Pre-Joining Topics

Week 1: MYSQL JOINS

JOINs

MySQL JOIN is a fundamental feature that combines rows from two or more tables based on a related column between them. It allows for efficient **data retrieval** by enabling the extraction of related information from multiple tables in a single query.

MySQL JOIN

- A **MySQL JOIN** is a method to combine data from two or more tables in a **database** based on a related column between them.
- JOINs allow us to retrieve related data from multiple tables in a single query, avoiding the need for numerous separate queries.
- JOINs help maintain **referential integrity** by ensuring that relationships between tables are respected and **data consistency** is preserved.
- By allowing the use of normalized database structures (where data is split into different tables to reduce redundancy), JOINs help in efficiently **managing** and **organizing** data.
- JOINs are often used in conjunction with GROUP BY and aggregation functions (like **COUNT, SUM, AVG**) to perform comprehensive data analysis.

Syntax:

```
SELECT column_names
FROM table1
INNER JOIN table2
ON table1.common_column = table2.common_column;
```

where,

- **SELECT column_names**: This identifies the columns you want to get from the tables. One can select columns in both table1 and table2.
- **FROM table1**: This specifies the first table to begin joining from.
- **INNER JOIN table2:** Identifies we wan to join table1 and table2
- **ON table1.common_column = table2.common_column:** This specifies the condition on which the join is based

employee_id	name	department_id
1	Alice	1
2	Bob	2
3	Charlie	1
4	David	3
5	Eve	NULL

department_id	department_name
1	HR
2	Engineering
3	Marketing
4	Finance

INNER JOIN

- It Returns records that have matching values in both tables.
- ```
  SELECT employees.name, departments.department_name
  FROM employees
  INNER JOIN departments
  ON employees.department_id = departments.department_id;
  ```

Explanation:

- The INNER JOIN query that retrieves employee names and their corresponding department names
  is: SELECT **employees.name**, **departments.department_name FROM** employees **INNER JOIN** departments ON employees.department_id = **departments.department_id**;
- This joins two tables, '**employees**' and '**departments**', using the 'department_id' column to return only those records that have matching values in both tables.
- It gives a list of employees with the corresponding department names.

name	department_name
Alice	HR
Bob	Engineering
Charlie	HR
David	Marketing
  ```

LEFT JOIN

It returns all records from the Left table and matched records from the Right table. If there is no match, then NULL values are returned for Right table columns.

```
SELECT employees.name, departments.department_name
FROM employees
LEFT JOIN departments
ON employees.department_id = departments.department_id;
```

RIGHT JOIN

It Returns all the rows from the right table and the matched rows from the left table. NULL values will be returned for columns from the left table when there are no matches.

```
SELECT employees.name, departments.department_name
FROM employees
RIGHT JOIN departments
ON employees.department_id = departments.department_id;
```

FULL JOIN

It Returns all records when there is a match in either the left or the right table. In case of no match, NULL values are returned for columns that have no match in either table.

```
SELECT employees.name, departments.department_name
FROM employees
LEFT JOIN departments
ON employees.department_id = departments.department_id
UNION
SELECT employees.name, departments.department_name
FROM employees
RIGHT JOIN departments
ON employees.department_id = departments.department_id;
```