DBMS ASSIGNMENT

1) What do you understand By Database?

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).

2) What is Normalization?

Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency.

3) What is Difference between DBMS and RDBMS?

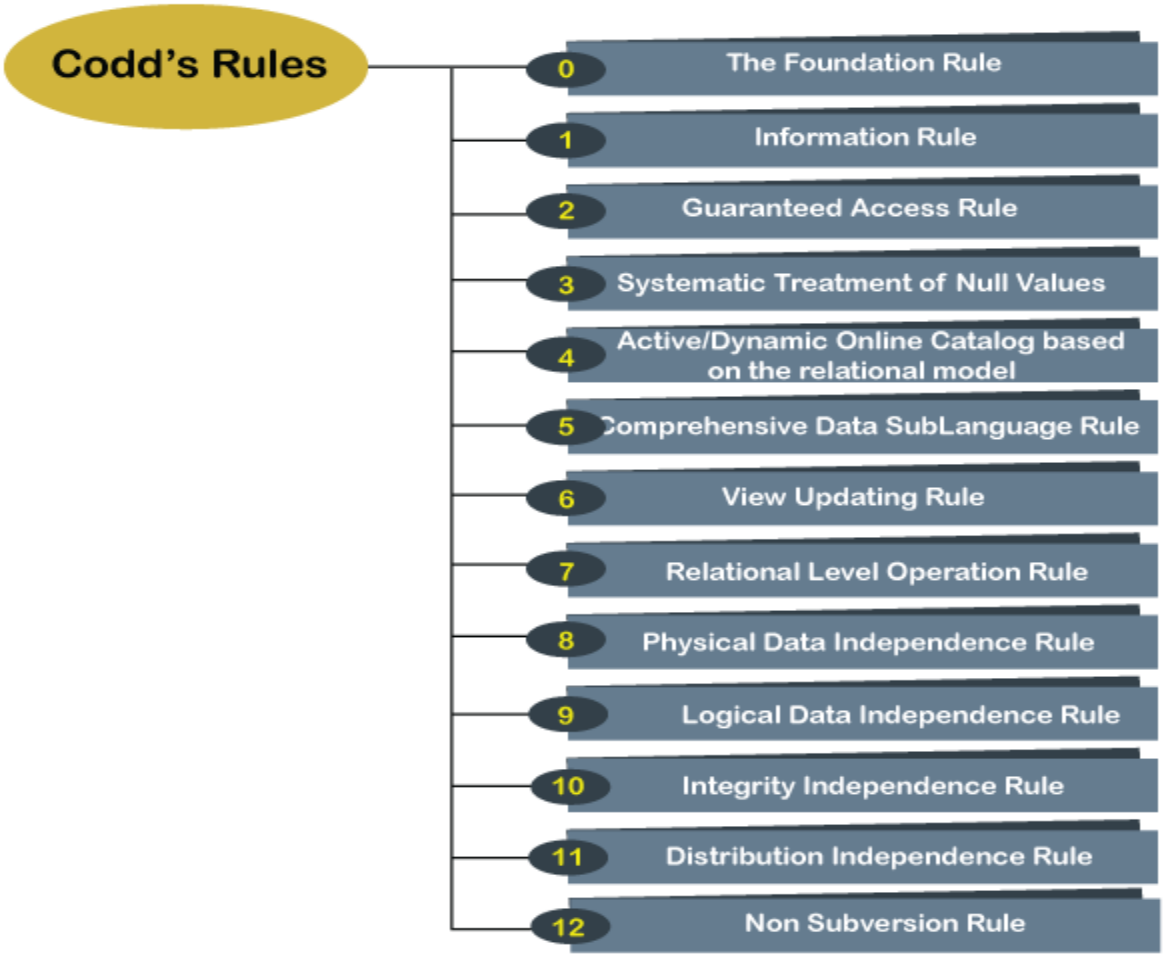
RDBMS	DBMS
Data stored is in table format	Data stored is in the file format
Multiple data elements are accessible together	Individual access of data elements
Data in the form of a table are linked together	No connection between data
Normalisation is not achievable	There is normalisation
Support distributed database	No support for distributed database

Data is stored in a large amount	Data stored is a small quantity
Here, redundancy of data is reduced with the help of key and indexes in RDBMS	Data redundancy is common
RDBMS supports multiple users	DBMS supports a single user
It features multiple layers of security while handling data	There is only low security while handling data
The software and hardware requirements are higher	The software and hardware requirements are low
Oracle, SQL Server.	XML, Microsoft Access.

4) What is EF Cod Rule of RDBMS Systems?

EF Codd's Rules: Every database has tables, and constraints cannot be referred to as a rational database system. And if any database has only relational data model, it cannot be a Relational Database System (RDBMS). So, some rules define a database to be the correct RDBMS. These rules were developed by **Dr. Edgar F.**

Codd (E.F. Codd) in **1985**, who has vast research knowledge on the Relational Model of database Systems. Codd presents his 13 rules for a database to test the concept of DBMS against his relational model, and if a database follows the rule, it is called a **true relational database (RDBMS)**. These 13 rules are popular in RDBMS, known as **Codd's 12rules**.



Rule 0: The Foundation Rule

The database must be in relational form. So that the system can handle the database through its relational capabilities.

Rule 1: Information Rule

A database contains various information, and this information must be stored in each cell of a table in the form of rows and columns.

Rule 2: Guaranteed Access Rule

Every single or precise data (atomic value) may be accessed logically from a relational database using the combination of primary key value, table name, and column name.

Rule 3: Systematic Treatment of Null Values

This rule defines the systematic treatment of Null values in database records. The null value has various meanings in the database, like missing the data, no value in a cell, inappropriate information, unknown data and the primary key should not be null.

Rule 4: Active/Dynamic Online Catalog based on the relational model

It represents the entire logical structure of the descriptive database that must be stored online and is known as a database dictionary. It authorizes users to access the database and implement a similar query language to access the database.

Rule 5: Comprehensive Data SubLanguage Rule

The relational database supports various languages, and if we want to access the database, the language must be the explicit, linear or well-defined syntax, character strings and supports the comprehensive: data definition, view definition, data manipulation, integrity constraints, and limit transaction management operations. If the database allows access to the data without any language, it is considered a violation of the database.

Rule 6: View Updating Rule

All views table can be theoretically updated and must be practically updated by the database systems.

Rule 7: Relational Level Operation (High-Level Insert, Update and delete) Rule

A database system should follow high-level relational operations such as insert, update, and delete in each level or a single row. It also supports union, intersection and minus operation in the database system.

Rule 8: Physical Data Independence Rule

All stored data in a database or an application must be physically independent to access the database. Each data should not depend on other data or an application. If data is updated or the physical structure of the database is changed, it will not show any effect on external applications that are accessing the data from the database.

Rule 9: Logical Data Independence Rule

It is similar to physical data independence. It means, if any changes occurred to the logical level (table structures), it should not affect the user's view (application). For example, suppose a table either split into two tables, or two table joins to create a single table, these changes should not be impacted on the user view application.

Rule 10: Integrity Independence Rule

A database must maintain integrity independence when inserting data into table's cells using the SQL query language. All entered values should not be changed or rely on any external factor or application to maintain integrity. It is also helpful in making the database-independent for each front-end application.

Rule 11: Distribution Independence Rule

The distribution independence rule represents a database that must work properly, even if it is stored in different locations and used by different end-users. Suppose a user accesses the database through an application; in that case, they should not be aware that another user uses particular data, and the data they always get is only located on one site. The end users can access the

database, and these access data should be independent for every user to perform the SQL queries.

Rule 12: Non Subversion Rule

The non-submersion rule defines RDBMS as a SQL language to store and manipulate the data in the database. If a system has a low-level or separate language other than SQL to access the database system, it should not subvert or bypass integrity to transform data

.

5) What do you understand By Data Redundancy?

Ans: Data redundancy refers to the practice of keeping data in two or more places within a database or data storage system. Data redundancy ensures an organization can provide continued operations or services in the event something happens to its data -- for example, in the case of data corruption or data loss. The concept applies to areas such as databases, computer memory and file storage systems.

Data redundancy can occur within an organization intentionally or accidentally. If done intentionally, the same data is kept in different locations with the organization making a conscious effort to protect it and ensure its consistency. This data is often used for backups or disaster recovery.

If carried out by accident, duplicate data may cause data inconsistencies. Even though data redundancy can help minimize the chance of data loss, redundancy issues can affect larger data sets. For example, data that is stored in several places

takes up valuable storage space and makes it difficult for the organization to identify which data they should access or update.

The word redundant can also be used as an independent technical term to refer to the following:

Computer or network system components that are installed to back up primary resources in case they fail.

Redundant information that is unneeded or duplicated.

Redundant bits or extra binary digits that are generated and moved with a data transfer to ensure that no bits were lost during the data transfer.

Redundant data that protects a storage array against data loss in the event of a hard disk failure.

.

6) What is DDL Interpreter?

DDL Interpreter DDL expands to Data Definition Language. DDL Interpreter as the name suggests interprets the DDL statements such as schema definition statements like create, delete, etc. The result of this interpretation is a set of a table that contains the meta-data which is stored in the data dictionary.

7) What is DML Compiler in SQL?

DML Compiler DML expands to Data Manipulation Language in DBMS. DML Compiler again as the name suggests compiles(or translates) the DML statements such as select, update and delete statements into low-level instructions which is nothing but the machine-readable object code to make it executable.

8) What is SQL Key Constraints writing an Example of SQL Key Constraints

Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

Syntax

```
CREATE TABLE table_name (
   column1 datatype constraint,
   column2 datatype constraint,
   column3 datatype constraint,
```

....
);

SQL Constraints

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

Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted.

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

The following constraints are commonly used in SQL:

1) NOT NULL - Ensures that a column cannot have a NULL value
2) UNIQUE - Ensures that all values in a column are different
3) PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
4) FOREIGN KEY - Prevents actions that would destroy links between tables
5) CHECK - Ensures that the values in a column satisfies a specific condition
6) DEFAULT - Sets a default value for a column if no value is specified
7) CREATE INDEX - Used to create and retrieve data from the database very quickly\

9) What is save Point? How to create a save Point write a Query?

Ans: A savepoint is a special mark inside a transaction that allows all commands that are executed after it was established to be rolled back, restoring the transaction state to what it was at the time of the savepoint.

syntax:

SAVEPOINT SAVEPOINT_NAME;

This command serves only in the creation of a SAVEPOINT among all the transactional statements. The ROLLBACK command is used to undo a group of transactions.

The syntax for rolling back to a SAVEPOINT is as shown below. ROLLBACK TO SAVEPOINT_NAME;

Query:-

START TRANSACTION; SAVEPOINT ini;

INSERT INTO student VALUES (10, "Saurabh Singh", 54, "Kashmir", "198 9-01-06");

ROLLBACK TO ini ;

10) What is trigger and how to create a Trigger in SQL?

A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server. DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view

TRIGGER Example:

DELIMITER //

CREATE trigger TR

after insert ON Employee

for each row

BEGIN

insert into view values(new.Employee_id,new.First_Name,new.Last_Name,new.Salary,new.joining_date,new.department);

END; //