**Set Operations**

are used to combine(manipulate) the results of multiple queries.

Types of Set Operators are below

1. **UNION:**

* The UNION operator combines the results of two or more SELECT statements into a single result set.
* Select The columns in the result set are the union of the columns in the SELECT statements.
* Duplicate rows are removed unless the UNION ALL operator is used instead.

Syntax:

SELECT column1, column2, ... FROM table1

UNION

SELECT column1, column2, ... FROM table2

***Examples 1 :*** *Combine Employe + Manager*

CREATE TABLE employees (

name VARCHAR(50),

department VARCHAR(50),

salary DECIMAL(10, 2)

);

INSERT INTO employees (name, department, salary)

VALUES

('John Doe', 'Sales', 50000.00),

('Jane Smith', 'Marketing', 60000.00),

('Bob Johnson', 'IT', 70000.00),

('Mike Williams', 'HR', 55000.00);

CREATE TABLE managers (

name VARCHAR(50),

department VARCHAR(50),

salary DECIMAL(10, 2)

);

INSERT INTO managers (name, department, salary)

VALUES

('Sara Lee', 'Sales', 90000.00),

('Adam Smith', 'Marketing', 80000.00),

('David Lee', 'IT', 95000.00);

SELECT name, department, salary

FROM employees

UNION

SELECT name, department, salary

FROM managers;

1. **Intersect:**

🡪 The INTERSECT operator returns the common rows between two or more SELECT statements.

* The columns in the result set are the intersection of the columns in the SELECT statements.

Syntax:

SELECT column1, column2, ... FROM table1

INTERSECT

SELECT column1, column2, ... FROM table2

***Examples 1 :*** *Similar Record (Employe + SalesPeopel)*

CREATE TABLE employees (

name VARCHAR(50),

department VARCHAR(50)

);

INSERT INTO employees (name, department)

VALUES

('John Doe', 'Sales'),

('Jane Smith', 'Marketing'),

('Bob Johnson', 'IT'),

('Mike Williams', 'HR');

CREATE TABLE salespeople (

name VARCHAR(50),

department VARCHAR(50)

);

INSERT INTO salespeople (name, department)

VALUES

('John Doe', 'Sales'),

('Sara Lee', 'Sales'),

('Mark Jones', 'Sales');

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Intersect Query \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SELECT name, department

FROM employees

INTERSECT

SELECT name, department

FROM salespeople;

1. **Except:**

* Select Different Records

Syntax:

SELECT column1, column2, ... FROM table1

EXCEPT

SELECT column1, column2, ... FROM table2

***Examples 1 :***

CREATE TABLE employees (

name VARCHAR(50),

department VARCHAR(50),

salary DECIMAL(10, 2)

);

INSERT INTO employees (name, department, salary)

VALUES

('John Doe', 'Sales', 50000.00),

('Jane Smith', 'Marketing', 60000.00),

('Bob Johnson', 'IT', 70000.00),

('Mike Williams', 'HR', 55000.00);

CREATE TABLE managers (

name VARCHAR(50),

department VARCHAR(50),

salary DECIMAL(10, 2)

);

INSERT INTO managers (name, department, salary)

VALUES

('Sara Lee', 'Sales', 90000.00),

('Adam Smith', 'Marketing', 80000.00),

('David Lee', 'IT', 95000.00),

('Jane Smith', 'Marketing', 70000.00);

---\_\_\_\_\_\_\_\_\_\_\_\_ Except \_\_\_\_\_\_\_\_\_\_\_\_

SELECT name, department

FROM employees

EXCEPT

SELECT name, department

FROM managers;

1. **Union All :**

* Select All Similar , Non-Similar , Different Record

Syntax:

SELECT column1, column2, ... FROM table1

UNION ALL

SELECT column1, column2, ... FROM table2

***Examples 1 :***

CREATE TABLE employees (

name VARCHAR(50),

department VARCHAR(50)

);

INSERT INTO employees

VALUES

('John Doe', 'Sales'),

('Jane Smith', 'Marketing'),

('Bob Johnson', 'IT'),

('Mike Williams', 'HR');

CREATE TABLE contractors (

name VARCHAR(50),

department VARCHAR(50)

);

INSERT INTO contractors (name, department)

VALUES

('Sarah Lee', 'Sales'),

('Mark Jones', 'IT'),

('Julia Roberts', 'Marketing');

select \* from employees

---\_\_\_\_\_Will Give Seperate Recod\_\_\_\_\_\_\_

SELECT name,department

FROM employees

SELECT name, department

FROM contractors;

--\_\_\_\_\_\_ Join Records set by Union All SetOperation \_\_\_\_\_\_\_\_\_

SELECT name, department

FROM employees

UNION ALL

SELECT name, department

FROM contractors;

1. **CROSS Apply:**

* The Cross Match The 2 table Record with Each other Example
* 1st Function Created
* 2nd apply cross Join

Syntax:

SELECT column1, column2, ... FROM table1

CROSS APPLY

function\_name(column1, column2, ...)

***Examples 1 :***

CREATE TABLE employees (

id INT,

name VARCHAR(50),

department\_id INT

);

INSERT INTO employees (id, name, department\_id)

VALUES

(1, 'John Doe', 1),

(2, 'Jane Smith', 2),

(3, 'Bob Johnson', 1),

(4, 'Mike Williams', 3),

(5, 'Sara Lee', 1);

CREATE TABLE departments (

id INT,

name VARCHAR(50)

);

INSERT INTO departments (id, name)

VALUES

(1, 'Sales'),

(2, 'Marketing'),

(3, 'IT');

CREATE FUNCTION getEmployeesInDepartment(@deptName VARCHAR(50))

RETURNS @employees TABLE (name VARCHAR(50))

AS

BEGIN

-- Add ho gy gaa @emplyee Variable me after --> select Return

INSERT INTO @employees (name)

-- select name and Return to Insert

SELECT employees.name

FROM employees

JOIN departments ON employees.department\_id = departments.id

WHERE departments.name = @deptName

RETURN

END;

--\_\_\_ Execute Table Function \_\_\_

select name from getEmployeesInDepartment('fish')

select \* from employees

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Apply Cross Apply \_\_\_\_\_\_\_\_\_\_

SELECT departments.name, employees.name

FROM departments

CROSS APPLY getEmployeesInDepartment(departments.name) AS employees;

1. **Outer Apply:**

* The Outer Give matching or Non-matching Record from both Tables

Syntax:

SELECT column1, column2, ... FROM table1

OUTER APPLY

function\_name(column1, column2, ...)

***Examples 1 :***

CREATE TABLE orders (

order\_id INT,

customer\_id INT,

order\_date DATE,

amount DECIMAL(10, 2)

);

INSERT INTO orders (order\_id, customer\_id, order\_date, amount)

VALUES

(1, 1, '2022-02-01', 100.00),

(2, 1, '2022-02-05', 200.00),

(3, 2, '2022-02-08', 150.00),

(4, 3, '2022-02-10', 75.00);

CREATE TABLE customers (

customer\_id INT,

customer\_name VARCHAR(50),

customer\_email VARCHAR(50)

);

INSERT INTO customers (customer\_id, customer\_name, customer\_email)

VALUES

(1, 'John Doe', 'johndoe@example.com'),

(2, 'Jane Smith', 'janesmith@example.com');

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Outerf Apply \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SELECT o.order\_id, c.customer\_name, c.customer\_email, o.amount

FROM orders o

OUTER APPLY (

SELECT customer\_name, customer\_email

FROM customers c

WHERE o.customer\_id = c.customer\_id

) AS c;