SQL Theory

1. Delete Truncate and Drop

Delete	Truncate	Drop
DML	DDL	DDL
Rollback	Can't Rollback	Can't Rollback
Use by Where condition	Shema will be there	Shema Deleted

2. Join query -

```
Select  Name
from Emplyoyee
INNER JOIN Student
On emp.Id = std.Id
```

3. Top 3 Salary Employee -

```
Select Top 1 Salary from
(Select DISTINCT top 3 salary,name from Employee ORDER BY DESC)
a OREDR BY Salary ASC
```

4. Indexing ()
   Indexing@Medium

   Insertion and deletion should be faster
   Types -
   > Single-Column Indexes:
   > Composite Indexes:
   > Unique Indexes:
   > Full-Text Indexes:

5. Indexes -

   Cluster Index -
   Primary

   Non-Cluster Index -
   Unique Key

6. Normalization

   "Normalization is a process in database design that organizes data into logical and efficient structures"

   for more

SQL Theory

```
Create procedure p1
as
begin
sql Statement
```

1. Self Join :

Self-join allows us to join a table itself. It is useful when a user wants to compare the data (rows) within the same table.

Syntax –

```
select select_list
from T t1 [Inner|Left] Join
on T t2
on join_predicate.
```

2. Cross Join :

Cross join allows us to join each and every row of both the tables. It is similar to the cartesian product that joins all the rows.

Syntax –

```
select
select_list
from T1 cross join T2
```

>> can we use group by and having clause together
Yes
SELECT column1, COUNT(*)
FROM table_name
GROUP BY column1
HAVING COUNT(*) > 1;

>> What is Temporary Table in SQL
>> It is used for the particular execution of session Transaction then it will

Double ## - Global Temporary Table
Visible to all sessions and connections. Once created, any session can access it.

SQL Theory

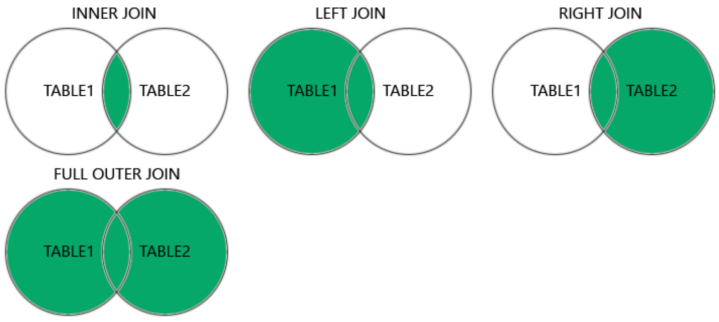
In SQL , Select 2nd Highest salary from each department Department1 and Department from Employee Table

```
        WITH RankedSalaries AS (
    SELECT
        Department,
        Salary,
        ROW_NUMBER() OVER (PARTITION BY Department ORDER BY Salary DESC) AS RowNum
    FROM Employee
)
SELECT
    Department,
    Salary AS SecondHighestSalary
FROM RankedSalaries
WHERE RowNum = 2;
```

\\\\\\\\

Joins

1. **Inner Join**
2. **Left (Outer) Join**
3. **Right (Outer) Join**
4. **Full (Outer Join)**



(INNER) JOIN: Returns records that have matching values in both tables

LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table

RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table

FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

Important
Note: The LEFT JOIN keyword returns all records from the left table (Customers), even if there are no matches in the right table (Orders).

SQL Theory

Note: The RIGHT JOIN keyword returns all records from the right table (Employees), even if there are no matches in the left table (Orders).

Note: The FULL OUTER JOIN keyword returns all matching records from both tables whether the other table matches or not. So, if there are rows in "Customers" that do not have matches in "Orders", or if there are rows in "Orders" that do not have matches in "Customers", those rows will be listed as well.

A **typed table** in SQL, also known as **a structured or strongly-typed table**, is a table that is explicitly defined by a user-defined type (UDT) or structured type. This type enforces a specific structure on the table, providing a way to ensure that all rows in the table **adhere to a defined schema.**

The MERGE statement in SQL is a powerful and flexible command that allows you to perform **insert, update, and delete operations in a single statement based on the results of a join** with another table or data source. It is particularly useful for synchronizing two tables, performing upserts (update if exists, insert if not), and ensuring that data remains consistent across different tables.
