**Basics of Database**

**Q.1** What do you understand By Database

**Ans.** A database is a logical grouping of data. It contains the data associated with one application or with a group of related applications. It includes a collection of related table spaces and index spaces. Databases are designed to support data integrity, security, and concurrency control, ensuring that data remains accurate and accessible to users and applications. Common database systems include relational databases (like MySQL, PostgreSQL), NoSQL databases (like MongoDB, Redis), and others tailored for specific data types and use cases.

**Q.2** What is Normalization?

**Ans.**

* Database normalization is the process of removing redundant data from your tables to improve storage efficiency, data integrity, and scalability.
* In the relational model, methods exist for quantifying how efficient a database is. These classifications are called normal forms (or NF), and there are algorithms for converting a given database in normal form.

Definition:

In relational database design, the process of organizing data to minimize redundancy. Normalization usually involves dividing a database into two or more tables without losing information and defining relationships between the tables.

The goal of normalization process are:

* To minimize data redundancy.
* To minimize update, deletion and insertion anomalies.
* Improve data integrity, scalability and data consistency.
* Reduces disk space.
* The Normal Form is a state of a relation that results from applying some criteria on that relation.

**Q.3**  What is Difference between DBMS and RDBMS?

**Ans.**

|  |  |
| --- | --- |
| **DBMS** | **RDBMS** |
| DBMS store date as file. | RDBMS store data in tabular form. |
| Normalization is not present. | Normalization is present. |
| It deals with small amount of data. | It deals with large Amount of data. |
| It supports single user | It Supports multiple user |
| File storage | Tabular structure |
| DBMS will not support | RDBMS provides complete support |
| Lack of security | Good data security due to several log files |

**Q.4** What is MF Cod Rule of RDBMS Systems?

**Ans.**

Codd's original 12 rules were intended to establish the theoretical foundation and requirements for relational database systems. Here is a summary of some of the key principles from Codd's rules:

1. Information Rule
2. Systematic Treatment of NULL Values
3. Guaranteed Access Rule
4. Comprehensive Data Sublanguage Rule
5. Integrity Independence
6. High-Level Insert, Update, and Delete Rule
7. Non-Subversion Rule
8. Active Online Catalog
9. View Updating Rule

10. Physical Data Independence

 12. Distribution Independence

**Q.5** What do you understand By Data Redundancy?

**Ans.** Data redundancy refers to the situation where the same piece of data is stored in multiple places within a database or across multiple databases. This redundancy can occur intentionally or unintentionally due to poor database design or application architecture.

Data redundancy can lead to several issues:

1. **Wasted Storage Space:** Storing the same data multiple times consumes unnecessary storage space, which can increase costs and resource usage.

2. **Data Inconsistency:** When the same data is stored in multiple locations, there is a risk of inconsistencies arising if the data is updated in one place but not updated in another. This can lead to discrepancies and incorrect information being used in different parts of the system.

3. **Increased Complexity:** Managing redundant data adds complexity to the database design and application logic. It can make it harder to maintain and update the system.

4. **Decreased Data Integrity:** Redundant data can compromise data integrity, as updates or deletions of data may not be properly synchronized across all instances of the data.

**Q.6** What is DDL Interpreter?

**Ans.**

DDL (Data Definition Language) interpreter is a component of a database management system (DBMS) that processes DDL statements to define or modify the structure of a database. It translates DDL commands like `CREATE`, `ALTER`, `DROP`, and `TRUNCATE` into low-level actions that create, modify, or delete database objects such as tables, indexes, and views. The DDL interpreter updates metadata to reflect these changes and enforces data integrity constraints specified in the DDL statements. Its main role is to manage the schema and structure of the database based on user-defined commands.

**Q.7** What is DML Compiler in SQL?

**Ans.**

DML is short name of Data Manipulation Language which deals with data manipulation, and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE etc, and it is used to store, modify, retrieve, delete and update data in database.

* SELECT – retrieve data from one or more tables.
* INSERT – insert data into a table.
* UPDATE – updates existing data within a table.
* DELETE – delete all records from a table.
* MERGE – UPSERT operation (insert or update)
* CALL – call a PL/SQL or Java subprogram.
* EXPLAIN PLAN – interpretation of the data access path.
* LOCK TABLE – concurrency control.

**Q.8** What is SQL Key Constraints writing an Example of SQL Key Constraints

**Ans.** SQL constraints are used to specify rules for the data in a table.

Constraints can be column level or table level. Column level constraints apply to a column, and table level constraints apply to the whole table.

* Primary key:-

A primary key constraint uniquely identifies each record in a table and ensures that there are no duplicate values in the specified column or columns.

- Example:

CREATE TABLE Students (

student\_id INT PRIMARY KEY,

student\_nameVARCHAR(50),

age INT

);

* Unique key:-

A unique key constraint ensures that all values in a specified column or columns are unique (i.e., no duplicates), but unlike a primary key, it allows `NULL` values (except for columns defined as `NOT NULL`).

- Example:

CREATE TABLE Employees (

employee\_id INT UNIQUE,

employee\_nameVARCHAR(50),

department\_id INT

);

Foreign key:-

A foreign key constraint establishes a relationship between two tables by enforcing referential integrity. It ensures that values in a column (or columns) of one table match the values in another table's primary key or unique key.

- Example:

CREATE TABLE Orders (

order\_id INT PRIMARY KEY,

customer\_id INT,

order\_date DATE,

FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id)

);

CREATE TABLE Customers (

customer\_id INT PRIMARY KEY,

customer\_nameVARCHAR(50),

email VARCHAR(100)

);

**Q.9** What is save Point? How to create a save Point write a Query?

**Ans.** The SQL 'SAVEPOINT' is a transactional control language command that allows you to specify a point in a transaction that you can roll back to without affecting the entire transaction. This is particularly useful when you have a large transaction and you want to avoid having to redo the whole transaction if a part of it fails.

* Here's an example of how to create and use a save point within a transaction:

Start a transaction

START TRANSACTION;

* Execute SQL statements within the transaction

INSERT INTO employees (id, name, salary) VALUES (1, 'John', 50000);

INSERT INTO employees (id, name, salary) VALUES (2, 'Jane', 60000);

* Create a savepoint named 'before\_update'

SAVEPOINT before\_update;

* Update salary for employee with ID 1

UPDATE employees SET salary = 55000 WHERE id = 1;

* Check the updated records

SELECT \* FROM employees;

* Check the records after rolling back

SELECT \* FROM employees;

* Commit the transaction

COMMIT;