Database Management Systems (DBMS)

1. Database Concepts

Database: is an organized collection of structured and interrelated data, stored and accessed electronically.

Database Management System (DBMS): It is a software that allows users to define, create, maintain, and control access to the database.

Examples of some DBMS softwares: MySQL, Oracle, PostgreSQL, Microsoft SQL Server.

Relational Databases (RDBMS):

* Data is organized into tables (relations) consisting of rows and columns.
* Relationships between tables are established using keys.

Example: MySQL, Oracle.

Need for a Database:

- Reduces Data Redundancy: Minimizes duplication of data.
- Eliminates Data Inconsistency: Ensures multiple copies of the same data are identical.
- Data Sharing: Allows multiple users to access data concurrently.
- Data Integrity: Ensures the accuracy and consistency of data over its lifecycle.
- Data Security: Provides access control and authorization.
- Backup and Recovery: Protects data from loss.

2. Relational Data Model

This model organizes data into one or more tables (relations) . A relation is made of rows and columns.

Relation: A table with rows and columns.

Attribute: A column header in a table that describes the data held in that column. Example: `Roll_No`, `Name`.

Tuple: A single row in a table, representing a complete record. Example: (1, 'Anya Sharma', '2005-08-14') .

Domain: The set of all possible values that an attribute can hold. Example: The domain for `Gender` can be `{'M', 'F', 'Other'}`.

Degree: The total number of attributes (columns) in a relation.

Ex: A table with 4 columns has a degree of 4.

Cardinality: The total number of tuples (rows) in a relation.

Ex: A table with 100 rows has a cardinality of 100.

3. Keys in a Relational Database: Keys are fundamental for establishing relationships and ensuring data integrity.

Candidate Key:

* An attribute (or a set of attributes) that uniquely identifies each tuple in a relation.
* A table can have more than one candidate key.

Properties: Must be Unique and Non-redundant (i.e., no subset of the key should be unique).

Example: In a `Student` table, both `Roll_No` and `Email` could be candidate keys.

Primary Key:

* The candidate key that is chosen to uniquely identify tuples in a relation.
* A table can have only one primary key.
* It cannot contain NULL values.

Example: From the candidate keys, we choose `Roll_No` as the Primary Key.

Alternate Key:

* The candidate keys that are not selected as the primary key.

Example: If `Roll_No` is the primary key, then `Email` becomes the alternate key.

Foreign Key:

* An attribute (or set of attributes) in one table that refers to the Primary Key of another table.
* It is used to create a link between two tables (relations).
* It enforces Referential Integrity (a foreign key value must match an existing value in the referenced table or be NULL).

Example: A `DepartmentID` in an `Employee` table that references the `DepartmentID` (Primary Key) in a `Department` table.

4. SQL Commands: SQL (Structured Query Language) is used to communicate with a database.

(A) DDL (Data Definition Language): These commands are used to define and modify the database structure (schema).

(i) CREATE: Creates a new database object (e.g., a table).

```
CREATE TABLE Student (
    Roll_No INT PRIMARY KEY,
    Name VARCHAR(50)
);
```

(ii) ALTER: Modifies an existing database object (e.g., add/drop a column).

```
ALTER TABLE Student ADD COLUMN Phone VARCHAR(15);
```

(iii) DROP: Deletes an existing database object entirely.

```
DROP TABLE Student;
```

(B) DML (Data Manipulation Language) : These commands are used for managing data within the database objects.

(I) INSERT: Adds new rows of data to a table.

```
INSERT INTO Student (Roll_No, Name) VALUES (1, 'Rahul');
```

(ii) UPDATE: Modifies existing data in a table.

```
UPDATE Student SET Name = 'Rahul Kumar' WHERE Roll_No = 1;
```

(iii) DELETE: Removes rows from a table.

```
DELETE FROM Student WHERE Roll_No = 1;
```

(iv) SELECT: Retrieves data from one or more tables. (Sometimes classified as DQL - Data Query Language).

```
SELECT Name FROM Student WHERE Roll_No = 1;
```

(C)TCL (Transaction Control Language): These commands manage transactions in the database (a transaction is a logical unit of work).

(i) COMMIT: Permanently saves all changes made during the current transaction.

```
COMMIT;
```

(ii) ROLLBACK: Undoes all changes made during the current transaction, reverting the data to its state before the transaction began.

```
ROLLBACK;
```

(iii) SAVEPOINT: Sets a point within a transaction to which you can later roll back.

```
SAVEPOINT S1;
```

5. Common SQL Data Types: Data types define the kind of data an attribute (column) can hold.

(i) CHAR(n): Fixed-length character string. If the data is shorter than `n`, it is padded with spaces.

Example: `CHAR(10)` for a 10-letter code. Efficient for fixed-length data.

(ii)VARCHAR(n): Variable-length character string. Uses only as much space as needed, up to a maximum of `n` characters.

Example: `VARCHAR(100)` for a name. Efficient for variable-length data.

(iii) INT or INTEGER: Stores whole numbers (without decimals).

Example: `INT` for age, roll number.

(iv) FLOAT: Stores approximate numerical values with floating-point decimals.

Example: `FLOAT` for scientific data, percentages.

(V)DATE: Stores calendar date (Year, Month, Day).

Format: 'YYYY-MM-DD'

Example: `DATE` for date of birth.

6. SQL Constraints: Constraints are rules enforced on data columns to ensure the accuracy and reliability of the data.

(i) NOT NULL: Ensures that a column cannot have a NULL value.

```
CREATE TABLE Student (
    Roll_No INT NOT NULL,
    Name VARCHAR(50) NOT NULL
);
```

(ii) UNIQUE: Ensures all values in a column are different.

```
CREATE TABLE Student (
    Email VARCHAR(100) UNIQUE
);
```

(iii) PRIMARY KEY: A combination of `NOT NULL` and `UNIQUE`. Uniquely identifies each row in a table.

```
CREATE TABLE Student (
    Roll_No INT PRIMARY KEY
);
```

Note: A table can have only one primary key.