

Algebra Relacional

1. Proyección $P(L)(R)$

Filtrar por atributos

Columnas Tabla

Ventas				Artículos	
DNI	CA	CT	NU	CA	CAT
1111	C1	T1	5	C1	Limpieza
2222	C1	T2	1	C2	Menaje
3333	C2	T1	2	C1	Alimentación

2. Selección $S(F)(R)$

Extrae de una tabla R las filas que cumplen el predicado lógico F.

Ejemplo:

- $P(DNI) S(CA = 'C1')(VENTAS)$
- $S(CA = 'C1')(VENTAS)$

DNI	CA	CT	NU	DNI
1111	C1	T1	5	1111
2222	C1	T2	1	2222

3. Unión RUS

Las columnas o dominios deben coincidir y su orden

$$\{ \text{Carla, Himar} \} \cup \{ \text{Carla, Sergio} \} \rightarrow \{ \text{Carla, Himar, Sergio} \}$$

4.1 Intersección RNS

Devuelve lo común de ambas tablas y los mismos atributos

$$\{ \text{Carla, Himar} \} \cap \{ \text{Carla, Sergio} \} \rightarrow \{ \text{Carla} \}$$

$$P(\text{Ventas} \cap \text{Artículos})$$

$$\begin{matrix} CA \\ C1 \\ C1 \end{matrix}$$

5. Diferencia R-S

Se muestra el contenido de R que no se encuentra en S.

$$\{ a, b, c \} - \{ a, c, d \} = \{ b \} \neq \{ b, d \} \text{ (solo resta, no añade)}$$

6. Producto cartesiano R×S

Combinar todas las filas de la tabla R con la tabla S

Ej: $\begin{array}{|c|c|c|} \hline \text{DNI} & \text{Nombre} & \text{Edad} \\ \hline 1111 & \text{Himar} & 24 \\ \hline 2222 & \text{Lucia} & 36 \\ \hline 3333 & \text{Carlos} & 22 \\ \hline \end{array}$

$$A = P(\text{DNI})(\text{PERSONA})$$

$$B = P(\text{Nombre})(\text{PERSONA})$$

$$\rightarrow A \times B$$

DNI	Nombre
1111	Himar
2222	Lucia
3333	Carlos
2222	Himar
2222	Lucia
:	
3333	Carlos

7. Unión natural R*S

Devuelve las filas cuyos atributos con mismo nombre y mismo dominio tengan valores iguales, y en caso de haber columnas repetidas (mismo dominio y valor) las elimina.

- Hace una unión (U) por el campo común.

$$\text{VENTA}(\text{DNI}, \text{CT}, \text{CA}, \text{F})$$

- Unión y producto cartesiano al mismo tiempo.

$$\text{PERSONA}(\text{DNI}, \text{NOMBRE}, \text{EDAD})$$

Ejemplo: Edad de las personas que han comprado algún artículo el 01-03-2023

$$= \begin{array}{l} P(\text{Edad}) S((F = '01-03-2023') \wedge (\text{VENTA.DNI} = \text{PERSONA.DNI})) (\text{PERSONA} \times \text{VENTA}) \\ - P(\text{Edad}) S(F = '01-03-2023') (\text{PERSONA} * \text{VENTA}) \end{array}$$

8. Cociente R/S

Verificar que el esquema de S tiene que estar contenido en el esquema de R.

Solo se puede usar cuando los atributos de S (ds) son un subconjunto de los atributos de R (dr).

do que queremos probar/evaluar

se pone el 'todas'

Ejemplos: VENTA(DNI, CT, CA, F); PERSONA(DNI, NOMBRE, EDAD); TIENDA(CT, Dirección)

a) Personas que han comprado algún artículo en todas las tiendas.

$$P(DNI)(VENTA)$$

$$P(CT)(TIENDA)$$

b) Personas que han comprado algún artículo de todas las categorías

$$P(DNI, CAT)(VENTA * ARTICULOS)$$

$$P(CAT)(ARTICULOS)$$

c) Personas que han comprado todos los artículos de todas las categorías

$$P(DNI, CA, CAT)(VENTA * ARTICULOS)$$

$$P(CAT, CA)(ARTICULOS)$$

9. Fórmula:

$$P(DNI) \left(P(DNI, CAT)(ALQUILERES * COCHES) - P(DNI, CAT) \left(\begin{array}{l} P(DNI)(ALQUILERES) \times P(DNI, M, AT)(ALQUILERES * COCHES) \\ - \\ P(DNI, M, AT)(ALQUILERES * COCHE) \end{array} \right) \right)$$

Calculo Relacional de Tuplas

Se trata de un tipo de cálculo relacional donde las variables representan Tuplas de una tabla. Sigue la siguiente estructura:

$\{t \mid P(t)\}$ = Conjunto de todas las Tuplas que satisfacen el predicado lógico.

- t = Variable libre que queremos hallar

- $P(t)$ = Predicado lógico (condición) a trabajar.

Calculo Relacional de Dominios

SQL - DML

3.1. SELECT

SELECT [ALL | UNIQUE | DISTINCT]

- UNIQUE: Obtener valores únicos de la columna de una tabla.
- DISTINCT: Obtener valores únicos de una columna que puede tener valores duplicados.

3.2. FROM

Indica las tablas que se utilizarán en la consulta, así como la relación entre ellas

SELECT parametros

FROM nombre_tabla;

Se puede indicar otro tipo de relaciones:

- CROSS JOIN: producto cartesiano a las tablas indicadas. X
- NATURAL JOIN: unión natural a las tablas indicadas. *
- INNER JOIN: unión según el predicado indicado.

3.3. WHERE

La cláusula WHERE es igual al operador SELECCION en álgebra relacional. Debe aparecer después de la cláusula FROM. Permite utilizar los siguientes elementos:

SELECT argumento

FROM tabla

WHERE operador_lógico

- Operadores aritméticos: (), +, -, *, /
- Operadores comparacionales: =, <>, !=, <, <=, >, >=
- Operadores lógicos: AND, OR, NOT

3.3.1. BETWEEN

Sirve para comprobar si una expresión está comprendida entre dos valores. Debe colocarse dentro del WHERE.

SELECT DNI

FROM CUENTA

WHERE CS BETWEEN 1 AND 10;

3.3.2. IN

Sirve para comprobar si una expresión está dentro de un conjunto. Debe colocarse dentro del WHERE.

SELECT DNI

FROM CUENTA

WHERE CS IN (SELECT CS
FROM CUENTA
WHERE DNI = 111);

3.3.3. ANY y ALL

- El operador ANY verifica si la expresión cumple un operador comparacional para cualquier valor del conjunto indicado.
- El operador ALL es igual al ANY pero para todos los valores del conjunto.

SELECT DNI

FROM CUENTA

WHERE SLD >= [ANY | ALL] (SELECT SLD
FROM CUENTA
WHERE DNI = 1111);

3.3.4. LIKE

Sirve para comprobar si una expresión se asemeja a un patrón indicado

SELECT DNI

%: serie de 0 a N caracteres arbitrarios

FROM CUENTA

_: Único carácter arbitrario.

WHERE N LIKE '%';

3.3.6. EXISTS

Comprueba si la tabla tiene al menos una fila que existe. Debe colocarse dentro WHERE

SELECT DNI

FROM CLIENTE C1

WHERE NOT EXISTS (SELECT *

FROM CUENTA C2

WHERE (CS=1)

AND (C1.DNI = C2.DNI);

3.4. Operadores de conjunto

Permiten realizar operaciones de conjunto a los datos resultantes de una consulta.

3.4.1. UNION

SELECT *

FROM CUENTA

UNION

SELECT *

FROM PRESTANDO;

3.4.2. INTERSECT

SELECT DNI

FROM CUENTA

WHERE CS=1

INTERSECT

SELECT DNI FROM

PRESTANDO WHERE CS=1;

3.4.3. Diferencia (MINUS)

SELECT UP

FROM TRABAJA

MINUS

SELECT NP FROM

TRABAJA NATURAL JOIN LOCALIZACION

WHERE CDC != 'La Laguna';

3.6. ORDER BY

Ordena las filas que se mostrarán en la tabla resultados. Debe ser la última cláusula en toda la consulta.

SELECT * FROM Products

ORDER BY Price [ASC | DESC]

3.7. Funciones agregadas.

Permite convertir o resumir todo un grupo de datos en un único valor. Solo se aplica a **SELECT** y **HAVING**

- AVG: media conjunto datos
- COUNT: cuenta nº filas
- MAX: máximo del conjunto
- MIN: mínimo del conjunto
- SUM: suma valores de un conjunto.
- VARIANCE: varianza conjunto datos.
- STDDEV: desviación típica conjunto de datos.

3.7.1. GROUP BY

Se usa comúnmente con funciones agregadas para agrupar el resultado del conjunto en una o más columnas. Cláusula opcional y debe ir después del WHERE.

`SELECT COUNT(DISTINCT NC)`

`FROM CUENTA`

`GROUP BY CS;`

3.7.2. HAVING

Se añadió a SQL porque WHERE no puede utilizarse con funciones agregadas

`SELECT CS, NC`

`FROM CUENTA`

`GROUP BY CS, NC`

`HAVING COUNT(*) >= 2;`

3.8. Herramientas adicionales

3.8.1. Operaciones sobre cadenas

- UPPER(x): devuelve x en mayúscula
- LOWER(x): devuelve x en minúscula
- LENGTH(x): devuelve longitud x
- CONCAT(x,y): concatena x e y
- INITCAP(x): devuelve la cadena x en formato mayus-minus

3.8.2. Operaciones numéricas

- SIN(x)
- FLOOR(x)
- COS(x)
- ROUND(x)
- TAN(x)
- SIGN(x)
- SQRT(x)
- POWER(x,e)
- ABS(x)
- LOG(m,n)
- CEIL(x)
- MOD(m,n)

3.8.3. Operaciones sobre fechas

Tipo dato para fechas → DATE

- SYSDATE
- CURRENT_DATE
- SYSTIMESTAMP
- CURRENT_SYSTIMESTAMP

`SELECT TO_CHAR(SYSDATE, 'HH-MI-SS')`

`FROM DUAL`

SQL - DDL

4.1. Crear una vista

```
CREATE VIEW VISTA-NOMBRE AS (
    ---- SUBCONSULTA ----
);
```

4.2. Modificación de la estructura de una tabla

- Añadir columnas a la estructura de una tabla:

```
ALTER TABLE TABLA-NOMBRE ADD (
    NUEVA-COLUMNNA VARCHAR2([NUM.LONG.MAX.STRING])(IS|NOT) NULL
);
```

- Añadir una restricción a una tabla usada.

```
ALTER TABLE NOMBRE-TABLA CONSTRAINT NOMBRE-CONSTRAINT CHECK (
    (NOT EXISTS() | Subconsulta)
);
```

→ Expresión booleana

4.3. Modificación de las filas de una tabla

```
UPDATE NOMBRE-TABLA
SET columna = valor
WHERE Condicion;
```

```
UPDATE empleados
SET salario = 55000, puesto = 'Informático'
WHERE id = 3;
```

4.4. Insertar nuevas filas SQL

```
INSERT INTO NOMBRE-TABLA(C1,C2,...)
VALUES (V1,V2,...)
```

4.5. Creación de una assertión

Creación de una restricción de la misma manera que el `ALTER TABLE`, pero con la diferencia de que para este caso realiza e implementa una restricción para la base de datos entera, es decir, más de 1 tabla usada para establecer dicha restricción.

```
CREATE ASSERTION NOMBRE-ASSERTION CHECK NOT EXISTS (
    subconsulta
);
```

4.6. Creación de un trigger

Es un conjunto de instrucciones que se ejecutan automáticamente en respuesta a ciertos eventos en una tabla o vista.

```
CREATE TRIGGER NOMBRE-TRIGGER
[BEFORE | AFTER | ] [DELETE | INSERT | UPDATE | ] ON NOMBRE-TABLA
REFERENCING [NEW | OLD] AS NOMBRE-FILA
FOR EACH ROW
[WHEN (NOMBRE-FILA.C1 = ...)]
```

```
BEGIN
```

```
:  
:  
:  
:
```

```
END;
```