Set Operators

Set operators combine the result sets of two or more `SELECT` statements. The queries involved must have the same number of columns, and corresponding columns must have compatible data types.

- `UNION`: Combines the result sets and removes duplicate rows.

```
SELECT column1, column2 FROM table1
UNION
SELECT column1, column2 FROM table2;
```

- `UNION ALL`: Combines the result sets, including all duplicate rows.

```
SELECT column1, column2 FROM table1
UNION ALL
SELECT column1, column2 FROM table2;
```

- `INTERSECT`: Returns only the distinct rows that appear in both result sets.

```
SELECT column1, column2 FROM table1
INTERSECT
SELECT column1, column2 FROM table2;
```

- `MINUS` (or `EXCEPT` in some SQL dialects): Returns the distinct rows from the first query that are not present in the second query's result set.

```
SELECT column1, column2 FROM table1
MINUS
SELECT column1, column2 FROM table2;
```

Nested Queries (Subqueries)

A nested query, or subquery, is a `SELECT` statement embedded within another SQL statement (e.g., `SELECT`, `INSERT`, `UPDATE`, `DELETE`). The inner query executes first, and its result is used by the outer query.

- **Example**: Finding employees who work in a specific department.

```
SELECT employee_name
FROM employees
WHERE department_id IN (SELECT department_id FROM departments WHERE
department_name = 'Sales');
```

Join Queries

Join queries combine rows from two or more tables based on related columns between them.

- `INNER JOIN`: Returns only the rows where there is a match in both tables based on the join condition.

```
SELECT orders.order_id, customers.customer_name
FROM orders
INNER JOIN customers ON orders.customer_id = customers.customer_id;
```

- `LEFT JOIN` (or `LEFT OUTER JOIN`): Returns all rows from the left table, and the matching rows from the right table. If no match is found in the right table, `NULL` values are returned for right table columns.

```
SELECT products.product_name, order_items.quantity
FROM products
LEFT JOIN order_items ON products.product_id = order_items.product_id;
```

- `RIGHT JOIN` (or `RIGHT OUTER JOIN`): Returns all rows from the right table, and the matching rows from the left table. If no match is found in the left table, `NULL` values are returned for left table columns.

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

- `FULL JOIN` (or `FULL OUTER JOIN`): Returns all rows when there is a match in one of the tables. If no match is found, `NULL` values are returned for the columns of the non-matching table.

```
SELECT customers.customer_name, orders.order_id
FROM customers
FULL JOIN orders ON customers.customer_id = orders.customer_id;
```