`

Database Management System

1.1 Introduction to data, database, Database system, DBMS

Data

Data is a collection of raw (plain) and unorganized facts that need to be processed. They do not have specific meaning and can exist in any form such as numbers, text, graphics, audio, video etc.

Information

When data are processed, organized, structured or presented in a given context so as to make them useful, they are called Information. Information is meaningful to the users.

For example, researchers who conduct a market research survey might ask a member of the public to complete questionnaires about a product or a service. These completed questionnaires are data; they are processed and analyze in order to prepare a report on the survey. This resulting report is information.

Database

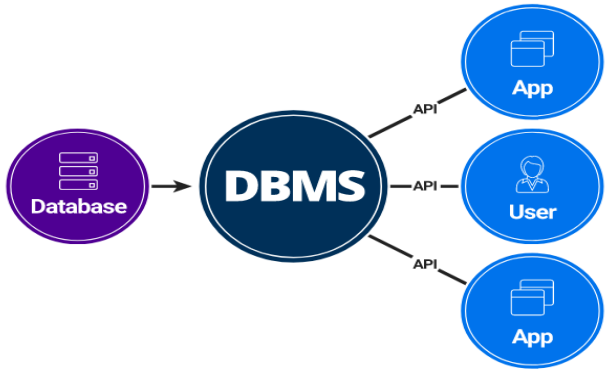
Database is a systematically organized or structured repository of data files that allows easy retrieval, updating, analysis, and output of data. These are stored usually in a computer, in the form of graphics, reports, scripts, tables, text, etc., representing almost every kind of information. It enables management of large volume of data in an organization and makes data management easy. For example, your school database maintains information about students, courses, teachers' information, and results.

Database Management System

A Database Management System (DBMS) is a software package designed to store and manages databases. In other words, It allows users to create, maintain, manipulate and retrieve data and information from a single table or from a group of inter-related tables to prepare a report or to answer the related queries and for many other applications.

The primary objective of the DBMS is to provide a convenient and effective method of defining, storing and retrieving the information contained in the database. The DBMS interfaces with the application programs, so that the data contained in the database can be used by multiple applications and users. Some examples of DBMS software are Oracle, MySQL, DB2, Microsoft SQL Server, Ms-Access, DBase etc.

`



Objectives of DBMS

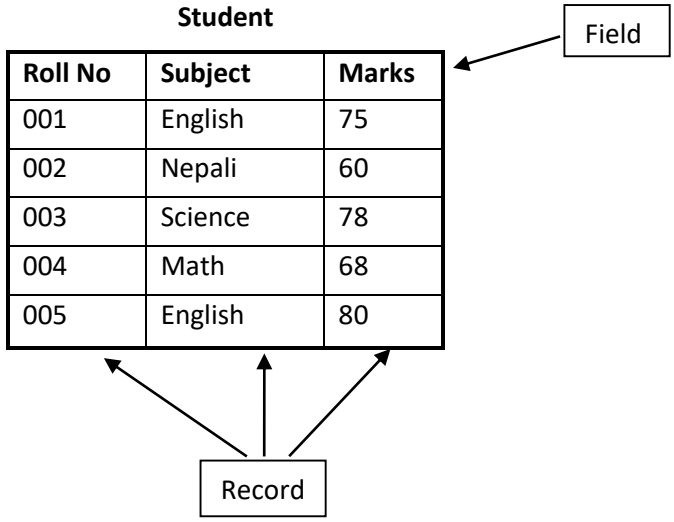
- To protect the data from physical harm and unauthorized access.
- To allow multiple users to be active at a time.
- To eliminate data redundancy.
- To allow growth in the database system.
- To enable easier data access to user.
- To provide mass storage of relevant data.
- To maintain data integrity i.e. accuracy and consistency.

1.2 Field, Record, Objects, Primary Key, Alternate Key, Candidate Key

A relational database is a common type of database that stores data in a structured format i.e. table (relation) with rows and columns. This makes it easy to locate and access specific values within the database.

Example: Let's take a relation Student

`



Field

Field is a set of data values, of the same data type in a table. It is also referred as a column or an attribute. Examples: Roll No, Subject and Marks are field names and the data 001, 002, 003, 004, 005 are the field value in above table.

Record

Record is a collection of interrelated fields, it contains data of different data type and is also referred as row or tuple. In above relation there are five records. Example: 001, English, 75 is a first record.

Objects

Objects in database are logical units within the database that are used to store information and are referred to as the back-end database. They are used to store or reference data. Some examples of database objects include tables, views, sequences, indexes, and synonyms.

- **Table**: Tables are the basic unit of data storage in database system with rows and columns. It contains all the data in a database and is also referred as relation.
- **View:** A view is a logical table which do not store data physically like tables but is like a window through which data from tables can be viewed or changed. It is stored in database in the form of query.
- **Sequence:** A sequence is a user created database object that can be shared by multiple users to generate unique integers. A typical usage for sequences is to create a primary key value, which must be unique for each row.
- **Index:** An index provides direct and fast access to rows in a table. It uses pointer to speed up the retrieval of rows. If you do not have an index on the column, then a full table scan occurs.

`

- **Synonym:** Synonym is an alternate name for table, view, sequence and other database object. It is used to create indexes in a database. With synonyms you can easily access objects and are very useful with lengthy object names such as views.

Keys

Keys play an important role in relational database. Keys in DBMS are an attribute or set of attributes which helps to identify a row in a relation (table). It helps to uniquely identify a row in a table and help to establish and identify relationships between tables.

Let us consider a relation **Student**

Stu_id	Name	Phone	Address
001	Bishal	98150905	Dharampur
002	Ram	98345677	Damak
003	Sita	98112223	Kathmandu
004	Hemanta	98178974	Bhaktapur
005	Rupen	98789643	Bhaktapur

In above table Stu_id is a key because it is unique for each student.

Types of keys used are as follows:

- Super key
- Candidate key
- Alternate key
- Primary key
- Composite key
- Foreign key

We will discuss about three keys used in database.

Candidate Key

Candidate key is a set of attributes that uniquely identify rows in a table. It is an attribute or a set of attributes that can act as a Primary Key for a table to uniquely identify each record in that table. Candidate key can be more than one in a table, i.e. it can be a combination of more than one columns.

From our given relation, Student_id and Phone both are candidate keys because both the attributes can uniquely identify each record.

`

Primary Key

Primary key is a set of attributes which uniquely identify tuples in a table. Candidate key is the one which is most suitable to be a main key to identify each record in a table. There can be many candidate keys but one of the key should be selected as a primary key. The database designers chooses the primary key. Primary key must satisfy following two characteristics:

- It cannot be null.
- It cannot be duplicate.

From the relation Student, Student_id and Phone are the Candidate keys. Student_id should be the primary key as it is more suitable then Phone.

Alternate Key

Alternate key is also known as secondary key or alternative key. They are candidate key which are not selected as primary key.

From the relation Student, Phone is the alternate key because Stu_id is the primary key.

Foreign Key

Foreign key is an attribute or combination of attributes that is used to create a relationship between two tables. A set of attributes that references primary key of another table is called primary key.

Let us take two tables to understand foreign key concept.

Employee

Emp_id	Name	Phone	Dept_id
001	Bishal	98150905	C001
002	Ram	98345677	C002
003	Sita	98112223	C002
004	Hemanta	98178974	C001
005	Rupen	98789643	C003

Department

Dept_id	Department
C001	Construction
C002	Sales
C003	Marketing

In above example, Dept_id is the foreign key in table Employee and both the tables are related.

1.3 Advantages of using DBMS

- **Reducing data redundancy:** Data redundancy refers to the duplication of same data in different data files in different places, which is a major problem with traditional file processing. DBMS can reduce such redundancy problem.

`

- **Data sharing:** The data stored in database can be shared among different users simultaneously if correct authorization protocols are followed. Even the data can be accessed remotely without any interference.
- **Data integrity:** Integrity of data means to bring the accuracy and consistency, i.e. correctness of data before and after execution of transaction. For this, DBMS gives power to its user that they can establish the validation controls.
- **Avoid data inconsistency:** Data inconsistency occur with duplicate data, if data modification is reflected in one site but not in another. DBMS avoids such data inconsistency.
- **Improved data security:** DBMS allows only authorized users to access certain data. The data in organization can be confidential so the DBA can specify account restrictions which allow different authority to different users.
- **Providing backup and recovery:** DBMS provides periodic backup of data and incase of any system failure during program execution, recovery subsystem restores the database to the earlier state.
- **Standards can be enforced:** Since DBMS is a central system, standard can be enforced easily in naming the format or structure of data. The standardized data is very helpful during migration or interchanging of data.

Limitations of DBMS

- **Expensive:** The initial setup cost is expensive. Some database software for large systems with rich functionality is very expensive. The maintenance cost is also high.
- **Requires trained manpower**: Trained technical manpower such as Database Administrator, programmers, data entry operators etc. are required to handle database systems.
- **Cost of training staff:** Staffs need to be trained properly to work with database system. Organization should invest a lot in training employees.
- **Risk of database damage:** Database stores all the data so if due to any reasons database is damaged then the data may be lost forever.

Database languages

The languages that are used to interact with database management systems are called database language. They contain the set of statements used to define and manipulate database. Types of database languages are listed below:

- Data Definition Language (DDL)
- Data Manipulation Language (DML)

`

1.4 DDL (Data Definition Language) and DML (Data Manipulation Language)

Data Definition Language (DDL)

DDL is used by the database administrators and programmers to specify the content and the structure of database. It is used to create, delete or modify tables within a database. We can also define indexes, specify links between tables and define integrity constraints in tables using DDL. DDL contains a predefined syntax for describing data. Structured Query Language supports following DDL statements:

- CREATE statement: It is used to create tables, views or other database objects.
- ALTER statement: It is used to alter structure of database tables, views or other database objects
- DROP statement: It is used to delete database tables, views or other database objects.

Syntax of **CREATE** statement is

CREATE table table_name (field1 data_type, field2 data_type,……);

Example: Statement to create a table Studentinfo with field name S_id, Name and Rollno.

CREATE table Studentinfo (S_id NUMBER (3), Name CHAR (20), Rollno NUMBER(4));

Data Manipulation Language (DML)

Data Manipulation Language has a set of statements that allows users to access and manipulate the data in a database. DML allows the insertion, deletion, modification and retrieval of data from database. Query language is also a part of DML. The DML statements are listed below:

- SELECT statement: It is used to retrieve data from the a database
- INSERT statement: It is used to insert data into a table.
- UPDATE statement: It is used to update existing data within a table.
- DELETE statement: It is used to delete records from a database table.

DML are basically divided into two categories.

- **Procedural DML:** The DML in which user specifies what data are required and how to get those data.
- **Nonprocedural DML:** The DML in which user specifies what data are required without specifying how to get those data. Example is SQL.

`

Syntax of **INSERT** statement is

INSERT into table_name VALUES (List of values);

Example: Statement to insert values 1, Sita and 12 in a table Studentinfo.

INSERT into Studentinfo VALUES (1, 'Sita', 12);

Syntax of **DELETE** statement

DELETE from table_name WHERE (condition);

Example: Statement to delete student whose rollno is 5 from table Studentinfo.

DELETE from Student_info WHERE Rollno = 5;

1.5 Database Model: Network model, Hierarchical model, Relational database model

Database Model describes the structure of the database. It is basically a collection of logical tools that are used to describe the overall structure of the database. Data modelling includes the all essential like data item, and their relationships, used to express the database. Furthermore it also facilitates the storage and retrieval of information. The primary difference between different data models is in the method of expressing the relationships and constraints among the data elements.

To describe the structure of database, several models have been introduced, and the Data Base Task Group is still trying to enhance the models. There are four typical implementation models of databases:

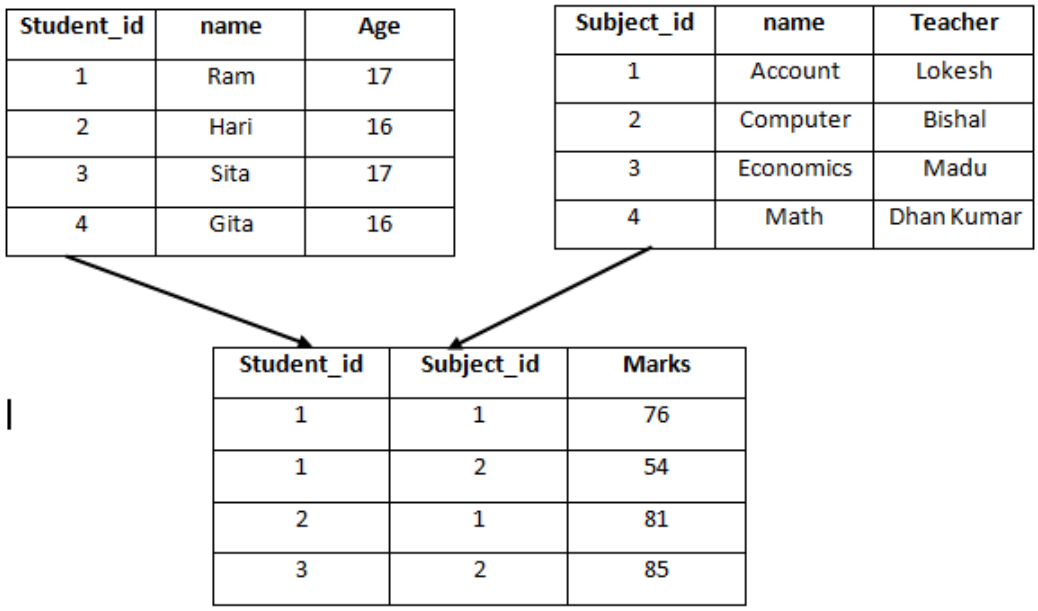
1. Relational model
2. Hierarchical model
3. Network model
4. Entity Relationship model

Relational model

The relational database model is developed by E. F. Codd in 1970 and is currently one of the most used model. The relational database structure manages data in more than one file at once, and these files are treated as tables with rows and columns, rather than as list of records. Each table column represents a data base field, and each row a database record – sometime called a Tuple. The relationship between two tables is implemented through a common attribute which makes querying much easier than in hierarchical or network database model.

`

Figure below displays the relational model of online student information system.



Student_id	name	Age
1	Ram	17
2	Hari	16
3	Sita	17
4	Gita	16

Subject_id	name	Teacher
1	Account	Lokesh
2	Computer	Bishal
3	Economics	Madu
4	Math	Dhan Kumar

Student_id	Subject_id	Marks
1	1	76
1	2	54
2	1	81
3	2	85

Advantages:

- Structural independence.
- Improved conceptual simplicity.
- Easier database design, implementation, management, and use
- Query capability (SQL) and integrity rules can be implemented easily.
- Data redundancy is minimal.

Disadvantages:

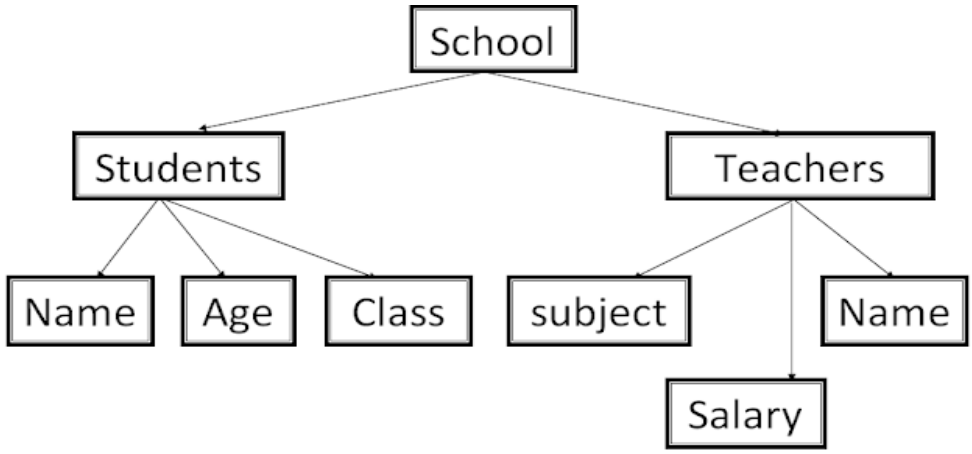
- Substantial hardware and system software overhead.
- Possibility of poor design and implementation.
- It is more complex than other models.

Hierarchical model

Hierarchical database model was developed by IBM and is the oldest database model. The data stored in database looks like a family tree with one root and a number of branches or subdivisions. It organizes data elements as tabular rows, one for each entity; the row position implies a relationship to the other rows. In this type of data structure, data are stored in

hierarchy or tree structure form. The relationship between the rows (records) is a parent child relationship in which any child record relates to a single parent type record. Any record other than the root of the tree must be connected with a superior record (the parent). Thus, to insert a record, the user must select the parent record first. When a record is deleted, all the descendants of the record are also deleted.

Figure below displays the hierarchical database model



Advantages

- it is one of the simplest database model.
- It supports one-to-one and one-to-many relationship.
- The searching is fast and easy, if parent is known.
- Efficient when dealing with a large database.

Disadvantages

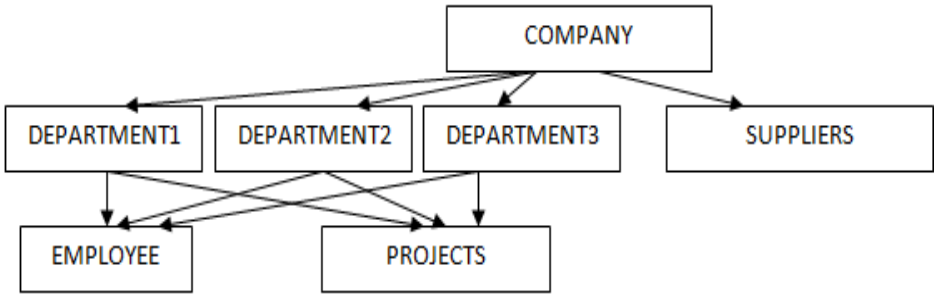
- It is old, traditional database model.
- It does not support many-to-many relationship.
- Because of dependency of parent node, when parent node is deleted, all the child nodes are also deleted.
- It increases redundancy because of the repetition of same data in different location.

Network model

This model replaces the hierarchical tree with a graph. Data are represented by collections of records. Relationships among data are represented by links. Network model is the generalization of the hierarchical model. It describes data and relations between data by using graph rather than tree like structure. The network database is like a hierarchical database but

with one modification: you can relate any record types or files with any other record types or files; that is, the data structure is many-to-many instead of one-to-many.

Figure below displays the network database model



Advantages

- It is more flexible as it accepts many-to-many relationship.
- Data redundancy is lower because data shouldn't be repeated if same data is required.
- Searching is faster because of multidirectional pointers.

Disadvantages

- It is a complex database model.
- Security is low as data can be shared.
- Lack of structural independence.

Entity Relationship model

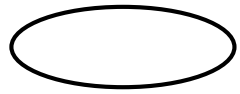
The Entity-Relationship (ER) model, a high-level data model that is useful in developing a conceptual design for a database. The ER model is well-suited to data modeling for use with databases because it is fairly abstract and is easy to discuss and explain. Database can be modeled as a collection of entities and relationship among these entities. ER models are readily translated to relations. The basic components of ER diagram are:

1. **Entity:** An entity is a real-world item or concept that exists on its own. Each entity has **attributes**, or particular properties that describe the entity type. Each entity has attributes, or particular properties that describe the entity. It is represented by rectangle.
   An entity might be

`

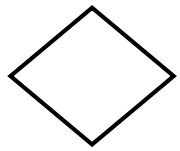
- An object with physical existence. E.g. a lecturer, a student, a car
- An object with conceptual existence. E.g. a course, a job, a position

2. **Attribute:** Entity has a set of properties which are termed as attributes. Attributes are also called elements or fields. For example, an Employee entity can have attributes (Name, Address, Age, and Salary). It is represented by ellipse.

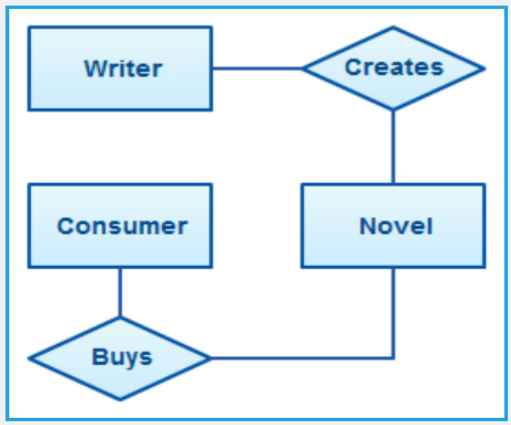


3. **Link:** The flow of information in ER diagram is indicated by the links. It connects entities, attributes and relationship. It is represented by a straight line.

4. **Relationship:** Relationship is an associations or interactions between entities. It is represented by a diamond. For example a relationship between teachers and students is teaching.

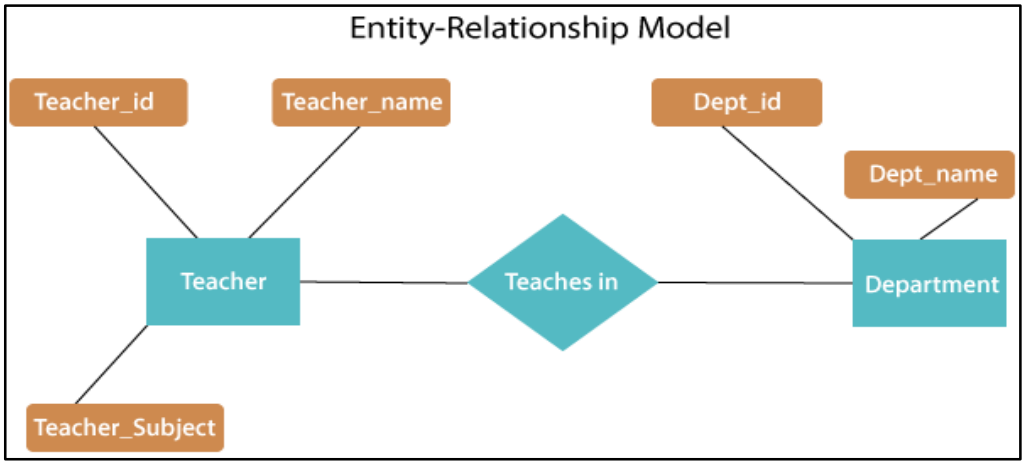


For example, the elements writer, novel, and consumer may be described using ER diagrams this way:



ER diagram with basic objects

Typical example of ER diagram



1.6 concept of Normalization

The main aim of database is to reduce redundancy i.e. the information to be stored once and use it at various places. Storing information several times leads the wastage of storage space and increases the size of data stored. Thus during the updating, database leads to be inconsistent.

Normalization is defined as the process of efficiently organizing data in a database by breaking complex database tables into simple separate tables. There are ultimately two goals of the normalization process. The first is to eliminate redundant data. Redundant data is defined as storing the same data in more than one table. The second is to ensure that data dependencies make sense by having only related data stored in the same table. Both of these goals are important since they reduce the amount of space a database consumes and ensures that data is logically stored.

Advantages of normalization

- Data redundancy is minimized.
- It helps to maintain data consistency in the database.
- It improves faster sorting and index creation.
- It helps avoid certain updating, inserting and deleting anomalies.
- It improves performance of database system
- It facilitates the enforcement of data constraints.

The most widely used 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)

We will be discussing about three normal forms.

1. First Normal Form (1NF)

First Normal Form (1NF) is the most basic of normal forms- each cell in a table (relation) must contain only one piece of information, and there can be no duplicate rows. One of the most common errors in a database is to repeat fields within a table. To achieve first normal form, we should eliminate the repeating fields. A table is in 1NF if it contains only atomic values (values that cannot be further broken down) and there are no repeating groups.

Example: Let us consider a un-normalized table STU_RESULT.

STU_RESULT

Roll No	Subject	Marks
001	English, Science	75
002	Nepali	60
003	Science	78
004	Math, Nepali	68
005	English	80

Above table is not in first normal form because the [Subject] column can contain multiple values. For example, the first row includes values "English" and "Science."

To bring this table to first normal form (1NF), we split the table into two tables and now we have the resulting tables:

`

STU_RESULT_MARKS

Roll No	Subject
001	75
002	60
003	78
004	68
005	80

STU_RESULT_SUBJECTS

Roll No	Subject
001	English
001	Science
002	Nepali
003	Science
004	Math
004	Nepali
005	English

Now first normal form is satisfied, as the columns on each table all hold just one value.

2. Second Normal Form (2NF)

A relation is said to be in 2NF, if it is in 1NF and all non-key attributes are fully functionally dependent on the primary key. . If a table has some attributes which is partially dependent on the primary key of that table then it is not in 2NF.

Example: Let us consider the following table CLASS_DETAILS.

CLASS_DETAILS

Stu_ID	Room_No	Subjects
100	550	Math
100	600	English
101	550	Math
102	575	Science
102	600	English
103	600	English

This table has a composite primary key [Stu_ID, Room_No]. The non-key attribute is [Subjects]. In this case, [Subjects] only depends on [Room_No], which is only part of the primary key. Therefore, this table does not satisfy second normal form.

To bring this table to second normal form, we break the table into two tables, and now we have the following:

`

CLASS_DETAILS_ROOM

Stu_ID	Room_No
100	550
100	600
101	550
102	575
102	600
103	600

CLASS_DETAILS_SUBJECT

Room_No	Subject
550	Math
575	Science
600	English

What we have done is to remove the partial functional dependency that we initially had. Now, in the table [CLASS_DETAILS_SUBJECT], the column [Subject] is fully dependent on the primary key of that table, which is [Room_No].

3. Third Normal Form (3NF)

A relation is said to be in 3NF, if it is in 2NF and if there is no transitive functional dependency. If table contains non key attributes that is transitively dependent upon primary key, then the table should be split to bring into 3NF. A transitive dependency exists if among three attributes A, B and C, A B, B C then C A. 3NF eliminates fields of records that can be derived from existing fields.

Example: Let us consider following table Student.

Student

Stu_id	Name	Age	Department	Building_No
PCC001	Saroj	17	Engineering	Eng101
PCC002	Jay	18	Engineering	Eng101
PCC003	Prajwal	19	Computer Science	CS11
PCC004	Nabin	20	Computer Science	CS11

The above relation contains transitive dependency;

Stu_id → Department,

Department → Building_No and

Stu_id → Building_No. Hence, it is not in 3NF. So, we convert it into 3NF by splitting the relation as below:

`

Student

Stu_id	Stu_id	Age	Department
PCC101	PCC101	17	Engineering
PCC102	PCC102	18	Engineering
PCC101	PCC101	19	Computer Science
PCC103	PCC103	20	Computer Science

Department

Department	Building_No
Engineering	Eng101
Computer Science	CS11

1.7 Centralized Vs. Distributed Database

Centralized Database

A centralized database holds all of an organisation's data on a central computer, whether mainframe or server. Dummy terminals on a mainframe system and PCs on a network system can access it. These terminals and PCs can be some distance from the central database, but the point is that all the data is together at the same place.

The main advantage of a centralized database is that it is far easier to manage and control if all the data is in one location. For example, in an ordering system, a customer may phone orders and gives notice that their address has changed; when the order operator changes that address it has been changed across the system. If the accounts department then wishes to send a statement to the customer they will have the customer's new address.

Another advantage is that the database is far easier to back up when it is centralised than if it was kept on different computers; a suitable backup strategy can easily be implemented. It is suitable for small organizations such as schools, banks, hospital etc.

`

Advantages

1. It is a simplest form of database system.
2. Maintenance of database is easier because of central storage.
3. Data integrity is maximized as the whole database is stored at a single physical location.
4. The centralized database is much secure.
5. It has a low cost of implementation and operation.

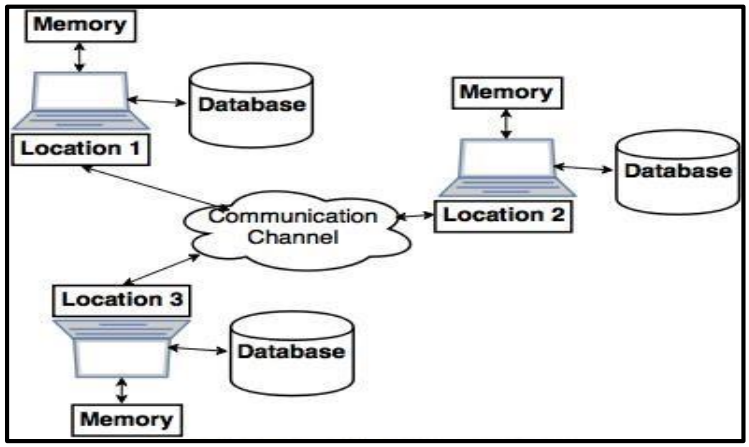
Disadvantages

1. Failure of central device such as server will leads to the disruption in whole database system.
2. If the network is slow, data access becomes very difficult.
3. When number of user increases, performance will degrade.
4. It is not suitable for larger organizations.

Distributed Database

A distributed database is a database that consists of two or more files located at different sites on a computer network. Because the database is distributed, different users can access it without interfering with one another. The DBMS must periodically synchronise the scattered databases to make sure they all have consistent data.

In a distributed database system, the database is stored on several computers. The computers in a distributed system communicate with one another through various communication media, such as high-speed networks or telephone lines. They do not share main memory or disks. The computers in a distributed system may vary in size & function, ranging from workstations up to mainframe systems.



`

Advantages

1. Expansion is database is easier as it is already spread across multiple systems.
2. Failure of one system will not affect the operation of other computer system.
3. Due to the use of multiple servers, it can support large number of users.
4. It is suitable for larger organizations with large number of users.

Disadvantages

1. It is quite complex and difficult to make sure that a user gets a uniform view of the database because it is spread across multiple locations.
2. This database is more expensive as it is complex and hence, difficult to maintain.
3. Data security is a concern as the database system is distributed in different locations.
4. Chances of data redundancy in the database as they are stored at multiple locations.

Difference between Centralized and Distributed database system

Centralized database system	Distributed database system
The databases are stored in only one site.	The databases are stored in different site and through network it can be accessed.
Managing of data is easier due to one database file.	Due to multiple database files, it requires time to synchronize data.
If centralized system fails, entire system fails.	If one system fails, system continues its operation with other site.
Requires more time for accessing data.	No effect on data access time because it can be retrieved from nearest database file.
It is suitable for small organizations	It is suitable for larger organizations
Data security is high.	High risk of data loss; hacking, data theft etc.
Cost of centralized database system is low.	Cost of centralized database system is high.

Database Administrator (DBA)

DBA is a person with a specialized computer skill responsible for maintaining a successful database environment by directing or performing all operations to keep the data secure. He/she has central control over both data and programs. DBA implements different strategies and

policies to ensure data is secure from unauthorized access and available to users without any hassle. Some of the roles and responsibilities of DBA are listed below:

- DBA works with the development team during database design process so that the potential problems can be avoided later.
- DBA is responsible for the initial installation and configuration of database server. As the updates and patches are required, DBA handles this during maintenance process.
- DBA creates backup and recovery plans and procedures. In case of server failure or other form of data loss, the DBA will use existing backups to restore lost information to the system.
- DBA is responsible for establishing and monitoring the security of the database. He/she should define integrity rules for checking accuracy and validity of data.
- DBAs are responsible for troubleshooting any problems that occurs such as problem in database connectivity, data loss, and problem with database system.
- DBA is responsible for monitoring overall activities in the system such as addition, modification and deletion of data from the database.

1.8 Database Security

Database stores sensitive information which is very vital to an organization. So, these data should only be within the access of authorized users. Database security refers to the protection of data against unauthorized access, alteration or destruction. It encompasses the range of tools, controls, and measures designed to establish and preserve database confidentiality, integrity, and availability. While designing a database, security should be the key factor and the unauthorized access to database should be checked but also this restriction should not be affecting the authorized users to perform their actions. Database integrity is key to protect improper modifications of data. DBA is responsible for maintaining overall security of the database system.

Database security can be classified into following two types:

1. **Physical security:** It refers to the security of the hardware associated with the system and the protection of the site where the computer resides. Natural events such as fire, floods and earthquakes can be considered as some of the physical threats.
2. **Logical security:** It refers to the security of software of an organization's systems, including user identification and password access, authenticating, access rights and authority levels. It ensures that only authorized users are able to perform actions or access information in a network or a workstation.

`

Importance of database security

- Database system are based on internet, data transportation is done through web. So, to protect data against hackers database security is required.
- As the data can be shared among multiple users at a time, data consistency should be maintained. For which database security is required.
- Users should be allowed to access only the information that is pertinent to their jobs by granting privileges which refers to authorization. Authorization is important for database security as in many cases the insiders are responsible for data loss.
- Prevent malware or virus infections which can corrupt data, bring down a network which leads to loss of data.

Database security measures

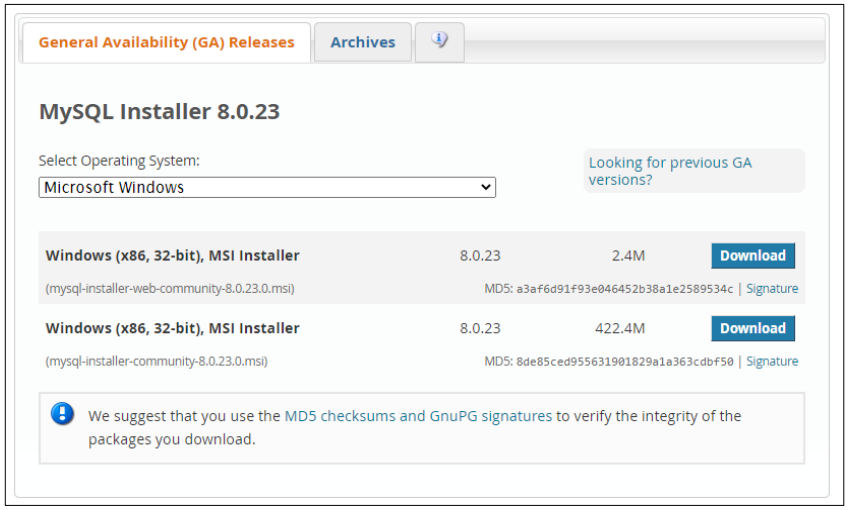
- **DBMS configuration:** DBMS should be properly configured and hardened to take advantage of security features and limit privileged access. Monitoring the DBMS configuration helps the database remain consistent.
- **Authentication:** It is process by which a system verifies a user's identity. User is verified based on the credentials stored in database restricting any unauthorized users to access the data.
- **Authorization:** After a user is authenticated, users go through another layer of security which is authorization. By granting privileges users are allowed to access only the part of database that is pertinent to their job.
- **Backups:** Backup is a process of making a copy of your data in separate location or system. It is important because it allows us to recover lost data.
- **Encryption:** While transmitting a data they are encrypted with the encryption algorithm. Encryption is a means of securing data by converting it into cipher text.
- Application security: Database and application security framework measures can help protect against common known attacker from attacks such as access controls, including SQL injection.

1.9 Getting started with MySQL

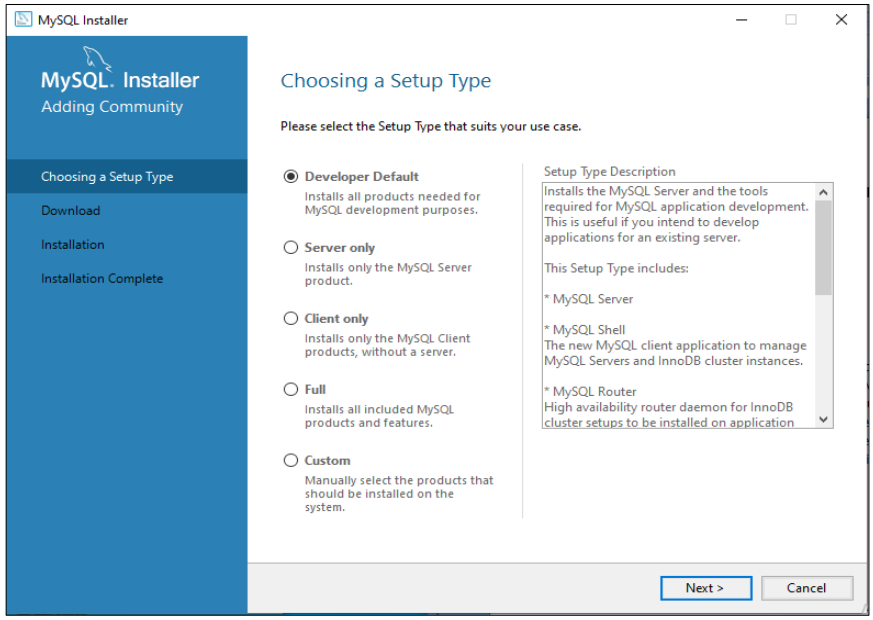
Install MySQL

MySQL is a standard part of the typical Linux server build (or LAMP stack) but is also available for use on Windows operating systems. Installing MySQL is a simple process. You can download the MySQL Installer from dev.mysql.com. You will get two download options:

1. **Web-community** version will only download the server, by default, but you can select other applications (like Workbench) as desired.
2. **The full installer** will download the server and all the recommended additional applications.



Run the installer that you download from its location on your server, by double clicking. Next you will get different setup. Choose which one suits you the best.



Developer Default: it is the full installation of MySQL Server and the other tools needed for development. If you will be managing the data directly in the database, it best suits you.

Server Only: If you only need MySQL Server installed for use with other application and will not be managing the database directly. It best suits your requirement.

`

Custom: It allows you to customize every part of the installation from the server version to whichever additional tools you select.

Install appropriate setup and complete the configuration process by following the instructions.

SQL (Structured Query Language)

SQL is a database language is a database language designed for the retrieval and management of data in a relational database management system (RDBMS). All the RDBMS like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language. SQL offers a mechanism of universal database access. It allows users to retrieve data from the database without being concerned about the DBMS processes and the structure of the records. It is also used for the manipulation of data in a database.

Data types in SQL

SQL data type is an attribute that specifies the type of data of any object. These data types are used when creating a table. You need to choose an appropriate data type for each column based on the requirement. Commonly used data types are:

1. Char (size): A fixed length character string with user specified size.
2. Varchar (size): A variable length character string with user specified maximum length.
3. Text (size): Holds a string with a maximum of 65,535 bytes.
4. Int: It represents a whole number. It is a finite subset of the integers whose range is machine dependent.
5. Numeric (p,d): A fixed point number with user-specified precision. The total numbers of digits on both sides of the decimal are the precision. ‘p’ represents the total number of digits and ‘d’ of the ‘p’ digits is to the right of the decimal point.
6. Float (n): A floating-point number, with precision of at least n digits. Precision refers to the total number of digits on both sides of the decimal.
7. Date: It represents calendar date with year, month and day format. Example: 2021-1-11.
8. Time: it represents clock time with hour, minute and second format. Example: 4:22:24

Let’s work with some DDL statements:

1. CREATE statement

CREATE statement is used to create tables, views, sequences indexes etc.

`

Syntax:

```
CREATE table table_name
(
        field1 data_type,
        field2 data_type,
        ……..
        Field n data_type,
);
```

Example:

```
CREATE TABLE Studentinfo
(
        S_id INT,
        Name CHAR (20),
        Class VARCHAR(12),
);
```

The above query when executed creates the following table

Studentinfo

S_id	Name	Class

While creating a table we can also define the integrity constraints for columns in a table.

Integrity constrains are rules or conditions applied to databases in order to maintain accuracy and correctness of data.

NOT NULL constraint: it enforces column to not accept NULL values.

UNIQUE constraint: it imposes that every value in a column must be unique and null values are allowed

Primary Key constraint: The column set to primary key will not accept NULL values and the duplicate values. Only one primary key can be defined for each table.

`

Example: Let's write a query to create a table with the constraints applied.

```
CREATE TABLE Student
(
        Sid int Primary key,
        Rollno varchar (10) Unique,
        Name varchar (20) Not Null,
        Age int
);
```

When the query above executes then, Sid is set to Primary key, Rollno column should have non repetitive values but it allows us to leave the field empty, and Name column should not be empty, all the names should be inserted.

2. ALTER statement

ALTER statement is used to change the structure of database tables, views or other database objects, i.e. It is used to add, delete, or modify columns in an existing table.

- **Adding new column**

Syntax:

ALTER TABLE table_name

ADD column_name datatype;

Example:

```
ALTER TABLE Studentinfo
ADD (Age INT);
```

The above query when executed gives us the following table

Studentinfo

S_id	Name	Class	Age

`

- **Removing column**

Syntax:

ALTER TABLE table_name

DROP (column_name);

Example:

```
ALTER TABLE Studentinfo
DROP (Age);
```

The above query when executed gives us the following table

Studentinfo

S_id	Name	Class

- **Modifying column definition**

Syntax:

ALTER TABLE table_name

MODIFY(column_name data_type);

Example:

Modify table by changing the data type of Name to varchar(25)

```
ALTER TABLE Studentinfo
MODIFY (Name VARCHAR (25));
```

3. DROP statement

DROP statement allows us to remove an existing table, view or other database object from the database.

Syntax:

DROP TABLE table_name

Example:

Remove table Studentinfo from database

```
DROP TABLE Studentinfo
```

`

Let's work with some DML statements

1. SELECT statement

SELECT statement is used to fetch the data from database table. The data returned is stored in a result table.

Syntax:

The basic syntax of SELECT statement is

SELECT column1, column2, columnN FROM table_name;

Here, column1, column2.....columnnN are the fields of a table whose values you want to fetch.

If you want to fetch all the data from the table, then use the following syntax.

Syntax:

SELECT * FROM table_name;

Example: Let us consider a table **Employees**

Emp_id	Name	Age	Address	Salary
101	Ram	32	Damak	25000
102	Shyam	31	Dharampur	30000
103	Hari	29	Manglabare	22000
104	Sita	25	Birtamod	15000

- Write SQL query to find the Name, Age and Address of the employees

  SELECT Name, Age, Address FROM Employees;

The result will be

Name	Age	Address
Ram	32	Damak
Shyam	31	Dharampur
Hari	29	Manglabare
Sita	25	Birtamod

`

- Write SQL query to display all the records from table Employees
  SELECT * FROM Employees;

WHERE clause

WHERE clause is optional and it is used to filter rows from tables specified in FROM clause. It always follows FROM clause. WHERE clause is followed by a logical expression also known as perdicate. Comparisons are performed.

SQL supports following comparision

- Equals to (=)
- Greater than (>)
- Less than (<)
- Greater than or equals to (>=)
- Less than or equals to (<=)
- Not equals to (< > or !=)

Example: Let us again consider a table Employees

Emp_id	Name	Age	Address	Salary
101	Ram	32	Damak	25000
102	Shyam	31	Dharampur	30000
103	Hari	29	Manglabare	22000
104	Sita	25	Birtamod	15000

- Write a query to find id and name of employees having age more than 30.
  SELECT Emp_id, Name FROM Employees WHERE Age > 30;

Emp_id	Name
101	Ram
102	Shyam
103	Hari
104	Sita

- Write query to find the name and address of employees whose age is greater than 30 and salary greater than or equals to 25000.

`

SELECT Name, Address FROM Employees WHERE Age > 30 AND Salary>=25000;

Name	Address
Ram	Damak
Shyam	Dharampur

2. INSERT statement

INSERT statement is used to add one or more rows into database table.

Syntax:

INSERT INTO table_name (column_list) VALUES (value_list)

When the INSERT query is executed, a single row is added into the table with the values specified for each column. Only the columns listed will be assigned values. Unlisted columns are set to null.

Example 1:

Insert values into table Studentinfo. We know, Studentinfo contains three fields; S_id, Name and Class.

```
INSERT INTO Studentinfo (S_id, Name, Class) VALUES (101, "Bishal", "Ten")
                                    OR
INSERT INTO Studentinfo VALUES (101, "Bishal", "Ten")
```

The above query when executed gives us the following table

Studentinfo

S_id	Name	Class
101	Bishal	Ten

Example 2:

Let us insert values for two more rows in Studentinfo table.

```
INSERT INTO Studentinfo VALUES
        (102, "Prajwal", "Eleven"),
        (103, "Sita", "Ten");
```

`

The above query when executed gives us the following table

Studentinfo

S_id	Name	Class
101	Bishal	Ten
102	Prajwal	Eleven
103	Sita	Ten

3. UPDATE statement

UPDATE statement is used to modify the existing records in a table. WHERE clause can also be used to update the selected rows, otherwise all the rows will be affected.

Syntax:

UPDATE table_name SET column1 = value1, column2 = value2……., column = valueN WHERE condition;

Example:

Let us take table Employees

Emp_id	Name	Age	Address	Salary
101	Ram	32	Damak	25000
102	Shyam	31	Dharampur	30000
103	Hari	29	Manglabare	22000
104	Sita	25	Birtamod	15000

- Write a query to update Address to Kathmandu of an employee whose ID is 103.

  UPDATE Employees SET Address = “Kathmandu” WHERE Emp_id = 103;

If you want to update all the column values in the table, then WHERE clause is not required.

```
UPDATE Employees SET Address = “Kathmandu”, Salary = 30000;
```

`

The updated table would have following records

Emp_id	Name	Age	Address	Salary
101	Ram	32	Kathmandu	30000
102	Shyam	31	Kathmandu	30000
103	Hari	29	Kathmandu	30000
104	Sita	25	Kathmandu	30000

4. DELETE statement

DELETE Query is used to delete the existing records or rows from a table. WHERE clause can be used to delete the selected rows, otherwise all the records will be deleted.

Syntax:

DELETE FROM table_name WHERE condition;

Example 1:

Let us refer the same table **Employees.**

```
DELETE FROM Employees
```

The above query when executed will delete all the records from table Employees.

Example 2:

Delete record of employees whose age is greater than 30.

DELETE FROM Employees WHERE age>30;

The resultant table will be

Emp_id	Name	Age	Address	Salary
103	Hari	29	Manglabare	22000
104	Sita	25	Birtamod	15000

`

EXERCISE

Short answer questions

1. Define the terms Database and DBMS.
2. Write the objectives of DBMS.
3. Define keys in database. Explain the types of keys.
4. Write the use of Primary key and Foreign key.
5. Explain the advantages and limitations of DBMS.
6. Define objects in database. Explain the types of objects.
7. Define DDL and DML with along with their statements.
8. Define database model. Explain about the types of database models used in database.
9. Define relational model with its advantages.
10. Compare hierarchical model and network model.
11. What is normalization? Explain why it is necessary?
12. Explain normalization process with examples.
13. Define 1NF, 2NF and 3NF normalization process.
14. Differentiate Centralized database and Distributed database models.
15. Who is Database Administrator (DBA)? State the major roles and responsibilities of DBA.
16. What is database security? What is the importance of database security?
17. List the ways of securing a database.
18. Define SQL. List the data types used in SQL.
19. Write short notes on:
    a. Objects in database
    b. Data and information
    c. Candidate key
    d. Entity Relationship model
    e. Distributed database

`

Lab Work

1. Consider the following table:

Student-Information		
Roll No	**Name**	**Address**
1	Ram	Dharampur
2	Shyam	Bhaktapur
3	Hari	Kathmandu
4	Sita	Damak
5	Gita	Dharampur

   a. Write SQL query to create a table Student-Details.
   b. Write SQL query to insert records in a table.
   c. Write SQL query to display all the records from the table.
   d. Write SQL query to display Roll No and Name.
   e. Write SQL query to remove the table from database.

2. Consider following table:

Employee-Information			
Emp No	**Name**	**Address**	**Department**
101	Ram	Dharampur	Sales
102	Shyam	Bhaktapur	Marketing
103	Hari	Kathmandu	HR
104	Sita	Damak	Marketing
105	Gita	Dharampur	Sales

   a. Write SQL query to create a table Employee-Information.
   b. Write SQL query to insert all the records in a table.
   c. Write SQL query to display name and address of the employees working in Sales and HR department.

`

d. Write SQL query to find the employee number and Department of employees living in Dharampur Kathmandu.
e. Write SQL query to delete record of an employee whose address is Bhaktapur.

3. Consider following table:

Employee Details				
Emp No	**Name**	**Address**	**Salary**	**Age**
101	Ram	Dharampur	10000	25
102	Shyam	Bhaktapur	20000	29
103	Hari	Kathmandu	23000	31
104	Sita	Damak	30000	25
105	Gita	Dharampur	15000	27

a. Write SQL query to create a table. Consider following information:
   - *Following Integrity constraints are defined for given columns*
     Emp No – Primary Key
     Name – Not null
b. Write SQL query to add new column "Department" in a table.
c. Write SQL query to change address to Dharampur of an employee whose Emp No is 104.
d. Write SQL query to update salary of all employees to 40000.
e. Write SQL query to delete records of employees whose salary is less than 25000 and age is greater than 25 years.