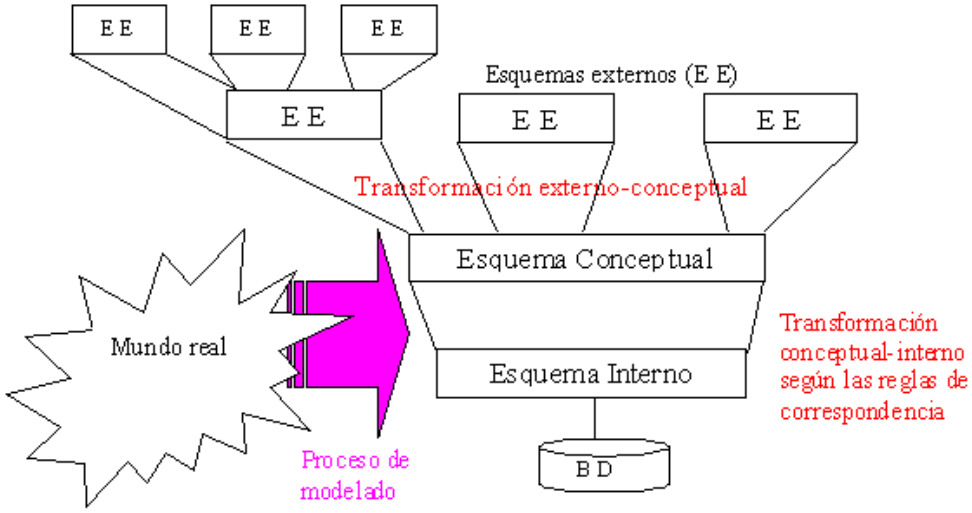


UNIVERSIDAD DE LOS ANDES

Bases de datos Unidad 3

Universidad de Los Andes
Escuela de Ingeniería de Sistemas
Departamento de Computación

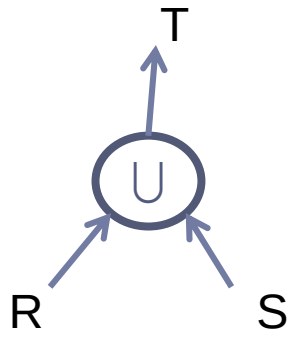
Tema 2. Lenguaje de consulta del modelo relacional y objeto-relacional

Tema 2. Lenguajes de consulta del modelo relacional y objeto-relacional

- **Contenido:**
  - Algebra relacional
  - SQL3
- **Objetivo:**
  - Desarrollar habilidades en el uso de los lenguajes de consulta de las bases de datos objeto-relacionales
- **Actividades:**
  - Leer: Elmasri y Navathe, cap. 6, 8 y 9
  - Realizar los ejercicios de este documento
  - Realizar el ejercicio 3

Lenguajes de manipulación de datos

- **Algebra relacional:**
  - colección de operaciones formales sobre las relaciones
- **Las operaciones básicas son de dos tipos: unarias y binarias**
- **Unión:**
  - La unión de dos relaciones R y S con el mismo esquema es una relación T con el mismo esquema y con el conjunto de tuplas que pertenecen a R, a S o a ambas
  - Notación: $T = R \cup S = UNION(R, S)$



Unión – operación unaria

C1	placa	marca	color
	'LAB384'	'ford'	'verde'
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'

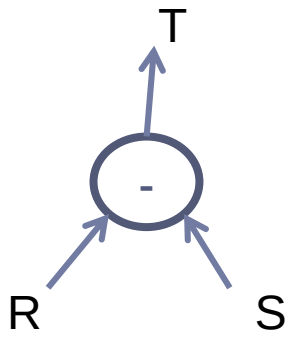
C2	placa	marca	color
	'LAB384'	'ford'	'verde'
	'XSG230'	'ford'	'gris'

$C = C1 \cup C2$

C	placa	marca	color
	'LAB384'	'ford'	'verde'
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'

Diferencia – operación binaria

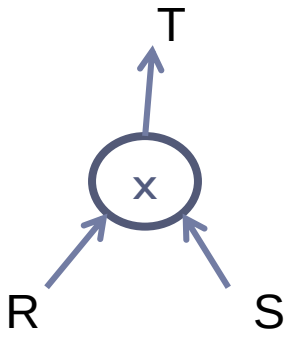
- **La diferencia (R – S) de dos relaciones R y S con el mismo esquema es una relación T con el mismo esquema que contiene las tuplas que pertenecen a R y no pertenecen a S**
- **Notación: T = R – S = MINUS(R-S)**
- **Ejemplo: C3 = C1 – C2**



C3	placa	marca	color
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'

Producto cartesiano – operación binaria

- El producto cartesiano de dos relaciones R y S de cualquier esquema, es una relación T que contiene los atributos de R concatenados con los de S y sus tuplas son todas las formadas por la concatenación de una tupla de R con todas las tuplas de S
- Notación: T = R x S = PRODUCT(R, S)
- Ejemplo: C4 = P1 x C3



Producto cartesiano – operación binaria

P1	cedula	nombre	apellido
	'V4567345'	'Pedro'	'Diaz'
	'V8234125'	'Juana'	'Perez'

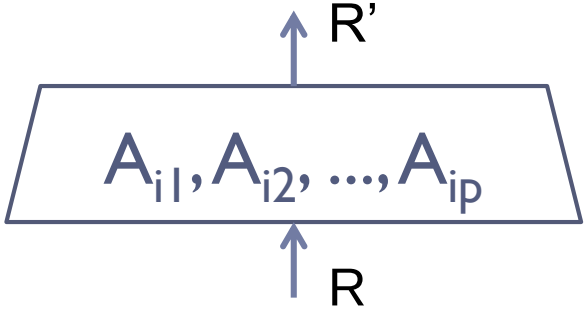
C4 = P1 x C3

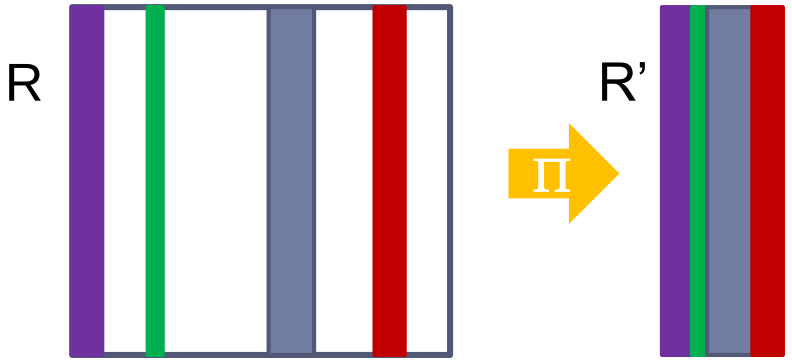
C3	placa	marca	color
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'

C4	cedula	nombre	apellido	placa	marca	color
	'V4567345'	'Pedro'	'Diaz'	'LAM112'	'toyota'	'azul'
	'V4567345'	'Pedro'	'Diaz'	'LGR889'	'toyota'	'azul'
	'V8234125'	'Juana'	'Perez'	'LAM112'	'toyota'	'azul'
	'V8234125'	'Juana'	'Perez'	'LGR889'	'toyota'	'azul'

Proyección – operación unaria

- La proyección de una relación $R(A_1, A_2, ..., A_n)$ sobre los atributos $A_{i1}, A_{i2}, ..., A_{ip}$, con $ij \neq ik$, es una relación R' con esquema $R'(A_{i1}, A_{i2}, ..., A_{ip})$ obtenida por eliminación de los valores de los atributos de R que no están en R' y la supresión de las tuplas duplicadas
- Notación: $R' = \prod A_{i1}, A_{i2}, ..., A_{ip} (R)$
  $= PROJECT(R/ A_{i1}, A_{i2}, ..., A_{ip})$





Proyección

C4	cedula	nombre	apellido	placa	marca	color
	‘V4567345’	‘Pedro’	‘Diaz’	‘LAM112’	‘toyota’	‘azul’
	‘V4567345’	‘Pedro’	‘Diaz’	‘LGR889’	‘toyota’	‘azul’
	‘V8234125’	‘Juana’	‘Perez’	‘LAM112’	‘toyota’	‘azul’
	‘V8234125’	‘Juana’	‘Perez’	‘LGR889’	‘toyota’	‘azul’

R1 = Π marca, color (C4)

R1	marca	color
	‘toyota’	‘azul’

Criterios de selección

- **Fórmula de cualificación conjuntiva**
- **Uniatributo:**
  - $Q = A_{i1}\ \theta_1\ C_1 \wedge A_{i2}\ \theta_2\ C_2 \wedge \ldots \wedge A_{in}\ \theta_n\ C_n$ donde:

$A_{i1}, A_{i2}, \ldots, A_{ip}$ son atributos

$A_{i1}, A_{i2}, \ldots, A_{ip}$ son constantes

θ i $\in \{<, >, \geq, \neq, \leq, =\}$ $\wedge$ es el y lógico

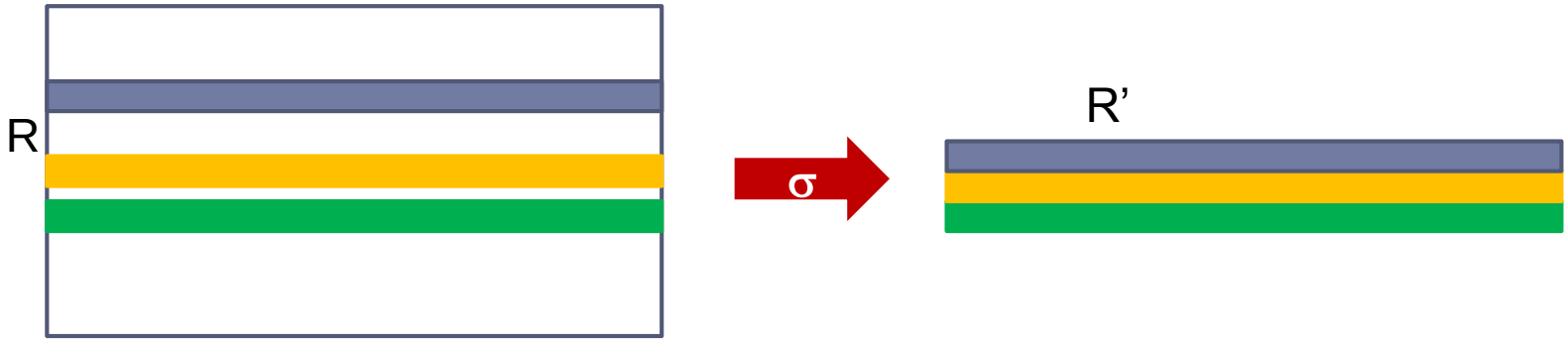
Ejemplo: Q = placa = 'LWQ381'

Q1 = modelo = 'corolla' $\wedge$ color $\neq$ 'rojo'

- **Multiatributos:**
  - $Q = A_{i1}\ \theta_1\ B_1 \wedge A_{i2}\ \theta_2\ B_2 \wedge \ldots \wedge A_{in}\ \theta_n\ B_n$ donde: $A_i \in R$ y $B_j \in S$

Restricción – operación unaria

- **La restricción de una relación R por un criterio de selección Q es una relación R' con el mismo esquema de R y cuyas tuplas son aquellas que pertenecen a R y satisfacen Q**
- **Notación: R' = R [Q] = σ_Q (R) = RESTRICT(R/Q)**



Restricción – operación unaria

W	cedula	nombre	apellido	genero	feNac	ciudad
	'V04567345'	'Pedro'	'Dunas'	'M'	'24/2/56'	'Cumana'
	'V06732179'	'Juan'	'Diaz'	'M'	'5/7/60'	'Valencia'
	'V08234125'	'Juana'	'Perez'	'F'	'6/9/64'	'Barinas'
	'V10238125'	'Juan'	'Lacoste'	'M'	'5/4/70'	'Caracas'

Z = W [apellido < 'Mendez']

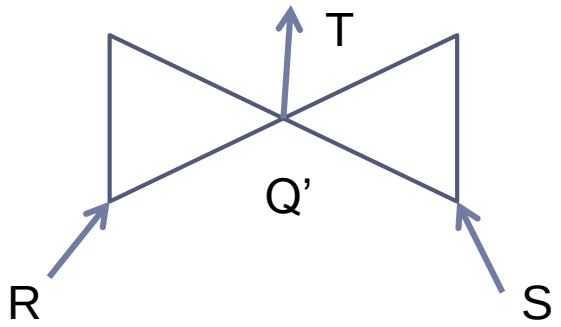
Z	cedula	nombre	apellido	genero	feNac	ciudad
	'V04567345'	'Pedro'	'Dunas'	'M'	'24/2/56'	'Cumana'
	'V06732179'	'Juan'	'Diaz'	'M'	'5/7/60'	'Valencia'
	'V10238125'	'Juan'	'Lacoste'	'M'	'5/4/70'	'Caracas'

Operaciones adicionales

- **Se deducen de las anteriores**

Producto – operación binaria:

- **El producto de dos relaciones R y S según Q' es un conjunto de tuplas del producto cartesiano R x S que satisfacen Q'**
- **Notación:** $T = R \bowtie_{Q'} S = JOIN(R, S / Q')$
- **Ejemplo:** $T1 = P1 \bowtie_{apellido<'Mendez'} C3 = R2$



Producto (*Join*)

P1	cedula	nombre	apellido
	'V4567345'	'Pedro'	'Diaz'
	'V8234125'	'Juana'	'Perez'

$P1 \bowtie_{apellido<'Mendez'} C3 = R2$

C3	placa	marca	color
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'

R2	cedula	nombre	apellido	placa	marca	color
	'V4567345'	'Pedro'	'Diaz'	'LAM112'	'toyota'	'azul'
	'V4567345'	'Pedro'	'Diaz'	'LGR889'	'toyota'	'azul'

UNIVERSIDAD DE LOS ANDES

Casos de productos

1. **Equiproducto o reunión natural: de $R(A_1, A_2, ..., A_n)$ y $S(B_1, B_2, ..., B_p)$ con $Q' = (A_i = B_j)$, $R \bowtie_{Ai = Bj} S$**
2. **Producto: de $R(A_1, A_2, ..., A_n)$ y $S(B_1, B_2, ..., B_p)$ con $Q' = (A_i \theta B_j)$, $R \bowtie_{Ai \theta Bj} S$**
3. **Autoproducto: de $R(A_1, A_2, ..., A_n)$ sobre A_i $R \bowtie_{Ai} R$**
4. **Equiproducto a la izquierda: de $R(A_1, A_2, ..., A_n)$ y $S(B_1, B_2, ..., B_p)$ con $Q' = (A_i = B_j)$, expresándose como: $R \ltimes_{Ai = Bj} S$ y cuyo resultado sólo contiene el esquema de R, desechando las columnas correspondientes de S**
5. **Equiproducto a la derecha: de $R(A_1, A_2, ..., A_n)$ y $S(B_1, B_2, ..., B_p)$ con $Q' = (A_i = B_j)$, que se expresa como: $R \rtimes_{Ai = Bj} S$ cuyo resultado sólo contiene el esquema de S, desechando el de R**

Casos de productos

6. **Semiproducto: en sus dos versiones, a la izquierda y a la derecha, son los descritos en los puntos 4 y 5 anteriores, pero donde el criterio de selección es cualquiera de los definidos como Q', a saber:**

- **T = R $\ltimes_{Q'}$ S semiproducto a la izquierda con el esquema de R**
- **T = R $\rtimes_{Q'}$ S semiproducto a la derecha con el esquema de S**

Ejemplo: T=R2 $\ltimes_{\text{marca} >\text{'ford'}}$ R1

Semi-producto a la izquierda

R2	cedula	nombre	apellido	placa	marca	color
	‘V4567345’	‘Pedro’	‘Diaz’	‘LAM112’	‘toyota’	‘azul’
	‘V4567345’	‘Pedro’	‘Diaz’	‘LGR889’	‘toyota’	‘azul’

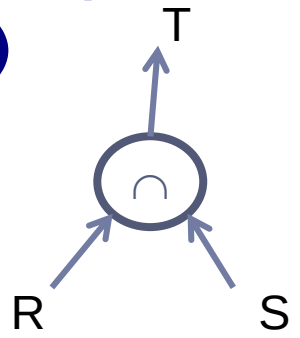
Ejemplo: $T = R2 \ltimes_{marca > \text{'ford'}} R1$

R1	marca	color
	‘toyota’	‘azul’

T	cedula	nombre	apellido	placa	marca	color
	‘V4567345’	‘Pedro’	‘Diaz’	‘LAM112’	‘toyota’	‘azul’
	‘V4567345’	‘Pedro’	‘Diaz’	‘LGR889’	‘toyota’	‘azul’

Intersección – operación binaria

- **La intersección de dos relaciones R y S con el mismo esquema es una relación T con el mismo esquema, que contiene las tuplas que pertenecen a R y a S a la vez**
- **Notación:** $T = R \cap S = INTERSECT(R, S)$
- $T = R - (R - S) = S - (S - R)$
- **Ejemplo:** $T2 = C1 \cap C2$



C1	placa	marca	color
	'LAB384'	'ford'	'verde'
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'

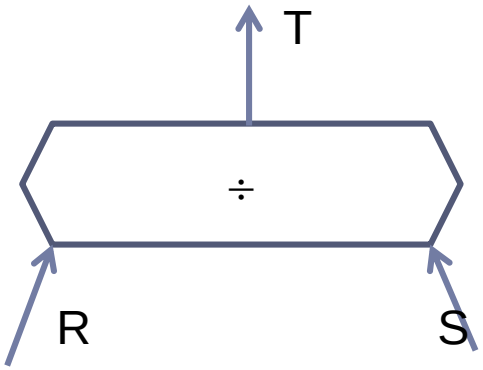
C2	placa	marca	color
	'LAB384'	'ford'	'verde'
	'XSG230'	'ford'	'gris'

T2	placa	marca	color
	'LAB384'	'ford'	'verde'

División – operación binaria

- El cociente de $R(A_1, A_2, ..., A_n)$ por la subrelación $S(A_p, ..., A_n)$ es una relación $T(A_1, A_2, ..., A_{p-1})$ formada por las tuplas que concatenadas a cada una de las tuplas de S da siempre una tupla de R
- Notación: $T = R \div S = DIVISION(R, S)$
- $R \div S = T - W$, donde

$T = \prod A_1, A_2, ..., A_{p-1}(R)$ $W = \prod A_1, A_2, ..., A_{p-1}((T \times S) - R)$



División – operación binaria

Ejemplo: T3 = C5 ÷ C6

C5	año	marca	potencia	color
	'2005'	'ford'	1.8	'verde'
	'2003'	'toyota'	1.6	'azul'
	'2003'	'ford'	1.6	'verde'
	'2005'	'toyota'	1.8	'azul'

C6	año	potencia
	'2005'	1.8
	'2003'	1.6

T3	marca	color
	'ford'	'verde'
	'toyota'	'azul'

Extensión del algebra relacional

- **Para soportar expresiones y funciones (Baralis y Widom, ACM TODS 25(3):269-332. Sep-2000) en SQL y QUEL, excepto las consultas que incluyen ordenamiento**
- **Operador ε: operador unario que calcula expresiones aplicadas a cada tupla de R, produciendo un resultado temporal (no es una tabla en la BD) con el esquema de R unido a X, expresado como:**

ε [X = expresión] R donde:

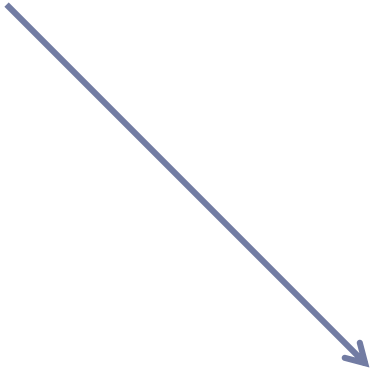
expresión puede contener atributos y constantes, obteniendo un valor para cada tupla, el cual se incluye como un nuevo atributo de R denominado X

ε - operador unario

- **Calcular el valor de la potencia por 10 para cada tupla de C5**

  Se expresa como: T4 = ε [pt = potencia*10] C5

C5	año	marca	potencia	color
	'2005'	'ford'	1.8	'verde'
	'2003'	'toyota'	1.6	'azul'
	'2003'	'ford'	1.6	'verde'
	'2005'	'toyota'	1.8	'azul'



T4	año	marca	potencia	color	pt
	'2005'	'ford'	1.8	'verde'	18
	'2003'	'toyota'	1.6	'azul'	16
	'2003'	'ford'	1.6	'verde'	16
	'2005'	'toyota'	1.8	'azul'	18

UNIVERSIDAD DE LOS ANDES

f - operador unario

- **Funciones: max, min, avg, sum, count; para encontrar el máximo, el mínimo, el promedio, la suma acumulada y contar tuplas, respectivamente**
  - Calcula la función escogida sobre particiones de R o sobre R completa, produciendo como resultado un esquema igual al de R unido a X
- **Se expresa como: f [X = $\wp$(A); B] R donde**
  - B es opcional y define el conjunto de atributos que divide R en particiones y cada grupo en la partición contiene todas las tuplas con el mismo valor de B
  - Si se omite B NO se realiza el agrupamiento y la función se aplica a R completa, obteniendo un único valor de X que se incluye en cada tupla de R
- **función que aplicada a los valores contenidos en la proyección de cada partición sobre A, da un valor para cada partición que será el nuevo atributo X para cada tupla de la partición**

f - operador unario

Ejemplo: calcular la potencia máxima de C6

Esto es: T5 = f [maxPotencia = max(potencia)] C6

C6	año	potencia
	'2005'	1.8
	'2003'	1.6

T5	año	potencia	maxPotencia
	'2005'	1.8	1.8
	'2003'	1.6	

Ejemplo: calcular la potencia máxima de C5 por marca

Esto es: T6 = f [maxPot = max(potencia); marca] C5

T6	año	marca	potencia	color	maxPot
	'2005'	'ford'	1.8	'verde'	1.8
	'2003'	'ford'	1.6	'verde'	
	'2003'	'toyota'	1.6	'azul'	1.8
	'2005'	'toyota'	1.8	'azul'	

Inserción

- **Inserción de una nueva tupla en la base de datos, donde el esquema de la nueva tupla debe coincidir con el esquema de R**
- **Expresado como:** E_{ins} **(listaDeValores) R**

Ejemplo: Insertar una nueva tupla en C de un carro con placa SQQ522, marca fiat y color negro

- E_{ins} **= ('SQQ522', 'fiat', 'negro') C**

C	placa	marca	color
	'LAB384'	'ford'	'verde'
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'

ins

C	placa	marca	color
	'LAB384'	'ford'	'verde'
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'
	'SQQ522'	'fiat'	'negro'

Descomposición que preserva las DFs

Eliminación de una o más tuplas en la base de datos, donde el esquema debe coincidir con el esquema de R

Expresado como: $E_{del} = R[Q]$

Ejemplo: Eliminar los carros de color verde de C

$E_{del} = C\ [\ color = \text{'verde'}\ ]$

C	placa	marca	color
	'LAB384'	'ford'	'verde'
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'
	'SQQ522'	'fiat'	'negro'

del

C	placa	marca	color
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'
	'SQQ522'	'fiat'	'negro'

Actualización

- **Actualización de una o más tuplas en la base de datos, donde el esquema es el mismo de R unido a los nuevos valores de los atributos actualizados**

Expresado como: E_{upd} = Q' R[atributo=valor]

Ejemplo: Acaban de pintar el carro 'SQQ522' de blanco

$E_{upd} = \varepsilon$ [color='blanco'] C [placa= 'SQQ522']

C	placa	marca	color
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'
	'SQQ522'	'fiat'	'negro'

upd

C	placa	marca	color
	'LAM112'	'toyota'	'azul'
	'LGR889'	'toyota'	'azul'
	'XSG230'	'ford'	'gris'
	'SQQ522'	'fiat'	'blanco'

Composición de operaciones

- **Se encadenan las operaciones sobre las relaciones, haciendo posible responder la mayoría de las consultas hechas a la BDR**

  Ejemplo: ¿ Cuántos carros posee cada persona?

P = ∏nombre (*f*[nro = count(placa); cedula] C4)

C4	cedula	nombre	apellido	placa	marca	color
	'V4567345'	'Pedro'	'Diaz'	'LAM112'	'toyota'	'azul'
	'V4567345'	'Pedro'	'Diaz'	'LGR889'	'toyota'	'azul'
	'V8234125'	'Juana'	'Perez'	'LAM112'	'toyota'	'azul'
	'V8234125'	'Juana'	'Perez'	'LGR889'	'toyota'	'azul'

P	nombre	nro
	'Pedro'	2
	'Juana'	2

UNIVERSIDAD DE LOS ANDES

Autoevaluación

1. ¿Qué se entiende algebra relacional?
2. ¿Cuáles son las operaciones básicas del algebra relacional?
3. ¿Cuáles son los criterios de selección?
4. ¿ Cuáles son las operaciones adicionales del algebra relacional?
5. ¿Cuáles son los casos de productos?
6. ¿Cuáles son las operaciones de la extensión del algebra relacional?
7. ¿Cómo se componen las operaciones del algebra relacional?

UNIVERSIDAD DE LOS ANDES

Ejercicio

Esquema de la base de datos:

Dpto(#dpto, presupDpto, jefeDpto)

Emp(#emp, #pro, telEmp)

Pro(#pro, #dpto, prePro)

Ofic(#ofi, #dpto, area)

Tlfs(#tel, #ofi)

Responda lo siguiente en algebra relacional:

1. **¿Cuál es el jefe del departamento 14 y cuál es su número de oficina?**
2. **¿Cuáles son los proyectos cuyo presupuesto es mayor que 500.000,00 y cuáles son sus empleados asignados?**
3. **¿Cuáles son las oficinas con área menor a 50 y cuáles son sus teléfonos?**
4. **¿Cuál es el número de teléfono del empleado 756, en que proyecto y en que departamento trabaja?**
5. **¿Cuáles son los empleados, proyectos, oficinas y teléfonos del departamento 32?**
6. **¿Cuál es el área promedio de las oficinas del departamento 32?**
7. **Calcular los presupuestos de los departamentos en dólares al cambio de 1000 Bs/dólar**
8. **Inserte una nueva oficina para el departamento 32 con un área de 24.**
9. **Acaba de terminarse el proyecto 22, elimínelo junto con todos sus empleados**
10. **Acaba de ser recibida una donación anual para la compañía, por ello actualice los montos de los presupuestos de todos los departamentos para incrementarse en 50.000,00**

UNIVERSIDAD DE LOS ANDES

Lenguaje estructurado de consulta SQL

- **Versión comercial del lenguaje SEQUEL creado por IBM para el sistema R bajo DOS y VM**
- **Lenguaje estándar de bases de datos**
- **Lenguaje declarativo, interactivo, de programación, de definición de esquemas y de comunicación con servidores a través de redes LAN, MAN y WAN**
- **En su primera versión, el SQL-89 se tienen tres partes:**
  - Lenguaje de definición de datos (LDD): contiene todas las instrucciones para definir el esquema de una BD, como son: ***create***, ***alter*** y ***drop***
  - Lenguaje de manipulación de datos (LMD): tiene las instrucciones de manejo de las tablas como son: ***select***, ***insert***, ***delete*** y ***update***, y para control de concurrencia como: ***commit*** y ***rollback***
  - Lenguaje de control de datos (LCD): tiene aquellas para dar y revocar permisos de acceso a los datos de la BD, como son: ***grant*** y ***revoke***

Lenguaje estructurado de consulta SQL

- **Las instrucciones pueden ir embebidas en programas escritos en otros lenguajes de programación, como: Cobol, Fortran, Pascal y PL/I**
- **La segunda versión del lenguaje fue el SQL-92 o SQL2 donde se incluyó:**
  - el uso de agentes de software, nuevos tipos básicos de datos como: Date, Time, Timestamp, BLOB, Varchar
  - se establecen conexiones cliente-servidor en sesiones concurrentes
  - se tiene SQL dinámico
  - mayor granularidad a nivel de transacciones
  - se tienen nuevas versiones de la operación producto o conjunción
  - se usa un catálogo estandarizado, se manejan códigos de error estandarizados
  - se utilizan nuevos lenguajes de programación como: C, ada y mumps

Lenguaje estructurado de consulta SQL

- **La última versión SQL3 amplía o extiende el SQL2 para contener:**
  - tipos abstractos de datos definidos por el diseñador de la BD
  - manejo de roles de usuarios
  - consultas recursivas
  - disparadores o *triggers*
  - procedimientos almacenados
  - encadenamiento tardío
  - manejo de interoperabilidad a través de un API u ODBC para acceso a bases de datos basado en el estándar SAG (*SQL Access Group*)
  - manejo de transacciones anidadas

Lenguaje de definición de datos de SQL3

- **Esquema conceptual:**
- **Para crear y destruir objetos de la BD:**
  - create y drop
- **Los objetos de la BD en DB2 son:**
  - schema, table, index, view, alias, distinct type, function, trigger
- **Los objetos en Sybase son:**
  - table, index, view
- **Existen otros objetos que dependen del SMBDR que se tenga, por lo cual es necesario consultar el manual de usuario del SMBDR**
- **Para alterar objetos de la BD se utiliza alter al objeto table**

UNIVERSIDAD DE LOS ANDES

Definiciones SQL3

- **Atómico: Incapaz de ser dividido**
- **Unidad de compilación: segmento de código ejecutable que puede contener uno o más subprogramas**
- **Tipo de dato: conjunto de valores representables**
- **Identificador: medio por el cual se identifica algo**
- **Identificar: referenciar algo sin ambigüedad**
- **Instancia: representación física de un valor**
- **Valor nulo: valor especial utilizado para representar la ausencia de valor**
- **Persistencia: existencia indefinida hasta su eliminación deliberada. Acciones en cascada y referenciales son deliberadas**
- **Secuencia: colección ordenada de objetos no necesariamente distintos**

UNIVERSIDAD DE LOS ANDES

Definiciones SQL3

- **Sesión SQL: contexto en el cual un solo usuario, con un solo agente SQL ejecuta una secuencia de instrucciones SQL con una solo conexión SQL**
- **Conexión SQL: asociación entre un cliente SQL y un servidor SQL**
- **Ambiente SQL: contexto en el cual los datos SQL existen y las instrucciones SQL se pueden ejecutar**
- **Tabla: una colección desordenada de filas que contienen una colección ordenada de columnas, donde cada columna tiene un nombre y un tipo de dato asociado. Cada fila tiene en cada columna exactamente un valor definido en el tipo de dato de la columna.**
- **Fila: una secuencia de (campo, valor), donde el tipo de dato del valor está especificado en el tipo de fila**
- **Propiedad: un atributo, relación o cualidad de un objeto**

UNIVERSIDAD DE LOS ANDES

Definiciones SQL3

- **Agente SQL: hace que se ejecuten las instrucciones SQL**
- **Implementación SQL: procesador que ejecuta las instrucciones SQL requeridas por el agente SQL**
- **Cliente SQL: procesador percibido por el agente SQL como parte de una implementación SQL**
- **Servidor SQL: procesador percibido por el agente SQL como parte de una implementación SQL que maneja datos SQL (maneja las sesiones SQL y ejecuta instrucciones SQL)**
- **Identificador de usuario: representa al usuario**
- **Catálogo: nombre de una colección de esquemas SQL, descriptores de servidores externos y descriptores de datos externos en una instrucción SQL.**
- **Esquema SQL: colección de descriptores con nombre**
- **Servidor externo: procesador que no forma parte de la implementación SQL**

UNIVERSIDAD DE LOS ANDES

Definiciones SQL3

- **Tabla base: tabla en el catálogo**
- **Consulta (*query*): operación que referencia 0 o más tablas base y regresa una tabla derivada temporal que contiene el resultado de la consulta**
- **Vista (esquema externo): es una consulta con nombre, que puede ser invocada por su nombre y cuyo resultado es una tabla vista**
- **Valor: cada valor pertenece a un tipo de dato**
- **Tipo atómico: es un tipo de dato cuyos valores no están compuestos de valores de otros tipos de datos**
- **Tipo compuesto: es un tipo de dato donde sus valores están compuestos de 0 o más valores cada uno declarado en un tipo de dato**
- **Valor nulo: valor especial contenido en cada tipo de dato y a veces indicado con la palabra NULL**

UNIVERSIDAD DE LOS ANDES

Tipos de datos

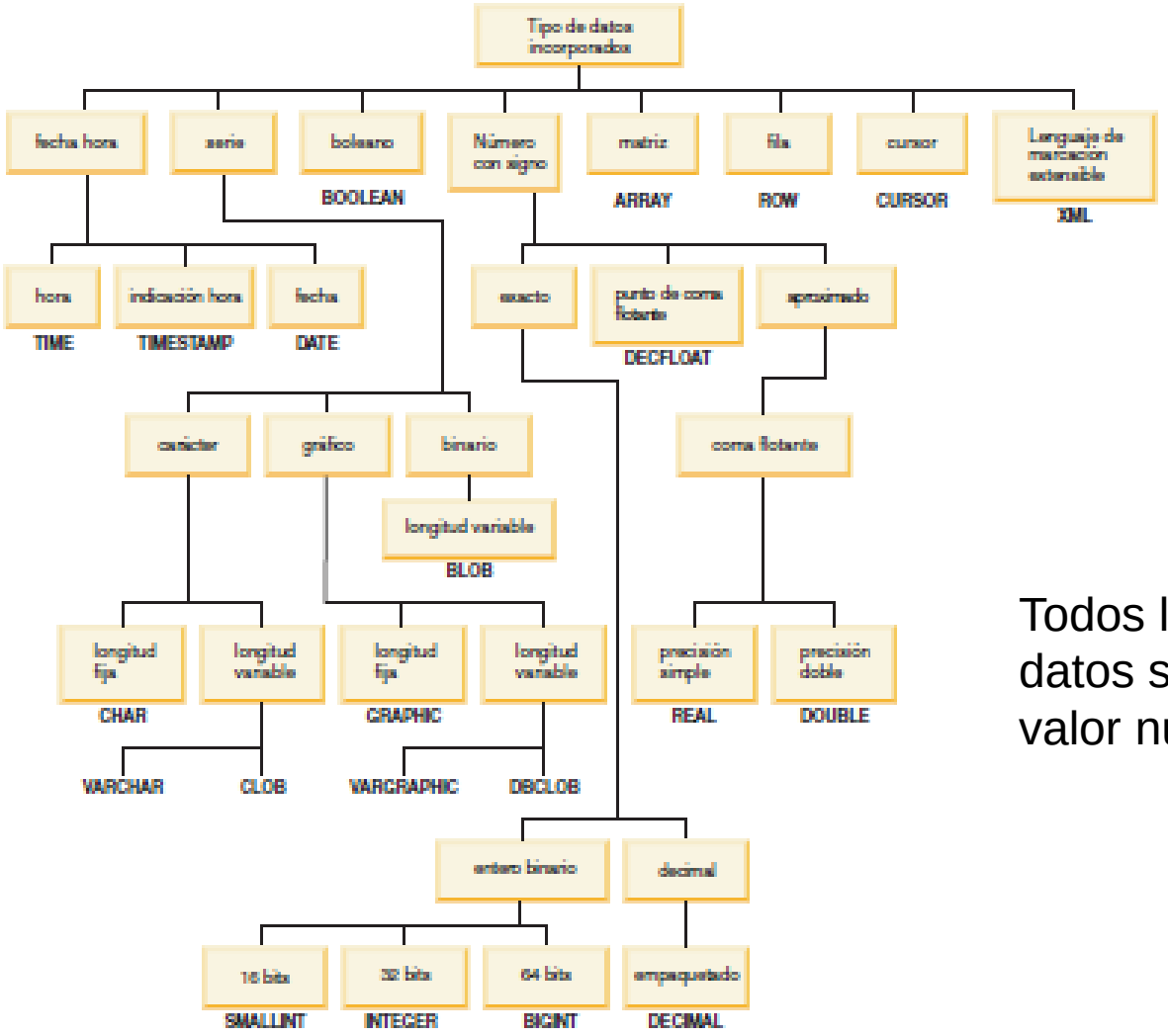
- **Numéricos:**
  - Exactos: enteros o tipos con precisión y escala especificada
  - Aproximados o punto flotante
- **Cadenas de caracteres o bits: longitud fija o variable**
- **Lógicos o booleanos**
- **Fecha, Hora y FechaHora: con año entre 0001 y 9999, calendario gregoriano y escogiendo la zona**
- **Intervalo temporal**
- **XML**
- **Referencias a tipos**
- **Colecciones: arreglos (*array*) y bolsas (*multiset*)**
- **Fila: secuencia de 1 o más (nombreCampo, tipoDeDato)**
- **Campo: (nombreCampo, tipoDeDato)**

Tipos de datos de PostgreSQL v9.4

Name	Aliases	Description
bigint	int8	signed eight-byte integer
bigserial	serial8	autoincrementing eight-byte integer
bit [(n)]		fixed-length bit string
bit varying [(n)]	varbit	variable-length bit string
boolean	bool	logical Boolean (true/false)
box		rectangular box on a plane
bytea		binary data ("byte array")
character varying [(n)]	varchar [(n)]	variable-length character string
character [(n)]	char [(n)]	fixed-length character string
cidr		IPv4 or IPv6 network address
circle		circle on a plane
date		calendar date (year, month, day)
double precision	float8	double precision floating-point number (8 bytes)
inet		IPv4 or IPv6 host address
integer	int, int4	signed four-byte integer
interval [fields] [(p)]		time span
line		infinite line on a plane
lseg		line segment on a plane
macaddr		MAC (Media Access Control) address
money		currency amount
numeric [(p, s)]	decimal [(p, s)]	exact numeric of selectable precision
path		geometric path on a plane
point		geometric point on a plane
polygon		closed geometric path on a plane

json	textual	JSON data
jsonb	Binary	JSON data, decomposed
real	float4	single precision floating-point number (4 bytes)
smallint	int2	signed two-byte integer
smallserial	serial2	autoincrementing two-byte integer
serial	serial4	autoincrementing four-byte integer
text		variable-length character string
time [(p)] [without time zone]		time of day (no time zone)
timestamp [(p)] with time zone	timestamptz	date and time, including time zone
uuid		universally unique identifier
xml		XML data

Tipos de datos de DB2 v9.7



Todos los tipos de datos soportan el valor nulo (NULL)

UNIVERSIDAD DE LOS ANDES

SQL3

- **Lugar (*Site*): lugar donde se encuentra una instancia de un valor de un tipo de dato**
- **Asignación: operación que reemplaza una instancia en un lugar (*target*) con una nueva instancia del valor (*source*)**
- **Cada esquema tiene un nombre y sus clases son:**
  - Tablas base y vistas
  - Dominios y tipos definidos por los usuarios
  - Restricciones de tablas, de dominios y aserciones
  - Módulos servidores SQL y Gatillos o *triggers*
  - Rutinas invocadas, conjuntos de caracteres, operaciones para ordenar caracteres (*collations*), traductores y generadores de secuencias

UNIVERSIDAD DE LOS ANDES

SQL3

- **Dominio: es un objeto definido por el usuario que consiste de un tipo de dato con cero o más restricciones**
- **Tipo estructurado: tipo definido por el usuario que puede ser subtipo de otro tipo estructurado**
- **Atributo: componente con nombre de un tipo estructurado**
- **Creación de tablas: Crea una tabla <nomTabla> cuyo esquema es el conjunto de atributos con (<nomAt) <tipo>)**

CREATE TABLE <nomTabla> (<nomAt> <tipo> [NOT NULL | WITH DEFAULT],…);

```
CREATE TABLE     Persona
      (idPersona     SMALLINT        NOT NULL,
       nombre        VARCHAR(16) WITH DEFAULT 'Desconocido',
       apellido      VARCHAR(16) WITH DEFAULT 'Desconocido'
       fechaNac      DATE);
```

Ejemplo de sintaxis de PostgreSQL

```
CREATE [ [ GLOBAL | LOCAL ] { TEMPORARY | TEMP } | UNLOGGED ] TABLE [ IF NOT EXISTS ] table_
  { column_name data_type [ COLLATE collation ] [ column_constraint [ ... ] ]
    | table_constraint
    | LIKE source_table [ like_option ... ] }
    [, ... ]
] )
[ INHERITS ( parent_table [, ... ] ) ]
[ WITH ( storage_parameter [= value] [, ... ] ) | WITH OIDS | WITHOUT OIDS ]
[ ON COMMIT { PRESERVE ROWS | DELETE ROWS | DROP } ]
[ TABLESPACE tablespace_name ]
```

where *column_constraint* is:

```
[ CONSTRAINT constraint_name ]
{ NOT NULL |
  NULL |
  CHECK ( expression ) [ NO INHERIT ] |
  DEFAULT default_expr |
  UNIQUE index_parameters |
  PRIMARY KEY index_parameters |
  REFERENCES reftable [ ( refcolumn ) ] [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ]
    [ ON DELETE action ] [ ON UPDATE action ] }
[ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
```

and *table_constraint* is:

```
[ CONSTRAINT constraint_name ]
{ CHECK ( expression ) [ NO INHERIT ] |
  UNIQUE ( column_name [, ... ] ) index_parameters |
  PRIMARY KEY ( column_name [, ... ] ) index_parameters |
  EXCLUDE [ USING index_method ] ( exclude_element WITH operator [, ... ] ) index_parameters [ W
  FOREIGN KEY ( column_name [, ... ] ) REFERENCES reftable [ ( refcolumn [, ... ] ) ]
    [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ] [ ON DELETE action ] [ ON UPDATE action ]
```

UNIVERSIDAD DE LOS ANDES

Ejemplo

```
CREATE TABLE Carro
(placa      char(8) NOT NULL CONSTRAINT      agrupadoXplaca      PRIMARY KEY,
modelo CHAR(16) NOT NULL FOREIGN KEY (modelo) REFERENCES ModelosXmarca,
color colores);
CREATE TABLE ModelosXmarca (
            modelo CHAR(16) NOT NULL,
            marca CHAR(16),
            CONSTRAINT agrupadoXmodelo PRIMARY KEY (modelo))
```

diferible

- **Restricciones de tabla: (*deferrable*, *not deferrable*)**
  - única (*unique*) no hay dos filas con el mismo valor en esa columna,
  - clave primaria (*primary key*) única y no hay nulos,
  - clave foránea (*foreign key*) referencia a la clave primaria de otra tabla, aceptando nulos y
  - de verificación (*check*) especifica una condición de búsqueda

Mantenimiento de la integridad referencial

- **Políticas:**
  - Devolver actualizaciones que la violan (por omisión)
  - Actualizaciones en cascada (CASCADE)
  - Colocar en nulo (SET NULL)

```
create table Ventas
(nroVen        CHAR(8) UNIQUE,
 fechaVen      DATE NOT NULL,
 nomCli        VARCHAR(32),
 nroProVen CHAR(6) REFERENCES Producto(nroPro)
                ON DELETE SET NULL
                ON UPDATE CASCADE,
 cantVen int
);
```

No acepta nulos

Problema con referencias huérfanas

Restricciones sobre atributos en SQL

- **Condiciones SQL que se declaran con CHECK**

```
create table Ventas
(nroVen CHAR(8) UNIQUE,
 fechaVen DATE NOT NULL,
 nomCli VARCHAR(32),
 nroProVen CHAR(6) REFERENCES Producto(nroPro)
                ON DELETE SET NULL
                ON UPDATE CASCADE,
 cantVen INT CHECK (cantVen >= 0)
);
```

- **Si la condición refiere a otras relaciones o atributos de otras relaciones, debe declararse con una consulta**

Restricciones de dominio en SQL

- **Creación de dominios con restricciones**

```
CREATE DOMAIN colores CHAR(8)
   CHECK (VALUE IN ('verde','negro','rojo','azul'));

CREATE TABLE Producto
(nroPro CHAR(6) PRIMARY KEY,
 nombrePro VARCHAR(64),
 cantidad int,
 color colores
);
```

```
DROP DOMAIN [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```

CREATE TYPE colores AS ENUM ('verde', 'rojo', 'gris');

Restricciones globales en SQL

- **Para las tuplas de una relación: con la palabra CHECK**

```
create table Ventas
(nroVen CHAR(8) UNIQUE,
 fechaVen DATE NOT NULL,
 nomCli VARCHAR(32),
 nroProVen CHAR(6) REFERENCES Producto(nroPro)
                ON DELETE SET NULL
                ON UPDATE CASCADE,
 cantVen INT CHECK (cantVen >= 0),
 CHECK (fechaVen >= TODAY())
);
```

UNIVERSIDAD DE LOS ANDES

Restricciones globales en SQL

- **Aserciones: permites asegurar condiciones globales en toda la base de datos durante el tiempo de vida de la misma**
- **Aserción: restricción de verificación (solo es falsa si no es cierta o desconocida)**
- **Creación: con la sentencia**

ASSERTION nomAser CHECK (condición)

```
CREATE ASSERTION cantidades CHECK (cantVen <=
    (SELECT cantExistencia
     FROM Producto P, Ventas V
     WHERE p.nroPro = V.nroProVen));
```

UNIVERSIDAD DE LOS ANDES

Resumen

Tomado de Ullman y Widom, 1997

Tipo de restricción	Donde se declara	Cuando se activa	Garantía de cumplirse
CHECK en los atributos	Junto con el atributo	Cuando se inserte o se modifique el atributo	No, si hay subconsultas
CHECK en las tuplas	Elemento del esquema de la BD	Cuando se inserte o se modifique una tupla	No, si hay subconsultas
Aserciones	Elemento del esquema de la BD	Cuando cambie cualquier relación involucrada	Siempre

Modificaciones de las restricciones en SQL

- **Colocar nombre a cada restricción para luego poderlas cambiar o eliminar**
- **Sintaxis:**

  CONSTRAINT cantVendida CHECK (cantVen >= 0);

  CONSTRAINT fechaDeVenta CHECK (fechaVen >= TODAY());
- **Cambios: sólo anexar o quitar restricciones**

  ALTER TABLE Ventas DROP CONSTRAINT fechaDeVenta;

  ALTER TABLE Producto ADD CONSTRAINT cantidadValida CHECK (cantidad >= 0);

Modificaciones de dominios y tablas en SQL

- **Los dominios se pueden cambiar, anexar o quitar**
- **Sintaxis:**

  ALTER DOMAIN nomDominio ADD CONSTRAINT nomRest CHECK (condición);

  ALTER DOMAIN nomDominio DROP CONSTRAINT nomRest;

  DROP DOMAIN nomDominio;
- **Las aserciones sólo se pueden quitar**

  DROP ASSERTION nomAser;

```
ALTER TABLE [ IF EXISTS ] [ ONLY ] name [ * ]
    action [, ... ]
ALTER TABLE [ IF EXISTS ] [ ONLY ] name [ * ]
    RENAME [ COLUMN ] column_name TO new_column_name
ALTER TABLE [ IF EXISTS ] [ ONLY ] name [ * ]
    RENAME CONSTRAINT constraint_name TO new_constraint_name
ALTER TABLE [ IF EXISTS ] name
    RENAME TO new_name
ALTER TABLE [ IF EXISTS ] name
    SET SCHEMA new_schema

where action is one of:

    ADD [ COLUMN ] column_name data_type [ COLLATE collation ] [ column_constraint [ ... ] ]
    DROP [ COLUMN ] [ IF EXISTS ] column_name [ RESTRICT | CASCADE ]
```

UNIVERSIDAD DE LOS ANDES

Gatillos para tablas

- **Asociados a las tablas base, contienen:**
  - un evento (INSERT, DELETE, UPDATE),
  - una acción temporal (BEFORE, AFTER) y
  - 1 o más acciones disparadas (instrucción SQL o BEGIN ATOMIC seguido de 1 o más instrucciones SQL con ; END)

```
CREATE [ CONSTRAINT ] TRIGGER name { BEFORE | AFTER | INSTEAD OF } { event [ OR ... ] }
    ON table_name
    [ FROM referenced_table_name ]
    { NOT DEFERRABLE | [ DEFERRABLE ] { INITIALLY IMMEDIATE | INITIALLY DEFERRED } }
    [ FOR [ EACH ] { ROW | STATEMENT } ]
    [ WHEN ( condition ) ]
    EXECUTE PROCEDURE function_name ( arguments )

where event can be one of:

    INSERT
    UPDATE [ OF column_name [, ... ] ]
    DELETE
    TRUNCATE

DROP TRIGGER [ IF EXISTS ] name ON table_name [ CASCADE | RESTRICT ]
```

Creación de vistas o esquemas externos

- **Vistas: consulta con nombre, su valor es el resultado de la evaluación de la consulta**
- **Soportan los esquemas externos o vistas externas**

```
CREATE [ OR REPLACE ] [ TEMP | TEMPORARY ] VIEW name [ ( column_name [, ...] ) ]
    [ WITH ( view_option_name [= view_option_value] [, ... ] ) ]
    AS query

DROP VIEW [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```

UNIVERSIDAD DE LOS ANDES

Índices de tablas

- **Mejoran el desempeño de la BD y forman parte del esquema físico de la misma**
- **Un SMBDR siempre indexa la tabla por la clave primaria declarada**
- **La tabla siempre agrupará y ordenará sus tuplas según la clave primaria**
- **El diseñador puede declarar otros índices sobre otros atributos**

  CREATE [UNIQUE] INDEX nomIndice ON nomTabla(at1[,at2,…]);

  CREATE INDEX indXproducto ON Producto(nombrePro);

  DROP INDEX nomIndice;

```
DROP INDEX [ CONCURRENTLY ] [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```

UNIVERSIDAD DE LOS ANDES

Autoevaluación

1. ¿Qué es el lenguaje SQL y cuáles versiones tiene?
2. ¿Cuáles son las instrucciones de definición de datos del SQL?
3. ¿Qué se entiende por persistencia, catálogo, vista y tabla base en SQL3?
4. ¿Cuáles son los tipos de datos del SQL?
5. ¿Cuál es la diferencia entre dominio y tipo estructurado en SQL3?
6. ¿Cómo se crea y modifica una tabla en SQL3?
7. ¿Cómo se crea y modifica un dominio en SQL3?
8. ¿Cómo se crea y modifica una restricción en SQL3?

UNIVERSIDAD DE LOS ANDES

Ejercicios

1. Para el ejercicio anterior, escriba el programa en sql que crea la base de datos con los esquemas dados
2. Agregue al programa anterior un esquema externo según la consulta 5 del ejercicio
3. Agregue los índices adicionales para las consultas del ejercicio anterior
4. Verifique que se cumplan todas las restricciones de integridad referencial del esquema
5. Proponga al menos 2 gatillos en SQL para el esquema obtenido en 1.