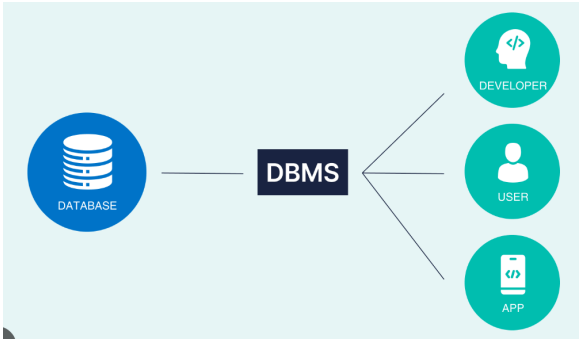
SQL REVISION

What is a Database?

Data is the new fuel of this world but only data is unorganized information, so to organize that data we make a database. A database is the organized collection of structured data which is usually controlled by a database management system (DBMS). Databases help us with easily storing, accessing, and manipulating data held on a computer.



Why is DBMS Required?

Database management system, as the name suggests, is a management system that is used to manage the entire flow of data, i.e, the insertion of data or the retrieval of data, how the data is inserted into the database, or how fast the data should be retrieved, so DBMS takes care of all these features, as it maintains the uniformity of the database as well as the faster insertions as well as retrievals.

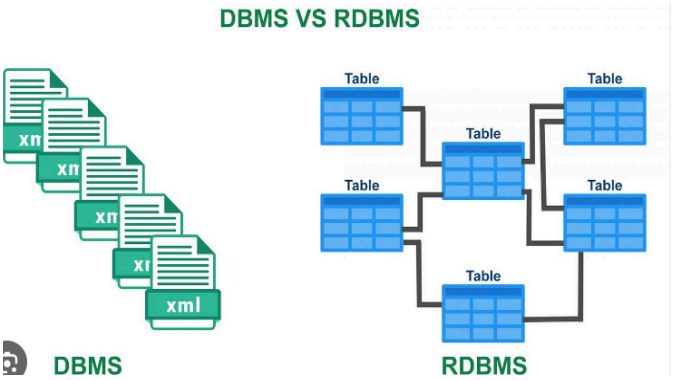
Why is RDBMS Required?

RDBMS on the other hand is a type of DBMS, as the name suggests it deals with relations as well as various key constraints. So here we have tables which are called schema and rows which are called tuples. It also aids in the reduction of data redundancy and the preservation of database integrity.

DBMS VS RDBMS

PARAMETERS	DBMS	RDBMS
STORAGE	IT STORES DATA AS A FILE.	IT STORES DATA IN THE FORM OF A TABLE.
NO. OF USERS	IT SUPPORTS ONE USER AT A TIME.	MULTIPLE USERS CAN ACCESS IT AT A TIME.
NORMALIZATION	IT DOESN'T SUPPORT NORMALIZATION	IT SUPPORTS NORMALIZATION.
DISTRIBUTED DATABASES	IT DOES NOT SUPPORT DISTRIBUTED DATABASES.	IT ALLOWS DISTRIBUTED DATABASES.
CLIENT-SERVER	IT DOESN'T SUPPORT CLIENT-SERVER ARCHITECTURE.	IT SUPPORTS CLIENT-SERVER ARCHITECTURE.
EXAMPLES	XML, FILE SYSTEM, WINDOWS REGISTRY, ETC.	SQL SERVER, MYSQL, ORACLE, ETC.

Made by Sejal Randive



What is SQL in DBMS?

Structured query language (SQL) is a programming language for storing and processing information in a relational database.

- SQL Commands

DDL	DML	TCL	DQL	DCL
Create	Insert	Commit	Select	Grant
Drop	Update	Savepoint		Revoke
Alter	Delete	Rollback		
Truncate				

1. DELETE :

It is used to delete one or more tuples of a table. With the help of the "DELETE" command, we can either delete all the rows in one go or can delete rows one by one. The TRUNCATE command does not remove the structure of the table.

Note – Here we can use the "ROLLBACK" command to restore the tuple because it does not auto-commit.

2. DROP :

It is a Data Definition Language Command (DDL). It is used to drop the whole table. By using this command the existence of the whole table is finished or lost.

Note – Here we can't restore the table by using the "ROLLBACK" command because it auto commits.

3. TRUNCATE :

DDL Command. It is used to delete all the rows of a relation (table) in one go. With the help of the "TRUNCATE" command, we can't delete the single row as here WHERE clause is not used. By using this command the existence of all the rows of the table is lost. It is comparatively faster than the delete command as it deletes all the rows fastly.

Note – Here we can't restore the tuples of the table by using the "ROLLBACK" command.

CLAUSES

What is the WHERE?

WHERE clause is used to filter the data in a table on the basis of a given condition, WHERE Clause cannot be used with AGGREGATE Functions. That's why we use the HAVING function.

What is the HAVING?

The HAVING clause is used in SQL to filter grouped data

SELECT select_list

FROM table_name

GROUP BY group_list

HAVING conditions;

- Group By Clause

It is used to group the data on the basis of one or multiple columns. To understand the HAVING we need to understand the Group by Clause. Group by is used for aggregate functions like sum, average, max, min, and count.

- GROUP (Aggregate) Functions

There are following aggregate or group functions available in MySQL:

Name	Description
AVG()	Returns the average value of the argument.
COUNT(DISTINCT)	Returns the COUNT of a number of different value
COUNT()	Returns a COUNT of the number of rows returned
MAX()	Returns the maximum value.
MIN()	Returns the minimum value.
SUM()	Returns the sum.

SELECT [Column] (Column), ...

[ORDER By Column]

FROM table [WHERE Condition]

[GROUP BY Column];

- Order of Execution

- The SQL LIKE Operator

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

JOINS

Why do we use JOINs in SQL?

The purpose of JOINs in SQL is to access data from multiple tables based on logical relationships between them. JOINS are used to fetch data from database tables and represent the result dataset as a separate table.

1. INNER JOIN:

Use an INNER JOIN when you want to retrieve only the rows where there is a match in both tables. In this example, you want to get a list of employees along with their corresponding department names.

SYNTAX:

```
SELECT*
FROM Table_A
JOIN Table_B;
SELECT *
FROM Table_A
INNER JOIN Table_B;
```

2. LEFT JOIN:

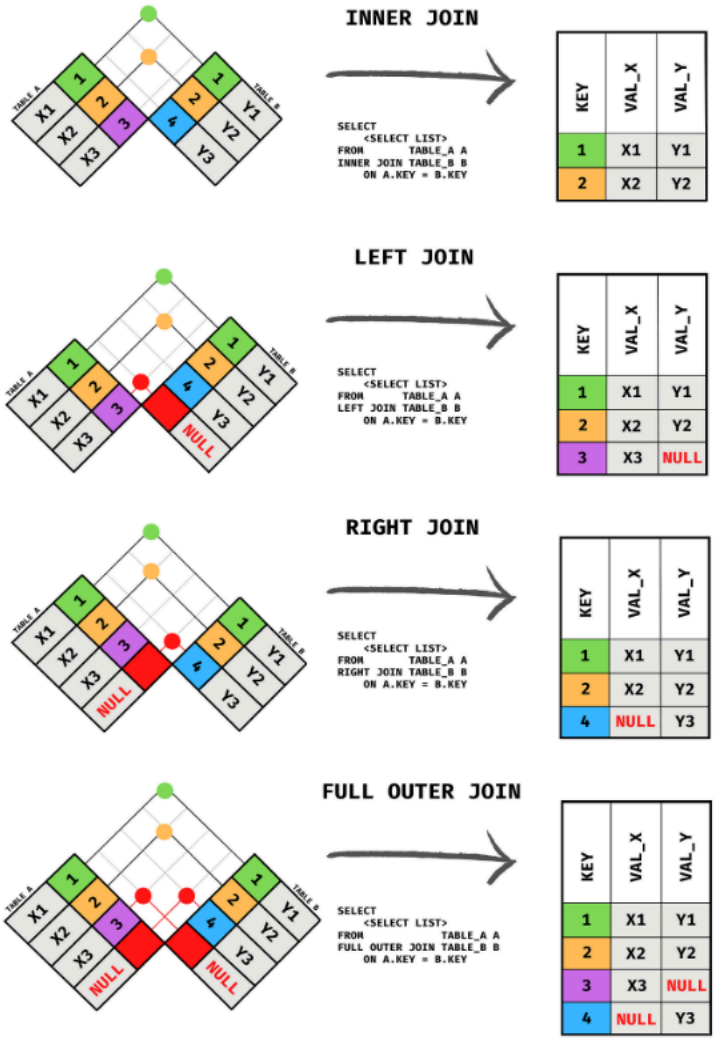
Condition: Use a LEFT JOIN when you want to retrieve all rows from the left table (employees) and the matching rows from the right table (departments). If an employee is not assigned to any department, the result will still include the employee information with NULL values for department details.

3. RIGHT JOIN:

Condition: Use a RIGHT JOIN when you want to retrieve all rows from the right table (departments) and the matching rows from the left table (employees). If there is a department with no assigned employees, the result will still include the department information with NULL values for employee details.

4. FULL JOIN:

Condition: Use a FULL JOIN when you want to retrieve all rows from both tables, whether there is a match or not. If an employee has no assigned department or a department has no assigned employees, the result will include both employee and department information with NULL values where there is no match.



5. CROSS JOIN:

Condition: Use a CROSS JOIN when you want to generate all possible combinations of rows from both tables. It's typically used sparingly, as it can result in a large number of rows in the output.

What is an Index? Explain its different types.

A database index is a data structure that provides a quick lookup of data in a column or columns of a table. It enhances the speed of operations accessing data from a database table at the cost of additional writes and memory to maintain the index data structure.

```
CREATE INDEX index_name /* Create Index */
ON table_name (column_1, column_2);
DROP INDEX index_name; /* Drop Index */
```

There are different types of indexes that can be created for different purposes:

- Unique and Non-Unique Index:

Unique indexes are indexes that help maintain data integrity by ensuring that no two rows of data in a table have identical key values. Once a unique index has been defined for a table, uniqueness is enforced whenever keys are added or changed within the index.

```
CREATE UNIQUE INDEX myIndex
ON students (enroll_no);
```

Non-unique indexes, on the other hand, are not used to enforce constraints on the tables with which they are associated. Instead, non-unique indexes are used solely to improve query performance by maintaining a sorted order of data values that are used frequently.

What is a Subquery? What are its types?

A subquery is a query within another query, also known as a nested query or inner query. It is used to restrict or enhance the data to be queried by the main query, thus restricting or enhancing the output of the main query respectively. For example, here we fetch the contact information for students who have enrolled for the maths subject:

```
SELECT name, email, mob, address FROM myDb.contacts WHERE roll_no IN ( SELECT roll_no FROM myDb.students WHERE subject = 'Maths');
```

What is a View?

A view in SQL is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

Why use views instead of tables?

Views provide a way to present a customized or filtered perspective of the data in a table, without creating a new

physical copy. They help simplify complex queries, enhance security by restricting access to specific columns, and make it easier to manage changes in the underlying structure.

What is a Window Function?

A window function lets you perform calculations on a specific group of rows in a table, rather than the entire table, providing context and insights by focusing on a subset of the data.

WHY WINDOW FUNCTION?

Window functions are useful because they allow you to perform calculations in a way that takes into account the context of each individual row within a dataset. Instead of treating each row in isolation, window functions consider a specified range or grouping of rows around each data point.