Asignatura: Bases de Datos II
Profesor: D. Sc. Gerardo García Gil
2022-A
Alumno: José Rafael Ruiz Gudiño
Ingeniería en Desarrollo de Software
Centro de Enseñanza Técnica Industrial (CETI)

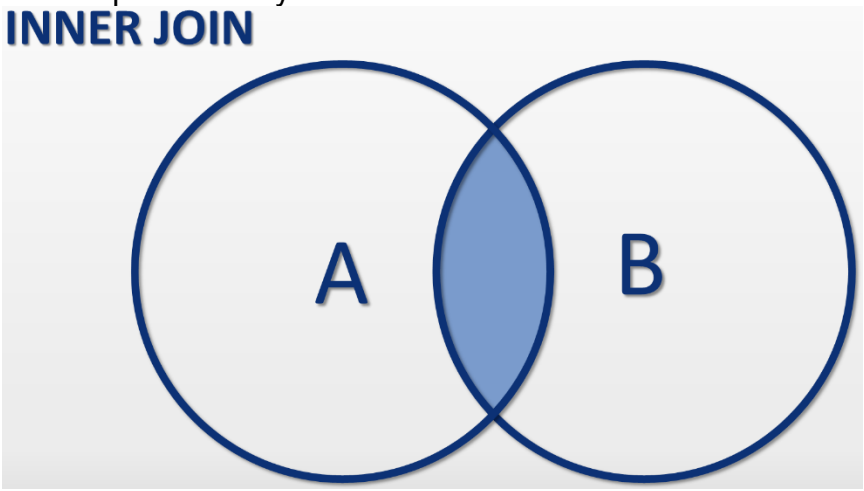
-Introducción

En el modelo de bases de datos relacionales, se recurre a las sentencias JOIN de SQL para consultar diferentes tablas de bases de datos. A excepción de CROSS JOIN, las sentencias de JOIN son una combinación de producto cartesiano y selección.
El Sistema Gestor de Bases de Datos (SGBD) primero presenta el producto cartesiano de dos tablas de bases de datos. A continuación, filtra el resultado según una condición de selección definida por el usuario a través de una sentencia SQL.

-Desarrollo

-Inner Join

El INNER JOIN se diferencia de todas las demás sentencias de JOIN por mostrar un conjunto de resultados mínimos, pues solo se muestran los registros de datos del producto cruzado que cumplen la condición de selección. Todo ello se presenta en una tabla de resultados llamada "View" que no incluye valores nulos.



-Implementación
La tabla "Empleados" contiene todos los empleados de una empresa, junto a sus números de identificación (e_id) y el número de departamento al que pertenecen (d_id).

e_id	Apellidos	Nombre	d_id
1	García Hurtado	Macarena	3
2	Ocaña Martínez	Francisco	1
3	Gutiérrez Doblado	Elena	1
4	Hernández Soria	Manuela	2
5	Oliva Cansino	Andrea	NULL

La tabla "Departamentos" enumera todos los departamentos de la empresa, incluyendo el número identificador de cada departamento y su ubicación

d_id	Denominación	Localización
1	Ventas	Sevilla
2	IT	Málaga
3	Recursos Humanos	Marbella
4	Investigación	Málaga

Ambas tablas están enlazadas por una relación de clave externa. El ID de departamento, que actúa como clave primaria en la tabla “Departamentos”, se ha integrado como clave externa en la tabla “Empleados”.

Cuando se consultan bases de datos relacionales, se suele definir como condición de selección la correspondencia entre una clave primaria y una externa. La condición se considera cumplida si la clave externa seleccionada de una tabla coincide con la clave primaria de la otra tabla (=), es decir, solo se emiten aquellos registros de datos que contienen valores en común.

Un INNER JOIN que combina las tablas a con la condición empleados.d_id = departmentos.d_id vuelca la siguiente tabla de resultados.

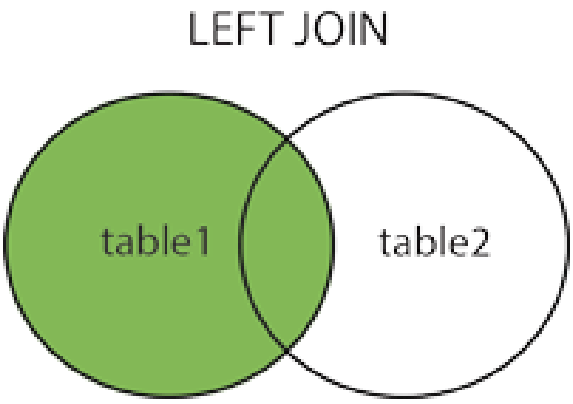
SELECT * FROM empleados INNER JOIN departamentos ON empleados.e_id = departamentos.d_id;

-Resultado

e_id	Apellidos	Nombre	Empleados.d_id	departamentos.d_id	Denominación	Localización
1	García Hurtado	Macarena	3	3	Recursos Humanos	Marbella
2	Ocaña Martínez	Francisco	1	1	Ventas	Sevilla
3	Gutiérrez Doblado	Elena	1	1	Ventas	Sevilla
4	Hernández Soria	Manuela	2	2	IT	Málaga

-Left Join

La sentencia LEFT JOIN combina los valores de la primera tabla con los valores de la segunda tabla. Siempre devolverá las filas de la primera tabla, incluso aunque no cumplan la condición.



-Sintaxis

SELECT * FROM tabla1
LEFT JOIN tabla2
WHERE tabla1.columna1 = tabla2.columna1

-Implementación

Tabla personas, con la clave primaria "per "

per	nombre	apellido1	apellido2	dep
1	ANTONIO	PEREZ	GOMEZ	1
2	ANTONIO	GARCIA	RODRIGUEZ	2
3	PEDRO	RUIZ	GONZALEZ	4

Tabla "departamentos", con la clave primaria "dep"

dep	departamento
1	ADMINISTRACION
2	INFORMATICA
3	COMERCIAL

SELECT nombre, apellido1, departamento
FROM personas
LEFT JOIN departamentos
WHERE personas.dep = departamentos.dep

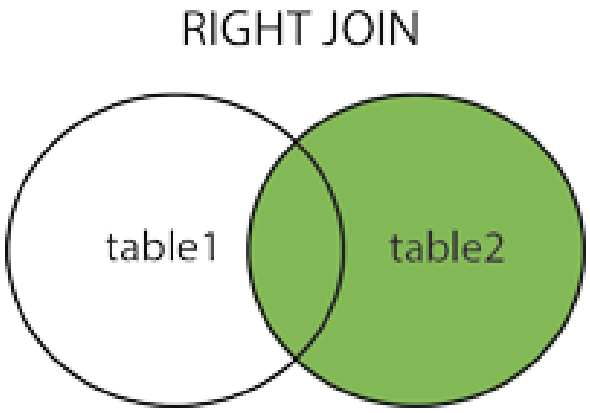
-Resultado

nombre	apellido1	departamento
ANTONIO	PEREZ	ADMINISTRACION
ANTONIO	GARCIA	INFORMATICA
PEDRO	RUIZ	

Aunque el departamento '4' de PEDRO RUIZ no existe en la tabla de departamentos, devolverá la fila con esa columna 'departamento' en blanco.

-Right Join

La sentencia **RIGHT JOIN** combina los valores de la primera tabla con los valores de la segunda tabla. Siempre devolverá las filas de la segunda tabla, incluso aunque no cumplan la condición.



-Sintaxis

SELECT * FROM tabla1 RIGHT JOIN tabla2 WHERE tabla1.columna1 = tabla2.columna1

-Implementación

Tabla personas, con la clave primaria "per "

per	nombre	apellido1	apellido2	dep
1	ANTONIO	PEREZ	GOMEZ	1
2	ANTONIO	GARCIA	RODRIGUEZ	2
3	PEDRO	RUIZ	GONZALEZ	4

Tabla "departamentos", con la clave primaria "dep"

dep	departamento
1	ADMINISTRACION
2	INFORMATICA
3	COMERCIAL

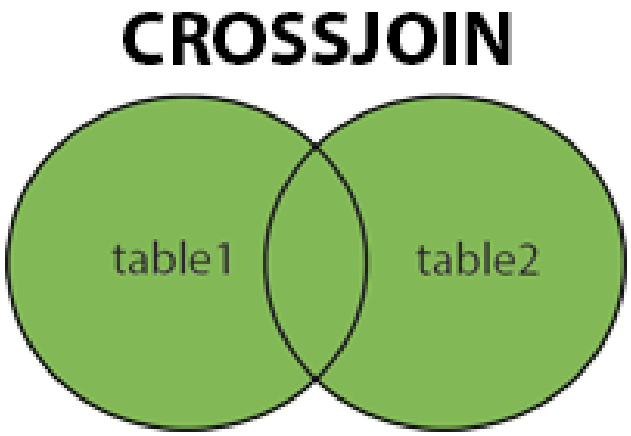
SELECT nombre, apellido1, departamento
FROM personas
RIGHT JOIN departamentos
WHERE personas.dep = departamentos.dep
-Resultado

nombre	apellido1	departamento
ANTONIO	PEREZ	ADMINISTRACION
ANTONIO	GARCIA	INFORMATICA
		COMERCIAL

Aunque no exista ninguna persona del departamento 'COMERCIAL' (3), esta fila aparecerá con las otras columnas en blanco

-CROSS JOIN

La sentencia **CROSS JOIN** combina los valores de la primera tabla con los valores de la segunda tabla. Siempre devolverá las filas de las dos tablas, aunque no tengan coincidencias.



-Sintaxis

SELECT *
FROM *tabla1*
CROSS JOIN *tabla2*;

-Implementación

Tabla personas, con la clave primaria "per "

per	nombre	apellido1	apellido2	dep
1	ANTONIO	PEREZ	GOMEZ	1
2	ANTONIO	GARCIA	RODRIGUEZ	2
3	PEDRO	RUIZ	GONZALEZ	4

Tabla "departamentos", con la clave primaria "dep"

dep	departamento
1	ADMINISTRACION
2	INFORMATICA
3	COMERCIAL

select * from persona cross join departamentos;

-Resultado

per	nombre	apellido1	apellido2	dep	dep	departamento
3	PEDRO	RUIZ	GONZALEZ	4	1	ADMINISTRACION
2	ANTONIO	GARCIA	RODRIGUEZ	2	1	ADMINISTRACION
1	ANTONIO	PEREZ	GOMEZ	1	1	ADMINISTRACION
3	PEDRO	RUIZ	GONZALEZ	4	2	INFORMATICA
2	ANTONIO	GARCIA	RODRIGUEZ	2	2	INFORMATICA
1	ANTONIO	PEREZ	GOMEZ	1	2	INFORMATICA
3	PEDRO	RUIZ	GONZALEZ	4	3	COMERCIAL
2	ANTONIO	GARCIA	RODRIGUEZ	2	3	COMERCIAL
1	ANTONIO	PEREZ	GOMEZ	1	3	COMERCIAL

Si agrega una cláusula WHERE (si tabla1 y tabla2 tienen una relación), CROSS JOIN producirá el mismo resultado que la cláusula INNER JOIN.

-Conclusiones

Las cláusulas JOIN son de gran utilidad para agrupar las diferentes tablas que estemos manejando en nuestro gestor de base de datos. Ya que como se mencionó anteriormente estas se basan en la algebra relacional y diagramas de ven y podemos ver gráficamente cómo funcionan. Ya que podemos seleccionar la parte izquierda, derecha, las dos partes completas e incluso seleccionar la intersección de las tablas dependiendo de las condiciones que pongamos. Dependiendo de la necesidad de la consulta estas pueden ser de gran ayuda para organizar nuestras tablas.

-Referencias

IONOS. (2019, 31 julio). *INNER JOIN: definición y aplicación*. IONOS Digitalguide. Recuperado 21 de abril de 2022, de https://www.ionos.mx/digitalguide/hosting/cuestiones-tecnicas/inner-join/

SQL LEFT JOIN. (s. f.). 11sql. Recuperado 21 de abril de 2022, de http://sql.11sql.com/sql-left-join.htm

w3schools. (s. f.-a). *MySQL CROSS JOIN Keyword*. Recuperado 21 de abril de 2022, de https://www.w3schools.com/mysql/mysql_join_cross.asp

w3schools. (s. f.-b). *SQL RIGHT JOIN Keyword*. Recuperado 21 de abril de 2022, de https://www.w3schools.com/sql/sql_join_right.asp