



Introducción al Álgebra Relacional

ING. VÍCTOR A. FUENTES T., MSC.

Álgebra Relacional

- ▶ Es un lenguaje teórico con operaciones que se aplican a una o más relaciones, con el fin de definir otra relación sin modificar las relaciones originales.
- ▶ Existen múltiples variaciones de las operaciones incluidas en el álgebra relacional.
- ▶ Existen ocho operaciones originales propuestas por Codd. Las cinco fundamentales son selección, proyección, producto cartesiano, unión y diferencia de conjuntos. Adicionalmente se definen las operaciones combinación, intersección y división.
- ▶ Las dos primeras operaciones son conocidas como operaciones unarias, y el resto como operaciones binarias, dado el número de relaciones que participan.

1. Selección (o Restricción)

- ▶ La operación de selección se aplica a una única relación R y define otra relación que contiene únicamente aquellas tuplas de R que satisfacen la condición (predicado) especificada

$$\sigma_{\text{predicado}}(R)$$

- ▶ Ejemplo. Enumerar todos los miembros del personal cuyo salario sea superior a 5000 dólares:

$$\sigma_{\text{salario} > 5000}(\text{Empleados})$$

- ▶ Se puede utilizar cualquier operador lógico.

Ejemplo Selección

ID	Nombre	Año	Índice
50	Amy	2	2.80
60	Mike	4	1.75
70	Sam	4	2.60
80	Tom	3	1.25

$\sigma_{\text{predicado (R)}}$

$\sigma_{\text{Índice} > 2.50}$ (Estudiante)

Se obtiene el siguiente resultado:

ID	Nombre	Año	Índice
50	Amy	2	2.80
70	Sam	4	2.60

Selectividad de la Selección

- ▶ Se puede mostrar la fracción de records (tuplas) que son seleccionadas a través de la expresión de un valor entre 0 y 1. A ello se le denomina selectividad de la selección.
- ▶ $\text{Selectividad de la Selección} = \frac{\text{Número de records de la selección}}{\text{Total de records en la tabla original}}$
- ▶ Si por ejemplo se tiene una tabla con 10 records y se seleccionan 2 de ellos en la selección, entonces:
- ▶ $\text{Selectividad de la Selección} = \frac{\text{Número de records de la selección}}{\text{Total de records en la tabla original}} = \frac{2}{10} = 0.2 = 20\%$

2. Proyección

- ▶ La proyección se aplica a una única relación R y define otra relación que contiene un subconjunto vertical de R, extrayendo los valores de los atributos especificados y eliminando los duplicados.

$$\pi_{a_1, \dots, a_n}(R)$$

- ▶ Ejemplo: Generar una lista de salarios de todo el personal, mostrando solamente los detalles referidos a los atributos idE, fName, lName, salario.

$$\pi_{\text{idE, fName, lName, salario}}(\text{Empleado})$$

Ejemplo Proyección

ID	Nombre	Año	Índice
50	Amy	2	2.80
60	Mike	4	1.75
70	Sam	4	2.60
80	Tom	3	1.25

$\pi_{ID, Nombre, Índice} (Estudiante)$

Se obtiene el siguiente resultado:

ID	Nombre	Índice
50	Amy	2.80
60	Mike	1.75
70	Sam	2.60
80	Tom	1.25

Compatibilidad de Unión

- ▶ Dos tablas manejan compatibilidad de unión si ambas tablas tienen el mismo grado y si el dominio de los correspondientes atributos es el mismo.
- ▶ Las siguientes dos tablas tienen compatibilidad de unión.

Id	Nombre	Salario
Integer	Char(20)	Decimal(7,2)

Código	Título	Precio
Integer	Char(20)	Decimal(7,2)

3. Unión

- ▶ La unión de dos relaciones R y S define una relación que contiene todas las tuplas R, de S o tanto de R como de S, eliminándose las tuplas duplicadas. R y S deben ser compatibles con respecto a la unión.

$R \cup S$

T1

ID	Name
1	Jason
2	Pam
3	Jack
4	Jill

T2

Num	Name
4	George
5	Sam
2	Pam

$T1 \cup T2$



ID	Name
1	Jason
2	Pam
3	Jack
4	Jill
4	George
5	Sam

4. Intersección

- ▶ La operación de intersección define una relación compuesta por el conjunto de todas las tuplas que existen tanto en R como en S. R y S deben ser compatibles con respecto a la unión.

$R \cap S$

T1

ID	Name
1	Jason
2	Pam
3	Jack
4	Jill

T2

Num	Name
4	George
5	Sam
2	Pam

T1 \cap T2

Num	Name
2	Pam

5. Diferencia de conjuntos

- ▶ La operación diferencia de conjuntos define una relación compuesta por las tuplas que se encuentran en la relación R pero no en S. R y S deben ser compatibles con respecto a la unión.

R – S

T1

ID	Name
1	Jason
2	Pam
3	Jack
4	Jill

T2

Num	Name
4	George
5	Sam
2	Pam

T1 – T2

ID	Name
1	Jason
3	Jack
4	Jill

6. Producto Cartesiano

- ▶ La operación de producto cartesiano define una relación que es la concatenación de cada tupla de la relación R con cada tupla de la relación S.
- ▶ El resultado será una tabla con la multiplicación de la cantidad de tuplas de cada una como tuplas totales y la suma del grado de cada tabla como el número de atributos.

Ejemplo Producto Cartesiano

$$|R_1| = k$$
$$|R_2| = l$$

$$\text{Grado}(R_1) = m$$
$$\text{Grado}(R_2) = n$$

$$\text{Grado}(R_1 \times R_2) = m + n$$
$$|R_1 \times R_2| = k \cdot l$$

$$\text{Grado}(T_1 \times T_2) = 2 + 2 = 4$$
$$|T_1 \times T_2| = 4 \cdot 3 = 12$$

T1

ID	Name
1	Jason
2	Pam
3	Jack
4	Jill

T2

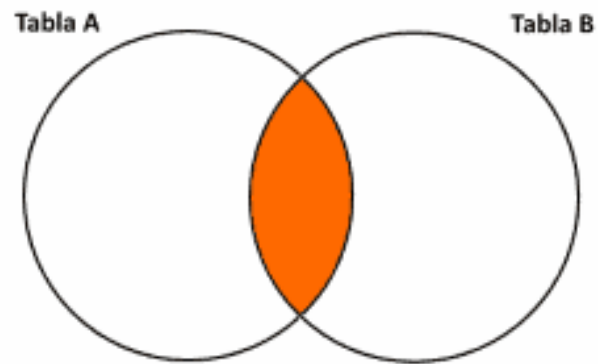
Num	Name
4	George
5	Sam
2	Pam

T1 x T2

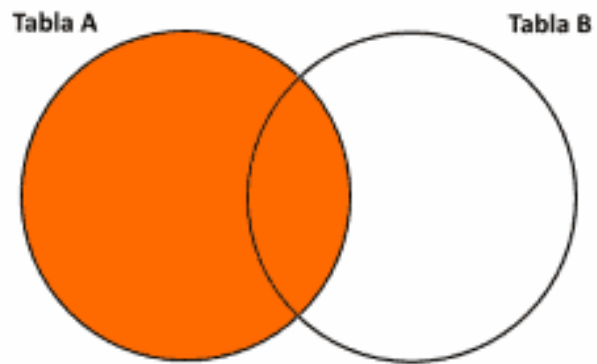
ID	Name	Num	Name
1	Jason	4	George
1	Jason	5	Sam
1	Jason	2	Pam
...
4	Jill	4	George
4	Jill	5	Sam
4	Jill	2	Pam

7. Join (Combinación)

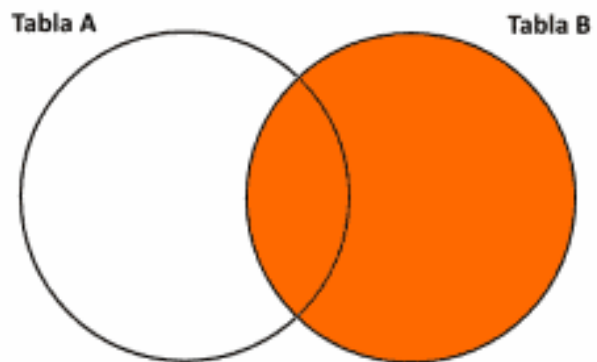
- ▶ La operación de JOIN o combinación se obtiene a través del producto cartesiano.
- ▶ Esta operación filtra los récords o tuplas resultantes del producto basados en una condición de combinación JOIN. Cada una de las tablas que participa en un JOIN debe tener un atributo en común que tenga el mismo dominio. Estos atributos se conocen como atributos JOIN.
- ▶ Las condiciones del JOIN pueden usar cualquiera de los siguientes operadores $=, \neq, <, >, \leq, \geq$. Cuando el operador usado es $=$ se conoce como equi-JOIN.
- ▶ Al usar equi-JOIN se tendrán dos columnas con los mismos valores. Se puede remover una de las columnas repetidas, proceso que se conoce como JOIN natural.



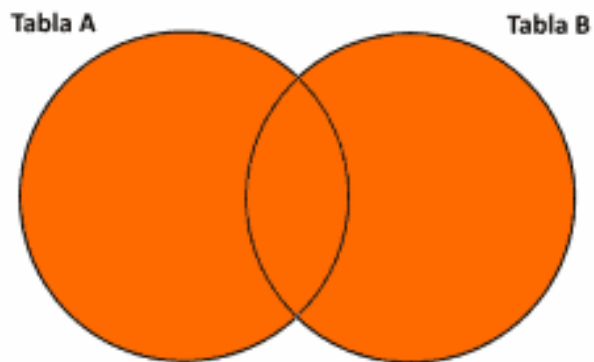
INNER JOIN



LEFT OUTER JOIN



RIGHT OUTER JOIN



FULL OUTER JOIN

Tipos de Join

7. Join (Combinación)

Formato: **R** ⋈_{Predicado} **S**

Aplicant

ID	Name	Skill
15	Jeff	Java
16	Mike	C++
17	Pam	Java
18	Dan	Oracle

Jobs

ID	Title	Req_Skill
100	Programmer	C#
101	Programmer	Java
102	DBA	Informix




Aplicant ⋈_{Skill=Req_Skill} **Jobs**

ID	Title	Skill	ID	Title	Req_Skill
15	Jeff	Java	101	Programmer	Java
17	Pam	Java	101	Programmer	Java



ID	Title	Skill	ID	Title
15	Jeff	Java	101	Programmer
17	Pam	Java	101	Programmer

Outer Join

Outer Join	Símbolo	Descripción
Left Outer Join		Toma el resultado de la operación JOIN y agrega las tuplas que no coincidieron con la condición de la tabla izquierda. Agrega valores nulos a las columnas del lado derecho.
Right Outer Join		Toma el resultado de la operación JOIN y agrega las tuplas que no coincidieron con la condición de la tabla derecha. Agrega valores nulos a las columnas del lado izquierdo.
Full Outer Join		Toma el resultado de la operación JOIN y agrega las tuplas que no coincidieron con la condición de la tabla izquierda y derecha. Agrega los valores nulos correspondientes.

Ejemplos (Left Outer Join)

Aplicant

ID	Name	Skill
15	Jeff	Java
16	Mike	C++
17	Pam	Java
18	Dan	Oracle

Jobs

ID	Title	Req_Skill
100	Programmer	C#
101	Programmer	Java
102	DBA	Informix

Aplicant ⋈_{Aplicant.Skill=Jobs.Req_Skill} **Jobs**

ID	Name	Skill	ID	Title	Req_Skill
15	Jeff	Java	101	Programmer	Java
17	Pam	Java	101	Programmer	Java
16	Mike	C++	Null	Null	Null
18	Dan	Oracle	Null	Null	Null

Ejemplos (Full Outer Join)

Aplicant

ID	Name	Skill
15	Jeff	Java
16	Mike	C++
17	Pam	Java
18	Dan	Oracle

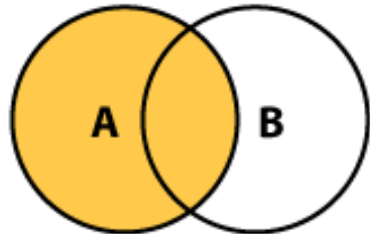
Jobs

ID	Title	Req_Skill
100	Programmer	C#
101	Programmer	Java
102	DBA	Informix

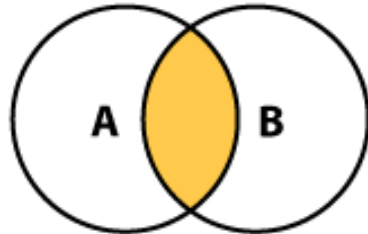
Aplicant ⋈ **Jobs** **Aplicant.Skill=Jobs.Req_Skill**

ID	Name	Skill	ID	Title	Req_Skill
15	Jeff	Java	101	Programmer	Java
17	Pam	Java	101	Programmer	Java
16	Mike	C++	Null	Null	Null
18	Dan	Oracle	Null	Null	Null
Null	Null	Null	101	Programmer	Java
Null	Null	Null	102	DBA	Informix

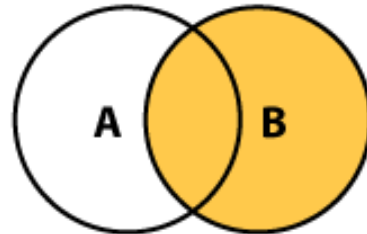
SQL JOINS



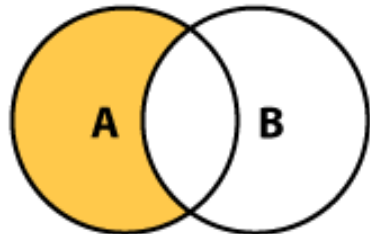
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key=B.Key
```



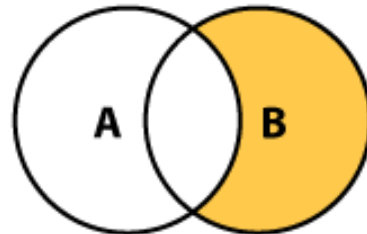
```
SELECT <select_list>  
FROM TableA A  
INNER JOIN TableB B  
ON A.Key=B.Key
```



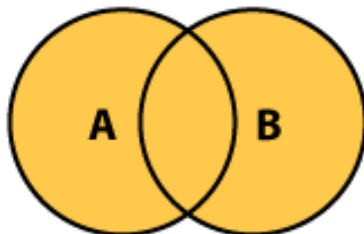
```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key=B.Key
```



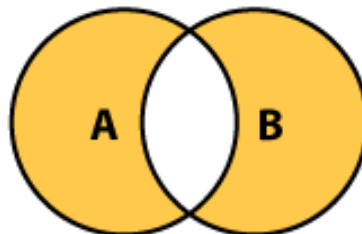
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key=B.Key  
WHERE B.Key IS NULL
```



```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key=B.Key  
WHERE A.Key IS NULL
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key=B.Key
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key=B.Key  
WHERE A.Key IS NULL  
OR B.Key IS NULL
```

Usos de Joins en SQL