**UNION ALL en SQL – Teoría y Uso**

**🔹 ¿Qué es UNION ALL?**

UNION ALL es una cláusula en SQL que se usa para combinar los resultados de dos o más consultas SELECT, **incluyendo los duplicados**.

**🔹 Diferencia entre UNION y UNION ALL**

| **Cláusula** | **¿Elimina duplicados?** | **Rendimiento** |
| --- | --- | --- |
| UNION | Sí (realiza una operación de DISTINCT) | Más lento, porque necesita filtrar duplicados. |
| UNION ALL | No (incluye duplicados) | Más rápido, porque solo une los resultados sin filtrarlos. |

**🔹 Sintaxis**

SELECT columna1, columna2 FROM tabla1

UNION ALL

SELECT columna1, columna2 FROM tabla2;

**Ejemplos Prácticos**

**1️⃣ UNION ALL con Datos Simples**

SELECT 'Pedro' AS nombre, 'Ventas' AS departamento

UNION ALL

SELECT 'Ana', 'Marketing'

UNION ALL

SELECT 'Pedro', 'Ventas';

🔹 **Salida:**

| Nombre | Departamento |

|--------|-------------|

| Pedro | Ventas |

| Ana | Marketing |

| Pedro | Ventas | <-- Se mantiene el duplicado

**🔸 Explicación:** Aquí Pedro aparece **dos veces** porque UNION ALL no elimina duplicados.

**2️⃣ UNION ALL con Tablas Reales**

SELECT ename, sal, job FROM emp WHERE job = 'MANAGER'

UNION ALL

SELECT ename, sal, job FROM emp WHERE job = 'CLERK';

🔹 **Explicación:** Une en una sola lista los empleados con cargo de MANAGER y CLERK.

Si algún empleado aparece en ambas consultas, también aparecerá **dos veces** en el resultado.

**3️⃣ UNION ALL con Columnas Diferentes**

Cuando las columnas de ambas consultas no coinciden en nombre, pero sí en tipo de datos, puedes usar alias:

SELECT ename AS empleado, sal AS sueldo FROM emp

UNION ALL

SELECT dname AS empleado, NULL AS sueldo FROM dept;

🔹 **Explicación:**

* ENAME de EMP y DNAME de DEPT se combinan en una columna llamada **empleado**.
* Como DNAME no tiene salario, usamos NULL en la segunda consulta para que coincida con la estructura.

**💡 Reglas Importantes de UNION ALL**

1️⃣ **Las consultas deben tener el mismo número de columnas.**

* ❌ Incorrecto:

SELECT ename FROM emp

UNION ALL

SELECT ename, sal FROM emp; -- ERROR: una consulta tiene 1 columna y la otra 2.

2️⃣ **Los tipos de datos de las columnas deben ser compatibles.**

* ❌ Incorrecto:

SELECT ename FROM emp

UNION ALL

SELECT hiredate FROM emp; -- ERROR: un campo es VARCHAR y el otro DATE.

3️⃣ **El orden final se hace con ORDER BY fuera del UNION ALL.**

* ✔ Correcto:

SELECT ename, sal FROM emp WHERE deptno = 10

UNION ALL

SELECT ename, sal FROM emp WHERE deptno = 20

ORDER BY sal DESC;

🔹 **Explicación:** Se ordena después de unir los resultados.

**📌 Cuándo Usar UNION ALL en un Ejercicio**

🔹 **Cuando quieres combinar consultas sin preocuparte por duplicados.**  
🔹 **Cuando necesitas mejor rendimiento y no es necesario eliminar valores repetidos.**  
🔹 **Cuando los datos provienen de diferentes fuentes o tablas y necesitas unirlos.**

Si necesitas ejercicios con UNION ALL, dime qué tipo de problema quieres resolver. 😊

**🔹 UNION en SQL y PL/SQL**

UNION es una operación en SQL que permite combinar los resultados de dos o más consultas SELECT en un solo conjunto de resultados. En PL/SQL, se usa dentro de consultas SQL dentro de bloques anónimos, procedimientos o funciones.

**🔹 Teoría**

**📌 Características de UNION**

* **Elimina duplicados**: Retorna solo los registros únicos de ambas consultas.
* **Las consultas deben tener el mismo número de columnas**.
* **Los tipos de datos de las columnas deben ser compatibles**.
* **Orden de las columnas debe coincidir en ambas consultas**.
* **Se pueden usar ORDER BY, pero solo en la última consulta**.

**🔹 Sintaxis de UNION**

SELECT columna1, columna2 FROM tabla1

UNION

SELECT columna1, columna2 FROM tabla2;

🔹 **Ejemplo**:

SELECT nombre, edad FROM clientes

UNION

SELECT nombre, edad FROM empleados;

Esto devuelve una lista única de nombres y edades de ambas tablas.

**🔹 UNION ALL**

Si necesitas incluir los duplicados, puedes usar UNION ALL, que es más rápido porque no hace la eliminación de duplicados.

SELECT columna1, columna2 FROM tabla1

UNION ALL

SELECT columna1, columna2 FROM tabla2;

🔹 **Ejemplo**:

SELECT nombre, edad FROM clientes

UNION ALL

SELECT nombre, edad FROM empleados;

Esto incluirá los registros repetidos en el resultado.

**🔹 UNION en PL/SQL**

En PL/SQL, UNION se usa en consultas dentro de procedimientos, funciones o bloques anónimos.

🔹 **Ejemplo en PL/SQL**:

DECLARE

CURSOR c\_union IS

(

SELECT nombre, edad FROM clientes

UNION

SELECT nombre, edad FROM empleados

);

v\_nombre clientes.nombre%TYPE;

v\_edad clientes.edad%TYPE;

BEGIN

OPEN c\_union;

LOOP

FETCH c\_union INTO v\_nombre, v\_edad;

EXIT WHEN c\_union%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Nombre: ' || v\_nombre || ', Edad: ' || v\_edad);

END LOOP;

CLOSE c\_union;

END;

/

Esto almacena los resultados de UNION en un cursor y los muestra con DBMS\_OUTPUT.PUT\_LINE.

**🔹 Diferencias clave entre UNION y UNION ALL**

| **Característica** | **UNION** | **UNION ALL** |
| --- | --- | --- |
| Elimina duplicados | ✅ Sí | ❌ No |
| Rendimiento | Más lento (porque elimina duplicados) | Más rápido |
| Uso de ORDER BY | Solo en la última consulta | Solo en la última consulta |

¿Necesitas ejemplos con condiciones (WHERE, ORDER BY), joins o en procedimientos almacenados? 😊