

Algebra Relacional

Introducción

El álgebra relacional es un conjunto de operaciones formales que se utilizan para manipular relaciones (tablas) en una base de datos relacional. Es la base teórica sobre la cual se construyen los lenguajes de consulta como SQL. En este documento, exploraremos los conceptos fundamentales del álgebra relacional, sus operaciones principales y cómo se aplican en el contexto de las bases de datos. Además, se incluirán ejemplos prácticos y ejercicios para reforzar el aprendizaje.

Usamos las sentencias de MySQL para los ejemplos. Los ejemplos en SQL son solo demostrativos y se usan para poder entender las operaciones del algebra relacional.

1. Conceptos Fundamentales

Relación

Una relación es una tabla compuesta por filas y columnas. Cada fila representa un registro (tupla), y cada columna representa un atributo (campo). Por ejemplo:

ID	Nombre	Edad	Ciudad
1	Juan	25	Bogotá
2	María	30	Medellín
3	Carlos	22	Cali

En este caso:

- *ID, Nombre, Edad y Ciudad* son atributos.
- Cada fila (*por ejemplo, (1, Juan, 25, Bogotá)*) es una tupla o instancia.

Esquema de una Relación

El esquema define la estructura de la relación, especificando los nombres de los atributos y sus tipos de datos. Por ejemplo:

Personas(ID: Entero, Nombre: Texto, Edad: Entero, Ciudad: Texto)

2. Operaciones del Álgebra Relacional

Las operaciones del álgebra relacional se dividen en dos categorías:

- **Operaciones básicas:** Selección, Proyección, Unión, Diferencia y Producto Cartesiano.
- **Operaciones derivadas:** Intersección, Join (combinación) y División.

2.1. Selección (Sigma σ)

La selección permite filtrar las tuplas de una relación que cumplen una condición específica.

Sintaxis:

$$\sigma_{condición}(R)$$

Ejemplo:

Dada la relación *Personas*:

ID	Nombre	Edad	Ciudad	
----	-----	-----	-----	
1	Juan	25	Bogotá	
2	María	30	Medellín	
3	Carlos	22	Cali	

Consulta: Seleccionar todas las personas cuya edad sea mayor a 24.

$$\sigma_{\text{Edad} > 24}(\text{Personas})$$

Resultado:

ID	Nombre	Edad	Ciudad	
-----	-----	-----	-----	
1	Juan	25	Bogotá	
2	María	30	Medellín	

SQL

Sintaxis:

*SELECT * FROM Tabla WHERE condición;*

Ejemplo:

*SELECT * FROM Personas WHERE Edad > 24;*

2.2. Proyección (π)

La proyección selecciona ciertos atributos (columnas) de una relación.

Sintaxis:

$$\pi_{\text{Atributos}}(R)$$

Ejemplo:

Usando la misma relación *Personas*, queremos obtener solo los nombres y ciudades de las personas.

$$\pi_{\text{Nombre, Ciudad}}(\text{Personas})$$

Resultado:

Nombre	Ciudad	
-----	-----	
Juan	Bogotá	
María	Medellín	
Carlos	Cali	

SQL

Sintaxis:

SELECT columnas FROM Tabla;

Ejemplo:

SELECT Nombre, Ciudad FROM Personas;

2.3. Unión ($\cup$)

La unión combina las tuplas de dos relaciones, eliminando duplicados. Las relaciones deben tener el mismo esquema (Las relaciones deben de tener el número de atributos y el mismo nombre, definidos en el mismo dominio).

.

Sintaxis:

$$R_1 \cup R_2$$

Ejemplo:

Sean las relaciones A y B :

A :

ID	Nombre
----	--------

----- -----	
1	Juan
2	María

B:

ID	Nombre
----- -----	
2	María
3	Carlos

Unión:

$A \cup B$

Resultado:

ID	Nombre
----- -----	
1	Juan
2	María
3	Carlos

SQL

Sintaxis:

*SELECT * FROM Tabla1*

UNION

*SELECT * FROM Tabla2;*

Ejemplo:

*SELECT * FROM A*

UNION

*SELECT * FROM B;*

2.4.Diferencia (-)

La diferencia devuelve las tuplas que están en una relación, pero no en otra. Si es $A - B$, devuelve las tuplas que están en A pero no en B. Las relaciones deben de tener el numero de atributos y el mismo nombre, definidos en el mismo dominio.

Sintaxis:

$$R_1 - R_2$$

Ejemplo:

Usando las mismas relaciones A y B:

$A - B$

Resultado:

ID	Nombre
----	--------

-----	-----
-------	-------

1	Juan
---	------

SQL

Sintaxis:

*SELECT * FROM Tabla1*

EXCEPT

*SELECT * FROM Tabla2;*

Ejemplo:

SELECT * FROM A

EXCEPT

SELECT * FROM B;

En MySQL

*SELECT * FROM A WHERE (id, nombre) NOT IN (SELECT id, nombre FROM B);*

MySQL no ha implementado la operación EXCEPT en su dialecto de SQL. En su lugar, debes usar las alternativas mencionadas anteriormente (NOT IN o NOT EXISTS).

Ejercicio:

¿Cuál sería el resultado si invertimos las relaciones?

$B - A$

¿Sería el mismo resultado de $A - B$?

2.5. Producto Cartesiano (×)

El producto cartesiano combina cada tupla de una relación con cada tupla de otra relación.

Sintaxis:

$R_1 \times R_2$

Ejemplo:

Sean las relaciones *C* y *D*:

C:

| ID | Nombre |

----	-----	
------	-------	--

1	Juan	
---	------	--

2	María	
---	-------	--

D:

Ciudad	
--------	--

-------	--

Bogotá	
--------	--

Medellín	
----------	--

Producto Cartesiano:

C X D

Resultado:

ID	Nombre	Ciudad	
----	--------	--------	--

----	-----	-----	
------	-------	-------	--

1	Juan	Bogotá	
---	------	--------	--

1	Juan	Medellín	
---	------	----------	--

2	María	Bogotá	
---	-------	--------	--

2	María	Medellín	
---	-------	----------	--

SQL

Sintaxis:

*SELECT * FROM Tabla1, Tabla2;*

Ejemplo:

*SELECT * FROM C, D;*

Ejercicios Prácticos

Los ejercicios deben de hacerlos primero en el cuaderno y comprobar el resultado en el phphmyadmin. Solo se revisara los ejercicios que estén resueltos en el cuaderno.

Ejercicio 1

Dada la relación *Productos*:

ID	Nombre	Precio	Stock
-----	-----	-----	-----
1	Camiseta	20	50
2	Pantalón	50	20
3	Zapatos	80	10

1. Obtener los productos con precio mayor a 30.
2. Proyectar solo los nombres de los productos.

Ejercicio 2

Dadas las relaciones *Clientes* y *Pedidos*:

Clientes:

ID	Nombre	Ciudad	
-----	-----	-----	
1	Ana	Bogotá	
2	Pedro	Medellín	

Pedidos:

ID_Cliente	Producto
-----	-----
1	Camiseta
2	Zapatos

Realizar un join entre `Clientes` y `Pedidos` usando `ID = ID\_Cliente`.

Anexos:

Operador relacional	Significado
=	Igual
<	Menor que
>	Mayor que
≤	Menor o igual
≥	Mayor o igual
<>	Diferente
!=	Diferente