can not be used for other applications without making major changes.

Q9.(b) Explain client server architecture to DBMS.

MDU BCA 2016

OR

Explain client server architecture.

MDU BCA 2015

Ans. Client server architecture

Client server architecture of database system has two logical components- client and server. A client is a computer system that sends request to the server connected to the network. A server is a computer system that receives the request, processes it and returns the requested information to the client. Client and server are usually present at different sites.

Clients are generally personal computers or workstations and server is large workstations, mini range or mainframe computer system. DBMS resides on server and performs all functions related to database management like data definition, manipulation, integrity, security etc. The applications and tools of DBMS run on one or more client platforms. These servers and client computers are connected to a network.

The client computers have user interfaces that help users to utilize the servers. Users can also run local applications on the client side. The processing power of the computer system at the user's end is utilized by processing the user- interface on that system.

Two approaches to implement client/server architecture are: two-tier architecture and three- tier architecture.

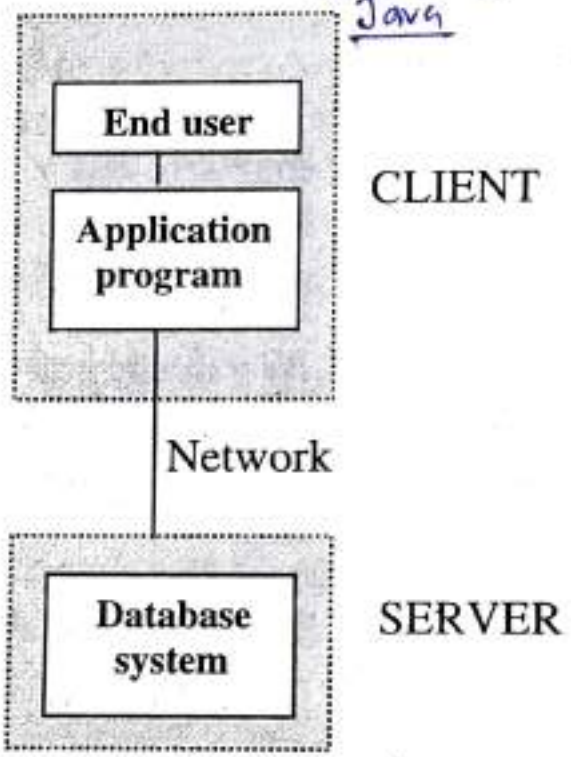
Two-tier architecture

In the two-tier architecture, the user interface and application programs are placed on the client side and

the database system on the server side. The application programs that reside at the client side invoke the DBMS at the server side.

Thus, in this type of architecture, the client invokes database system functionality at server machine through query language. ODBC, JDBC are used for interaction between the client and server.

Open Database connectivity
Java



(Two-tier architecture)

Advantages of two-tier architecture

Following are the advantages of two-tier architecture:

- ***Simplest to implement.***
- ***Performs well with moderate number of client.***

Disadvantages of two-tier architecture

Following are the disadvantages of two-tier architecture:

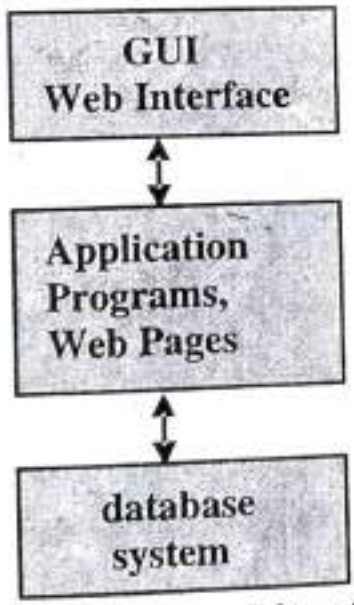
- ***It gives poor performance when large number of clients submits their requests.***

- ***Difficult to administer and maintain because when applications reside on the client, every upgrade must be delivered, installed and tested on each client.***

Three-tier architecture

In the three-tier architecture, an intermediate layer known as application server or web server is placed between the client and the database server.

The client communicates with the application server, which in turn communicates with the database server. When a client requests for information the application server accepts the request, processes it and sends corresponding database commands to database server. The database server sends the result back to application server which is converted into GUI format and presented to the client.



CLIENT

APPLICATIONS SERVER
OR WEB SERVER

DATABASE SERVER

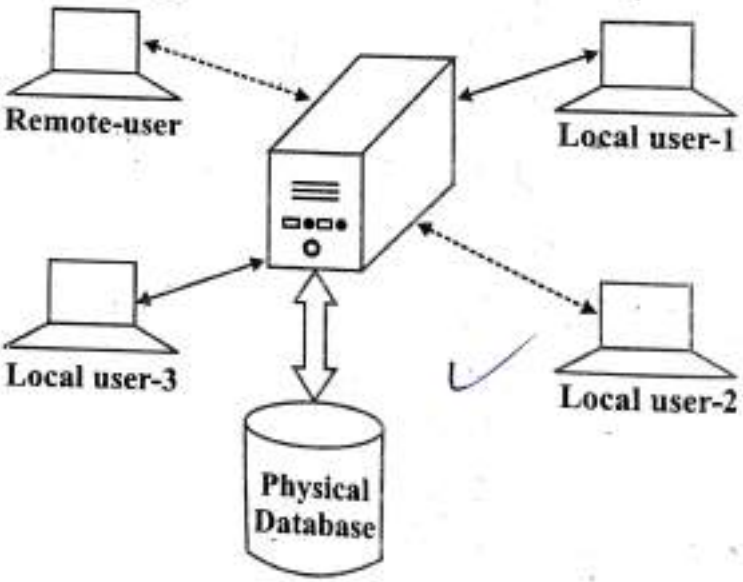
(Three-tier architecture)

The three tier architecture is used when an effective distributed client server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability and scalability, while hiding the complexity of distributed processing from the user.

Q9.(c) What is centralized DBMS? How is it useful and used? Explain its disadvantages with examples. **MDU BCA 2013**

Ans. The centralized database management system consists of a single processor together with its associated data storage devices and other peripherals. It is physically confined to a single location.

Centralized DBMS offers data processing capabilities to users who are located either at the same site or through remote terminals at geographically dispersed sites. The management of the system and its data are controlled centrally from any one or central site. This is shown below in the figure:



Usefulness of Centralized DBMS

- ***Single site provides high degree of security, concurrency, backup and recovery control.***
- ***Centralized control of data avoids unnecessary duplication of data and effectively reduces the total amount of data storage required.***

- *Centralized control can also ensure that adequate checks are incorporated in the DBMS to provide data integrity.*
- *It allows the sharing of data under its control by any number of application programs or users.*

Disadvantages of Centralized DBMS

Centralized DBMS has the following disadvantages:-

- *The site with the database can become a bottleneck.*
- *Performance degradation due to a growing number of remote locations over greater distances.*
- *Scalability problems associated with the physical limits imposed by a single location like power, temperature conditioning and power consumption.*
- *Possible availability problem. If the site with the database goes down, there can be no data access.*

Q10.(a) Define the term data model? Explain different types of data models. MDU BCA 2018

OR

Define the term data model. Explain different types of data model with examples. MDU BCA 2011

Ans. Data Model

Data model is a tool for describing data and relationships between data.

A number of models for data representation have been developed. The models can enforce a set of constraints to maintain the integrity of the data. Components of a data model are:

- ***A structural part which consists a number of rules for the construction of databases.***
- ***A manipulative part which defines the various operations that can be performed on the data.***

Data models should provide the basic concepts and notations that will allow database designers to communicate their understanding of the organizational data accurately.

The main purpose of a data model is to represent and make the data understandable.

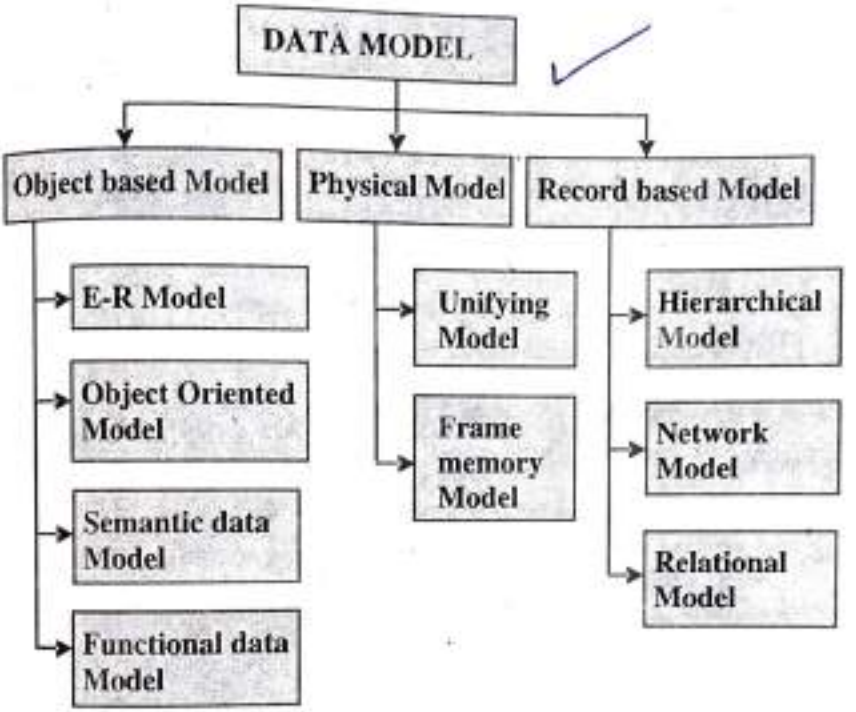
Thus in an organization, data model is an integrated collection of concepts for:

- ***Describing data.***
- ***Manipulating data.***
- ***Relationships between data and***
- ***A set of constraints to maintain the integrity of the data.***

Types of Data Model

The data models are basically divided into three groups:

1. Object based logical models
2. Physical models
3. Record based logical models



1. Object based logical models

These models are used to describe data at view level and logical level.

They use concepts like entities, attributes and relationships. They provide flexible structuring capabilities. Some of these are:

- Entity Relationship Model
- Object Oriented Model
- Semantic Data Model
- Functional Data Model

ER model has emerged as one of the main model to facilitate database design. The object oriented model is based on the collections of objects which describes attributes as the state of the object and the actions that are associated with the object as its behavior. For example dogs have state (name, colour, breed) and dogs have behavior (barking, fetching etc.). Cars have state (color, model etc.) and behavior (braking, changing gears, accelerating).

grammatic rules & regulations

Semantic data model considers the semantics at the time of operations on the data. Functional data model is categorized as small modules. Data is stored in database as per the module.

2. Physical Models

This data model is used to describe data at the physical level.

These models describe information like record structures, record ordering and access paths. Some of physical models are:

- Unifying Model
- Frame memory Model

3. Record based logical models

These models are used:

- ***To specify the overall logical structure of database.***
- ***To provide a higher-level description of the implementation.***

These are divided into three types:

- Hierarchical Model
- Network Model
- Relational Model

Q11.(a) Explain object based data models by giving suitable examples. IGU BCA 2018

OR

Describe object-based data model with examples. MDU BCA 2016, 2015, 2013

Ans. Object-based data model

Object based data model is a model that combines the DBMS characteristics and object oriented approach.

These models are used to describe a data at view level and logical level. They are characterized by the fact that they provide fairly flexible structuring capabilities and allow data constraints to be specified explicitly.

The various types of this model are:

1. Entity relationship model
2. Object oriented model
3. Semantic data model
4. Functional data model

1. Entity relationship model

Entity relationship data model is based on the perception of real world that consist of a collection of basic objects, such as entities, relationship among these objects.

An entity is a thing or object which shows its presence or existence in a nature. For example: student. It is represented by rectangle.

These entities are described in a database by a set of attributes. The attributes are the characteristics of an entity which may act as a field in tables of a database.

The relationship is an association between the entities. It is represented by diamond. The set of entities of a similar type is called as an entity set and set of relationships of similar type is called as relationship set.

The overall logical structure of database can be expressed graphically by an E-R diagram which is built on the following components:

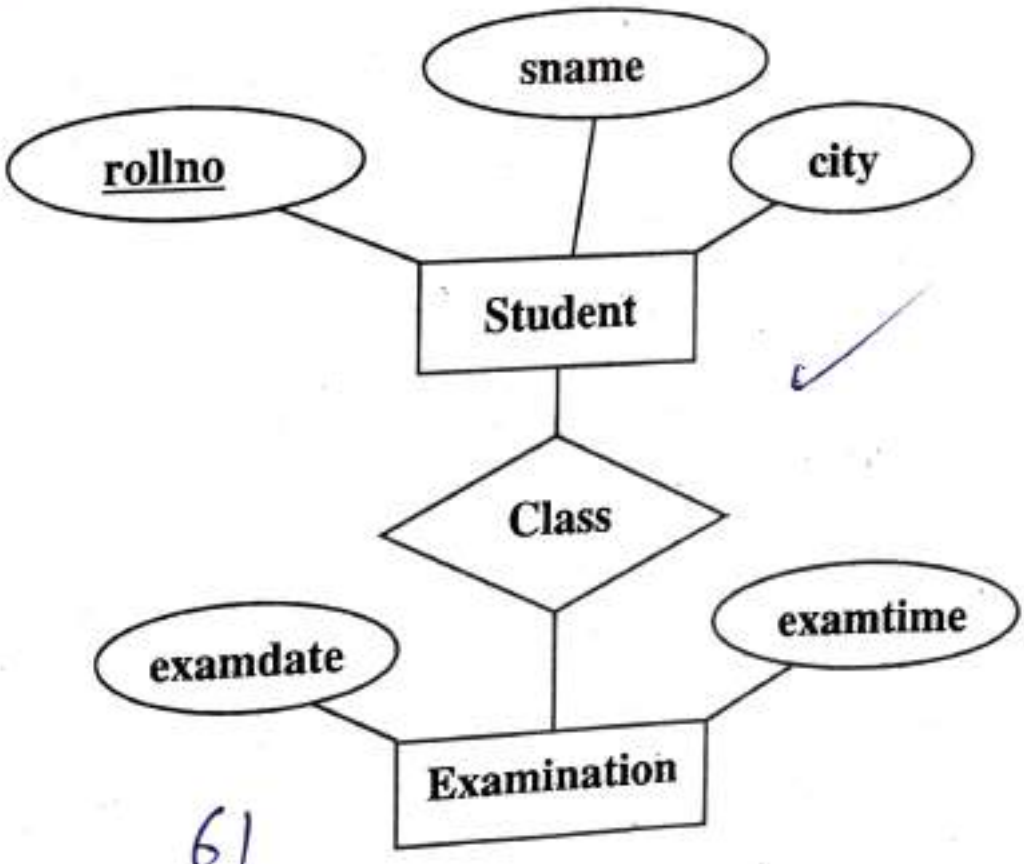
Rectangle: It represents the entity set.

Ellipse: It represents the attributes.

Diamond: It represents relationships among the entity sets.

Lines: Lines link attributes to entity sets and entity sets to relationships.

ER model represents the structure of a database graphically known as ER diagram. It can be shown as:



2. Object oriented model

Object oriented model is completely based on object oriented approach. This model is based on the collections of objects. Each object is an independent entity used to represent the real world thing. An object also contains a body of codes called as a method.

Similar values and methods of an object are grouped together to form a class. A class may be viewed as a type definition for objects, this combination of data and methods comprising a type definition which is similar to a language abstract data type.

The one way in which one object can access the data of another object is by invoking a method of that object. This action is called as sending a message of data. Unlike entities in a ER model each object has its own unique identity, independent of the value that it contains thus two objects containing the same values are nevertheless distinct.

3. Semantic data model

Semantic data model emphasize on semantic while working with data in database. Semantic means grammatical rules and regulations that apply on data. Each time when the data is operated in DBMS, the DBMS checks these semantics. It maintains the database integrity.

4. Functional data model

Functions are also called modules or subroutines. Functional model uses the predefined modules or subroutines while working with data in database. Functions store the data according to their needs and then they are checked and implemented by the DBMS.

Q11.(b) What is a hierarchical model? Also explain the various problems faces by this model in representing relationships.

Ans. Hierarchical Model

A model containing a collection of records connected in the form of a tree is called hierarchical model.

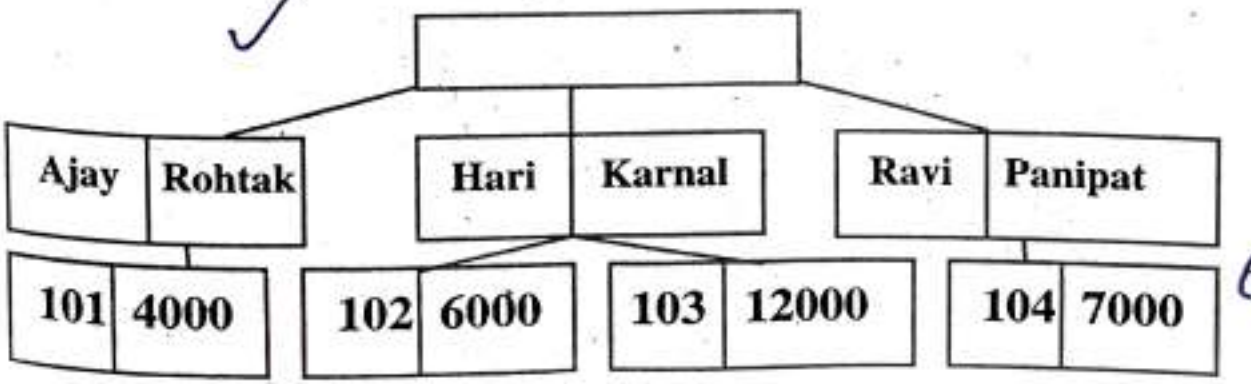
Hierarchical model is the oldest of the three record based logical models. One of the first hierarchical databases Information Management System (IMS) was developed by North American Rockwell company and IBM.

Hierarchical model uses the tree structure to organize the data. It shows the collection of records in the form of tree. The records work as nodes of tree and are also called segments. This structure shows the relationships among data as in the real world using tree structure, thus it is very efficient model.

It shows one to many relationships. It does not allow many to many relationships. It uses one root as parent and many branches as children. It describes the parent to child relationship that is one to many. Each child has only one parent. Child to parent relationship is one to one.

For example, consider a Client_account relationship in a banking system.

Sample database



There are two record types. One is Client with fields name and city, the other is Account with fields client_no and balance. Figure shows a sample database. It shows client Ajay has account no. 101, client Hari has account no.102 and 103, client Ravi has account no. 104. Here the root of the tree is a dummy node. This type of rooted tree is called a database tree and hence collection of such database trees forms a forest.

Various problem faces by this model in representing relationships

Various problem faces by this model in representing relationships are:

1. Implementation Complexity
2. Database management problem
3. Lack of Structural Independence
4. Operational anomalies
5. Inflexible
6. Program complexity
7. Implementation limitations
8. Hard code

1. Implementation Complexity

It is simple and easy to design but it is complex to implement. It requires proper knowledge of data storage characteristics.

2. Database management problem

If one makes any changes in the database structure then there is need to make changes in the entire application program that access the database. Thus to manage the database as well as application program is very difficult.

Q12.(a) What is mapping cardinalities? Explain various types of mapping cardinalities.

MDU BCA 2018

Ans. ***Mapping cardinalities or cardinality ratios express the number of entities to which another entity can be associated via a relationship.***

Mapping cardinalities are the most useful in describing binary relationship sets. For binary relationship set R between entity sets A and B, the mapping cardinality must be one of the following:

- **One to One:** An entity in A is associated with at most one entity in B and an entity in B is associated with at most one entity in A.
- **One to Many:** An entity in A is associated with any number of entities in B. An entity in B, however, can be associated with at most one entity in A.
- **Many to One:** An entity in A is associated with at most one entity in B. An entity in B, however, can be associated with any number of entities in A.
- **Many to Many:** An entity in A is associated with any number of entities in B, and an entity in B is associated with any number of entities in A.

For example, if the company policy does not allow more than 100 employees in a department, the cardinality rule governing the DEPARTMENT-EMPLOYEE relationship is expressed as "One department can have a maximum of 100 employees", Also the company policy allows each employee to join a maximum of 2 courses at a time. At present the company offers 10 courses. The cardinality is indicated by placing the appropriate numbers beside the entities as shown below:

Q12.(b) What do you mean by attributes? Describe the types of attributes that occur in the E-R model by giving suitable examples.

Ans. Attributes

Attributes are the properties or characteristics of an entity i.e. every entity has some basic attributes that characterize it.

In simple words, an attribute of an entity is a particular property that describes the entity.

For example, an employee entity may have employee number, name, address etc. as attributes. A student entity may have rollno, name, class, semester as attributes.

The attribute value is the actual data contained in the attribute. For example, 101, RAVI, BCA, I are the actual data stored in the attributes rollno, name, class and semester.

Thus, an attribute is description that is used to identify, qualify, classify or otherwise express the set of an entity occurrence. For each attribute, set of permitted values are available called as domain. The attribute which uniquely defines the occurrence of an entity is called primary key.

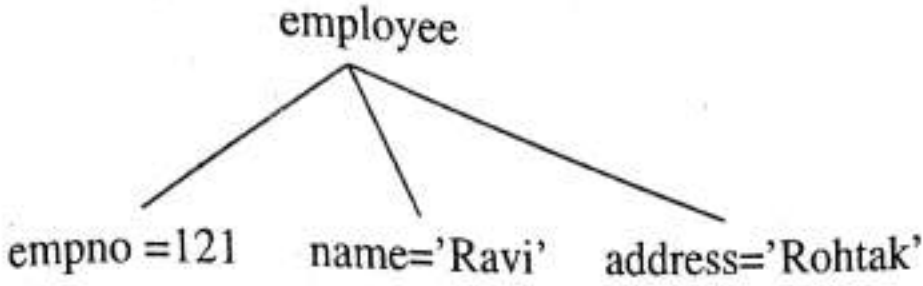
The attribute values describing an entity will constitute a significant portion of the data stored in the database.

Several types of attributes occur in the E-R model. These are:

1. Simple and Composite attribute

In simple attribute each entity has a single atomic value for the attribute. In other words the attribute which cannot be divided into smaller subparts is called simple attribute.

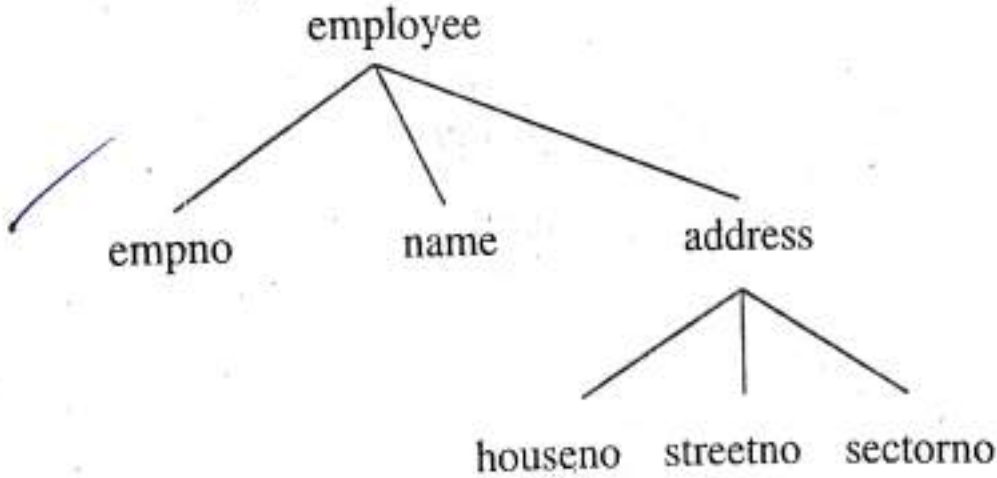
For example, shown below is an entity employee with its attribute values



Thus, simple attribute represents a single property, i.e., it is not divided into subparts. Simple attributes are sometimes called atomic attributes.

The attributes, which can be divided into smaller independent attributes, are called composite attributes.

For example, composite attribute address can be further divided into houseno, streetno, sectorno attributes.



Composite attribute appears as a hierarchy and helps to group together related attributes.

2. Single-valued and multi-valued attributes

An attribute which has a single value for a particular entity is known as single valued attribute.

For example, the attribute 'empno' can have a single number only.

An attribute which has more than one values is called multivalue attribute.

For example, PreviousDegrees attribute of a Student entity may have more than one values B.A. and M.A. etc.

A multiple value attribute is enclosed between braces as {PreviousDegrees}

3. Null attributes

A null value is used when an entity does not have a value for an attribute.

For example, if a student does not have any degree then degree attribute value will be null or if it is not known it is again null.

4. Stored attribute & Derived attribute

An attribute which cannot be derived from another attribute(s) is known as stored attribute.

If the value of attribute is derived from the values of other related attributes it is called derived attribute.

For example, consider the employee entity set with attributes start_date and employment_period. start_date represents the joining date and employment_period represents his total service. The value of employment_period can be derived from the value for start_date and the current_date. Here start_date is referred as stored attribute.

Q13.(a) What is ER diagram? Draw the ER diagram of library management system.

MDU BCA 2016

Ans. ER diagram

An ER diagram is a graphical tool that demonstrates the interrelationships among various entities of a database. It is used to represent the overall logical structure of the database.

Thus a visual way to represent a database is with an entity-relationship (E-R) diagram. While designing ER diagrams, the emphasis is on the schema of the database and not on the instance. **This** is because the schema of the database is changed **rarely;** however, the instances in the [illegible] and relationship **sets** change frequently. So, ER diagrams are **more useful** in designing the database.

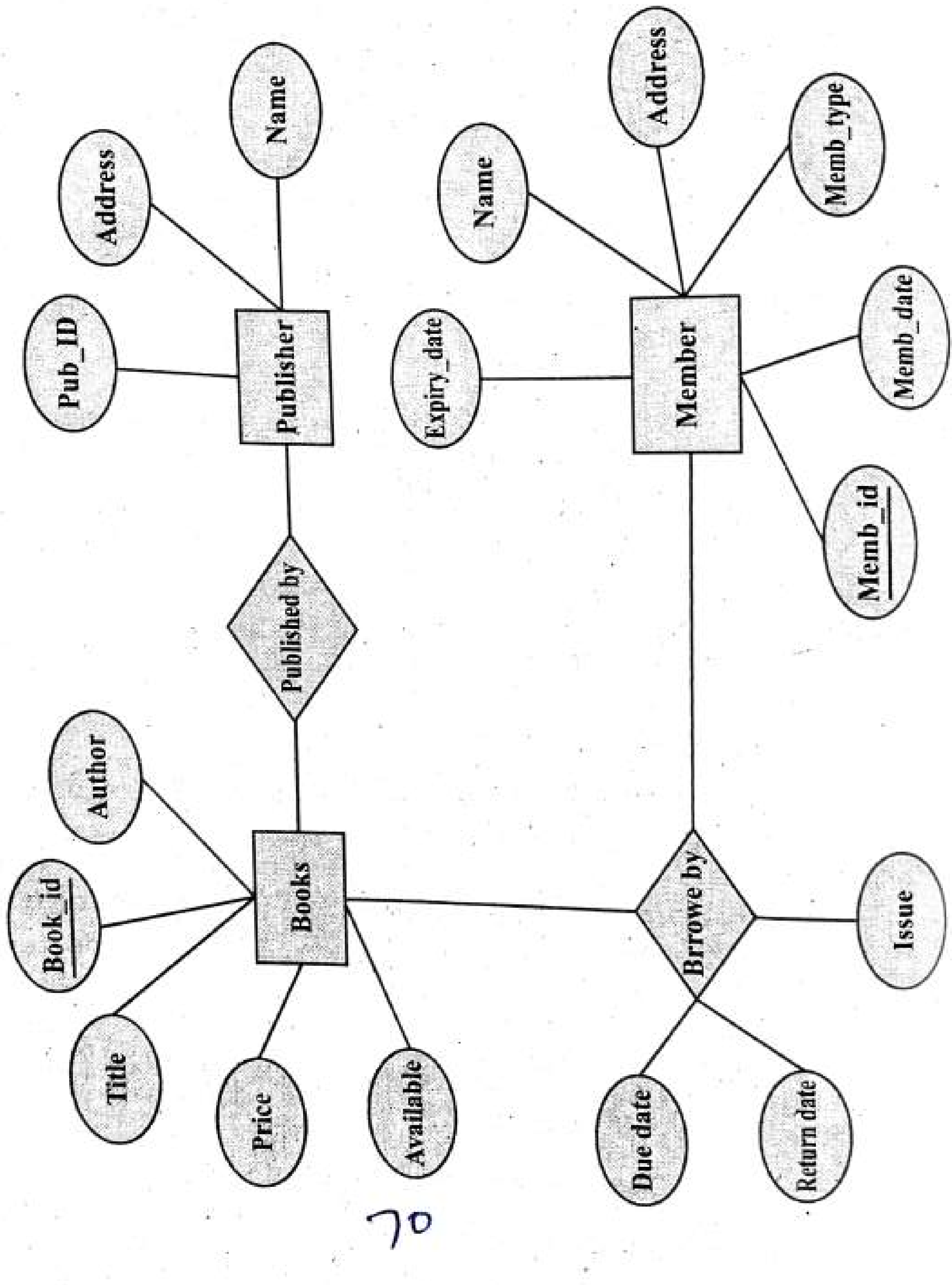
An ER diagram serves several purposes as given below:

- ***It is used to communicate the logical structure of the database to the end users.***
- ***It helps the database designer in understanding the information to be contained in the database.***
- ***It serves as a documentation tool.***

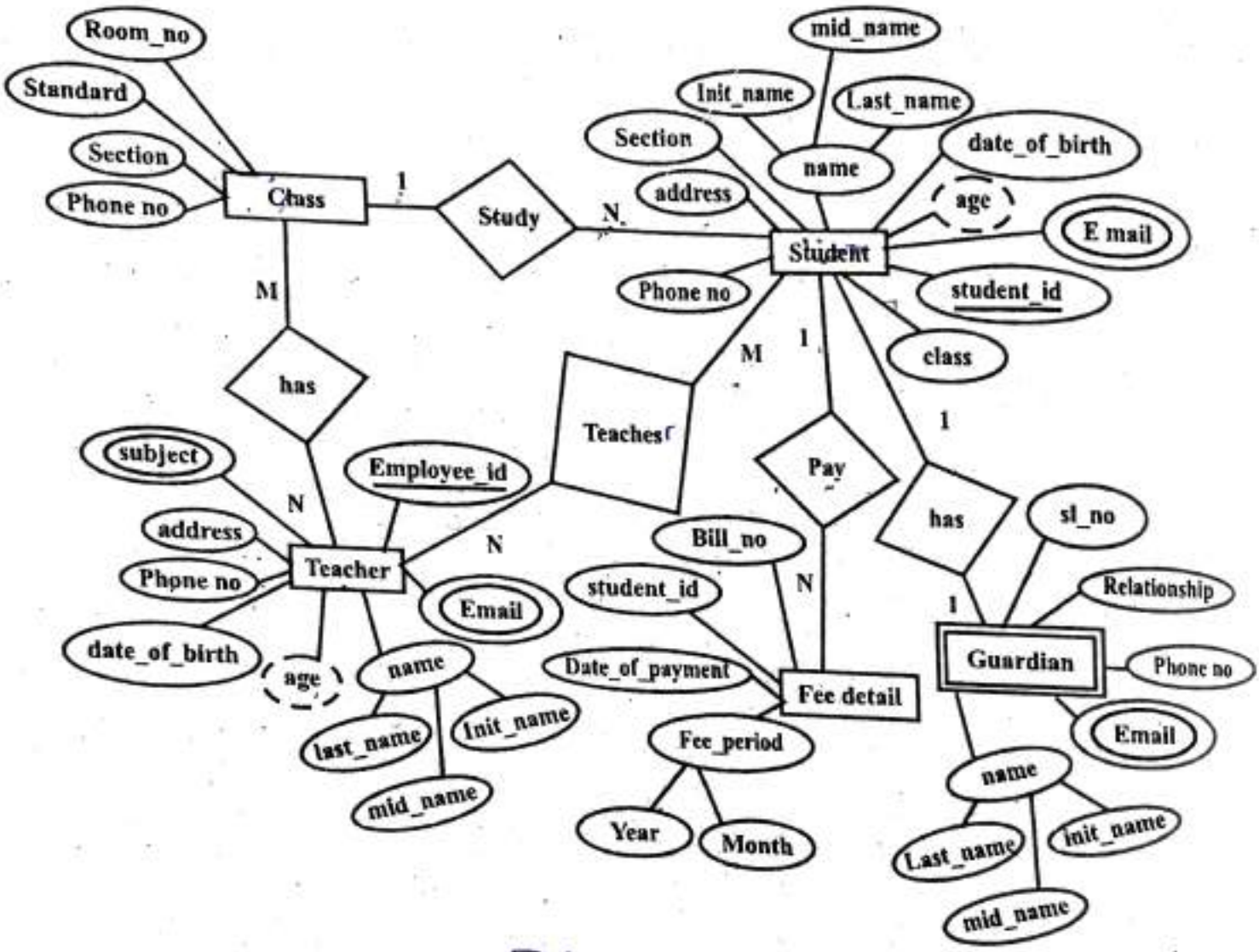
Thus, ER diagram is a graphical logical structure of a database which represents:

- ***All the entities involved in the logical structure.***
- ***Constituents of the entities.***
- ***Relationship between the entities.***
- ***Types of mapping between relations.***

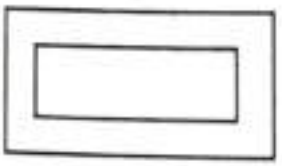
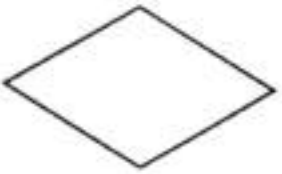
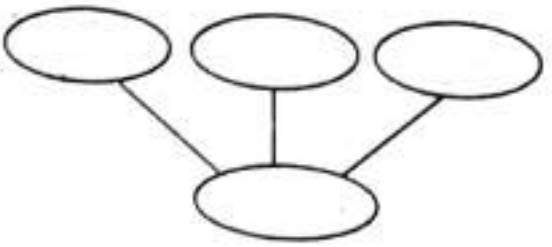
ER diagram of library management system



ER diagram of school management system



The symbols used to design an ER diagram are shown below:

Symbol	Meaning
	Entity Type
	Weak Entity Type
	Relationship Type
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Entity-2 is totally participating in Relationship type R

Q13.(d) **Write note on Data Abstraction and Data Integrity/Integration.** **MDU BCA 2010**

Ans. **Data Abstraction**

Abstraction is hiding of implementation details from the users.

Since non-computer professionals also use database, thus database complexity must be hidden from them. Moreover users are of different capabilities and data needed by them is also of different levels. Thus for smooth running of database systems, hide the complexity of database system from the end users.

Thus abstraction principle is applied to the DBMS and DBMS hides the complex details of how the data is stored in DBMS. It allows different users to concentrate only on their respective subparts.

The various levels of abstraction are:

1. Physical Level
2. Logical Level
3. View Level

1. Physical Level

This is the lowest level close to hardware. This level is also known as internal level or storage level.

It describes the physical layout of the data on the storage devices i.e. it describes how data is actually stored on the storage devices.

Since it contains the details of how the data are actually stored, it hides minimum information. It includes:

- ***Data structure.***

- ***File organizations used to store data on storage devices.***
- ***Database Definition Tables.***
- ***Definition of Access paths to files containing the data of the database.***

It interfaces with the operating system access methods to place the data on the storage devices, build the indexes, retrieve the data and so on. The security of data is also defined at this level. This view is seen by the system programmer and the system designers.

2. Logical Level

The middle level in the three level architecture is the logical level. It is the next higher level of abstraction.

It is the overall design of the database. It is a complete view of the data requirements of the organization.

Logical level is also called conceptual level. It hides the details of physical storage structures. At the logical level, following are defined:

- ***All record types.***
- ***Relationship existing among these records.***
- ***The constraints on the data.***
- ***Security and integrity information.***

It describes what data is stored without taking physical storage details.

For example, the description of an entity should contain only data types of attributes like integer, real, character; and their length like maximum number of digits or characters; but not any storage considerations like the number of bytes occupied.

Thus, it describes:

- ***Entities***
- ***Data types***
- ***Relationships***
- ***Constraints***

This view is seen by database administrator (DBA) who decides what information is to be kept in the database.

3. View Level

This is the highest level close to user. View level is also known as external view.

It describes only that part of the database, which is actually needed by the user.

User Interface is placed at this level. This level deals with the way in which individual users view data. Individual users are given different views according to their requirements. A number of users of a database system are not concerned with all the information contained in the database. Instead, they need a part of database. So they will usually access a subset of total data. At view level, parts of the database are described. Thus, view level provides many views of the same database.

View level increases the security of the database. It does not allow the user to access the unnecessary parts of the data.

For example, in an examination branch of the University, a clerk can access student information but not staff information. In addition different views may have different representations of the same data. For

example, one user may view data in the form (day, month, year) while another may view date as (month, day, year).

Data Integrity/Integration

Data Integrity means data in database is always accurate. It also implies that incorrect information cannot be stored in the database.

In a database system, better data integrity is obtained by imposing consistency constraints on the database.

For example:

1. A student database has the information regarding marks obtained in an examination. A student cannot obtain marks more than100 - the maximum marks in one subject. Integrity constraints can be specified so that database will accept marks only in the range of 0 to 100 per subject.
2. The database of a college till graduation will have the data for BA, BSc, BCA, Bcom classes only. If a user enters the data for MSc/MCA class then this incorrect/invalid data entry must not be stored in the database. To enforce this integrity constraints are imposed easily on the database instead of file system where these constraints are imposed on all the applications separately.

Q14.(a) Define the term 'data model'. Explain Hierarchical model and Network model in detail. **MDU BCA 2016**

Ans. Data Model

Data model is a tool for describing data and relationships between data.

A number of models for data representation have been developed. The models can enforce a set of constraints to maintain the integrity of the data.

Components of a data model are:

- ***A structural part which consists a number of rules for the construction of databases.***
- ***A manipulative part which defines the various operations that can be performed on the data.***

Data models should provide the basic concepts and notations that will allow database designers to communicate their understanding of the organizational data accurately.

The main purpose of a data model is to represent and make the data understandable.

Thus in an organization, data model is an integrated collection of concepts for:

- ***Describing data.***
- ***Manipulating data.***
- ***Relationships between data and***
- ***A set of constraints to maintain the integrity of the data.***

Hierarchical Model

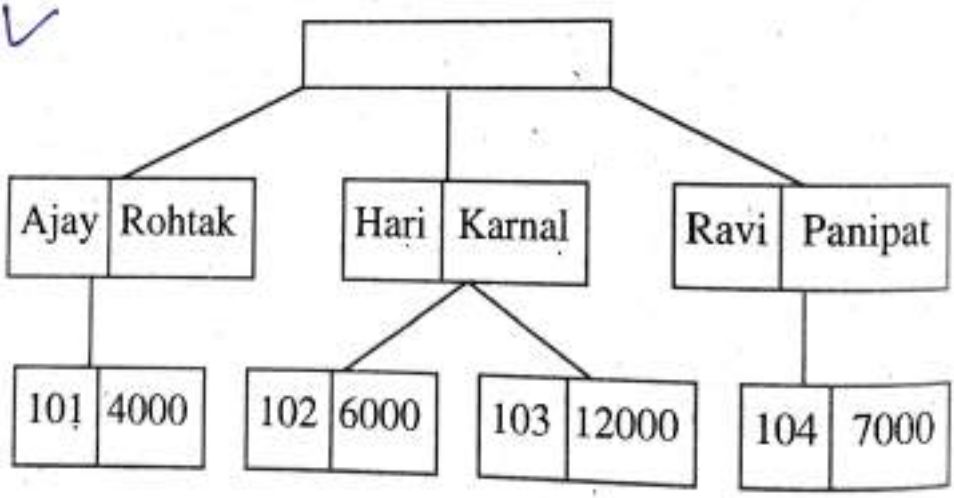
A model containing a collection of records connected in the form of a tree is called hierarchical model.

Hierarchical model is the oldest of the three record based logical models. One of the first hierarchical databases Information Management System (IMS) was developed by North American Rockwell company and IBM.

For example, consider a Client_account relationship in a banking system. There are two record types. One is Client with fields name and city, the other is Account with fields client_no and balance.

Figure shows a sample database. It shows client Ajay has account no. 101, client Hari has account no.102 and 103, client Ravi has account no. 104

Sample database



Here the root of the tree is a dummy node. This type of rooted tree is called a database tree and hence collection of such database trees forms a forest.

Operations on Hierarchical Model

The operations that can be performed on Hierarchical model are:

(i) Insert operation

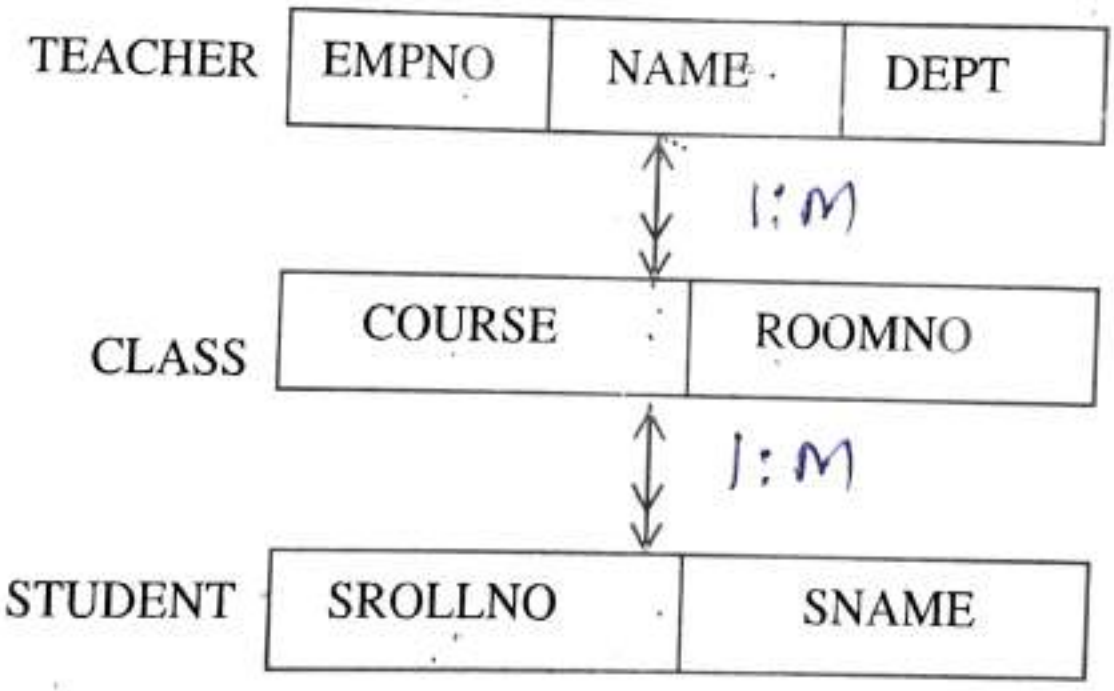
(ii) Delete operation

(iii) Update operation

To understand the operations on Hierarchical model, consider the following data structure diagram for a tree representing the TEACHER, CLASS & STUDENT.

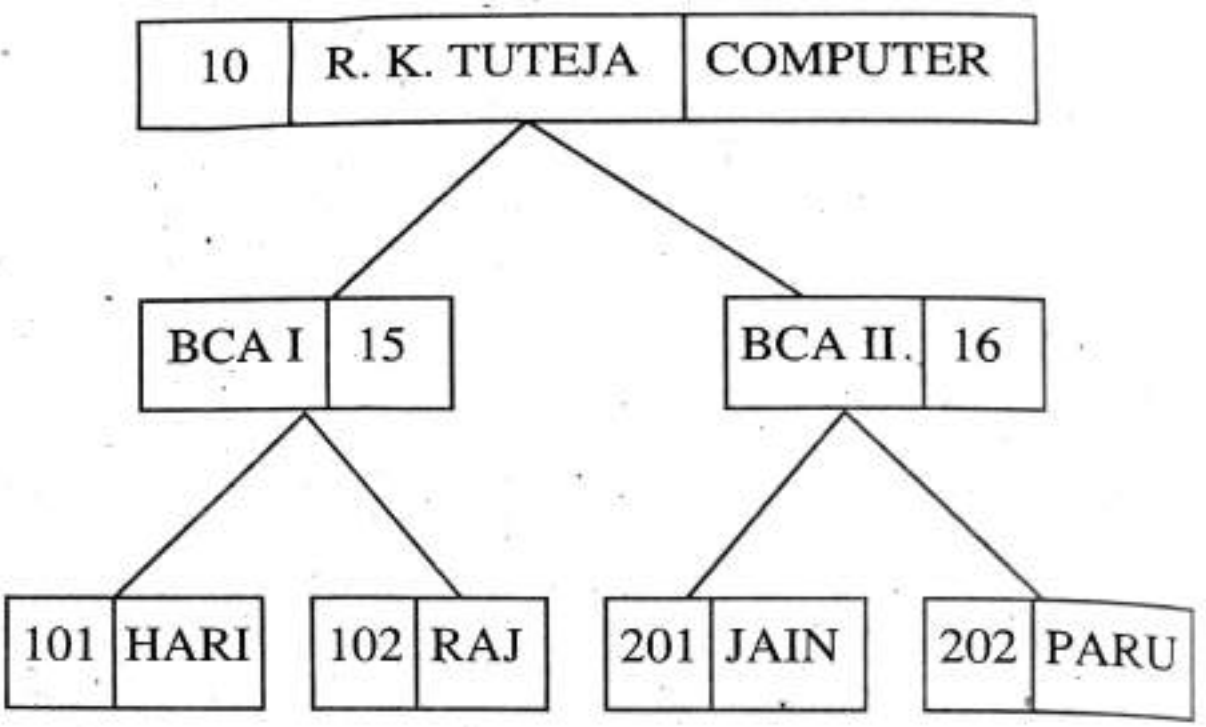
Here, the three record types are:

- TEACHER with fields EMPNO, NAME, DEPT
- CLASS with fields COURSE, ROOMNO
- STUDENT with fields SROLLNO, SNAME



TEACHER is the root node. CLASS as a child of TEACHER which shows there is one to many relationship between each TEACHER record and its associated CLASS records. Similary STUDENT is a child of CLASS.

Figure shows a sample database:



(i) Insert operation

It is not possible to insert a new class BCA III unless some teacher is available at root level. A node cannot exist without a root. Thus insert anomaly exists for those children which has no corresponding parents.

(ii) Delete operation

In hierarchical model, deletion of parent causes the deletion of children records. For example to remove the class BCA II causes the deletion of student with SROLLNO 201 and 202. Thus deletion is difficult.

(iii) Update operation

Update operation is time consuming because:

- ***Search problem will occur.***
- ***Multiple updations for different occurrences.***

Thus, hierarchical model suffers from insertion anomalies, deletion anomalies and updation anomalies.

Advantages of Hierarchical Data Model

Advantages of hierarchical data model are:

- *Simple and easy to use.* Any 3
- *Promotes data integrity as child is referred by its parents.*
- *Easy access to data.*
- *First model that is provided and enforced, data security by the DBMS.*
- *It is very efficient when large number of 1:N relationship exist and when user require large number of transaction using data whose relationship is fixed.*
- *Data with hierarchical relationships can be mapped on this model.*

Limitations of Hierarchical Model

Limitations of Hierarchical model are:

- *Non-hierarchical relationships are difficult to represent on this model. Thus it is inflexible.*
- *It is complex to design, develop and implement. The database designers must have complete knowledge of the physical data storage characteristics.*
- *The processing is sequential along the branches of the tree. Thus it takes more time to access the nodes.*
- *Data independence is limited because of structured dependence.*
- *Requires extra storage space for the pointers from parents to the children.*

- ***Deletion of parent node deletes its children nodes.***
- ***Changes in relationship are time-consuming and expensive.***
- ***Many to many relationship which is common in real life is difficult to implement.***

Examples of Hierarchical Model

IBM's, Information management system (IMS) and system 2000 are examples of Hierarchical database management system.

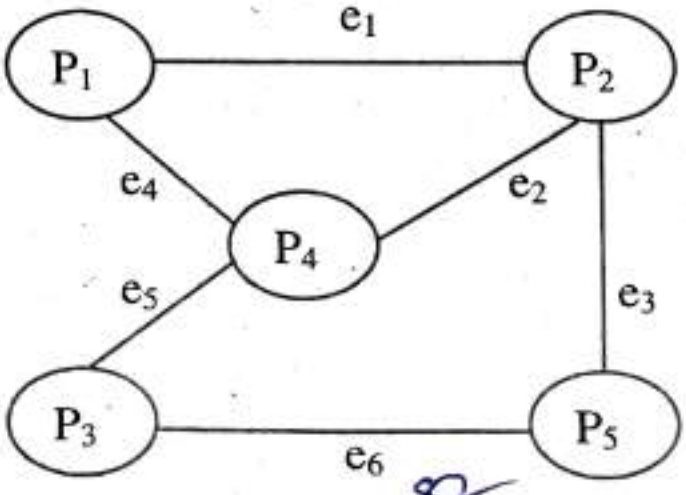
Network Model

A network model is a collection of records that are connected to each other through links. This model is an improvement over the hierarchical model.

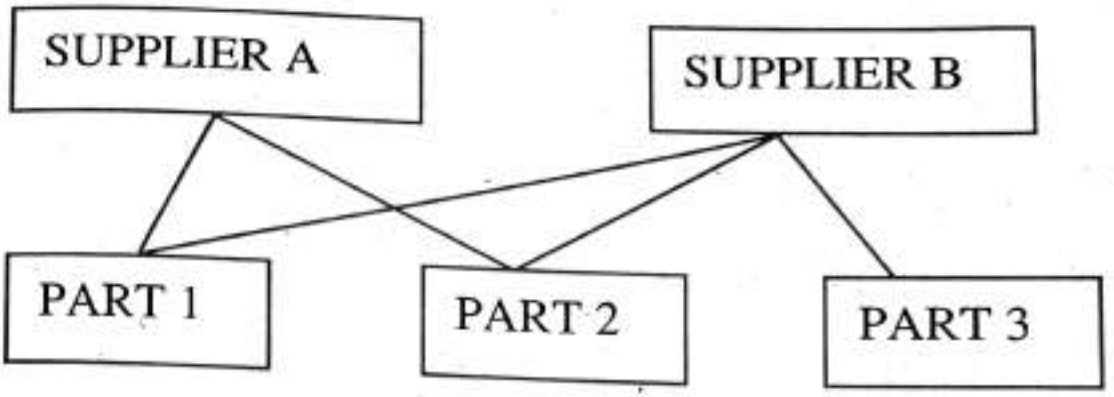
It differs from a hierarchical model as:

- ***No concept of root node.***
- ***Many-to-many relationship.***
- ***A node can have more than one parent node.***

Figure shows a network consisting of nodes P_1, P_2, P_3, P_4, P_5 and edges e_1, e_2, e_3, e_4, e_5 and e_6.



For example, a network structure with only two record types SUPPLIER and PART is shown below:



Operations on Network Model:-

The operations that can be performed on Network model are:

(i) Insert operation

(ii) Delete operation

(iii) Update operation

(i) Insert operation

From the figure, it is clear that new supplier or part can be easily inserted.

To insert a new record, create a new record occurrence. For example SUPPLIER C can be inserted in network model that does not supply any part. Similarly a new part can be inserted which is not supplied by any supplier.

(ii) Delete operation

Delete operation is very simple. To delete the information of any record occurrence that record can be deleted by removing the corresponding pointers and connectors.

For example to remove PART 1, delete the connector record occurrence linking this part.

(iii) Update operation

Updating a record is also easier process. For example suppose SUPPLIER A supplies PART 3 in place of PART 1. This change can be done simply by changing link of SUPPLIER B from PART 1 To PART 3.

Thus Network model does not suffer from the insert anomalies, delete anomalies and update anomalies. But since each above operation needs the modification of pointers, this makes the network model complicated and complex.

Advantages of Network data Model

Advantages of network data model are:

- ***Many-to-many relationship can be implemented.***
- ***It is efficient and flexible.***
- ***Data integrity i.e., it does not allow a member to exist without an owner.***
- ***Easier access as compared to hierarchical data model.***
- ***Simple and easy to design.***
- ***This model is based on the standards formulated by the DBTG. These standards included DDL (Data Definition Language) and DML (Data Manipulation Language) and thus enhancing database administration and portability.***

Limitations of Network Data Model

Limitations of network data model are:

- *Complexity increases as the number of links increases.*
- *Extra storage memory is required for pointers.*
- *General purpose Query facility is not available.*
- *Maintenance of this model is expensive because graph structures are made for each record.*
- *New queries can not be implemented.*
- *High level language is required to program the database.*
- *Insertion, deletion and updation operations of any record require large number of pointers adjustment.*

Example of Network Data Model

UNIVAC's DMS 1100 is an example of network database model.

Q14.(b) Explain Relational model in detail.

Ans. Relational Model

The hierarchical and network data models are unstable due to pointers. Incase of system errors, the chain of addresses between the records could be damaged which results in reduced data integrity.

Relational model is a record based logical model. It was proposed by Dr E. F. Codd in 1970. It has now gained wide acceptance.

A relational model is defined as a database that allows to group its data items into one or more independent tables that can be related to one another by using fields common to each related table.

Relational tables show only the logical relationship. End users need not know the exact physical structure of a table or relation. It is the primary data model for commercial data processing applications. Its success has led to its applications outside data processing in systems for computer-aided design and other environments.

Relational model simplifies the database structure. It uses tables. No physical connection like pointers are used. A table is called a relation and is assigned a unique name. A table consists of rows and columns. A row represents a record. Row is called a tuple. A tuple in a table represents a relationship among a set of values.

Column is called attribute. The column values must be single valued in the sense that it has no arrays or lists. The values in each column are of same type, may be integer or character etc. Each attribute has a set of permitted values called the domain of that attribute. The domain of all attributes is required to be atomic i.e. the

elements of a domain are to be indivisible. The number of tuples in a relation is known as cardinality. The number of attributes in a table is known as degree of the tuple or relation.

Operations on Relational Model

The operations that can be performed on Relational model are:

(i) Insert operation

(ii) Delete operation

(iii) Update operation

To understand the operations on Relational model, consider the relation STUDENT given below:

STUDENT

Rollno	Regdno	Name	Class	Address
101	05NRG01	HARI	BCA II	ROHTAK
102	05NRG02	RAVI	BCA II	ROHTAK
103	05NRG03	ANIL	BCA II	KARNAL
104	05NNG01	AJAY	BCA III	KARNAL
105	06NRG01	HARISH	BCA I	PANIPAT
106	06NRG02	MOHAN	BCA I	ROHTAK
107	06NRG03	VIJAY	BCA I	ROHTAK

(i) Insert operation

A new student record can be inserted very easily. So the insert operation can be performed without any anomaly.

(ii) Delete operation

Deletion is also easy. A student record can be deleted very easily. So the deletion operation can be performed without any anomaly.

(iii) Update operation

Updation is also easy. For example to change the Address of student whose Rollno is 102 from ROHTAK to KARNAL.

Advantages of Relational Model

Advantages of relational model are:

1. Ease of Use

Since tabular structure is simple and familiar, it is easy to use and understand. Even first time users find it attractive.

2. Flexibility

This model is very flexible because the information which is needed from different tables can be easily manipulated by using the operations like project and join.

3. Precision

Mathematical operations on tables can be applied. The usage of relational algebra and relational calculus for the manipulation of relations ensures clarity.

4. Data Independence

Data independence is achieved more easily with normalization in a relational database. Thus this model achieves data independence making the database design, maintenance and usage much easier than the other models.

5. Query capability

Relational data model provides excellent support for queries with the help of inbuilt query language. This

easy-to-use query capability is one of the main reasons for the popularity of this model.

6. Security

Allows security control and authorization by moving sensitive attributes in a table into a separate relation with its own authorization controls.

Limitations of Relational Model

The relational model has very minor limitations as compared to the advantages. Some of the limitations are:

- ***The size of database becomes large, it increases the response time.***
- ***It does not maintain physical connection among the records.***

Example of Relational Model

Oracle is example of a relational database management system.

Q14.(c) Differentiate between Network model and Hierarchical model. MDU BCA 2015

Ans. Following are the differences between network and hierarchical model:

Network Model	Hierarchical Model
1. Network model is a collection of records connected to each other through links, i.e., it is based on graph structure.	1. Hierarchical model is a collection of records connected in the form of a tree, i.e., it is based on tree structure.
2. Many–to–many relationship can be expressed in this model.	2. One–to–many relationship can be expressed in this model.
3. Relationship between records is in the form of links or pointers.	3. Relationship between records is parent to child.
4. Implementation is complex due to the use of pointers. Complexity increases as the number of link increases.	4. It is simple, easy to use and natural method of implementing hierarchical relationships.
5. Network model does not suffer from the insert anomalies, delete anomalies and update anomalies.	5. Hierarchical model suffers from insertion anomalies, deletion anomalies and updation anomalies.

Network Model	Hierarchical Model
6. Searching a record is easy as there are multiple access paths. Thus in this model easier access as compared to hierarchical data model.	6. Searching a record i not easy since t e processing is sequential along the branches of the tree. Moreover it takes more time to access the nodes.
7. This type of model is useful for representing such records which have many to many relationships.	7. This type of model is useful only when there is some hierarchical character in the database.

Q14.(d) Differentiate between hierarchical model and relational model. IGU BCA 2018

Ans. Following are the differences between hierarchical model and relational model:

Hierarchical Model	Relational Model
1. Hierarchical model is a collection of records connected in the form of a tree.	1. Relational model stores records in the form of tables.
2. One–to–many relationship can be expressed in this model.	2. Many–to–many relationship can be easily expressed in this model.
3. Relationship between records is parent to child.	3. Relationship between records is represented by a relation which have a key field from each record involved in the relationship.
4. General purpose Query facility is not available.	4. General purpose Query facility is available.
5. To represent links among the records, pointers from parents to the children are used. Thus relations among the records are physical.	5. In relational model, relations among the records are not physical. Here data are stored in tables consisting of rows and columns.

Hierarchical Model	Relational Model
6. Searching a record is not easy since the processing is sequential along the branches of the tree. Moreover it takes more time to access the nodes.	6. Searching a record is very easy as a unique key field is used to search a data element.
7. Hierarchical model suffers from insertion anomalies, deletion anomalies and updation anomalies.	7. Relational model does not suffer from the insert anomalies, delete anomalies and update anomalies

Q15.(a) Discuss database relations and their properties with examples. MDU BCA 2014

OR

What is a relation? Enumerate the properties of a relation.

Ans. Relation

Relation is essentially a simple table. It may be defined as set of rows (tuples), each row therefore has the same columns (attributes). Since a relation is a sort of tuple, there is no ordering associated with the tuples.

A database relation is used to represent information about any entity and its relationship with other entities in the form of attributes (columns) and tuples (rows).

In relational data model, the data is organized into tables i.e. rows and columns. These tables are called relations. A row in a table represents relationship among a set of values.

Properties of relation

A relation has the following properties:

- ***In a relation rows can be in any order.***
- ***Columns can be present in any order.***
- ***All the rows should be different.***
- ***The value of each column should come from some common domain.***
- ***Number of columns of a relation is called its degree.***
- ***Number of rows of a relation is called its cardinality.***

Q15.(b) What do you mean by key? Explain different types of key with suitable example.

MDU BCA 2016, 2015

OR

Explain various types of keys with suitable examples. **MDU BCA 2013**

OR

What do you mean by Key? Explain in detail the different types of keys. **MDU BCA 2012**

Ans. Key

A key is an attribute (also known as column or field) or a combination of attribute that is used to identify records.

Sometimes one might have to retrieve data from more than one table, in those cases he/she requires to join tables with the help of keys. The purpose of the key is to bind data together across tables without repeating all of the data in every table.

For example if a table has id, name and address as the column names then each one is known as the key for that table. One can also say that the table has 3 keys as id, name and address. The keys are also used to identify each record in the database table.

Keys are fundamental to relational databases. In fact, without keys, relational databases will not be usable at all. The power of relational databases comes from the fact that one can split related data into different tables and logically link them together by using keys.

Different types of Keys

Consider a relation STUDENT. Let its attributes as Rollno, Regdno, Name, Class, Address.

STUDENT

Rollno	Regdno	Name	Class	Address
101	05NRG01	HARI	BCA II	ROHTAK
102	05NRG02	RAVI	BCA II	ROHTAK
103	05NRG03	ANIL	BCA II	KARNAL
104	05NNG01	AJAY	BCA III	KARNAL
105	06NRG01	HARISH	BCA I	PANIPAT
106	06NRG02	MOHAN	BCA I	ROHTAK
107	06NRG03	VIJAY	BCA I	ROHTAK

Primary Key

An attribute which recognize a tuple uniquely is called a primary key.

For example, Rollno is a primary key in STUDENT relation.

In some cases, two or more attributes together can form the primary key, such type of key is called concatenated key. In such cases no single attribute can be called as the primary key.

The value of primary key in any tuple cannot be null.

Using primary key in a relation:

- ***Relationship between the various relations of the database can be established.***
- ***The DBMS can quickly search and access data from the relations.***

Secondary Key

A secondary key does not uniquely identify a tuple of a relation.

For example, Name, Class and Address are the secondary keys in STUDENT relation.

Candidate Key

There may be more than one attributes in a relation that recognize a tuple uniquely. These attributes are called candidate keys.

For example, in the relation STUDENT, Rollno and Regdno are candidate keys.

Alternate Key

Out of candidate keys, one is chosen as a primary key. Candidate key which is not primary key is called alternate key.

For example, if Rollno is taken as primary key then Regdno is the alternate key.

The significance of defining an alternate key is as follows:

- ***The DBMS creates and uses an index based on the alternate key.***
- ***Searching based on an alternate key is quicker than searching based on a non-key column, because of the index.***
- ***Unlike a primary key, an alternate key can have duplicates.***

Foreign key

Foreign keys are extremely important in the context of table inter-relation-ships.

If a non key attribute in one relation appears as the primary key in another relation, it is called foreign key.

For example, consider relation Account and Branch of a banking organization database.

Account

Accountno	Name	Balance	Distt
101	RAVI	40500	ROHTAK
102	HARI	70000	ROHTAK
103	MOHAN	10000	HISSAR
104	RAVINDER	10761	SIRSA
105	HARPREET	9712	PANIPAT
106	JAGDEEP	625	ROHTAK

Branch

Distt	State
ROHTAK	HARYANA
SIRSA	HARYANA
PANIPAT	HARYANA
HISSAR	HARYANA
KARNAL	HARYANA
AMBALA	HARYANA
JHAJJAR	HARYANA

Distt of relation Account is foreign key since it is primary key of relation Branch.

Q16.(a) Discuss entity integrity and referential integrity constraints. Why each is considered important? MDU BCA 2017

OR

Explain integrity constraints. Describe their importance. MDU BCA 2015

Ans. Integrity Constraints

Integrity constraints are basically the rules. These rules must be followed to have an accurate database.

There are two integrity constraints:

1. Entity Integrity Rule
2. Referential Integrity Rule

1. Entity Integrity Constraint

No attribute of a primary key can have a null value. This is because primary key values are used to identify the individual tuples. This constraint is known as Entity Integrity Constraint.

Example:

Account

Accountno	Name	Balance	Distt
101	RAVI	40500	ROHTAK
102	HARI	70000	ROHTAK
103	MOHAN	10000	HISSAR
104	RAVINDER	10761	SIRSA
107	HARPREET	9712	PANIPAT
		625	ROHTAK
	JAGDEEP	7250	SONIPAT

Not allowed

Entity Integrity ensure that there is no duplicate records in the table and each field that recognizes each record in the table must have unique value and not having null values.

Entity Integrity specifies that every instance of entity have the unique values, i.e., primary key must be kept and must have the values other than null values.

2. Referential Integrity Constraint

It states that "If a foreign key in relation R_1 refers to the primary key of relation R_2, then every value of the foreign key in relation R_1 must be available in relation R_2".

Example:

Account

Accountno	Name	Balance	Distt
101	RAVI	40500	ROHTAK
102	HARI	70000	ROHTAK
103	MOHAN	10000	HISSAR
104	RAVINDER	10761	SIRSA
105	HARPREET	9712	PANIPAT
106	JAGDEEP	625	ROHTAK
107	HANI	7250	SONIPAT

Branch

Distt	State
ROHTAK	HARYANA
SIRSA	HARYANA
PANIPAT	HARYANA
HISSAR	HARYANA
KARNAL	HARYANA
AMBALA	HARYANA
JHAJJAR	HARYANA

Distt of relation Account is foreign key. Thus Distt Sonipat is not allowed in Account relation because it is not available in Branch relation.

Referential Integrity is a property of data which when satisfied, requires every value of one attribute of a relation to exist as a value of another attribute in a different relation.

For referential integrity to hold in a relational database, any field in a table that is declared a foreign key can contain only values from a parent table's primary key. For instance, deleting a record that contains a value referred to by a foreign key in another table would break referential integrity.

Referential Integrity is used to maintain the consistency among tuples of the two relations.

Importance of integrity constraints

Integrity constraints are an important component of the relational database machinery. They are often defined as conditions that a database instance must satisfy at all times. Thus, integrity constraints has permeated database practice and nowadays their use is everywhere in all modern database management systems.

Q17.(a) What are the various operators in relational algebra? Explain with examples.

MDU BCA 2015

Ans. Relational Algebra

Relational algebra is a set of operations that can be used to manipulate relations.

It is a procedural query language. It takes one or more than one relations as input and produces a new relation as output. Thus the result of a query in relational algebra is always a relation.

The basic operations in relational algebra are:

1. Select
2. Project
3. Cartesian Product
4. Union
5. Set Difference
6. Set Intersection
7. Join
8. Division

1. Select (σ) Operation

To select tuples of a relation depending upon a condition being true, the operation is known as a select operation.

The symbol sigma (σ), a Greek letter, is used to denote select operation. The format of select operation is given below:

$\sigma_{<cond>}$ (<relation>)

where

<cond> : is the condition that must be satisfied.

<relation> : is the relation on which the select operation is applied.

e.g. To find all the tuples of relation Student where age is greater than 17

Student

Rollno	Name	Class	Age
101	Hari	BCA I	17
102	Ravi	BCA I	18
103	Mohan	BCA II	18
104	Ajay	BCA II	18
105	Anil	BCA I	17
106	Harish	BCA III	19
107	Anshul	BCA I	18

A query in relational algebra will be expressed as:

$$\sigma_{age>17}(Student)$$

The result of such a query will be the following relation:

Rollno	Name	Class	Age
102	Ravi	BCA I	18
103	Mohan	BCA II	18
104	Ajay	BCA II	18
106	Harish	BCA III	19
107	Anshul	BCA I	18

The comparison operators allowed in the condition expression are:

$$=,\ \neq,\ <,\ \leq,\ >,\ \geq$$

Connectives and ($\wedge$), or ($\vee$) can be used to combine two or more conditions.

e.g. To find all the tuples of relation Student where age is equal to 17 and class is 'BCA I'.

For this a query in relational algebra will be expressed as:

$$\sigma_{\text{age}=17 \wedge \text{class} = \text{"BCA I"}} (\text{Student})$$

The result of such a query will be the following relation:

Rollno	Name	Class	Age
101	Hari	BCA I	17
105	Anil	BCA I	17

2. Projection (π) Operation

A projection operation on a relation produces a relation with certain columns left out i.e. it produces the relation minus certain columns.

Projection is denoted by the symbol π (Pi), a Greek letter. The attribute list is specified as a subscript to π that should appear in the result.

The format of this operation is given below:

$\pi_{a1, a2, \ldots, an}$ (<name>)

where

a1, a2, ..., an : is the list of attributes to be projected

<name> : is the name of the relation.

e.g. (i) To have a relation showing rollno and name from the relation Student, the query is:

$$\pi_{\text{rollno, name}} (\text{Student})$$

The result is the following relation:

Rollno	Name
101	Hari
102	Ravi
103	Mohan
104	Ajay
105	Anil
106	Harish
107	Anshul

(ii) To find the rollno, name, class of the students with age greater than 17, the query is:

$$\pi_{\text{rollno, name, class}} (\sigma_{\text{age} > 17}(\text{Student}))$$

The result is the following relation:

Rollno	Name	Class
102	Ravi	BCA I
103	Mohan	BCA II
104	Ajay	BCA II
106	Harish	BCA III
107	Anshul	BCA I

(iii) To find class of the students, the query is:

$$\pi_{\text{class}} (\text{Student})$$

The result is the following relation:

Class
BCA I
BCA II
BCA III

Thus, the resulting relation has only the name of classes with the duplicate entries removed.

3. Cartesian Product (×) Operation

The Cartesian product of two relations A and B is a relation which is concatenation of every tuple of relation A with every tuple of relation B.

The cartesian product of relation A having m tuples and relation B having n tuples has m × n tuples. The product is denoted as A × B.

e.g.(i) Consider two relations A and B and their cartesian product C then $C = A \times B$

A

a_1	a_2	a_3
x_1	x_2	x_3
y_1	y_2	y_3
z_1	z_2	z_3

×

B

b_1	b_2
p_1	p_2
q_1	q_2

=

C

a_1	a_2	a_3	b_1	b_2
x_1	x_2	x_3	p_1	p_2
x_1	x_2	x_3	q_1	q_2
y_1	y_2	y_3	p_1	p_2
y_1	y_2	y_3	q_1	q_2
z_1	z_2	z_3	p_1	p_2
z_1	z_2	z_3	q_1	q_2

Thus, the relation C has the attributes of both the relations A and B and tuple of relation A appears with all tuples in relation B.

(ii) Let relation Student has four tuples and relation Test has two tuples as shown below:

Student

Rollno	Name	Class	Age
101	HARI	BCA I	17
102	RAVI	BCA I	18
103	MOHAN	BCA II	18
104	AJAY	BCA II	18

Test

Date	Time
30/03/08	9.00
03/04/08	14.00

Student × Test will have 8 tuples as shown below:

Rollno	Name	Class	Age	Date	Time
101	HARI	BCA I	17	30/03/08	9.00
101	HARI	BCA I	17	03/04/08	14.00
102	RAVI	BCA I	18	30/03/08	9.00
102	RAVI	BCA I	18	03/04/08	14.00
103	MOHAN	BCA II	18	30/03/08	9.00
103	MOHAN	BCA II	18	03/04/08	14.00
104	AJAY	BCA II	18	30/03/08	9.00
104	AJAY	BCA II	18	03/04/08	14.00

4. Union (∪) Operation

Two relations are union compatible if

- ***The degree of two relations is same***
- ***Their corresponding attributes are defined on the same domain.***

The union operation can be applied to two relations if they are union compatible. The union of two relations A and B, denoted as A∪B, is the set of all tuples that belong to A or B or both. In the resultant relation, the duplicate tuples are eliminated.

e.g. Consider two relations Musicgp and Sportgp as given below:

Musicgp

Rollno	Name	Class	Age
101	HARI	BCA I	17
102	RAVI	BCA I	18
103	MOHAN	BCA II	18
104	AJAY	BCA II	18
105	ANIL	BCA I	17
106	HARISH	BCA III	19
107	ANSHUL	BCA I	18

Sportgp

Rollno	Name	Class	Age
104	AJAY	BCA II	18
201	BAJENDER	BCA III	20
205	RAJVIR	BCA III	20
211	RAMU	BCA I	17
225	RAJU	BCA III	19

Musicgp ∪ Sportgp =

Rollno	Name	Class	Age
101	HARI	BCA I	17
102	RAVI	BCA I	18
103	MOHAN	BCA II	18
104	AJAY	BCA II	18
105	ANIL	BCA I	17
106	HARISH	BCA III	19
107	ANSHUL	BCA I	18
201	BAJENDER	BCA III	20
205	RAJVIR	BCA III	20
211	RAMU	BCA I	17
225	RAJU	BCA III	19

5. Set Difference (–) operation

The set difference operation can be applied to two relations if they are union compatible.

The difference between two relations A and B is the set of all tuples that belong to A and not to B.

e.g. Consider two relations Musicgp and Sportgp.

(i) Musicgp – Sportgp is the set of tuples for students who are in Musicgp and are not in Sportgp.

Musicgp – Sportgp =

Rollno	Name	Class	Age
101	HARI	BCA I	17
102	RAVI	BCA I	18
103	MOHAN	BCA II	18
105	ANIL	BCA I	17
106	HARISH	BCA III	19
107	ANSHUL	BCA I	18

(ii) To obtain the name of the students who are in music group but not is sport group, the query is written as:

$$\pi_{\text{name}}(\text{Musicgp}) - \pi_{\text{name}}(\text{Sportgp})$$

The result of this querry is

Name
HARI
RAVI
MOHAN
ANIL
HARISH
ANSHUL

6. Set intersection (∩) Operation

If two relations A and B are union compatible then they can participate in an intersection operation.

The intersection of two relations A and B is defined as the set of all tuples that belong to both A and B. It is denoted by the symbol '∩'.

e.g. Consider two relations Musicgp and Sportgp.

(i) Musicgp ∩ Sportgp

Rollno	Name	Class	Age
104	AJAY	BCA II	18

(ii) To obtain the name of the students who are in music group as well as in sport group, the query is written as:

$$\pi_{name}(\text{Musicgp}) \cap \pi_{name}(\text{Sportgp})$$

The result of this query is

Name
AJAY

7. Join (⋈) Operation

It is a binary operation and combines two relations into a single relation. It is denoted by the symbol ⋈.

It is equivalent to taking the Cartesian product of two relations and then performing a suitable selection from the Cartesian product. There are many ways to combine two relations to form a new relation. In general each join operation takes tuples from two relations and combine them together to form a single tuple where the tuples satisfy the specified condition(s).

Equijoin:

In equijoin operation the join condition includes one or more equality comparison involving attributes from two relations A and B.

e.g. Consider the relations Student and Position.

Student

Rollno	Name	Class	Age
101	HARI	BCA I	17
102	RAVI	BCA I	18
103	MOHAN	BCA II	18
104	AJAY	BCA II	18
105	ANIL	BCA I	17

Position

SID	Rank
103	1
105	2

The Cartesian product of Student and Position is shown below:

Student × Position

Rollno	Name	Class	Age	SID	Rank
101	HARI	BCA I	17	103	1
101	HARI	BCA I	17	105	2
102	RAVI	BCA I	18	103	1
102	RAVI	BCA I	18	105	2
103	MOHAN	BCA II	18	103	1
103	MOHAN	BCA II	18	105	2
104	AJAY	BCA II	18	103	1
104	AJAY	BCA II	18	105	2
105	ANIL	BCA I	17	103	1
105	ANIL	BCA I	17	105	2

To know the rank of each student in the class, join Student tuples with matching Position tuples. Such is denoted as Student tuple to an Position tuple of student equals SID of Position.

The query is written as:

Student $\bowtie$ Rollno = SID (Position)

Rollno	Name	Class	Age	SID	Rank
103	MOHAN	BCA II	18	103	1
105	ANIL	BCA I	17	105	2

(Equijoin)

Two attributes Rollno and SID are identical.Natural join eliminates the redundant join attributes.

Rollno	Name	Class	Age	Rank
103	MOHAN	BCA II	18	1
105	ANIL	BCA I	17	2

(Natural Join)

Normally, unless otherwise specified, the join means natural join.

8. Division (÷) Operation

Division operation is applied to queries that have the phrase "for all". It is denoted by ÷

The division operator divides a dividend relation A of degree m+n by a divisor relation B of degree n, the result produced is a relation of degree m.

e.g. Consider the following two relations Branch and Account

Branch

Branch_name	State
Rohtak	Haryana
Karnal	Haryana
Jalandhar	Punjab
Ludhana	Punjab

Account

Accountno	Customer_name	Branch_name
27421	HARI	Rohtak
28242	RAVI	Karnal
27437	JAGIT	Rohtak
47232	MANPREET	Jalandhar
48247	MANJEET	Ludhana
26242	HARI	Karnal

Consider a query, to find the names of those who are customers for all the branches in the state Haryana.

(i) First to find all the branch_names in Haryana, the query is written as:

$\pi_{\text{branch\_name}} (\sigma_{\text{state = "Haryana"}} (\text{Branch}))$

The result of this query is:

Branch_name
Rohtak
Karnal

(ii) Secondly, find the customer_name and branch_name from the relation Account.

The query is written as

$\pi_{\text{customer\_name, branch\_name}} (\text{Account})$

The result of this query is:

Customer_name	Branch_name
HARI	Rohtak
RAVI	Karnal
JAGIT	Rohtak
MANPREET	Jalandhar
MANJEET	Ludhana
HARI	Karnal

(iii) Finally, to find the names of those who are customers for all the branches in the state Haryana, the query is:

$$\pi_{\text{customer\_name, branch\_name}} (\text{Account}) \div \pi_{\text{branch\_name}} (\sigma_{\text{state = "Haryana"}} (\text{Branch}))$$

The result of this query is:

Customer_name
HARI

Q17.(b) **Explain the operators of relational calculus with the help of examples.**

Ans. **Relational Calculus**

Relational Calculus is a non-procedural query language. In it, the query is expressed as a formula consisting of a number of variables and an expression involving these variables. How this formula is evaluated, there is no specified rule. Everything depends upon the DBMS to transfer these non procedural queries into equivalent procedural queries.

The two forms of relational calculus are:

1. Tuple relational calculus
2. Domain relational calculus

1. Tuple relational calculus

Tuple relational calculus is basically used to select a set of tuples for which a given formula is true.

The formula is specified in expression. The expression is used in defining the result of a query, updating the query or viewing the query. Any query in tuple relational calculus is expressed as: { **a** | P(**a**) } i.e. the set of all tuples **a** such that the formula or predicate P is true for **a**.

Tuple relational calculus may include the following:

(a) Comparison Operators

Comparison operators are:

= equal to.	¬ = not equal to
> greater than	>= greater than or equal to
< less than	<= less than or equal to

(b) Boolean Operators

Boolean operators are:

- $\wedge$ AND
- $\vee$ OR
- $\neg$ NOT

(c) Quantifiers

Quantifiers are:

- $\exists$ there exists
- $\forall$ for all

Formula or predicate are built from atoms using the following rules:

- An atom is a formula.
- If F_1 and F_2 are formulae then $\neg F_1$, $\neg F_2$, $F_1 \wedge F_2$, $F_1 \vee F_2$ and $F_1 \Rightarrow F_2$ are also formulae.

An atom may have the following forms:

(i) $a \in R$

where

- a is tuple variable.
- R is relation.

(ii) $a_1[K_1]$ © $a_2[K_2]$

(iii) $a_1[K_1]$ © constant

where

- K_1 is an attribute corresponding to tuple variable a_1.
- K_2 is an attribute corresponding to tuple variable a_2.
- © is comparison operator.

(iv) **a**[K] denotes the value of tuple **a** on attribute K.

Example: Consider the following relation Student

Student

Rollno	Name	Class	Age
101	HARI	BCA I	17
102	RAVI	BCA I	18
103	MOHAN	BCA II	18
104	AJAY	BCA II	18
105	ANIL	BCA I	17
106	HARISH	BCA III	19
107	ANSHUL	BCA I	18

(i) To find the tuples of Student relation where the class is BCA I. The query is formed as:

$$\{ a \mid a \in \text{Student} \wedge a[\text{class}] = \text{"BCA I"} \}$$

This query has two atoms

- $a \in$ Student &
- a[class] = "BCA I"

connected by 'and' operator.

This query selects the tuples that belong to relation Student and have "BCA I" in class attribute.

(ii) To find the tuples of Student relation that have age greater than 17. The query is formed as

$$\{ a \mid a \in \text{Student} \wedge a[\text{age}] > 17 \}$$

(iii) To get only the names of all the students who have age greater than 17. The query is formed as

$$\{a \mid \exists\, b \in \text{Student}\ (a[\text{name}] = b[\text{name}] \wedge b[\text{age}] > 17)\}$$

This query is read as:

"The set of all tuples a such that there exist tuple b in relation Student for which the value of tuple a and tuple b for the name attribute are equal and tuple b has value of age attribute greater than 17."

The result of this query has only one attribute name since a is defined only on name attribute.

Free Variable

A variable is said to be free variable if it is not quantified by '∃' or '∀' operators.

e.g. $\{ a \mid a \in \text{Student} \wedge a[age] > 17 \}$

In this example a is a free variable.

Bound Variable

A variable is said to be bound variable if it is quantified by '∃' or '∀' operator.

e.g. $\{a \mid \exists\, b \in \text{Student}\ (a[name] = b[name] \wedge b[age] > 17)\}$

In this example a is a bound variable.

2. Domain relational calculus

In domain relational calculus, domain variables are used which can take values from an attribute domain, instead of entire tuple or tuple elements.

Any query in domain relational calculus is expressed as:

$\{ < a_1, a_2, ..., a_n > \mid f(a_1, a_2, ..., a_n) \}$

where

- $a_1, a_2, ..., a_n$ are domain variables.
- $f(a_1, a_2, ..., a_n)$ represents a formula.

It may include the following:

(a) Comparison Operators

Comparison operators are:

= equal to.	¬ = not equal to
> greater than	>= greater than or equal to
< less than	<= less than or equal to

(b) Boolean Operators

Boolean operators are:

- ∧ AND
- ∨ OR
- ¬ NOT

(c) Quantifiers

Quantifiers are:

- ∃ there exists
- ∀ for all

Formula or predicate are built from atoms using the following rules:

- An atom is a formula.
- If F_1 and F_2 are formulae then $\neg F_1$, $\neg F_2$, $F_1 \wedge F_2$, $F_1 \vee F_2$ and $F_1 \Rightarrow F_2$ are also formulae.

An atom of domain relational calculus has one of the following forms:

(i) $a_1, a_2, \ldots, a_n \in R$

where

- $a_1, a_2, \ldots, a_n$ are domain variables or domain constants corresponding to n attributes of relation R
- R is relation.

(ii) a_1 © a_2

where

- a_1 and a_2 are domain variables.
- © is a comparison operator.

(iii) a_1 © c

where

- a_1 is domain variable.
- c is a constant in the domain of the attribute for which a_1 is a domain variable.
- © is a comparison operator.

Example: Consider the relation Student(rollno, name, class, age)

(i) To find the rollno, name, class and age of relation Student having age greater than 17, the query is formed as:

$$\{ < a_1, a_2, a_3, a_4> | < a_1, a_2, a_3, a_4> \in \text{Student} \wedge a_4 > 17\}$$

Here a_1, a_2, a_3, a_4 are domain variables corresponding to attributes rollno, name, class, age.

(ii) To find all classes in which student have age greater than 17. the query is formed as

$$\{< a_3> | \exists\, a_1, a_2, a_4\ (< a_1, a_2, a_3, a_4> \in \text{Student} \wedge a_4 > 17\}$$

Here a_1, a_2, a_3, a_4 are domain variables corresponding to attributes rollno, name, class, age.

Q18.(a) **What is the need of normalization? Define and discuss 1NF, 2NF, 3NF and BCNF by giving suitable example for each.** MDU BCA 2018

OR

Define normalization. How is it useful and used? Explain by giving examples, the conditions that are necessary for a relation to be in 1NF, 2NF, 3NF, BCNF. MDU BCA 2016

OR

Define normalization. Explain by giving examples, the conditions that are necessary for a relation to be in 1NF, 2NF, 3NF, BCNF, 4NF. MDU BCA 2015

OR

What is normalization? How is it used and useful?[You may go up to 3NF] MDU BCA 2013

OR

What is normalization? Explain the different types of normalization with suitable example. MDU BCA 2012

Ans. **Normalization**

It is a design technique that is widely used in designing relational model.

Normalization is a process that removes the following undesirable properties in a relation:

- *Repetition of information i.e. redundancy.*
- *Inability to store some information i.e. insertion anomaly.*

असंगत, असंबद्धता

- *Loss of information in deletion i.e. deletion anomaly.*
- *Problem in updations.*

Normalization is basically a decomposition method in which a relation having undesirable dependency is split into two or more relations and thus removes the undesirable dependencies.

Need of normalization

The need of normalization is:

- *To minimize redundancy by storing each fact within the database only once.*
- *To remove, insert, delete and update anomalies during database activities.*
- *To reduce the need to reorganize data when it is modified or enhanced.*
- *To maintain data consistency in the database.*
- *To avoid unnecessary coding. Extra programming can be required to handle the non-normalized data and this in turn can impair performance significantly.*

A relation may exist in different normal forms and a particular normal form satisfies a specific set of constraints. The different normal forms are:

1. First normal form (1NF).
2. Second normal form (2NF).
3. Third normal form (3NF).
4. Boyce-Codd normal form (BCNF).
5. Fourth normal form (4NF).
6. Fifth normal form (5NF).

Each of these normal forms is more refined in that order. Thus 2NF is a better design than 1NF, 3NF is a better design than 2NF and so on. 2NF is always in 1NF, 3NF is always in 2NF etc as shown on the next page. Normalization of a relation is continued starting from 1NF until a desired normal form is obtained for a particular application.

1. First Normal Form (1NF)

A relation is said to be in 1NF if the values in the domain of each attribute of the relation are atomic.

A value is said to be atomic only if it is indivisible, for example, set of integers. Thus, any normalized relation is in 1 NF.

Example: Consider the relation FIRST (S#, STATUS, CITY, P#, QTY) .

FIRST

S#	STATUS	CITY	P#	QTY
S_1	30	Delhi	P_1	275
S_1	30	Delhi	P_2	175
S_1	30	Delhi	P_3	375
S_1	30	Delhi	P_4	175
S_1	30	Delhi	P_5	100
S_1	30	Delhi	P_6	100
S_2	20	Rohtak	P_1	275
S_2	20	Rohtak	P_2	375
S_3	20	Rohtak	P_2	175
S_4	30	Delhi	P_2	175
S_4	30	Delhi	P_4	275
S_4	30	Delhi	P_5	375

Here it is assumed that:

- ***Each supplier has a unique supplier number (S#).***

- *A supplier can supply different parts.*
- *STATUS is functionally dependent on CITY.*
- *The primary key of FIRST relation is the combination of (S#, P#).*

The following functional dependencies hold in FIRST relation:

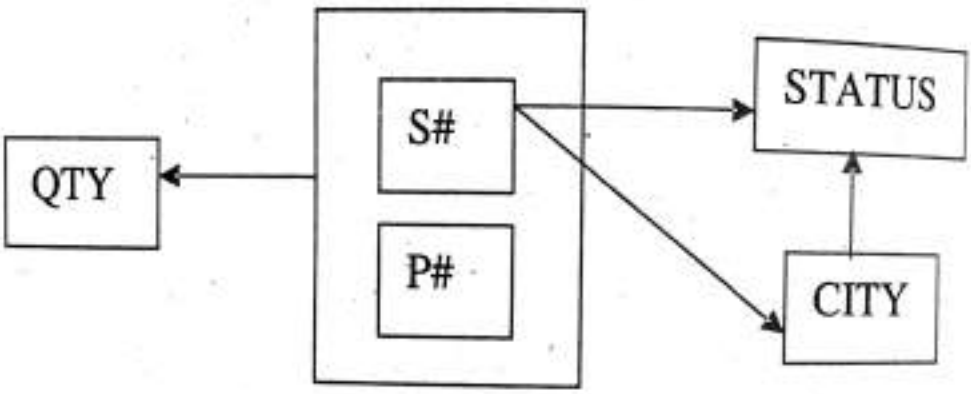
S# ⟶ CITY

S# ⟶ STATUS

CITY ⟶ STATUS

(S#, P#) ⟶ QTY

Figure shows the functional dependency diagram for this relation.



The relation FIRST is in 1NF.

Consider another relation Bookrecord (Bookno, Book_name, Authors)

Bookrecord

Bookno	Book_name	Authors
101	Fundamental	Suman, Ajay
102	Data Structure	Supriya, Anshul
103	DBMS	Ravi, Hari

The above relation is not in 1 NF since the values in the attribute Authors are not atomic.

A relation that is only in 1 NF leads to complex functional dependencies. It also leads to a number of other anomalies in updation.

The relation FIRST which is in 1 NF has the following anomalies:

(i) Inserting

The facts like a particular supplier is located in a particular city cannot be entered until that supplier supplies at least one part.

For example, the relation FIRST does not show supplier S_5 is located in Karnal. The reason is that S_5 does not supplies any part.

(ii) Deleting

If a particular supplier is deleted, the information that the supplier is located in a particular city is also destroyed. For example, if the tuple with S# value S_3 and P# value P_2 is deleted, the information that S_3 is located in Rohtak is also lost.

(iii) Updating

The CITY value for a given supplier appears many times in the relation FIRST. This redundancy causes update problems.

The solution of these problems is 2 NF.

2. Second Normal Form (2NF)

A relation is said to be in second normal form (2 NF) if it satisfies the following conditions:

- *It is in 1 NF.*
- *Every non-key attribute should be fully functional dependent on primary key.*

An attribute is non-key if it does not participate in the primary key.

The relation FIRST is not in 2 NF though it is in 1 NF. It can be reduced into relations that are in 2 NF. Infact every relation that is in 1 NF can be reduced into a set of relations that are in 2 NF form.

Reduction of 1 NF relation into 2 NF relation is done by projecting some attributes into another relation.

The relation FIRST is reduced into two relations SECOND(S#, STATUS, CITY) and SP(S#, P#, QTY).

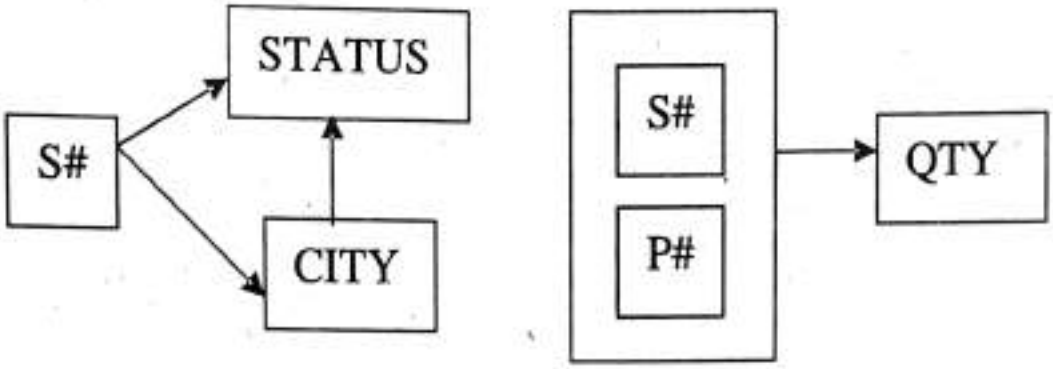
SECOND

S#	STATUS	CITY
S1	30	Delhi
S2	20	Rohtak
S3	20	Rohtak
S4	30	Delhi
S5	15	Karnal

SP

S#	P#	QTY
S_1	P_1	275
S_1	P_2	175
S_1	P_3	375
S_1	P_4	175
S_1	P_5	100
S_1	P_6	100
S_2	P_1	275
S_2	P_2	375
S_3	P_2	175
S_4	P_2	175
S_4	P_4	275
S_4	P_5	375

The functional dependency diagrams for these two relations are:



These two relations SECOND and SP satisfy the conditions of 2 NF and hence are in 2 NF. These relations overcome all the update problems involved with relation FIRST.

Now

(i) Even though S_5 does not supply any part, the information that S_5 is located in Karnal can be entered, by inserting the appropriate tuple in the **relation SECOND.**

(ii) One **can delete the** tuple with S# value S_3 and P# **value P_2** by deleting the appropriate tuple from relation SP. There is no lose of information that S_3 is located in Rohtak.

(iii) The city for a given supplier appears once. Thus one can change the city for a supplier by changing it once.

Relation SP is alright but relation SECOND still causes problems due to transitive functional dependence on S#. In SECOND, S# determines CITY and CITY determines STATUS then transitively, S# determines STATUS.

The relation SECOND which is in 2 NF has the following anomalies:

(i) Inserting

One cannot enter the fact that a particular city has a particular status value until a supplier from that

particular city. For example, one can not enter the status of Panipat until a supplier from Panipat.

(ii) Deleting

To delete a particular supplier, the information that the city has that particular status value is destroyed. For example, if one deletes the tuple S_5, the information that the status for Karnal i.e. 15 will be lost.

(iii) Updating

The status value for a given city in relation SECOND appears many times. This redundancy causes update problems.

The solution of these problems is 3 NF.

3. Third Normal Form

A relation is said to be in third normal form (3 NF) if

- ***It is in 2 NF.***
- ***Every non-key attribute is non-transitively dependent on primary key.***

The relation SECOND is in 2 NF but not in 3 NF. Thus this relation is reduced into SC(S#, CITY) and CS(CITY, STATUS) as shown below:

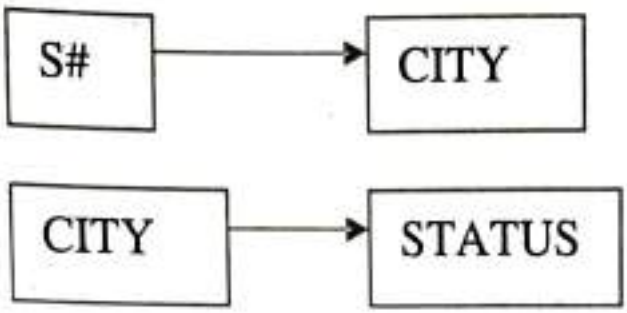
SC

S#	CITY
S1	Delhi
S2	Rohtak
S3	Rohtak
S4	Delhi
S5	Karnal

CS

CITY	STATUS
Delhi	30
Rohtak	20
Karnal	15

The corresponding functional dependency diagrams are shown below:



Relation SC, CS and SP are in 3 NF.

4. Boyce-Codd Normal Form (BCNF)

A normalized relation is in Boyce-Codd Normal form if every determinant is a candidate key.

A determinant is an attribute in a relation on which some other attribute of the relation is fully functionally dependent.

Every BCNF is in 3 NF. Relation FIRST has three determinants. S#, CITY and the combination (S#, P#). Out of these only (S#, P#) is a candidate key. Hence FIRST is not in BCNF. Relation SECOND is also not in BCNF because the determinant CITY is not a Candidate key. Relations SP, SC and CS are each in BCNF because in each case the primary key is the only determinant in the relation.

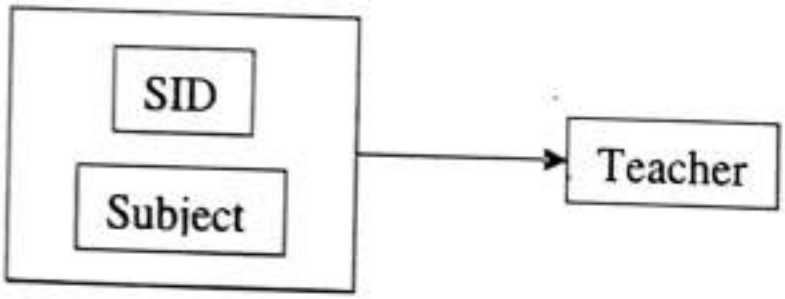
For example, consider the relation Guide(SID, Subject, Teacher)

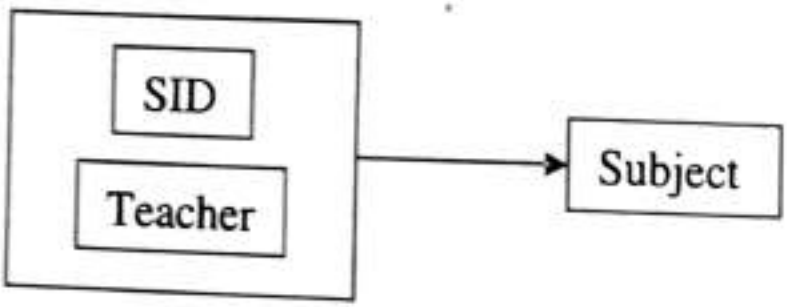
- It is assumed that
  - A student can select one or more subjects.
  - A teacher teaches only one subject.
  - A subject can be taught by many teachers.

- Since students can select several subjects, SID does not determine subject. Also students can be taught by several teachers, SID does not determine Teacher. Thus SID cannot be a primary key.
- The two possible composite keys are (SID, Subject) and (SID, Teacher).
- Since a teacher teaches only one subject. Thus teacher determine subject i.e. teacher is a determinant.

Guide

SID	Subject	Teacher
101	English	Ravi
101	Maths	Hari
102	English	Ravi
102	Hindi	Ramesh
103	Maths	Ravinder





In the relation Guide, there are three determinant (SID, SUBJECT), (SID, Teacher) and Teacher. Out of these (SID, Subject), (SID, Teacher) can be candidate key. Hence relation Guide is not BCNF.

Moreover if one deletes student 103's tuple, there is loss of information that Ravinder teaches Maths. This is deletion anomaly. Similarly one can not store the information that Harnaam teaches Computer if there is no student in computer subject. This is insertion anomaly.

Thus Guide relation can be decomposed into two relations Std_guide (SID, Subject) and Teacher_sub (Teacher, Subject)

Std_guide

SID	Subject
101	English
101	Maths
102	English
102	Hindi
103	Maths

Key: (SID, Subject)

Teacher_sub

Teacher	Subject
Ravi	English
Hari	Maths
Ramesh	Hindi
Ravinder	Maths

Key : Teacher

Thus relations Std_guide and Teacher_sub are in BCNF.

5. Fourth Normal Form

A relation is said to be in fourth normal form (4 NF) if

- ***It is in BCNF.***
- ***It has no multi valued dependencies.***

Given a relation R with attributes A, B and C. Multivalued dependency represents a dependency between these attributes such that

- A leads to multiple values of C and B
- B and C are independent of each other.

MVD between attributes A, B and C in a relation R can be indicated as

$$A \longrightarrow\!\!\!\!\!\rightarrow B \quad (A \text{ multidetermines } B)$$

$$A \longrightarrow\!\!\!\!\!\rightarrow C \quad (A \text{ multidetermines } C)$$

Multivalued dependency in a relation is due to first normal form, which disallows an attribute in a row from having a set of values.

For example, consider the relation Student(SID, Subject, Activity)

Student

SID	Subject	Activity
101	English	Skating
101	Maths	Skating
101	English	Cricket
101	Maths	Cricket
102	Computer	Swimming

Here SID leads to multiple values of Subject and multiple values of Activity. Subject and Activity are independent of one another.

These multivalued dependencies can be indicated as:

$$SID \longrightarrow\!\!\!\!\!\rightarrow Subject$$

$$SID \longrightarrow\!\!\!\!\!\rightarrow Activity$$

To remove this problem decompose the relation as:

SID	Subject
101	English
101	Maths
102	Computer

SID	Activity
101	Skating
101	Cricket
102	Swimming

Usefulness of Normalization

Normalization is useful to remove the various anomalies present in the relation. Normalization of a database helps in modifying the design at later times and helps in being prepared if a change is required in the database design. Normalization raises the efficiency of the database in terms of management, data storage and scalability.

Q18.(b) Write short note on various types of functional dependencies with examples. MDU BCA 2010

OR

Explain functional dependency, full functional dependencies, partial dependencies, transitive dependencies and multi valued dependencies with the help of suitable examples.

Ans. Functional Dependency

Functional dependency is the relationship between the attributes or fields of a relation.

An attribute A_2 of relation R is said to be functionally dependent on another attribute A_1 of the same relation if there is only one value of A_2 corresponding to A_1.

In other words whenever two tuples of R, agree on their A_1 value then they also agree on their A_2 value. Mathematically for any two tuples t_1 and t_2 of R

$$t_1[A_1] = t_2[A_1] \Rightarrow t_1[A_2] = t_2[A_2]$$

Diagrammatically this can be shown as:

Saying that A_2 is functionally dependent on A_1 is same to say that A_1 determines A_2.

Example: Consider a relation Item (Item_code, Item_name, Price, Qty).

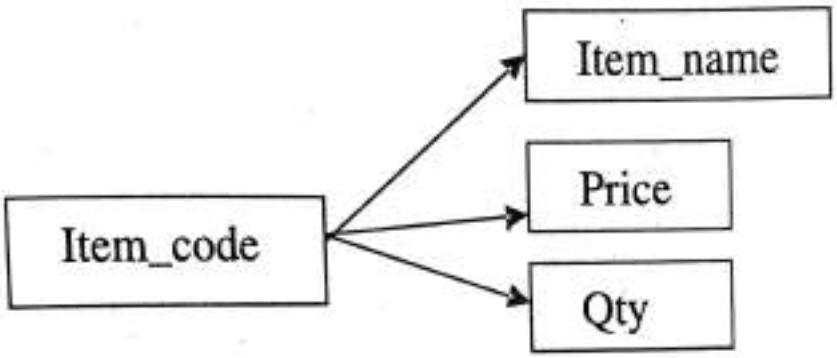
Item

Item_code	Item_name	Price	Qty
101	Pen	10	50
102	Pencil	2	100
103	Eraser	1	50
104	Sharpener	3	75

Here, each of the attributes Item_name, Price, Qty is functionally dependent on the attribute Item_code.

Thus Item_code → Item_name
Item_code → Price
Item_code → Qty

The functional dependency diagram of relation Item can be shown as:



Types of functional dependencies

Various types of functional dependencies are:

1. Fully functional dependency
2. Partial dependency
3. Transitive dependency
4. Multi valued dependency
5. Join dependency

1. Fully functional dependency

In a relation R, an attribute or a collection of attributes Y is said to be fully functionally dependent on another collection of attributes X if it is functionally dependent on whole of X but not functionally dependent on any subset of X.

Example: Consider the relation Result(Rollno, Course_code, Term, Grade)

Result

Rollno	Course_code	Term	Grade
5001	101	1	A
5001	102	2	B
5001	103	3	A
5002	101	1	C
5002	104	2	B
5002	107	3	A
5003	101	1	A
5003	105	2	A
5003	106	3	C

Here, it is assumed that:

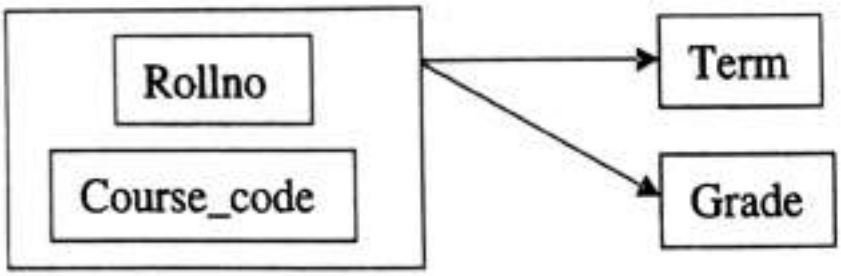
- A course can be offered to different students in different terms and
- A student can take a course only once.

Thus, the attributes Term and Grade are individually fully functional dependent upon the Primary key (Rollno,Course_code).

Rollno, Course_code ⟶ Term

Rollno, Course_code ⟶ Grade

The fully functional dependency diagram of relation Result can be shown as:



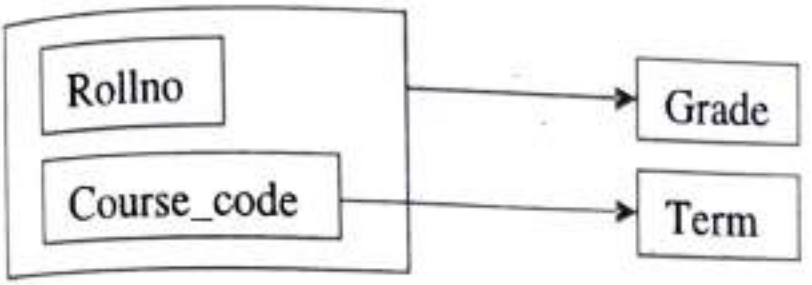
2. Partial Dependency

If a dependency will not follow the rules of fully functional dependency then it is called partial functional dependency.

For example, if it is assumed that a course is offered in only one term, then the attribute Term becomes functionally dependent on Course_code only.

Thus it is a partial functional dependency because Term is not fully functionally dependent on the primary key(Rollno, Course_code).

In this case, the functional dependency diagram will be:



3. Transitive dependency

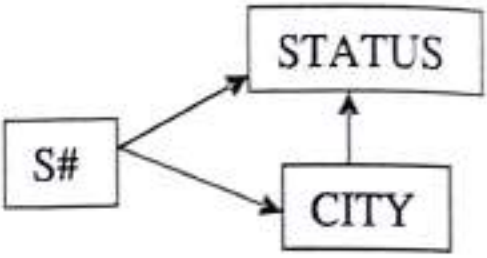
Let A, B and C are the set of attributes of the relation R. Let A → B, B → C and A → C. If all these conditions are true, then attribute C is transitively dependent on attribute A.

If any of these functional dependencies is not true then attribute C is not transitively dependent on attribute A.
Consider the following relation SECOND.
Here S# → City, City → STATUS and S# → STATUS

SECOND

S#	STATUS	CITY
S1	30	Delhi
S2	20	Rohtak
S3	20	Rohtak
S4	30	Delhi
S5	15	Karnal

Figure shows the functional dependency diagram for this relation.



Attribute STATUS is transitively dependent on attribute S#. If the relation SECOND is reduced into SC(S#, CITY) and CS(CITY, STATUS), then there does not exist any transitively dependency as shown below:

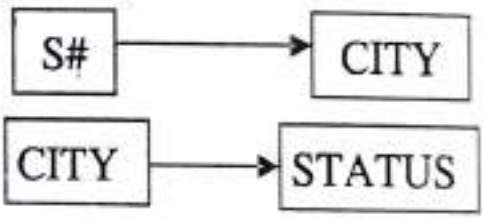
SC

S#	CITY
S1	Delhi
S2	Rohtak
S3	Rohtak
S4	Delhi
S5	Karnal

CS

CITY	STATUS
Delhi	30
Rohtak	20
Karnal	15

The corresponding functional dependency diagrams are shown below:



4. Multivalued Dependencies

Given a relation R with attributes A, B and C. Multivalued dependency represents a dependency between these attributes such that

- A leads to multiple values of C and B
- B and C are independent of each other.

MVD between attributes A, B and C in a relation R can be indicated as

$A \longrightarrow\!\!\!\!\!\rightarrow B$ (A multidetermines B)

$A \longrightarrow\!\!\!\!\!\rightarrow C$ (A multidetermines C)

Multivalued dependency in a relation is due to first normal form, which disallows an attribute in a row from having a set of values.

Example: Consider the relation

Student(SID, Subject, Activity)

Student

SID	Subject	Activity
101	English	Skating
101	Maths	Skating
101	English	Cricket
101	Maths	Cricket
102	Computer	Swimming

Here SID leads to multiple values of Subject and multiple values of Activity. Subject and Activity are independent of one another.

These multivalued dependencies can be indicated as:

$SID \longrightarrow\!\!\!\!\!\rightarrow$ Subject

$SID \longrightarrow\!\!\!\!\!\rightarrow$ Activity

5. Join dependency

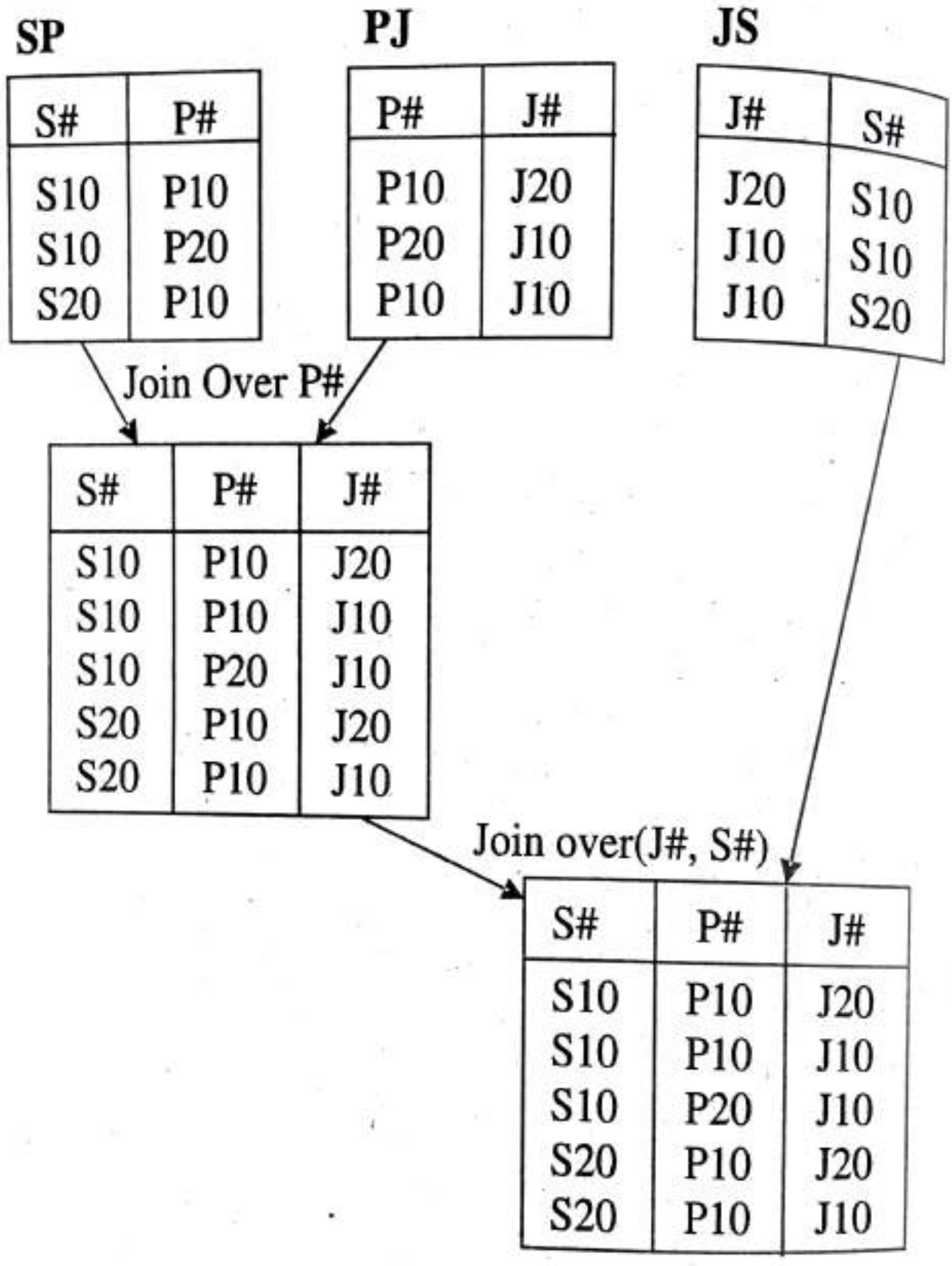
Let R be a relation and it is decomposed into {R1, R2, R3, ..., Rn}. The relation R satisfy the join dependency if and only if joining R1 to Rn = R.

Example: Consider the relation SPJ(S#, P#, J#)

SPJ

S#	P#	J#
S10	P10	J20
S10	P20	J10
S20	P10	J10
S10	P10	J10

Project it into 3 projections as: SP, PJ and JS



When joined these three relations give original form of SPJ.

Q19.(a) What is SQL? Explain its use, purpose and importance through appropriate examples.

MDU BCA 2010

Ans. SQL

Structured Query Language (SQL) is an advanced relational database language to help users to extract information from a database easily.

It was originally developed in 1970 at IBM's San Jose Research laboratory. It was originally called **S**tructured **E**nglish **QUE**ry **L**anguage or SEQUEL. The capability of an RDBMS is "Structured Query Language" as it caters to most of the routine queries placed by the user on the database.

The American National Standards Institute (ANSI) adopted SQL as the standard language for RDBMS in 1986. In 1987, SQL was adopted for use with RDBMSs, as an international standard by ISO (International Organization for Standardization).

Purpose of SQL

A database language like SQL is needed:

1. ***To create database objects such as relations, indexes etc.***
2. ***To retrieve or extract data from the database which is called querying the database.***
3. ***To perform data management tasks like insertion, modification and deletion of data from the database.***

Importance/Use of SQL

1. ***SQL is a well-defined language to interact with the database in a simple and efficient manner.***
2. ***SQL is easy to learn and can manage complex problems.***

3. *SQL is an English like language as it uses words like Select, Insert, Delete, From, Where etc. as part of its command set.*
4. *SQL is a non-procedural language. In it, the information to be retrieved is specified, rather than how to retrieve it. In other words SQL does not require to specify access methods to data in the database.*
5. *SQL provides automatic navigation to the data.*
6. *SQL provides a small and concise set of commands.*
7. *SQL saves time and reduces the amount of programming required to perform complex queries.*
8. *SQL processes set of records rather than just one record at a time.*
9. *SQL can be used by a range of users like:*
   - *Application programmers*
   - *Management personnels,*
   - *DBA and*
   - *All other types of end users*
10. *Applications written in SQL can be easily ported across systems.*
11. *SQL provides a variety of commands for tasks like:*
    - *Inserting, deleting and updating rows in a table.*
    - *Security administrators.*
    - *Querying data.*
    - *To Guarantee database consistency.*
    - *Controlling access to the database.*

Thus SQL is a simple and powerful language used to create, access and manipulate data and structure in the database.

Q19.(b) What is SQL? What are the major components of SQL? **MDU BCA 2018**

Ans. SQL

Structured Query Language (SQL) is an advanced relational database language to help users to extract information from a database easily.

It was originally developed in 1970 at IBM's San Jose Research laboratory. It was originally called **S**tructured **E**nglish **QUE**ry **L**anguage or SEQUEL. The capability of an RDBMS is "Structured Query Language" as it caters to most of the routine queries placed by the user on the database.

The American National Standards Institute (ANSI) adopted SQL as the standard language for RDBMS in 1986. In 1987, SQL was adopted for use with RDBMSs, as an international standard by ISO (International Organization for Standardization).

Components of SQL

The components of SQL are:

1. Data Definition Language (DDL)

DDL helps a user in defining the data structure in the database. It provides commands for defining relations, deleting relations, creating indices and modifying relations. It defines the physical characteristics of each record. It is used to define content and structure, format of the database. It describes data type and name of each element in the database.

Thus, if a user wants to create a relation then it can be done with the help of DDL. DDL commands are:

- CREATE TABLE
- ALTER TABLE
- DROP TABLE
- CREATE VIEW

DDL also includes commands for specifying access rights to relations and views.

2. Data Manipulation Language (DML)

DML helps a user in manipulation of data in the database i.e. It provides the way to process the data that is already existing in the database.

It provides commands for accessing, inserting, deleting and updating the rows of tables of the database. It accesses the data by the name specified in DDL. The DML statements issued in the form of queries are handled by a DML compiler.
DML commands are:

- INSERT
- DELETE
- UPDATE
- SELECT

There are basically two types of DML:

(i) Interactive DML

It includes commands for accessing, inserting, deleting and updating rows.

(ii) Embedded DML

It can be used in general purpose programming languages such as C, C++, VC++, VB, Pascal etc.

Q19.(c) Write SQL command syntax for create, alter, drop, insert, select using suitable examples.

Ans. 1. CREATE TABLE command

All data in a relational database is stored in the form of relations, also called tables. CREATE TABLE command is used to create the empty table i.e. a table without rows.

Syntax of CREATE TABLE command are:

Syntax 1

```
CREATE TABLE <table_name>
(
field_name1  field_type (width),
field_name2  field_type (width),
          :
          :
field_namen  field_type (width)
);
```

<table_name and field_names are supplied by the users. Field_type is the data type of the corresponding attribute or column with which it is specified. Field_type of a column is chosen according to the requirement. For example emp_no must be of NUMBER data type.

```
e.g. SQL> CREATE TABLE  employee
                    (
                        emp_no  number (5),
                        name  char (15),
                        joining_date  date
                    );
```

This will create the employee table with three columns emp_no, name and joining_date as shown below:

employee

emp_no	name	joining_date

When a table is created, it must have atleast one column. Also in the beginning, table is empty, it has no data, only the definition of table is stored in the database. This definition is called schema of that table.

Syntax 2

As the fields are named, certain constraints can be specified to ensure accuracy and consistency of data in a relational database. Constraints are the means by which invalid data entry into the table can be prevented. The common constraints include PRIMARY KEY, NOT NULL and CHECK constraints.

```
CREATE TABLE <table_name>
 (
 field_name1  field_type (width)  PRIMARY KEY,
 field_name2  field_type (width)  NOT NULL,
                    :
                    :
 field_namen  field_type (width), CHECK (field_namen
                                    in ('a', 'b', 'c'))
 );
```

PRIMARY KEY Constraint

It specifies one or more columns that make up the primary key of the table.

The primary key constraint ensures that the values of column cannot be null and also the values in the column will be unique.

NOT NULL Constraint

A NULL is a value that is unknown, unavailable, unassigned or inapplicable. A NULL is neither a zero nor a blank space because zero is a numeric value and blank space is a character. Also, columns of any data type can contain NULLS.

The NOT NULL constraint ensures that the values of a column can not be null.

CHECK Constraint

CHECK constraint is used to check the validity of data i.e. when data is entered into particular table column, the data in the column is limited to specific values like a, b or c.

2. ALTER TABLE Command

ALTER TABLE command can be used to make the following changes to any table:

- ***Add new columns.***
- ***Drop a column.***
- ***Modify existing columns***
  - ***Decrease length (Here all values in the column must be null)***
  - ***Change data type (Here all values in the column must be null)***
- ***Add new integrity constraints***

Syntax of ALTER TABLE command are:

```
ALTER TABLE <table_name>
        ADD ( field_name field_type (width));
```

```
ALTER TABLE <table_name>
        MODIFY ( field_name  field_type (width));

ALTER TABLE <table_name>
        DROP ( field_name);
```

e.g. To add a new column Pincode to the table employee, the command is:-

```
ALTER TABLE employee
        ADD ( Pincode  number (6));
```

3. DROP TABLE Command

DROP TABLE command is used to delete the table. A table can be deleted when it is no longer required.

Dropping a table not only deletes the data contained in the table but it also removes the definition of its structure from the database.

Syntax is

```
DROP TABLE <table_name>;
```

<table_name> is the name of table which is no longer required. Once this command is executed, the identity of the specified table is lost from the database. This command also frees the memory occupied by the table definition and data contained.

e.g. To delete the table employee, the command is:

```
DROP TABLE  employee;
```

Above command will remove the employee table from the database.

4. INSERT Command

This command inserts rows in a table. Values can be inserted for all the columns or for the selected columns. The syntax of this command is given below:

It is used to retrieve information already stored in the database. The syntax of SELECT command is:

```
SELECT ……..
FROM ……..
[WHERE ……..]
[GROUP BY ……..
[HAVING……..]
[ORDER BY …….]]
```

The main component of a SELECT statement are:

- ***The SELECT clause which shows the list of columns to be listed.***
- ***The FROM clause which enables to select database table to get the information from.***

All other clauses of SELECT statement are optional. The simplest form of SELECT statement is:

```
SELECT (column_name1, column_name2,……,
        column_namen) FROM tablename;
```

e.g. Consider the following table employee:

employee

Emp_no	Name	Design	Basic	Address
101	RAJIV	CLERK	5850	ROHTAK
102	MOHAN	SALESMAN	6400	ROHTAK
103	RAVINDER	MANAGER	9000	KARNAL
105	BALVINDER	CLERK	5850	PANIPAT
106	RAJBIR	CLERK	5850	ROHTAK
108	ASHU	SALESMAN	6400	ROHTAK

(i) Selecting all columns

Query:- What data are in the employee table?

```
SQL > SELECT *
      FROM employee ;
```

Once the above command is obeyed, the table will be:

employee

Emp_no	Name	Design	Basic	Address
101	RAJIV	CLERK	5850	ROHTAK
102	MOHAN	SALESMAN	6400	ROHTAK
103	RAVINDER	MANAGER	9000	KARNAL
105	BALVINDER	CLERK	5850	PANIPAT
106	RAJBIR	CLERK	5850	ROHTAK
108	ASHU	SALESMAN	6400	ROHTAK

The asterisk (*) is used to select all columns in the table. This is very useful when the columns names are not known. Thus, a wild card (*) character indicates to view data from every column in the table.

(ii) Selecting a specific column

```
SQL > SELECT Name
      FROM employee ;
```

Once the above command is obeyed, the table will be:

Name
RAJIV
MOHAN
RAVINDER
BALVINDER
RAJBIR
ASHU

(iii) Selecting multiple columns

```
SQL > SELECT Emp_no, Name, Design
      FROM employee ;
```

Once the above command is obeyed, the table will be:

Emp_no	Name	Design
101	RAJIV	CLERK
102	MOHAN	SALESMAN
103	RAVINDER	MANAGER
105	BALVINDER	CLERK
106	RAJBIR	CLERK
108	ASHU	SALESMAN

(iv) Select from where

It is one of the most important clause available in SQL. By specifying a WHERE clause, specific rows can be chosen. When a WHERE clause is present, the database program goes through the entire table one row at a time and checks each row to determine if the condition is true with respect to that row.

e.g.

```
SQL > SELECT  Name
            FROM  employee
                WHERE Basic = 6400 ;
```

Once the above command is obeyed, the table will be:

Name
MOHAN
ASHU

Q20.(a) What is query processing? How is it used and useful? Discuss its strategies with examples.

MDU BCA 2014

Ans. Query Processing

A user may specify a query to the system, it is necessary to find the best method to respond the query using the existing database.

Query processing is the technique of selecting the best method to be used in responding to a database request.

Generally there are different methods for computing the query. Each method of expressing the query suggests a plan to find the answer.

Firstly a user's query is converted into an internal standard representation. Subsequently, it is modified to an equivalent query which can be computed more efficiently. Then an access plan for evaluating the query is determined and the query is executed.

Thus query processing refers to a set of activities involved in transforming a high level query written in SQL, QUEL, QBE etc. into some internal representation.

When a query is presented to the system, it is the responsibility of the system to transform the query into an equivalent query which can be computed more efficiently.

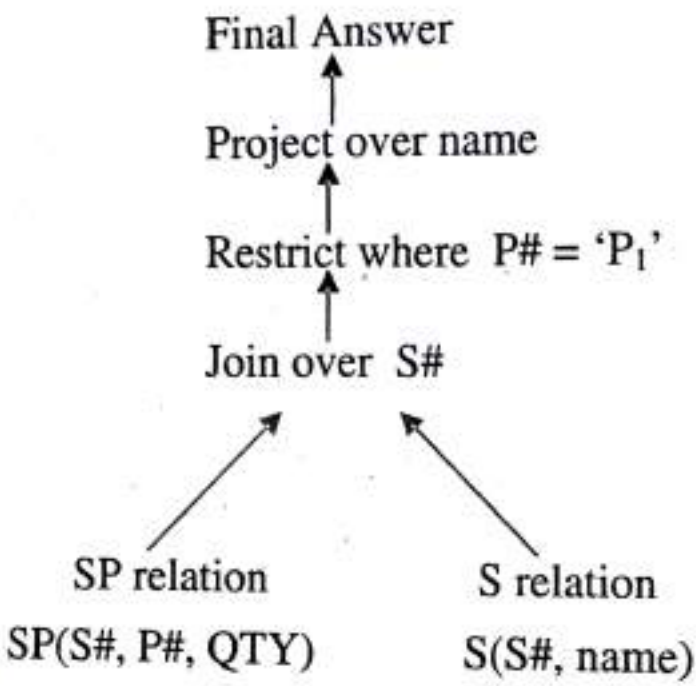
Steps in Query Processing

Query processing is a stepwise process:

(i) The first step is to transform the query into a standard starting point

First the query is to be converted into some internal form i.e. more suitable for machine manipulation. The internal form is chosen as a tree data structure called a query tree. It is also possible to represent the query using a graph data structure called a query graph.

e.g. Name of suppliers who supplies part P_1.



(ii) Transform the query

The query is transformed by replacing expression in the query with those that increase performance.

e.g. (P JOIN Q) with some restriction on P.

(P with some restriction on P) JOIN Q is more efficient than the previous one.

(iii) Simplifying the query

The query is then simplified by removing redundant and useless operations.

(iv) Prepare alternate access plans

In this step various plans, to evaluate the transformed query, are generated. The cost of each plan is estimated and the optimal one is chosen and executed.

General processing Strategies

The general query processing strategies that reduce the size of intermediate and final results as well as the processing costs are the following:

(i) Perform selection as early as possible

Selection operation reduces the cardinality of the relation, thus perform selection operations as early as possible.

e.g. Consider the relations

Branch (branch_name, assets, branch_city)

Deposit(branch_name, customer_name, balance)

Customer(Customer_name, Customer_city)

The query is "To find the assets and name of all the banks for depositors living in Rohtak".

$$\pi_{\text{branch\_name, assets}} (\sigma_{\text{customer\_city = "Rohtak"}} (\text{Customer} \bowtie \text{Deposit} \bowtie \text{Branch}))$$

Here, the intermediate result (Customer $\bowtie$ Deposit $\bowtie$ Branch) is very large to keep in main memory. It must be stored on secondary storage device. Thus the system has to access the secondary storage device for reading or writing the intermediate result.

The size of the intermediate result can be reduced by taking only those tuples of the customer relation where Customer_city = "Rohtak".

Now the query is

$$\pi_{\text{branch\_name, assets}} ((\sigma_{\text{customer\_city = "Rohtak"}} (\text{Customer})) \bowtie \text{Deposit} \bowtie \text{Branch})$$

(ii) Perform projection early

Projection operation also reduces the size of the relations. Consider the same query as above:

$$\pi_{branch\_name,\ assets}\ (\ \sigma_{customer\_city\ =\ \text{"Rohtak"}}\ (\text{Customer} \bowtie \text{Deposit} \bowtie \text{Branch}))$$

Here the subexpression

$$((\ \sigma_{customer\_city\ =\ \text{"Rohtak"}}\ (\text{Customer})) \bowtie \text{Deposit})$$

results a relation with attributes (Customer_name, Customer_city, Branch_name, Balance)

The only column needed is Branch_name. Eliminate the not required attributes Customer_name, Customer_city, Balance. Eliminating this will reduce the size of the intermediate result.

Thus the modified query is

$$\pi_{branch\_name,\ assets}\ (\pi_{branch\_name}\ ((\ \sigma_{customer\_city\ =\ \text{"Rohtak"}}\ (\text{Customer})) \bowtie \text{Deposit}) \bowtie \text{Branch})$$

(iii) Combine unary operations

Combine unary operations like

$$\sigma_X\ (\sigma_Y\ (R)) \equiv \sigma_X \wedge \sigma_Y\ (R)$$
$$\pi_X\ (\ \pi_Y\ (R)) \equiv \pi_{X \cap Y}\ (R)$$

where $X, Y \subseteq R$

(iv) Compute common expressions only once

Common expressions that appear in a query may be computed only once if the cost of storage is less than recomputing it.

Q20.(b) What is query optimization? Discuss the significance and different steps followed during query optimization. MDU BCA 2017

OR

Explain query optimization briefly with examples. MDU BCA 2013

Ans. Query Optimization

To improve the strategy for processing a query is called query optimization. In other words, choosing a suitable one out of many possible execution strategies for processing a query is known as query optimization.

Thus, the goal of the query optimizer is to choose an efficient execution strategy for processing a query. Major focus of the optimizer is to minimize the use of various computer resources such as CPU utilization time, disk I/O, memory etc.

In query optimization:

- First to find an expression that is equivalent to the given expression.
- Second to find the detailed strategy for processing the query which includes
  - (i) How the query will be executed.
  - (ii) The order in which tuples are processed.
  - (iii) How many times the disk access required.

Various Query Optimization Techniques

When a query is presented to the system, it is the responsibility of the system to transform the query into an equivalent query that can be computed more efficiently.

Query optimization is useful as it gives much difference in execution time between a good strategy and a bad one. For optimization of queries, there are two techniques.

(i) Heuristic Rules

(ii) Cost Estimation

(i) Heuristic Rules

Heuristic means it is a rule that works in most of the cases but not guaranteed to work well in every case. This rule records the operations in the query tree or in a query graph to improve the expected performance of the query.

First the internal representation of the query is generated which is then optimized according to heuristic rules. After this, query execution plan is generated to execute groups of operations based on the access paths available on the files involved in the query.

The main heuristic rule is to apply SELECT and PROJECT operations before applying the JOIN operation or other binary operations. These operations reduce the size of a file and thus must be applied accordingly.

(ii) Cost Estimation:-

A query optimizer not only depends on Heuristic rules. Optimizer estimates and compare cost of executing a query using different execution strategies and selects the strategy with the lowest cost estimation. The strategy chosen for a query depends upon:

- ***The size of each relation.***
- ***The distribution of values within columns.***

Q20.(c) What is database security? What are the various techniques to secure the database? Explain in detail. IGU BCA 2018

OR

What is database security? Why is it important? Also discuss the various security issues. MDU BCA 2015

OR

Explain database security briefly with examples. MDU BCA 2013

Ans. Database Security

Data security means protection of the data in the database against:

- ***Unauthorized access.***
- ***Intentional destruction or alteration.***
- ***Accidental introduction of inconsistency or accidental disclosure.***

Data security is of prime importance in the DBMS.

Accidental loss of data consistency is due to:

- ***Anomalies due to concurrent access to the database.***
- ***Anomalies due to distribution of data over many computers.***
- ***Crashes during transaction processing.***

Security issues related to database security

1. Privacy

Privacy states that only authorized persons should be allowed to access the database.

2. Database integrity

Database integrity states that database should be protected from improper modifications, either intentional or accidental to maintain database integrity.

3. Database availability

Database availability states that security should not restrict the authorized persons to perform their actions on the part of the database available to them.

Various techniques of database security

Perfect security is unattainable, the objective of data security is to:

- ***Minimize the risk of data loss and disclosure of data to minimum level***
- ***Implement a full recovery if a loss occurs.***

Thus security measures must be taken at various levels to protect the database.

The following four levels of defence are generally recognized for database security:

(i) Human Factor

Users must be authorized carefully. They should be provided with passwords for identification. If the user do not provide satisfactory identification and authorization, the DBMS should sign off the user with a message and keep a record like terminal number, date and time etc.

Moreover disclosure, alteration and destruction is also done by programmers. For this the responsibilities of application programmers, system programmers and operators must be clearcut fixed.

(ii) Physical Security

The computer system must be physically protected from fire, theft or other forms of destruction. To avoid theft physical security include appropriate locks and keys and entry logs to computing facility and terminals.
Precautions against fire are:

- ***Smoking should not be allowed in computer room.***
- ***The site for the computer should be properly chosen so that this site is not adjacent to fire sensitive areas.***
- ***The emergency power-off switch must be easily accessible.***

(iii) Operating System Security

The operating system provides the following features for database security:

- ***Identification and verification of users by assigning an account number and a password to every user.***
- ***Avoids direct access to primary and secondary memories and thus provides protection to data.***

On the other hand, weakness in operating system security may serve as a means of unauthorized access to the database.

(iv) Database System

The users of the database are authorized to access only a limited portion of the database. An application programmer writes only application programs not system programs. Some users are allowed to ask queries but not allowed to change the data.

Thus it is the responsibility of the system to ensure that such types of restrictions are followed.

Importance of database security

Database security is an important issue in database management because of the sensitivity and importance of data and information of an organization. The goal of the database security is the protection of data against threats such as accidental or intentional loss.

In the case of shared data, multiple users try to access the data at the same time. In order to maintain the consistency of the data in the database, database security is needed.

Due to the advancement of internet, data are accessed through World Wide Web, to protect the data against hackers, database security is needed.

Q20.(d) What are different threats of the database? How can you protect the database against misuse? Also discuss the different access control policies.

Ans. Threats to the database

Threats are the problems which are related to database. Following are the threats to database:

1. Transaction failure
2. System Failure / System Crash
3. Media Failure

1. Transaction Failure

Two types of errors can cause a transaction failure:

- Logical error
- System error

Logical errors

Logical errors are like wrong input, data not found, overflow. When such error occurs, transaction can not continue with its normal execution.

System error

When system errors occur, the system enters into an undesirable state like deadlocks. Due to such errors also transaction can not continue with its normal execution.

2. System failure

System failure can occur due to:

- ***Network errors***
- ***Presence of bugs in database software***
- ***Malfunctioning of hardware***

The occurrence of such errors in a computer system leads to system failure.

3. Media failure

It includes the hardware crashes. It results in damage of data base portion currently in use. Head crash on the disk is an example of media failure.

DBA is responsible for overall security. DBA develop policies, procedures to control these threats. These are the different methods for controlling the user access to database for security.

Security should be apply to whole enterprise, the data, database, DBMS, users, applications.

To protect the database against misuse, the following are the methods:

1. Authorization
2. Authentication
3. Encryption
4. Firewalls
5. Access control

1. Authentication

The first requirement for security is to know the users. To determine the privileges and access rights of the users, they need to be identified first.

In other words, authentications is a process by which a system verifies a user's identity and a user can request to authenticate by providing a proof of identity or any authentication token.

Passwords are one of the basic forms of the authentication. Passwords are assigned when users are

created. A database stores the passwords of users in the data dictionary in an encrypted format. The users have to provide correct password for accessing database.

2. Authorization

It is a permission given to a user, to access an object or a set of objects.

It is the process through which system obtains information about the authenticated user. It includes information about which operation user may perform and which data objects user can access.

Driver's license is a perfect example of authorization.

A user may have different forms of authorization on database. These are authorization rights:-

- **Read authorization:-** allows only reading not modifications.
- **Insert authorization:-** allows insertion not modifications.
- **Update authorization:-** allows modification not deletion.
- **Delete authorization:-** allows deletion of data.

A user can be assigned all or a combination of these authorization.

3. Encryption

It is an effective and practical way to secure data transmission over network.

This process requires an encryption device that converts the original message in a code and decryption device converts this code to the original message.

4. Firewalls

It is the most important tool for network security and is used by organizations that provide dedicated internet access to their employees.

- ***Firewalls control the network traffic flows.***
- ***Firewalls can be a single PC or a set of computers.***
- ***Firewalls track all the details of the user requests on the network. As per firewall policy every request should be passed through firewall first.***

5. Access control (Different access control policies)

The two methods for access control are:

(i) Database privileges

Privileges means permission to access a particular object in a particular manner.

For example, right to create a table, right to connect to the database, right to select rows from another user's table. Privileges are given to users so that the user can accomplish tasks required for his/her job.

Two types of database privileges are:

- ***System privileges***
- ***Object privileges***

System privileges refers to the right to perform a particular action or action on the particular type of object.

For example, creating the table, deleting the row of any table.

Object privileges refers the right to perform a particular action on a specific table, view, sequence or procedure.

For example, privilege to delete rows from the table employee.

(ii) Roles

Roles are designed to ease the administrator of end user system. They can be granted to users. There are two types of roles:

- Application role
- Users role

Application role contains all privileges to run an application.

User role contains common privileges required for a group of database users.

Q21.(a) What is concurrency control? How is it implemented? Explain. **MDU BCA 2016**

Ans. Concurrency Control

When two or more than two transactions try to process the same data item simultaneously, such situation is called concurrent processing.

For this concurrent access to a database, it is necessary for the system must control the interaction among the concurrent transactions:

- ***To preserve the consistency of the database.***
- ***To ensure that the modifications made by the transactions are not lost.***

This control is achieved by various techniques called concurrency control techniques.

When many users use the database for read-only, there is no problem but if the transactions are running concurrently to modify some data items, it may lead to inconsistencies or lead to incorrect values. The effect of concurrent updates without concurrency control is explained in the following example.

Example: Two users are in the process of updating the same saving account record. In the beginning Customer A has balance of Rs. 4000.

(i) User1 reads this record in his work area and withdraws Rs. 2000.

(ii) Next, User2 reads the same record in his work area and deposits Rs. 1000.

(iii) User1 stores the record and the balance is Rs. 2000.

(iv) User2 stores the record on the record stored by User1 and the current balance is Rs. 5000. This

Locking Techniques for Concurrency Control

To avoid the concurrent update problem, the data which is accessed for update must not be shared among users. To stop this type of sharing, the DBMS must incorporate a resource-locking mechanism. That is, the data which are accessed by one user for updating must be "locked" or denied to other users until the update is completed.

A Lock is a variable associated with a data item. It describes the status of the item with respect to possible operations that can be applied to it.

Types of Locks

(i) Binary Locks

A binary lock can have two states. These are locked and unlocked. 1 is used for lock and 0 is used for unlock.

A different lock is associated with each database item X.

- ***If the value of lock on item X is 1, then it means that item X can not be accessed by a database operation.***
- ***If the value of lock on item X is 0, then it means that item X can be accessed when requested.***

Suppose the current value of the lock with item X is LOCK(X) and let the two operations used with binary lock are LOCK_ITEM(X) and UNLOCK_ITEM(X).

- **LOCK_ITEM(X):** To access an item X, the transaction request by using LOCK_ITEM(X) operation. Now two cases arises:

  (a) If LOCK(X) = 0, it will be set to 1 and the transaction will accesss the item X.

  (b) If LOCK(X) = 1, the transaction will have to wait until LOCK(X) = 0.

- **UNLOCK_ITEM(X):** UNLOCK_ITEM(X) operation sets LOCK(X) to 0 so that X may be accessed by other transaction.

Thus it is clear that a binary lock enforces mutual exclusion on data item i.e. no two transactions can access the same item simultaneously.

Every transaction has to obey the following rules when binary lock is used:

(a) A transaction T must issue

- ***The operation LOCK_ITEM(X) before any read_item(X) or write_item(X) operation.***
- ***The operation UNLOCK_ITEM(X) after all read_item(X) and write_item(X) operations are completed.***

(b) A transaction T will not issue

- ***The operation LOCK_ITEM(X) if it already holds the lock on item X.***
- ***The operation UNLOCK_ITEM(X) unless it already holds the lock on item X.***

Binary locks are simple to implement but the restriction is that at most one transaction can hold a lock on an item.

(ii) Shared/Exclusive OR Read/Write Locks

These locks are generally used in database locking schemes because they provide general locking capabilities.

The lock that allows to read a data item is called read_lock. It is also called shared_lock because other transactions are also allowed to read the item.

The lock that allows to write the data item is called write_lock. It is also called exclusive lock because only a single transaction exclusively holds the lock on the item. Thus

- If LOCK(X) = Read_lock
  then the value of locking transaction will be a list of one or more transactions.
- If LOCK(X) = write_lock
  then the value of locking transaction will be a single transaction.

(iii) Two-Phase Locking

When more transactions execute on a database concurrently, the results produced may not always be correct. A transaction will be correct when it transforms a database from one correct state to another correct state.

If the transactions are processed serially i.e. one at a time, the final result is bound to be correct provided all the transactions are correct individually.

Example: Consider a banking system where two-transactions T_0 and T_1 transferring funds from an account X to Y. The transactions are shown below:

T_0	T_1
Read X	Read X
X = X – 10	X = X – 50
Write X	Write X
Read Y	Read Y
Y = Y + 10	Y = Y + 50
Write Y	Write Y

Suppose current balance in account X is Rs. 100 and Y is also Rs. 100. If T_0 and T_1 execute serially then the final balance using the sequence

Types of Two-phase Locking

(i) Conservative two-phase locking

It requires a transaction to predeclare all the locks it requires before its execution. If any of the predeclared items cannot be locked the transaction waits until all items are locked. This locking protocol is deadlock free but it is difficult to use in practice.

(ii) Strict two-phase Locking

In this protocol, a transaction does not release any of its write lock until it commits or aborts.

(iii) Rigorous two-phase Locking

In this protocol, a transaction does not release any of its write lock and read locks until it commits or aborts.

Problems in Concurrency through Locking

Locking protocol introduces many problems which the system must be able to resolve. The major problems in concurrency through locking are:

(a) Deadlock

(b) Starvation

(a) Deadlock

Deadlock occurs when each transaction Ti in a set of two or more transactions is waiting for some item that is locked by some other transaction Tj in the set. It means that each transaction in the set is waiting for one of the other transaction in the set to release the lock on the required item.

Example: The following two transactions T_1 and T_2 are dead locked. Transaction T_1 is waiting for Y which is

deadlock. If deadlock exists the system must break or recover the deadlock. This approach involves two steps:

- ***First, the deadlocked processes are identified or detected.***
- ***The next step is to break or recover the deadlock.***

The various strategies for recovery of deadlock are:

- ***Abort all deadlocked processes.***
- ***Backup each deadlocked process to some previously defined checkpoint and restart.***
- ***Successively abort deadlocked processes until deadlock no longer exists.***
- ***Successively preempt resources until deadlock no longer exists.***

(b) Starvation

Starvation occurs when a transaction cannot proceed for an indefinite period of time while other transactions in the system continue normally.

Example: Suppose two transactions T_1 and T_2 try to put a lock on item X, the scheduler allows T_1 to lock it. Before the execution of Transaction T_1 another Transaction T_3 demands for item X. The scheduler allows T_3 to lock on item X. Again Transaction T_4 requests for item X and scheduler allocates item X to T_4, T_2 to wait further. Thus T_2 have to wait indefinitely while there is no deadlock. Such a situation is called starvation.

There are many algorithms to resolve this problem. These are:

- First come – first serve algorithm
- Priority algorithm.

First come – first serve algorithm

This algorithm allocates the resources according to the order of requests made by transactions.

Priority algorithm

This algorithm assigns dynamic priority to the requests and allocates resources according to the priority of the requesting transactions.

Concurrency Control based on Timestamp ordering

Concurrency control problem can be handled by assigning ordered timestamp values to transactions. Timestamp values are assigned to transactions as per their starting time. This means that time stamps are based on the order in which transactions started. In this technique locking is not used. Therefore deadlock can not occur.

Q21.(b) Explain database recovery briefly with examples. MDU BCA 2013

Ans. Database Recovery

Recovery in database system means, recovering the database itself i.e. restoring the database to the most recent consistent state just before the time of failure.

A transaction is a collection of operations that performs a single logical function in a database application. Each transaction is a unit of atomicity.

"All or none" requirement is called Atomicity. In other words atomicity of a transaction means either the transaction is fully completed on the database or none of its part is committed. To preserve the atomicity of a transaction is the main goal of recovery process.

Thus if a transaction does not complete its execution, the recovery process has to make sure that the transaction has no effects on the database by undoing it.

Recovery may be of three types:

(i) Transaction Recovery

(ii) System Recovery

(iii) Media Recovery

(i) Transaction Recovery

Transaction is:

- ***Updation of any information to database.***
- ***To insert some data to database.***
- ***To delete something from database or***
- ***To create or drop some table from database.***

If a transaction executes and a failure occurs before the transaction reaches its termination, it is called transaction failure.

Recovery from transaction failures means that the database is recovered to the recent consistent state just before the time of failure. For this, the system has to kept information about the changes that are applied to data items by the various transactions.

(ii) System Recovery

System failure affects all the transactions currently in progress but do not physically damage the database. Power failure is an example of system failure. In this case database becomes inconsistent and the strategy is to reverse any changes that caused the inconsistency by undoing some operations.

It may also be necessary to redo some operations at restart time that did successfully complete before crash in order to restore a consistent state of the database.

Thus

- ***redo all work done by transactions that completed successfully prior to the crash.***
- ***undo all work done by transactions that started but did not complete prior to the crash.***

(iii) Media Recovery

Media failures cause damage to the database or to some portion of it. Here database is physically damaged like a disk crash.

The recovery method restores a past copy of the database that was backed up to archival. Then using the log to redo all transactions that completed since that backup copy was taken.

Recovery procedures/methods/techniques

For database recovery there are many techniques and the selection of technique depends upon the nature of failure.

Log-based Recovery

The most widely used recovery technique is log-based recovery technique. A log is a file which records all the changes done on the database. The structure of a log-record is implementation dependent and it has the following fields:

- **Transaction name:** The unique name of the transaction. It distinguishes one transaction from another.
- **Data item name:** The unique name of the data item modified by the transaction.
- **Previous value:** The value of data item before it was modified.
- **Current value:** The value of data item after it was modified.

There are many other log-records to identify the data and time of commencement of a transaction, commit or abort of a transaction. Various types of log-records are denoted as:

- ***< Ti, start > Transaction Ti has started.***
- ***< Ti, Xj, V_1, V_2 > Transaction Ti has performed an operation on data item Xj, Xj had V_1 previous value and V_2 as current value.***
- ***< Ti, commit > Transaction Ti has committed.***