nextleap

# SQL 3 Data Analyst Fellowship


## Saksham Arora

**Software Engineer** Microsoft

Saksham Arora - Microsoft | LinkedIn

**1. Understanding Relationships Between Tables**

- **Topics:**
  - One-to-Many One-to-One, Many-to-Many Relationships

**2. Introduction to Joins**

- **Topics:**
  - Understanding the Concept of Joins in SQL
  - Basics of Inner Joins

**3. Exploring Joins**

- **Topics:**
  - Left Joins
  - Right Joins
  - Full Outer Joins

**4. Advanced Joins Techniques**

- **Topics:**
  - Cross Joins
  - Self Joins
  - Joins on Joins
  - Multiple Inner Joins
  - Order of Joins

# Relationships in Tables

Relationships between tables define how data in one table is related to data in another. The three most common types of relationships are **One-to-One**, **One-to-Many**, and **Many-to-Many**. These relationships are implemented using **primary keys** and **foreign keys.**

## 1. One-to-One Relationship

A **One-to-One** relationship means that for each row in one table, there is a maximum of one related row in another table, and vice versa. This type of relationship is less common but is used when data can be split across multiple tables for optimization or separation of concerns.

Example - Person table with their Aadhar

## 2. One-to-Many Relationship

A **One-to-Many** relationship is the most common type of relationship in databases. In this relationship, a row in one table can be related to multiple rows in another table, but a row in the second table is related to only one row in the first table.

Example - Each Student will be linked to Single class in a class table.

## 3. Many-to-Many Relationship

A **Many-to-Many** relationship occurs when multiple rows in one table can be related to multiple rows in another table. This relationship is typically implemented using a **junction table** (also called a **bridge** or **join table**) to break it into two one-to-many relationships.

Example - Multiple students can be enrolled into multiple courses.

# Why are we JOINING the tables ?

1. We need to refer to other tables to get meaningful outcomes.
2. Foreign keys are used to reference other tables.

**1. branch** 分公司

| branch-name | branch-city | assets |
|---|---|---|
| Brighton | Brooklyn | 7100000 |
| Downtown | Brooklyn | 9000000 |
| Mianus | Horseneck | 400000 |
| North Town | Rye | 3700000 |
| Perryridge | Horseneck | 1700000 |
| Pownal | Bennington | 300000 |
| Redwood | Palo Alto | 2100000 |
| Round Hill | Horseneck | 8000000 |

**2. customer** 客戶(存款戶,貸款戶)

| customer-name | customer-street | customer-city |
|---|---|---|
| Adams | Spring | Pittsfield |
| Brooks | Senator | Brooklyn |
| Curry | North | Rye |
| Glenn | Sand Hill | Woodside |
| Green | Walnut | Stamford |
| Hayes | Main | Harrison |
| Johnson | Alma | Palo Alto |
| Jones | Main | Harrison |
| Lindsay | Park | Pittsfield |
| Smith | North | Rye |
| Turner | Putnam | Stamford |
| Williams | Nassau | Princeton |

**3. depositor** 存款戶

| customer-name | account-number |
|---|---|
| Hayes | A-102 |
| Johnson | A-101 |
| Johnson | A-201 |
| Jones | A-217 |
| Lindsay | A-222 |
| Smith | A-215 |
| Turner | A-305 |

**4. borrower** 貸款戶

| customer-name | loan-number |
|---|---|
| Adams | L-16 |
| Curry | L-93 |
| Hayes | L-15 |
| Jackson | L-14 |
| Jones | L-17 |
| Smith | L-11 |
| Smith | L-23 |
| Williams | L-17 |

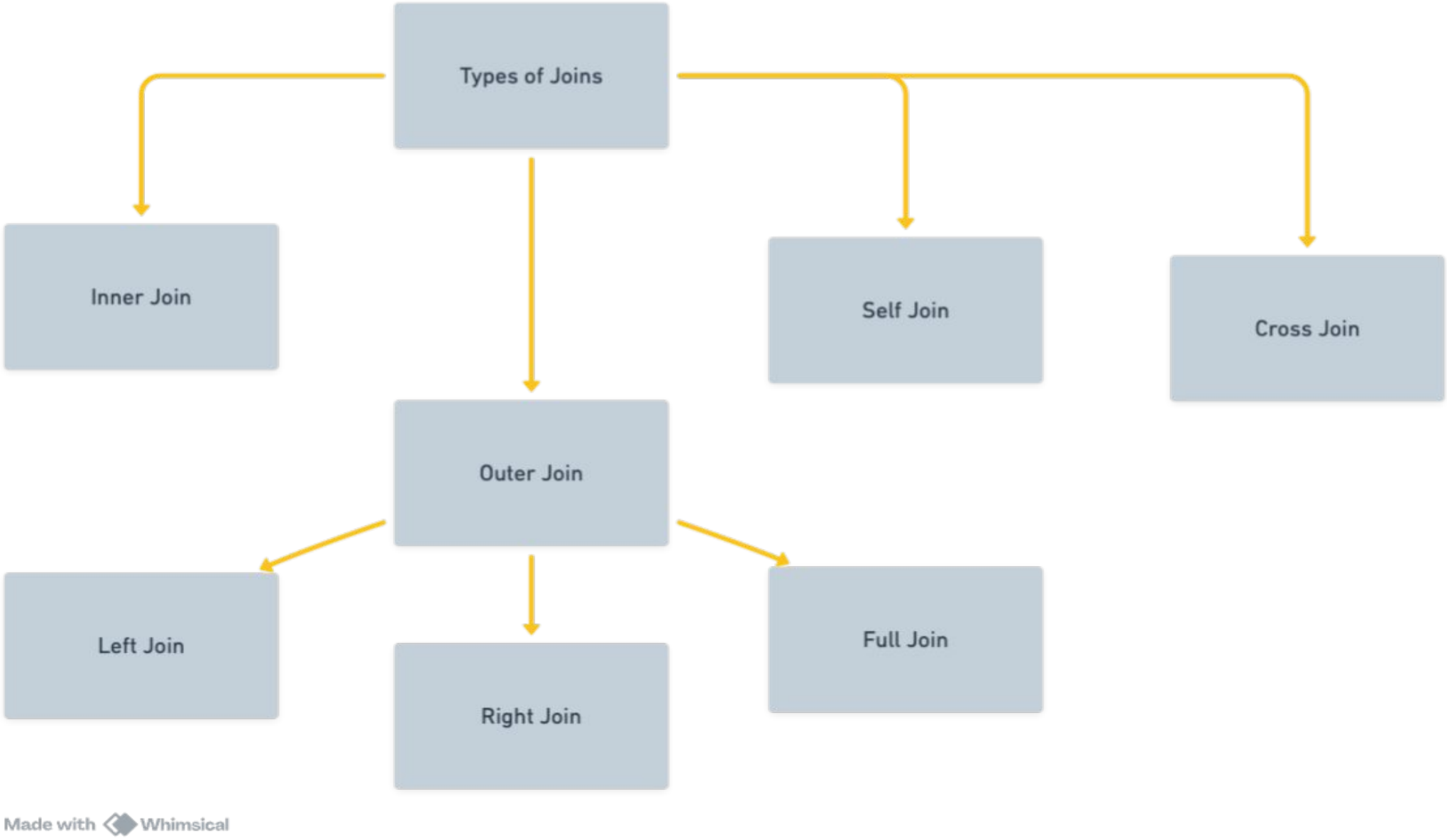
**5. account** 存款帳

| account-number | branch-name | balance |
|---|---|---|
| A-101 | Downtown | 500 |
| A-102 | Perryridge | 400 |
| A-201 | Brighton | 900 |
| A-215 | Mianus | 700 |
| A-217 | Brighton | 750 |
| A-222 | Redwood | 700 |
| A-305 | Round Hill | 350 |

**6. loan** 貸款帳

| loan-number | branch-name | amount |
|---|---|---|
| L-11 | Round Hill | 900 |
| L-14 | Downtown | 1500 |
| L-15 | Perryridge | 1500 |
| L-16 | Perryridge | 1300 |
| L-17 | Downtown | 1000 |
| L-23 | Redwood | 2000 |
| L-93 | Mianus | 500 |

# Types of JOINS



# Inner Joins

1. Returns a resultant table with matching values from the tables or all the tables.

2. SELECT column-list FROM table1 INNER JOIN table2 ON condition1 INNER JOIN table3 ON condition2

3. Alias in MySQL (AS)
    1. Aliases in MySQL are used to give a temporary name to a table or a column in a table for a particular query. It works as a nickname for expressing the tables or column names. It makes the query short and neat.
    2. SELECT col_name AS alias_name FROM table_name;

4. **"Inner join"** is a specific type of join that returns only matching rows from both tables. **"Join"** is a broader term that could refer to any type of join, but when used alone, it usually defaults to an inner join.

# Inner Join EXAMPLE

Let's look at a selection of the **Products** table:

| ProductID | ProductName | CategoryID | Price |
|---|---|---|---|
| 1 | Chais | 1 | 18 |
| 2 | Chang | 1 | 19 |
| 3 | Aniseed Syrup | 2 | 10 |

And a selection of the **Categories** table:

| CategoryID | CategoryName | Description |
|---|---|---|
| 1 | Beverages | Soft drinks, coffees, teas, beers, and ales |
| 2 | Condiments | Sweet and savory sauces, relishes, spreads, and seasonings |
| 3 | Confections | Desserts, candies, and sweet breads |

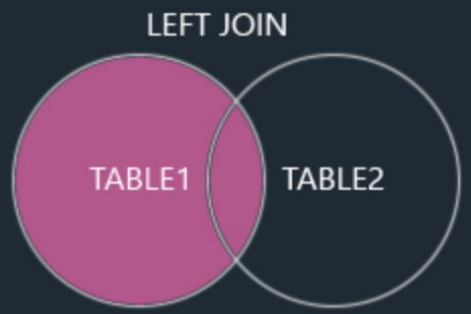
# OUTER JOINS

## LEFT JOIN

1. This returns a resulting table that contains all the data from the left table and the matched data from the right table.
2. SELECT columns FROM table LEFT JOIN table2 ON Join_Condition;

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

**Note:** In some databases LEFT JOIN is called LEFT OUTER JOIN.

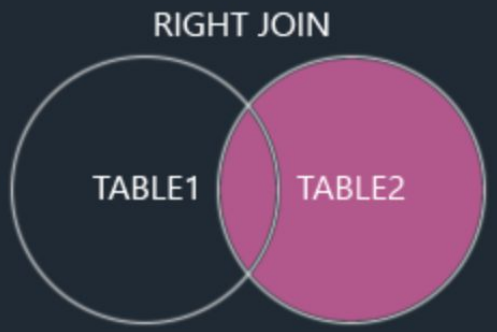


# RIGHT JOIN

1. This returns a resulting table that contains all the data from the right table and the matched data from the left table.
2. SELECT columns FROM table RIGHT JOIN table2 ON join_cond;

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name;
```

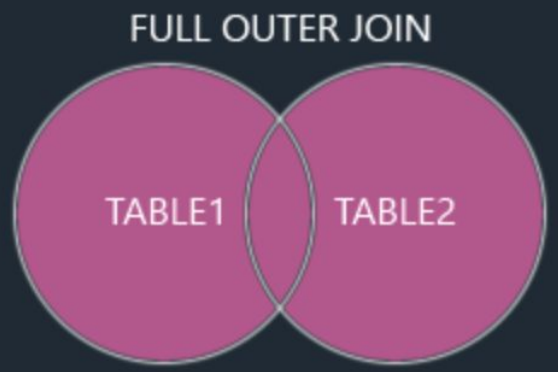
**Note:** In some databases `RIGHT JOIN` is called `RIGHT OUTER JOIN`.



# FULL JOIN

1. This returns a resulting table that contains all data when there is a match on left or right table data.

```
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name = table2.column_name
WHERE condition;
```




**Note:** `FULL OUTER JOIN` can potentially return very large result-sets!

# SELF JOIN

1. It is used to get the output from a particular table when the same table is joined.

Self Join Syntax

```
SELECT column_name(s)
FROM table1 T1, table1 T2
WHERE condition;
```

## Operators in SQL

### SET Operators

1. UNION
   - UNION will be used to combine the result of two select statements.
   - Duplicate rows will be eliminated from the results obtained after performing the UNION operation.
2. UNION ALL
   - This operator combines all the records from both the queries.
   - Duplicate rows will be not be eliminated from the results obtained after performing the UNION ALL operation.
3. INTERSECT
   - It is used to combine two SELECT statements, but it only returns the records which are common from both SELECT statements.
4. EXCEPT
   - It displays the rows which are present in the first query but absent in the second query with no duplicates.
   -

## Problem Statements

1. NextLeap | Bonus Report
2. NextLeap | Product Sale Details
3. High Earning Employees Relative to Their Managers
4. Products Ordered in a Month

Other questions to try - https://nextleap.app/interview-preparation/sql/questions

# Q&A

What’s on your mind?


SKELLYSCOO
I have several questions.

# Feed us back!