**What is data?**

* Data is raw facts and figures that provide information about something.
* Examples: names, numbers, dates, text, images, etc.
* Data can be:

1. **Structured Data**: Organized information, usually in rows and columns, like a spreadsheet or database.
2. **Unstructured Data**: Information without a clear format, like text, images, or videos.
3. **Semi-Structured Data**: Partly organized data with some structure, like JSON or XML files, which have tags but don’t fit neatly into rows and columns.

**Database**

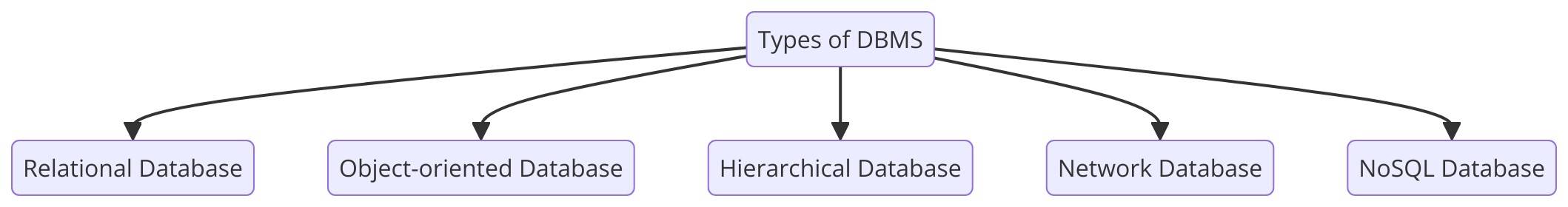
* Data base is an organized collection of data
* Collection of related data items stored in one place
* Database acts like logical collection of relevant data
* Database is designed for the purpose of storing, managing and retrieving information

The following are examples of database applications:

1. Computerized library systems
2. Automated teller machines
3. Flight reservation systems
4. Banking and Finance
5. Telecommunication

**Different Types of Database models**

* Hierarchical Model
* Network Model
* Relational Model
* Object Oriented Model
* NoSQL database



1. **Hierarchical database**

Hierarchical database follows a ranking order or a parent-child relationship to structure data.

1. **Network database**

The database is similar to a hierarchical database but has some changes. The network database connects the child record with various parent records, allowing two-directional relationships.

1. **Object-Oriented database**

In an object-oriented database, the system stores information in an object-like manner.

1. **Relational database**

A relational database is table-oriented, where every bit of data is linked to every other bit of data.

1. **Non-relational or NoSQL database**

A NoSQL database uses a variety of formats, such as documents, graphs, wide columns, etc., which offers excellent flexibility and scalability to database design.

**Database Management system**

**What is DBMS?**

* DBMS is the collection of software for organizing the information in a database that might contain routines of data input, verification, storage, retrieval, and combination.
* DBMS are applications that interact with user to others applications and database itself to capture and analyze data.
* DBMS is a suit of computer software providing interface between user's and the database.
* They are so closely related the term database when used casually often refers to both DBMS and the data it manipulates.
* A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data. A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as an interface between the database and end users or application programs ensuring that data is consistently organized and remains easily accessible.

**Work done by DBMS**

* The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema which defines the database's logical structure.
* These three foundational elements help provide concurrency, security, data integrity and uniform administration Procedures.
* The DBMS is perhaps most useful for providing a centralized view of data that can be accessed by multiple users, from multiple locations, in a controlled manner. E.g (Exam Result)
* A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing many views of a single database schema. End users and software

programs are free from having to understand where the data is physically located or on what type of storage media it resides because the DBMS handles all requests.

**Why Study DBMS**

Because now a days database is everywhere: -

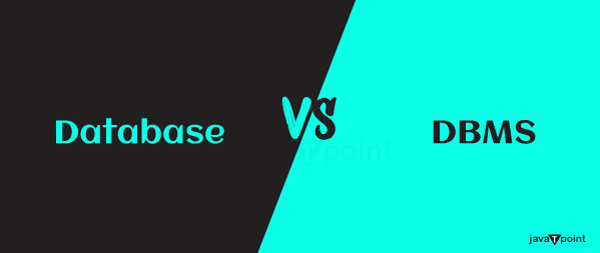
* Library Catalogue (OPAC)
* Institutional Repositories/ Digital Libraries
* Library Management System (KOНА)

**Example**: - In General (IRCTC Booking) Journey Bombay to Delhi

**Examples of DBMS:**

1. MySQL, PostgreSQL
2. Microsoft Access
3. SQL Serve
4. FileMaker
5. Oracle
6. RDBMS

**Difference Between Database And DBMS**



**Database**

* A collection of related data on individuals, places, or objects is called a database.
* The databases are intended for a relatively limited number of users-ideally a select few users-who access data at various times rather than for a huge number of users who can access data simultaneously.
* Very limited information may be changed at once when it comes to databases.
* Databases can be operated manually or by computers, however they can be quite slow if SQL is not utilized to access data.
* When an error occurs, the databases do not guarantee that the data will still be accessible.

**DBMS**

* A group of programs that allow us to create, handle, and make use of a database is called a database management system (DBMS).
* A big number of users can access the data simultaneously due to the architecture of the database management system.
* A large amount of data can be modified at once in a database management system (DBMS) because several users can access it simultaneously.
* Information can be retrieved quickly because a database management system uses a computer system.
* Data availability is guaranteed by the database management system (DBMS), even in the event of system failures.

**Relational Database**

* A relational database is a type of database that organizes data into structured tables, where each table consists of rows and columns.
* Each table holds data on a specific topic, such as customers, orders, or products.
* The rows represent individual records, while the columns capture the attributes of those records, like names, dates, or quantities.
* Relational databases establish connections between tables through relationships, often using primary keys to uniquely identify records and foreign keys to link data across tables.
* Structured Query Language (SQL) is used to interact with relational databases, enabling users to retrieve, update, and manage data effectively.
* Popular relational databases include MySQL, PostgreSQL, Oracle Database, and Microsoft SQL Server.
* These databases offer advantages like data consistency, support for complex queries, and data integrity, making them ideal for handling structured data and complex relationships in various applications.

**Relational Database Management System**

* A relational database management system (RDBMS) is a program used to create, update, and manage relational databases.
* Some of the most well-known RDBMSs include MySQL, PostgreSQL, MariaDB, Microsoft SQL Server, and Oracle Database.

**Data Organized in Tables**:

* Data in an RDBMS is stored in tables, which consist of rows and columns.
* Each table contains information about a specific entity (e.g., customers, orders, products).
* Rows represent individual records (tuples), while columns represent attributes (fields) of the data.

**Data Integrity:**

* Data integrity ensures accuracy and consistency of the data within the database.
* RDBMS enforces integrity through constraints such as:
  + - **NOT NULL**: Ensures a column cannot have a NULL value.
    - **UNIQUE**: Ensures all values in a column are distinct.
    - **PRIMARY KEY**: Ensures a unique identifier for a table.
    - **FOREIGN KEY**: Ensures referential integrity by linking tables.
    - **CHECK**: Ensures that values in a column meet specific criteria

**Normalization:**

* Normalization is the process of organizing data to reduce redundancy and improve data integrity.
* It involves dividing large tables into smaller, more manageable ones and linking them using relationships.
* Normal forms (e.g., INF, 2NF, 3NF) are used to define different levels of normalization.

**Transaction Management and ACID Properties:**

* RDBMS supports transactions, which are sequences of operations performed as a single logical unit of work.
* Transactions in an RDBMS follow the ACID properties:
* **Atomicity**: Ensures that a transaction is either fully completed or not executed at all.
* **Consistency**: Ensures that a transaction brings the database from one valid state to another valid state.
* **Isolation**: Ensures that transactions are executed independently of one another.
* **Durability**: Ensures that once a transaction is committed, it is permanently recorded in the database.

**Structured Query Language/ SQL**

SQL, or Structured Query Language, is a language used to work with data in relational databases. It lets you create, read, update, and delete data stored in tables. SQL is used in many applications to manage and organize data.

**Key features of SQL:**

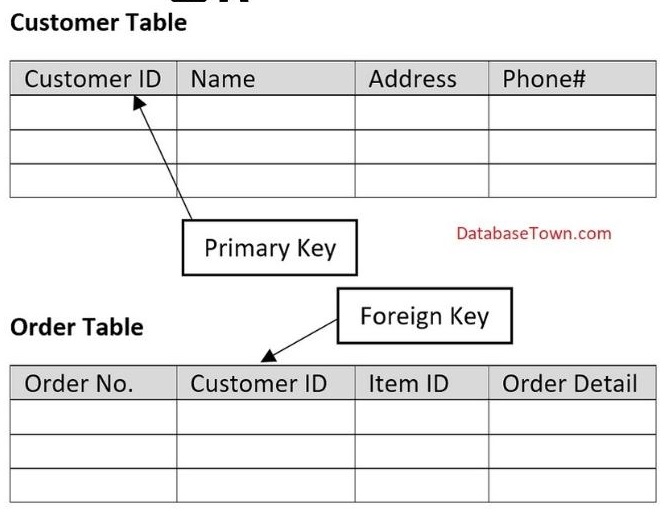
1. **Retrieve Data**: SQL can find specific data quickly using queries.
2. **Modify Data**: It allows you to add, change, or delete data in tables.
3. **Create Database Structures**: SQL can make new tables or change and delete existing ones.
4. **Control Access**: SQL helps set permissions to keep data secure and manage who can access it.

In short, SQL is a powerful tool for managing data in databases.

**Keys in Database:**

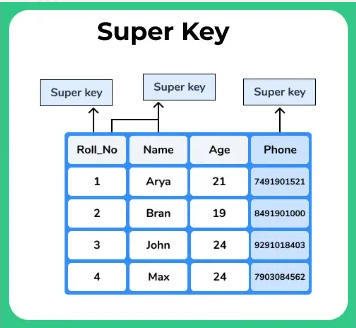
**Primary key and foreign key:**

|  |  |
| --- | --- |
| Primary key | A unique identifier for each record in a table. It cannot have a null value and must be unique across the entire table. |
| Foreign key | A field in a table that matches the primary key of another table. It creates a link between the two tables and is used to establish a relationship between them. |



**What is a Super Key?**

* A super key is a group of single or multiple keys which uniquely identifies rows in a table.
* A Super key***may have additional attributes that are not needed for unique identification.***

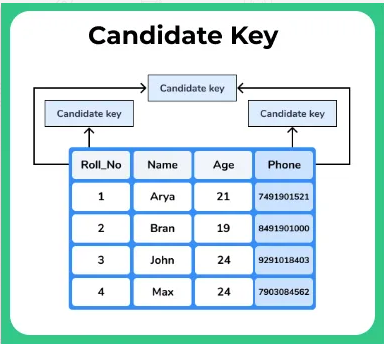


**Example of the super key**

* Consider the student table with columns [ID, NAME, PHONE]
* [ID]: As***no two students will have the same Id,*** it will help to uniquely access the student details, **hence it is a super key** Now we will identify all the Super Keys present in the table that is the set of attributes that will help to uniquely access a record
* [NAME, ID]: Even if more than one student has the same name then ***the combination name will help to recognize the record, as student id will break the tie*** and this is combination is a super key
* [PHONE]: As ***no two students will have the same phone number,*** a phone number will help to uniquely access the student details and hence ***phone number is a super key***

**What is a Candidate Key?**

* Candidate keys are selected from the set of super keys, the only thing that you should remember while selecting candidate keys is, it should not have any redundant attitude
* Definition of candidate key: Super key with no redundant attributes known as candidate key i.e. should not contain any column that contains duplicate data.



**Example for Candidate Key**

* Consider the student table with columns [ID, NAME, PHONE]
* First, identify all the super keys present in the table, then eliminate the super keys that contain a column with duplicate data. Then the remaining super keys that are left or nothing but candidate keys

1. {Id}:**ID column will contain all unique values** hence ID column is a candidate key
2. {phone}: As ***no two students have the same phone number, it is not redundant*** data column and hence phone column is a candidate key
3. {Id, phone}: As both***ID and phone are unique for all students*** this combination is a valid candidate key
4. {Id, Name}: This combination is not a candidate key because ***name column may have duplicate values***
5. {Id, phone, Name}: This combination is not a candidate key because***name column may have duplicate values***
6. {Name, Phone}: This combination is not a candidate key because ***name column may have duplicate values***

Candidate keys available is the table student

* {Id}
* {phone}
* {Id, phone}

**SQL Data Types**

In SQL, data types define the which kind of data that can be stored in a column or variable.

1. **CHAR**: String (0-255), can store characters of fixed length [e.g.: CHAR (50)]

2**. VARCHAR**: String (0-255), can store characters up to given length [e.g.: VARCHAR (50)]

3. **INT**: Integer (-2,147,483.648 to 2,147,483.647)

4**. TINYINT**: Integer (-128 to 127)

5. **BIGINT**: Integer (-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807)

6**. FLOAT**: Decimal number with precision to 23 digits

7. **DOUBLE**: Decimal number with 24 to 53 digits

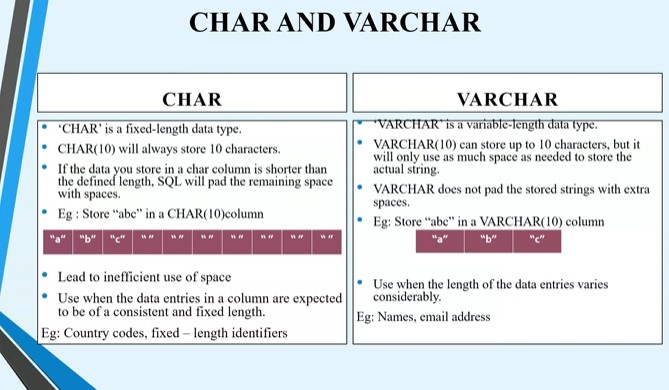
8. **BOOLEAN**: Boolean values 0 or 1

9. **DATE**: Date in format of YYYY-MM-DD ranging from 1000-01-01 to 9999-12-31

10. **TIME**: HH:MM: SS

11. **YEAR**: Year in 4 digits format ranging from 1901 to 2155Top of Form

Bottom of Form



**Relationship b/w Entities**

There are three main types of relationship:

* One-to-one
* One-to-many
* Many-to-many

**One-to-one relationships**

One-to-one relationships are those where one row of data in one table is related only to one row of data in another table. A practical example of a one-to-one relationship is the person to passport relationship.

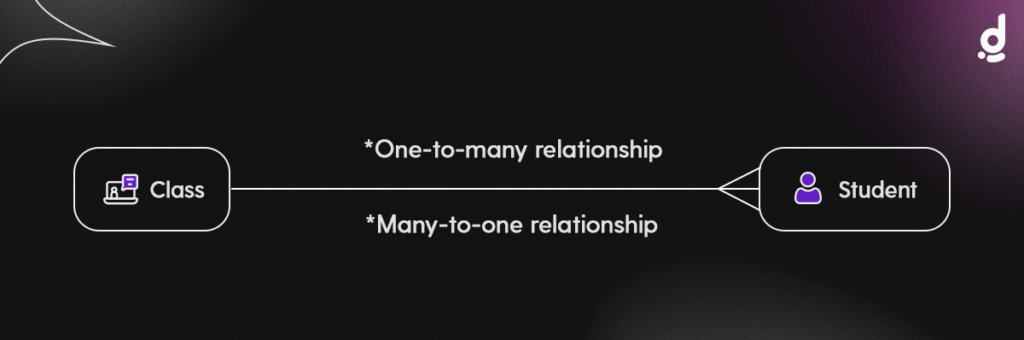
One person will have one passport – so one table storing a list of people & one storing a list of their passports.



**One-to-many relationships**

One-to-many relationships are those where one row of data in one table is related to many (one or more) rows of data in another table. A practical example of a one-to-many relationship is the class to student relationship.

At college, one student will take many classes. So, one table that contains a list of students, will be related to a table listing out a set of classes.



Notice that many to one relationship are a special case of one-to-many relationships that go in the opposite direction.

**Many-to-many relationships**

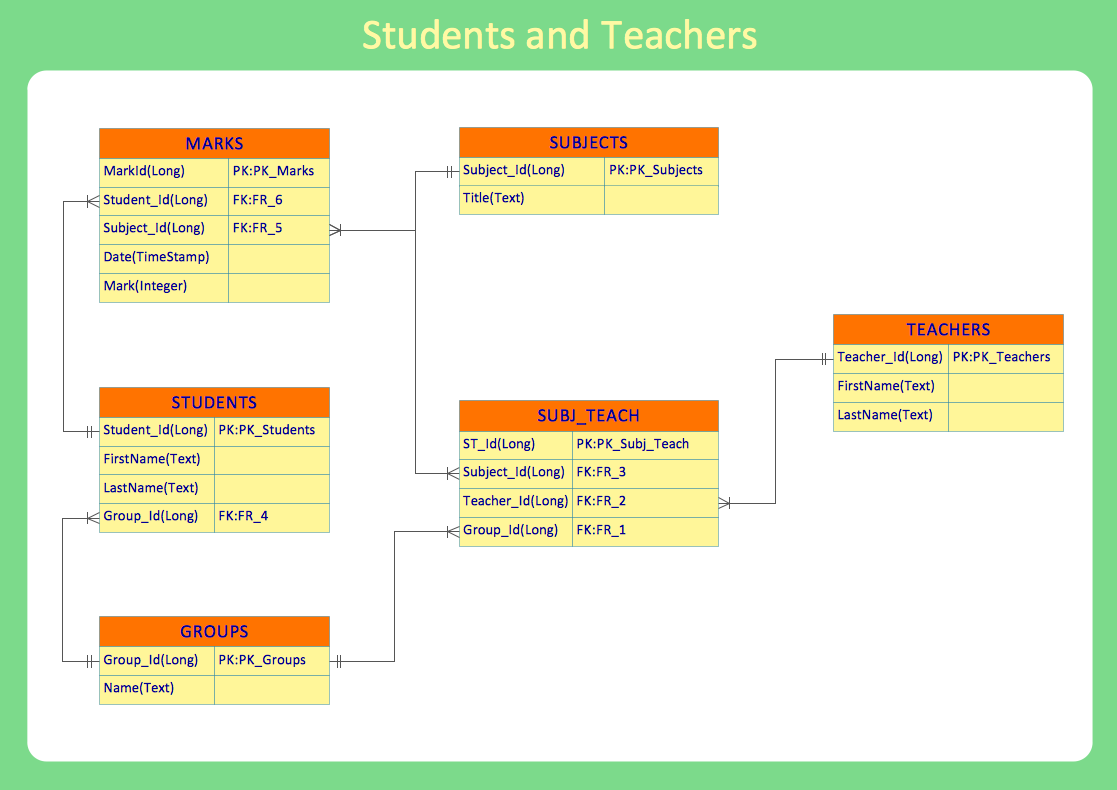
Many-to-many relationships are those where many (one or more) rows of data in one table are related to many (one or more) rows of data in another table.

A practical example of a many-to-many relationship is a table that stores a list of many universities courses & a table that stores a list of many students’ information. A Student can take multiple (many) Courses and a Course can have many attendant Students.



**What is an ER Diagram?**

An Entity Relationship Diagram (ER Diagram) pictorially explains the relationship between entities to be stored in a database. Fundamentally, the ER Diagram is a structural design of the database. It acts as a framework created with specialized symbols for the purpose of defining the relationship between the database entities. ER diagram is created based on three principal components: entities, attributes, and relationships.



**Types of SQL Commands**

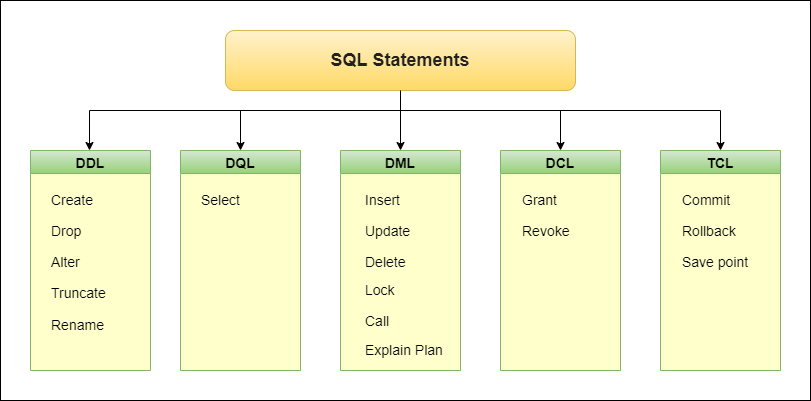
1. DDL (Data Definition Language)

2. DML (Data Manipulation Language)

3. DQL (Data Query Language)

4. DCL (Data Control Language)

5. TCL (Transaction Control Language)



**Data Definition Language (DDL)**

• It is a subset of SQL responsible for defining and managing the structure of databases and their objects.

• DDL commands enable you to create, modify, and delete database objects like tables, indexes, constraints, and more.

• **CREATE, DROP, ALTER, RENAME, TRUNCATE**

**SQL keywords are NOT case sensitive: “select” is the same as “SELECT”**

1. **CREATE Command in SQL**

SQL Create the database or its object (i.e. table, index, view, function, etc.).

**Syntax**

*CREATE DATABASE database name ;*

**Example**

CREATE DATABASE Student\_data;

**Syntax**

*CREATE TABLE table\_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype, ....);*

**Example**

CREATE TABLE Student (Student\_Id int, LastName varchar (255), FirstName varchar (255), Address varchar (255), Mark int);

1. **DROP Command in SQL**

Drop command helps to delete the object from the database (i.e. table, index, view, function, etc.).

**Syntax**

*DROP object objectname;*

**Example**

DROP TABLE Student;

**Syntax**

*DROP DATABASE database\_name;*

**Example**

DROP DATABASE Student\_data;

1. **ALTER Command in SQL**

Alter command is helpful to change or modify the structure of the database or its object.

**Syntax**

*ALTER TABLE table\_name ADD column\_name datatype;*

**Example**

ALTER TABLE Student ADD Total int;

**Syntax**

*ALTER TABLE table\_name DROP COLUMN column\_name;*

**Example**

​​​​​​​ALTER TABLE Student DROP COLUMN Mark;

ALTER TABLE Students RENAME COLUMN FirstName TO Name;

ALTER TABLE Students MODIFY COLUMN LastName varchar (200);

1. **TRUNCATE Command in SQL**

SQL Truncate command helps to remove all records from a table.

**Syntax**

*TRUNCATE TABLE table\_name;*

**Example:** TRUNCATE TABLE Student;

1. **RENAME Command in SQL**

SQL Rename is helpful to rename an object existing in the database.

**Syntax**

*ALTER DATABASE "Old\_DatabaseName" RENAME TO "New\_DatabaseName";*

**Example**

ALTER DATABASE "Student\_data" RENAME TO "Employee\_data";

**Data Query Language (DQL)**

**SELECT Command in SQL**

SQL SELECT a query on a table or tables to view the temporary table output from the database.

* **The SELECT statement is used to select data from a database.**
* **Select all records from the Customers table:**
* **SELECT \* FROM Customers;**
* **SELECT column1, column2, ...FROM table-name;**

**Here, column1, column2, ... are the field names of the table you want to select data from.**

**The table-name represents the name of the table you want to select data from.**