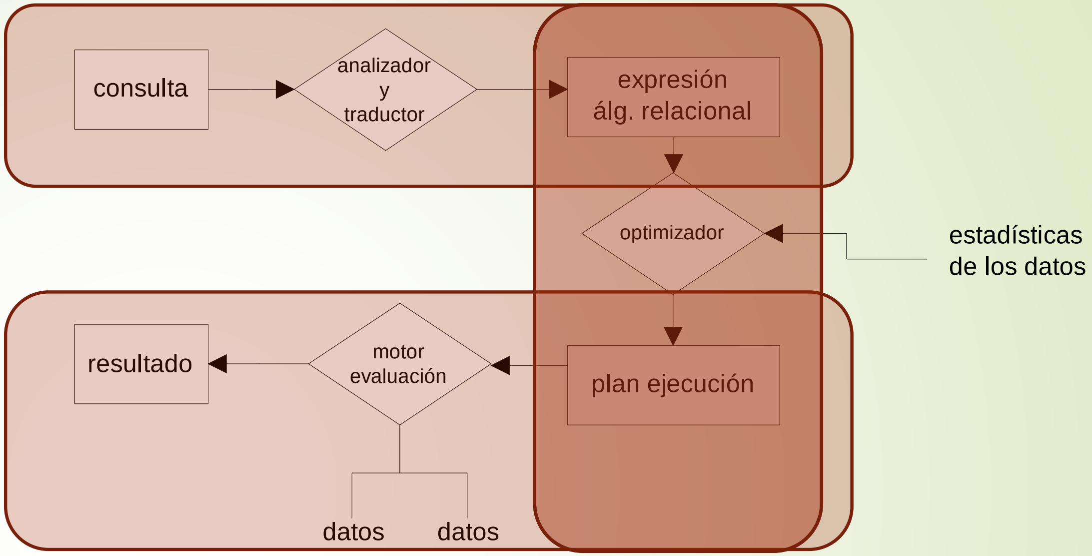
Tema.- Procesamiento y Optimización de consultas

Fundamentos de Bases de Datos (5º edic.).

A. Silberschatz. McGraw-Hill [caps. 13-14]

Procesamiento de consultas

- Pasos:



Análisis y traducción:

- Comprobación de sintaxis, verificación de nombres
- Construcción del árbol de consulta
- Transformación a una expresión en álgebra relacional extendido

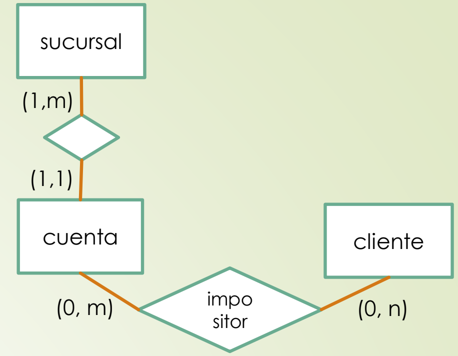
Optimización:

- Indicar cómo evaluar la expresión en álgebra relacional (algoritmo(s) a utilizar, índice(s) a aplicar,…) ⇒ **primitivas de evaluación**
- Determinar el **plan de ejecución/evaluación** (= secuencia de primitivas de evaluación) que minimice el coste de ejecución de la consulta

Evaluación:

- El motor de consultas ejecuta un plan de evaluación

Optimización de consultas



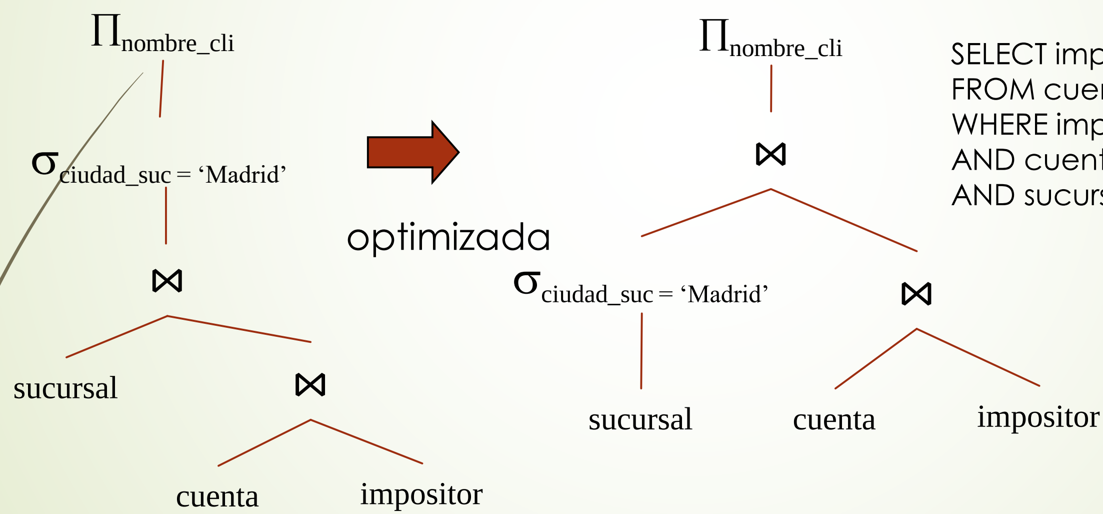
Nombre de clientes con cuenta en una sucursal de Madrid

$$\Pi_{nombre\_cli}\ (\sigma_{ciudad\_suc\ =\ \text{'Madrid'}}\ (sucursal \bowtie (cuenta \bowtie impositor)))$$

optimizada

$$\Pi_{nombre\_cli}\ (\sigma_{ciudad\_suc\ =\ \text{'Madrid'}}\ (sucursal) \bowtie (cuenta \bowtie impositor))$$

sucursal (nombre_suc, ciudad_suc, activos)
cuenta (num_cta, nombre_suc, saldo)
impositor (nombre_cli, num_cta)
cliente (nombre_cli, edad, ciudad_cli, sueldo)




```
SELECT impositor.nombre_cli
FROM cuenta, impositor, sucursal
WHERE impositor.num_cta = cuenta.num_cta
AND cuenta.nombre_suc = sucursal.nombre_suc
AND sucursal.ciudad_suc = "Madrid"
```

El optimizador diseña un plan de evaluación de consultas "óptimo" generando planes alternativos

1. **Heurística**: genera expresiones equivalentes mediante ***reglas de equivalencia***
2. **Coste**: estima el coste de cada plan (utilizando inf. estadística de tablas, índices, etc.)

Transformación de expresiones relacionales

- Las consultas se pueden expresar de varias maneras, con costes de evaluación diferentes
  - En vez de tomar la expresión relacional original se consideran expresiones alternativas equivalentes.

- 2 expresiones en álgebra relacional son **equivalentes** si generan el mismo conjunto de tuplas

- **Reglas de equivalencia** => 2 expresiones son equivalentes si se puede sustituir la primera expresión por la segunda, o viceversa

Reglas de equivalencia

- **CASCADA DE σ:** las operaciones de selección conjuntivas pueden dividirse en una secuencia de selecciones individuales => $\sigma_{\theta1 \wedge \theta2}(E) = \sigma_{\theta1}(\sigma_{\theta2}(E))$

  ej: $\sigma_{saldo>10 \wedge nombre\_suc=\text{“Madrid”}}(cuenta) = \sigma_{saldo>10}(\sigma_{nombre\_suc=\text{“Madrid”}}(cuenta))$

- **CONMUTATIVIDAD DE σ:** las operaciones de selección son conmutativas => $\sigma_{\theta1}(\sigma_{\theta2}(E)) = \sigma_{\theta2}(\sigma_{\theta1}(E))$

  ej: $\sigma_{saldo>10}(\sigma_{nombre\_suc=\text{“Madrid”}}(cuenta)) = \sigma_{nombre\_suc = \text{“Madrid”}}(\sigma_{saldo>10}(cuenta))$

- **CASCADA DE ∏:** solo es necesaria la última operación de proyección, las demás pueden omitirse => $\prod_{L1}(\prod_{L2}(\ldots\prod_{Li}(E))\ldots)) = \prod_{L1}(E)$

  ej: $\Pi_{num\_cta}(\Pi_{num\_cta,\ nombre\_suc}(\Pi_{num\_cta,\ nombre\_suc,\ saldo}(cuenta))) = \Pi_{num\_cta}(cuenta)$

Reglas de equivalencia

- **Las selecciones pueden combinarse con productos cartesianos y con zeta-joins**:

$\sigma_\theta(E_1 \times E_2) = E_1 \bowtie_\theta E_2$ $\qquad$ $\sigma_{\theta 1}(E1 \bowtie_{\theta 2} E2) = E1 \bowtie_{\theta 1 \wedge \theta 2} E2$

ej: $\sigma_{c.nombre\_suc = s.nombre\_suc}$ (cuenta x sucursal) = cuenta $\bowtie_{c.nombre\_suc = s.nombre\_suc}$ sucursal

ej: $\sigma_{c.saldo>s.activos}$(cuenta $\bowtie_{c.nombre\_suc = s.nombre\_suc}$ sucursal) =

= cuenta $\bowtie_{c.nombre\_suc = s.nombre\_suc \wedge c.saldo>s.activos}$ sucursal

- **CONMUTATIVIDAD DE ⋈**: los zeta-join son conmutativos => $E1 \bowtie_\theta E2 = E2 \bowtie_\theta E1$

ej: cuenta $\bowtie_{c.nombre\_suc = s.nombre\_suc}$ sucursal = sucursal $\bowtie_{c.nombre\_suc = s.nombre\_suc}$ cuenta

- **ASOCIATIVIDAD DE ⋈**: el join natural es asociativo => $(E1 \bowtie E2) \bowtie E3 = E1 \bowtie (E2 \bowtie E3)$

ej: (cuenta ⋈ sucursal) ⋈ impositor = cuenta ⋈ (sucursal ⋈ impositor)

Los zeta-join son asociativos en el siguiente sentido:

$(E1 \bowtie_{\theta 1} E2) \bowtie_{\theta 2 \wedge \theta 3} E3 = E1 \bowtie_{\theta 1 \wedge \theta 3} (E2 \bowtie_{\theta 2} E3)$ donde $\theta 2$ implica solo atributos de E2 y E3

ej: (cuenta $\bowtie_{c.num\_cta=i.num\_cta}$ impositor) $\bowtie_{i.nombre\_cli=cl.nombre\_cli \wedge c.saldo > cl.sueldo}$ cliente=

= cuenta $\bowtie_{c.num\_cta=i.num\_cta \wedge c.saldo > cl.sueldo}$ (impositor $\bowtie_{i.nombre\_cli=cl.nombre\_cli}$ cliente)

Reglas de equivalencia

- **Desplazar σ hacia hojas en ⋈**: La operación selección se distribuye en el zeta-join bajo 2 condiciones:

1. Si la selección θ solo implica atributos de una de las expresiones:

   $\sigma_{\theta 1}(E1 \bowtie_{\theta} E2) = \sigma_{\theta 1}(E1) \bowtie_{\theta} E2$ donde θ1 solo implica atributos de E1

   ej: $\sigma_{saldo>100}(cuenta \bowtie sucursal) = \sigma_{saldo>100}(cuenta) \bowtie sucursal$

2. Si la selección θ1 solo implica atributos de E1 y θ2 solo implica atributos de E2:

   $\sigma_{\theta 1 \wedge \theta 2}(E1 \bowtie_{\theta} E2) = \sigma_{\theta 1}(E1) \bowtie_{\theta} \sigma_{\theta 2}(E2)$

   ej: $\sigma_{(saldo>100) \wedge (activos=200)}(cuenta \bowtie sucursal) = \sigma_{saldo>100}(cuenta) \bowtie \sigma_{activos=200}(sucursal)$

Reglas de equivalencia

- **Desplazar $\prod$ hacia las hojas en $\bowtie$:** La operación proyección se distribuye en el zeta-join bajo 2 condiciones:

1. $\prod_{L1\cup L2}(E1 \bowtie_{\theta} E2) = \prod_{L1}(E1) \bowtie_{\theta} \prod_{L2}(E2)$ siendo L1 atributos de E1, L2 atributos de E2 y donde θ implica solo atributos de $L1\cup L2$

ej: $\prod_{nombre\_suc,\ activos}(sucursal \bowtie cuenta) = \prod_{nombre\_suc,\ activos}(sucursal) \bowtie \prod_{nombre\_suc}(cuenta)$

2. $\prod_{L1 \cup L2}(E1 \bowtie_{\theta} E2) = \prod_{L1 \cup L2}(\prod_{L1 \cup L3}(E1) \bowtie_{\theta} \prod_{L2 \cup L4}(E2))$

   donde L1 son atributos de E1

   L2 son atributos de E2

   L3 son atributos de E1 que están en θ pero no en L1

   L4 son atributos de E2 que están en θ pero no en L2

   ej: $\prod_{activos,\ saldo}(sucursal \bowtie cuenta) =$

   $\prod_{activos,\ saldo}(\prod_{nombre\_suc,\ activos}(sucursal) \bowtie \prod_{nombre\_suc,\ saldo}(cuenta))$

Reglas de equivalencia

- Un conjunto de reglas es MÍNIMO si no se puede obtener ninguna regla a partir de la unión de otras
- Los optimizadores utilizan conjuntos mínimos de reglas de equivalencia
- Los optimizadores generan sistemáticamente expresiones equivalentes a la consulta dada
- Una buena ordenación de los joins reduce el tamaño de los resultados

Ej:

$$\sigma_{ciudad\_suc = 'Madrid'} (sucursal) \bowtie (cuenta \bowtie impositor)$$

Como es probable que:

1) (cuenta ⋈ impositor) de lugar a una relación muy grande (tantas tuplas como impositores existan)
2) el número de cuentas de las sucursales de Madrid es pequeño

sería mejor aplicar la regla de asociatividad del join, dando lugar a la expresión:

$$(\sigma_{ciudad\_suc = 'Madrid'} (sucursal) \bowtie cuenta) \bowtie impositor$$

Tipos de optimización: Heurística

- Mediante la aplicación de reglas de equivalencia, reordena los componentes del árbol de consultas inicial para intentar reducir el coste de la optimización
- Pasos a seguir:

1. **Realizar las operaciones de selección tan pronto como sea posible**

CASCADA DE σ => $\sigma_{\theta1\wedge\theta2}(E) = \sigma_{\theta1}(\sigma_{\theta2}(E))$

CONMUTATIVIDAD DE σ => $\sigma_{\theta1}(\sigma_{\theta2}(E)) = \sigma_{\theta2}(\sigma_{\theta1}(E))$

$\sigma_{\theta0}(E1 \bowtie_{\theta} E2) = \sigma_{\theta0}(E1) \bowtie_{\theta} E2$ donde $\theta0$ implica solo atributos de E1

$\sigma_{\theta0}(E1 X E2) = \sigma_{\theta0}(E1) X E2$ donde $\theta0$ implica solo atributos de E1

$\sigma_{\theta1\wedge\theta2}(E1 \bowtie_{\theta} E2) = \sigma_{\theta1}(E1) \bowtie_{\theta} \sigma_{\theta2}(E2)$ donde $\theta1$ implica solo atributos de E1 y $\theta2$ atributos de E2

$\sigma_{\theta1\wedge\theta2}(E1 X E2) = \sigma_{\theta1}(E1) X \sigma_{\theta2}(E2)$ donde $\theta1$ implica solo atributos de E1 y $\theta2$ atributos de E2

Tipos de optimización: Heurística

2. Sustituir el producto cartesiano seguido de σ por $\bowtie$

$$\sigma_{\theta}(E_1 \times E_2) = E_1 \bowtie_{\theta} E_2$$

$$\sigma_{\theta 1}(E1 \bowtie_{\theta 2} E2) = E1 \bowtie_{\theta 1 \wedge \theta 2} E2$$

3. Determinar las operaciones σ y $\bowtie$ que producen menos tuplas, y ejecutarlas cuanto antes

CONMUTATIVIDAD DE σ $\Rightarrow \sigma_{\theta 1}(\sigma_{\theta 2}(E)) = \sigma_{\theta 2}(\sigma_{\theta 1}(E))$

ASOCIATIVIDAD DE $\bowtie$ $\Rightarrow (E1 \bowtie E2) \bowtie E3 = E1 \bowtie (E2 \bowtie E3)$

$$(E1 \bowtie_{\theta 1} E2) \bowtie_{\theta 2 \wedge \theta 3} E3 = E1 \bowtie_{\theta 1 \wedge \theta 3} (E2 \bowtie_{\theta 2} E3)$$

donde $\theta 2$ implica solo atributos de E2 y E3

4. Realizar las proyecciones tan pronto como sea posible

CASCADA DE Π $\Rightarrow \Pi_{L1}(\Pi_{L2}(\ldots\Pi_{Li}(E))\ldots)) = \Pi_{L1}(E)$

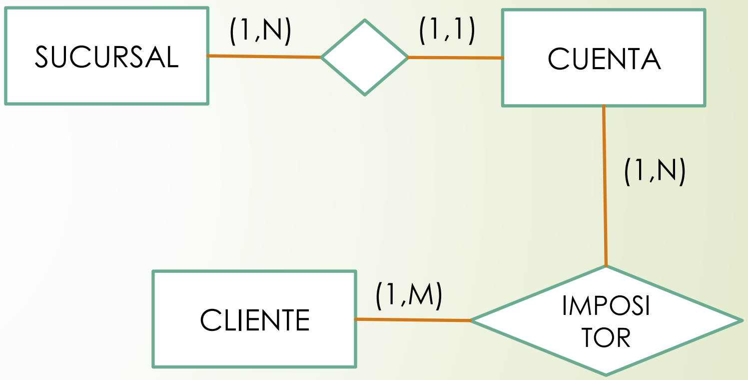
$$\Pi_{L1 \cup L2}(E1 \bowtie_{\theta} E2) = \Pi_{L1}(E1) \bowtie_{\theta} \Pi_{L2}(E2)$$

$$\Pi_{L1 \cup L2}(E1 \bowtie_{\theta} E2) = \Pi_{L1 \cup L2}(\Pi_{L1 \cup L3}(E1) \bowtie_{\theta} \Pi_{L2 \cup L4}(E2))$$

Ejemplo 1 uso reglas de equivalencia

- Representar mediante árboles las expresiones del álgebra relacional de la consulta siguiente y transformarla a una forma más eficiente. Enunciar las reglas de equivalencia utilizadas en cada uno de los pasos del proceso

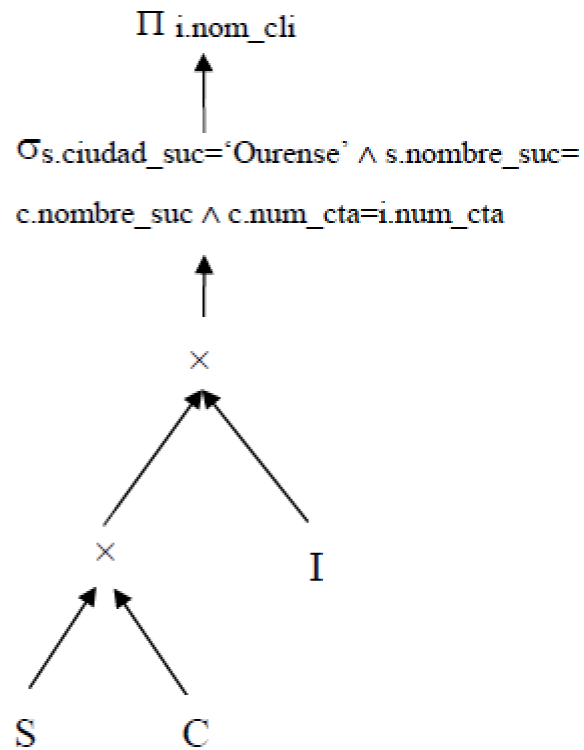
SUCURSAL (nombre_suc, ciudad_suc, activos)
CUENTA (num_cta, nombre_suc, saldo)
IMPOSITOR (nom_cli, num_cta)
CLIENTE (nom_cli, direccion, activos)



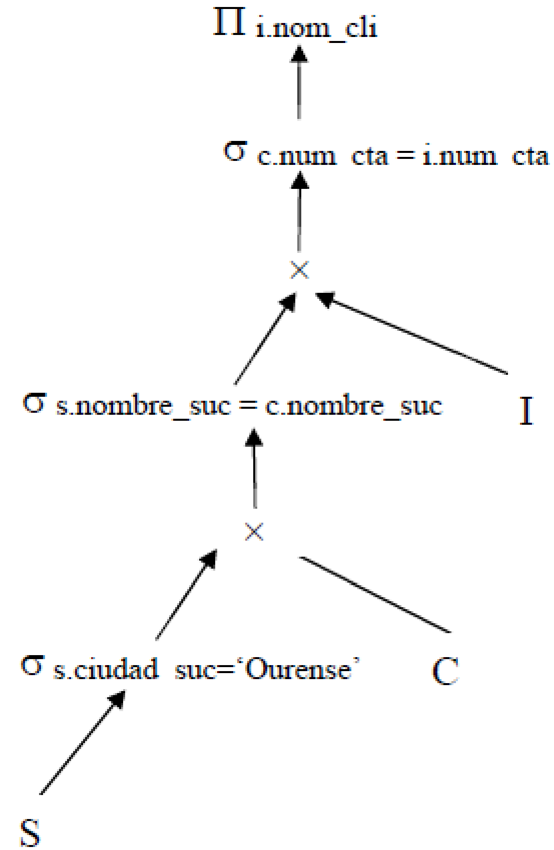
Nombre de clientes con cuenta en alguna sucursal de Ourense

```
SELECT   nom_cli
FROM     SUCURSAL s, CUENTA c, IMPOSITOR i
WHERE    s.ciudad_suc = 'Ourense' AND
         s.nombre_suc = c.nombre_suc AND
         c.num_cta = i.num_cta;
```

Ejemplo 1 uso reglas de equivalencia



Árbol inicial



1. DESPLAZAR σ HACIA HOJAS EN X:

CASCADA DE σ: $\sigma_{\theta1\wedge\theta2}(E) = \sigma_{\theta1}(\sigma_{\theta2}(E))$

CONMUTATIVIDAD DE σ: $\sigma_{\theta1}(\sigma_{\theta2}(E)) = \sigma_{\theta2}(\sigma_{\theta1}(E))$

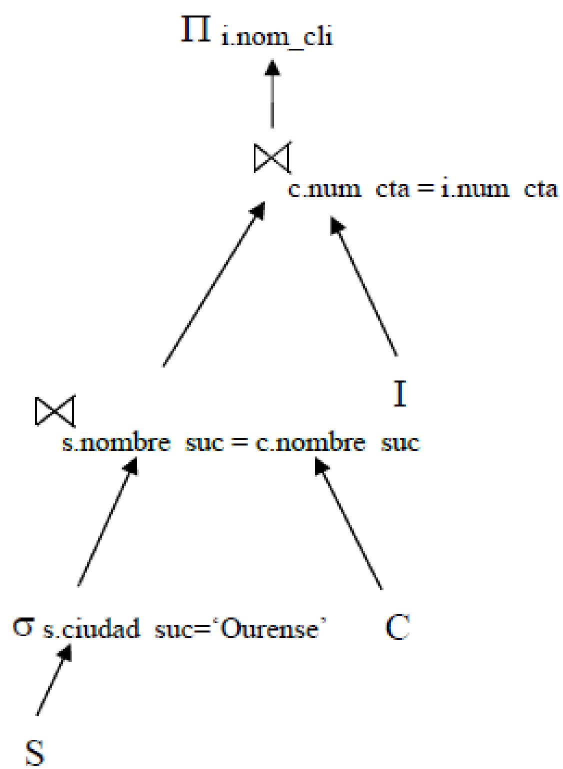
$\sigma_{\theta0}(E1 X E2) = \sigma_{\theta0}(E1) X E2$, θ0 implica solo atrib. de E1

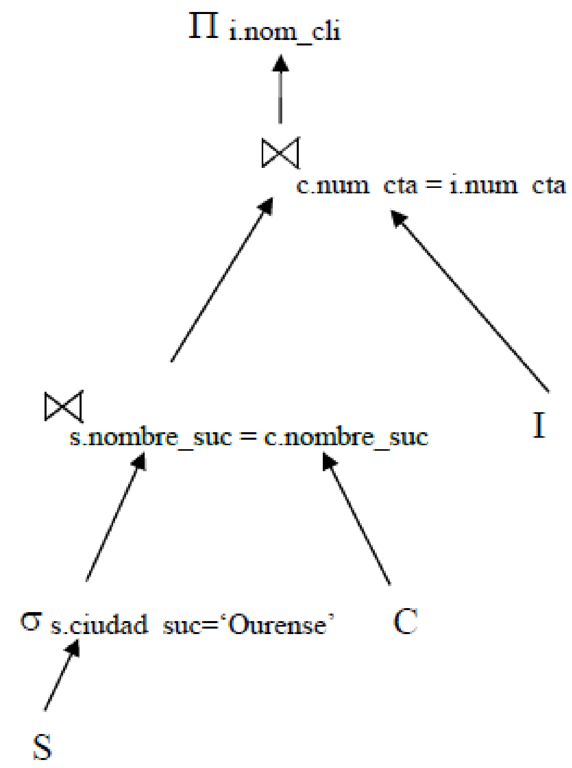
En este caso, no se utiliza $\sigma_{\theta1\wedge\theta2}(E1 X E2) = \sigma_{\theta1}(E1) X \sigma_{\theta2}(E2)$ donde θ1 implica solo atributos de E1 y θ2 atributos de E2

0. $\Pi_{i.nom\_cli}(\sigma_{s.ciudad\_suc='Ourense' \wedge s.nombre\_suc=c.nombre\_suc \wedge c.num\_cta=i.num\_cta}$ **(sucursal X cuenta X impositor))**

1. $\Pi_{i.nom\_cli}(\sigma_{c.num\_cta=i.num\_cta}(\sigma_{s.nombre\_suc=c.nombre\_suc}(\sigma_{ciudad\_suc='Ourense'}$ **(sucursal) X cuenta) X impositor))**

Ejemplo 1 uso reglas de equivalencia





2. Sustituir el producto cartesiano (X) seguido de σ por join:

$\sigma_\theta(E1 \; X \; E2) = E1 \bowtie_\theta E2$

3. Ejecutar antes las selecciones y los joins que producen menos tuplas

CONMUTATIVIDAD DE σ => $\sigma_{\theta 1}(\sigma_{\theta 2}(E)) = \sigma_{\theta 2}(\sigma_{\theta 1}(E))$

ASOCIATIVIDAD DE ⋈ =>

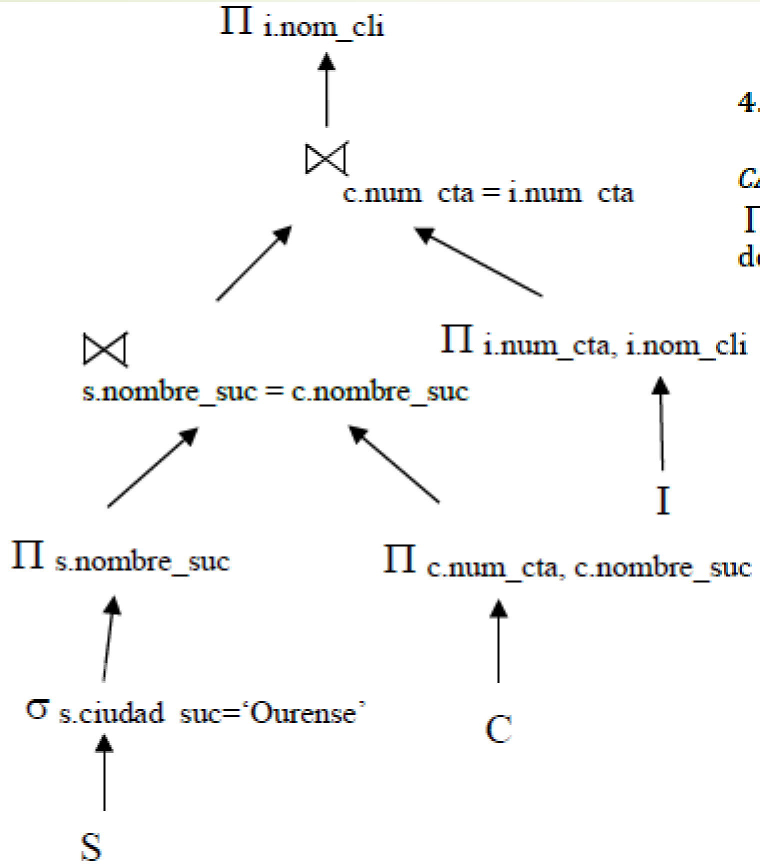
$(E1 \bowtie E2) \bowtie E3 = E1 \bowtie (E2 \bowtie E3)$

En este caso, no es necesario aplicar las reglas

2. $\Pi_{nom\_cli}((\sigma_{ciudad\_suc='Ourense'}$ **(sucursal) ⋈ cuenta) ⋈ impositor)**

3. $\Pi_{nom\_cli}((\sigma_{ciudad\_suc='Ourense'}$ **(sucursal) ⋈ cuenta) ⋈ impositor)**

Ejemplo 1 uso reglas de equivalencia



4. *DESPLAZAR Π HACIA HOJAS EN ⋈ ó X*:

CASCADA DE Π: $\Pi_{L1}(\Pi_{L2}(\ldots\Pi_{Li}(E))\ldots)) = \Pi_{L1}(E)$

$\Pi_{L1 \cup L2}(E1 \bowtie_{\theta} E2) = \Pi_{L1}(E1) \bowtie_{\theta} \Pi_{L2}(E2)$

donde θ implica solo atributos de L1∪L2

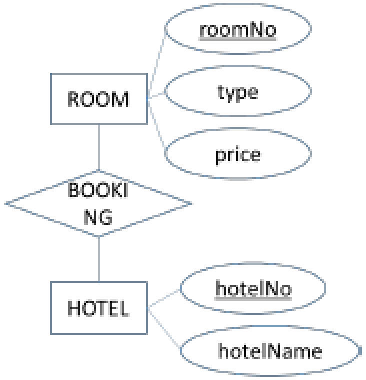
Expresión final optimizada

$\Pi_{nom\_cli}((\Pi_{nombre\_suc}(\sigma_{ciudad\_suc='Ourense'}(\text{sucursal})) \bowtie \Pi_{num\_cta,nombre\_suc}(\text{cuenta})) \bowtie \Pi_{num\_cta,nom\_cli}(\text{impositor}))$

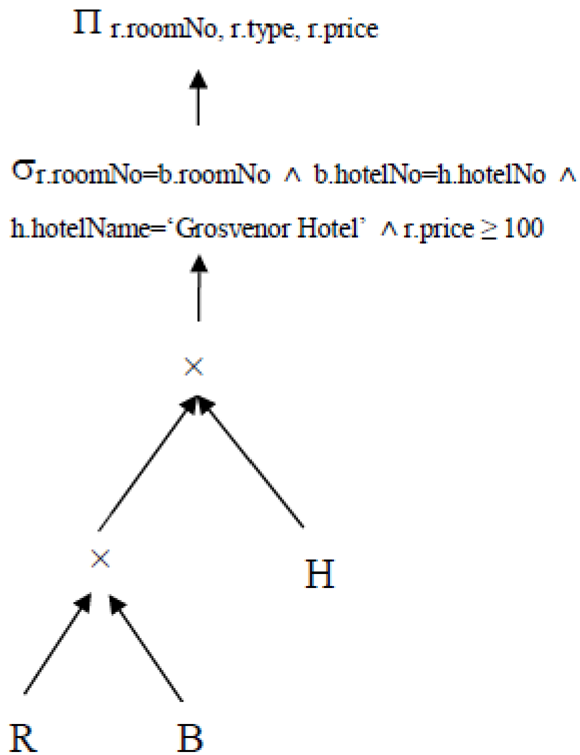
Ejemplo 2 uso reglas de equivalencia

1 – Representar gráficamente mediante árboles las expresiones del álgebra relacional de la consulta siguiente, y transformarla a una forma más eficiente. Enunciar las reglas de equivalencia utilizadas en cada uno de los pasos del proceso:

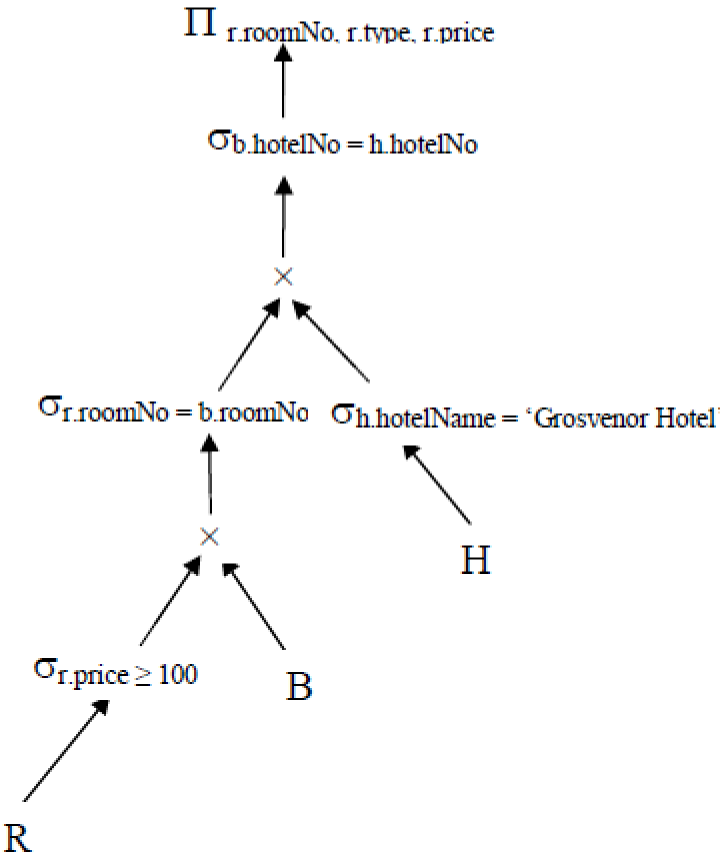
SELECT r.roomNo, r.type, r.price
FROM Room r, Booking b, Hotel h
WHERE r.roomNo = b.roomNo AND b.hotelNo = h.hotelNo AND
h.hotelName = "Grosvenor Hotel" AND r.price ≥ 100;



Ejemplo 2 uso reglas de equivalencia



Árbol inicial



1. ***DESPLAZAR σ HACIA HOJAS EN X:***

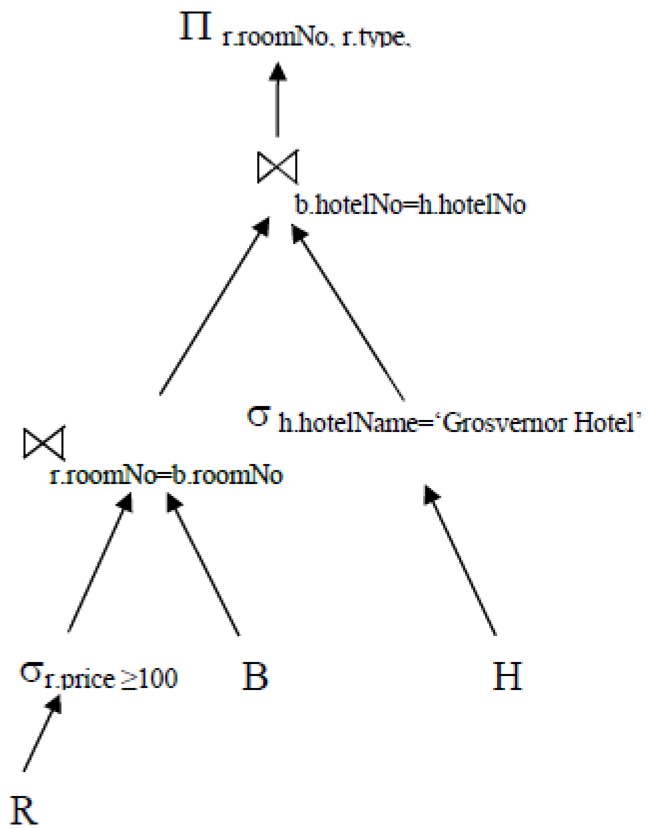
CASCADA DE σ: $\sigma_{\theta1\wedge\theta2}(E) = \sigma_{\theta1}(\sigma_{\theta2}(E))$

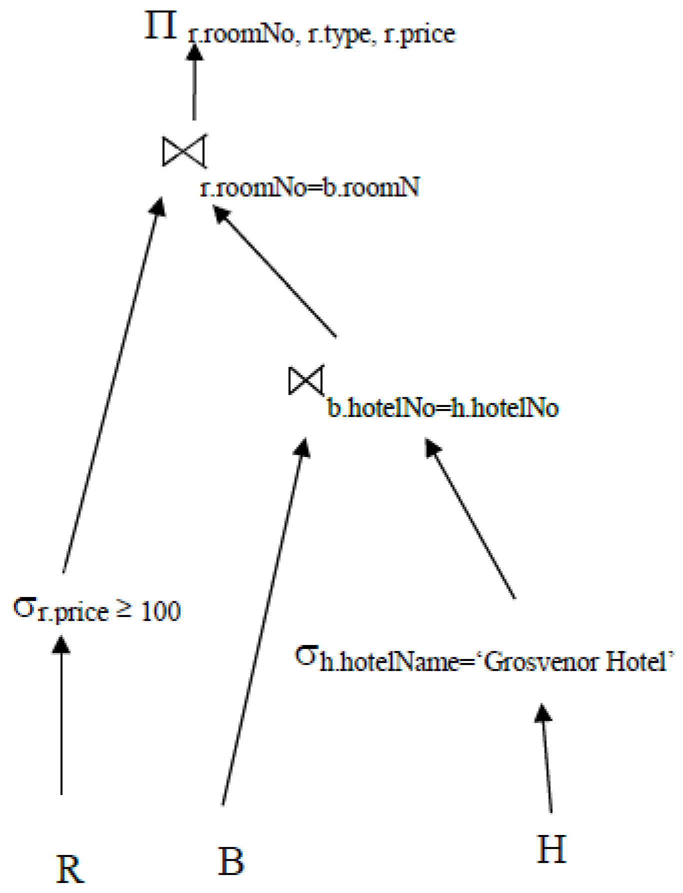
CONMUTATIVIDAD DE σ: $\sigma_{\theta1}(\sigma_{\theta2}(E)) = \sigma_{\theta2}(\sigma_{\theta1}(E))$

$\sigma_{\theta0}(E1XE2)=\sigma_{\theta0}(E1)X\,E2$, $\theta0$ implica solo atrib. de E1

$\sigma_{\theta1\wedge\theta2}(E1X\,E2) = \sigma_{\theta1}(E1)\ X\ \sigma_{\theta2}(E2)$ donde $\theta1$ implica solo atributos de E1 y $\theta2$ atributos de E2

Ejemplo 2 uso reglas de equivalencia





2. Sustituir el producto cartesiano (X) seguido de σ por join:

$\sigma_\theta (E1\ X\ E2) = E1 \bowtie_\theta E2$

3. Ejecutar antes las selecciones y los joins que producen menos tuplas

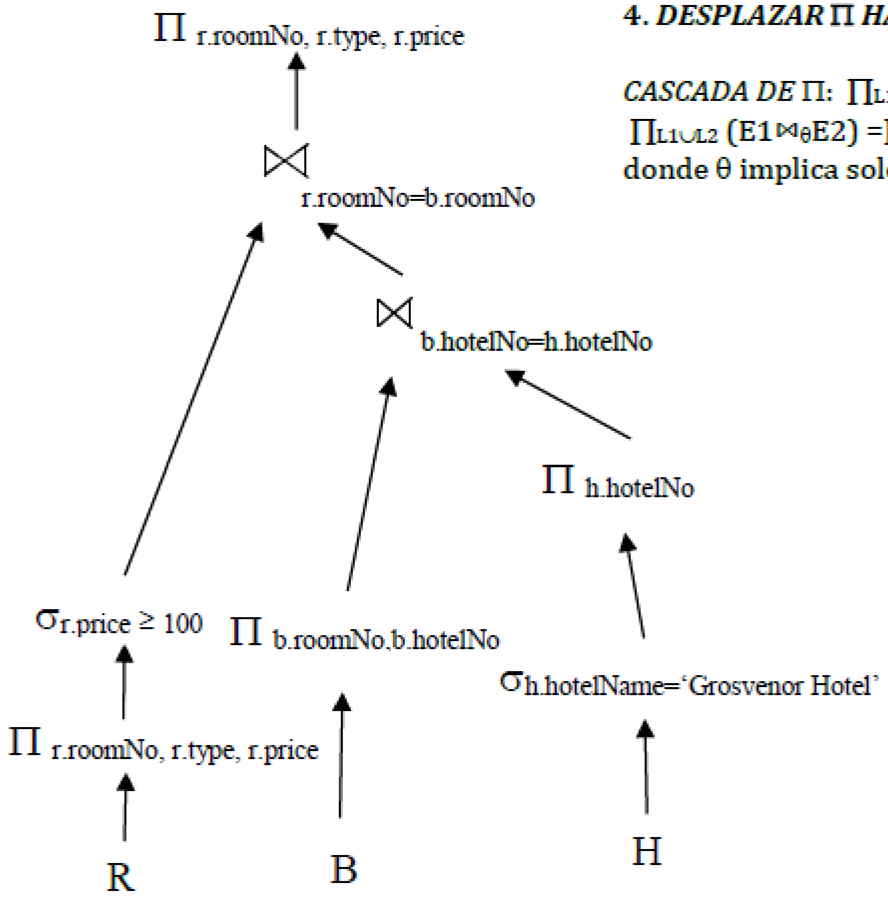
(en este caso, el nº hoteles es inferior al nº habitaciones)

CONMUTATIVIDAD DE σ => $\sigma_{\theta 1}(\sigma_{\theta 2}(E)) = \sigma_{\theta 2}(\sigma_{\theta 1}(E))$

ASOCIATIVIDAD DE ⋈ =>

$(E1 \bowtie E2) \bowtie E3 = E1 \bowtie (E2 \bowtie E3)$

Ejemplo 2 uso reglas de equivalencia



***4. DESPLAZAR Π HACIA HOJAS EN ⋈ ó X*:**

CASCADA DE Π: $\Pi_{L1}(\Pi_{L2}(\ldots\Pi_{Li}(E))\ldots)) = \Pi_{L1}(E)$

$\Pi_{L1\cup L2}(E1 \bowtie_{\theta} E2) = \Pi_{L1}(E1) \bowtie_{\theta} \Pi_{L2}(E2)$

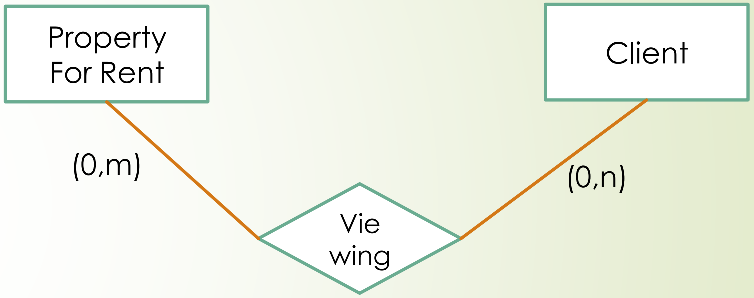
donde θ implica solo atributos de $L1\cup L2$

Expresión final optimizada

$\Pi_{\text{roomNo, type, price}}$ **(**$\sigma_{\text{price >= 100}}$ **(**$\Pi_{\text{roomNo, type, price}}$ **(Room))** $\bowtie$ **(**$\Pi_{\text{roomNo, hotelNo}}$ **(Booking)** $\bowtie$ Π_{hotelNo} **(**$\sigma_{\text{hotelName = `Grosvenor Hotel'}}$ **(Hotel))))**

Ejemplo 3 uso reglas de equivalencia

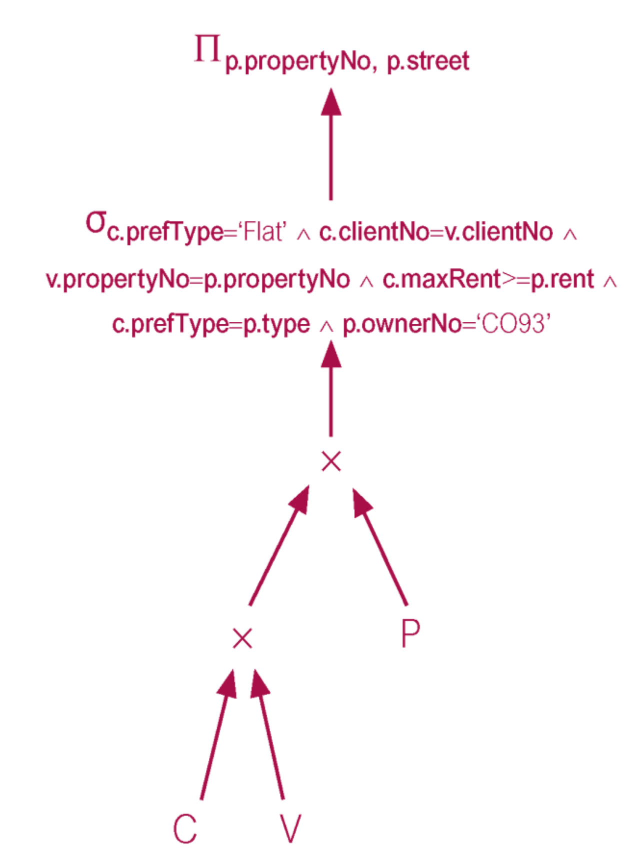
PropertyForRent (<u>propertyNo</u>, street, city, postcode, type, rooms, rent, ownerNo)
Client (<u>clientNo</u>, fName, lName, address, telNo, prefType, maxRent)
Viewing (<u>clientNo</u>, <u>propertyNo</u>, comment, viewDate)



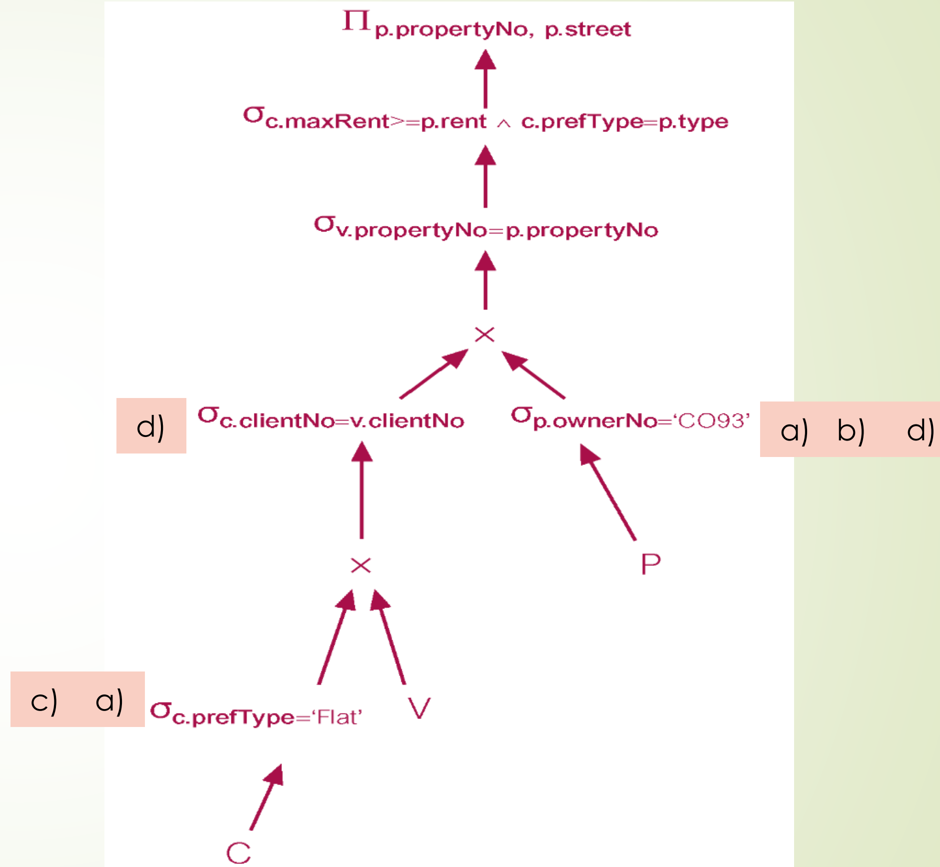
Para los clientes que buscan pisos, localizar los inmuebles que satisfacen sus requisitos y pertenecen al propietario CO93

```
SELECT    p.propertyNo, p.Street
FROM      Client c, Viewing v, PropertyForRent p
WHERE     c.prefType = 'Flat' AND
          c.clientNo = v.clientNo AND
          v.propertyNo = p.propertyNo AND
          c.maxRent >= p.rent AND
          c.prefType = p.type AND
          p.ownerNo = 'CO93';
```

Ejemplo 3 uso reglas de equivalencia



Árbol inicial



1. Desplazar σ hacia las hojas del árbol

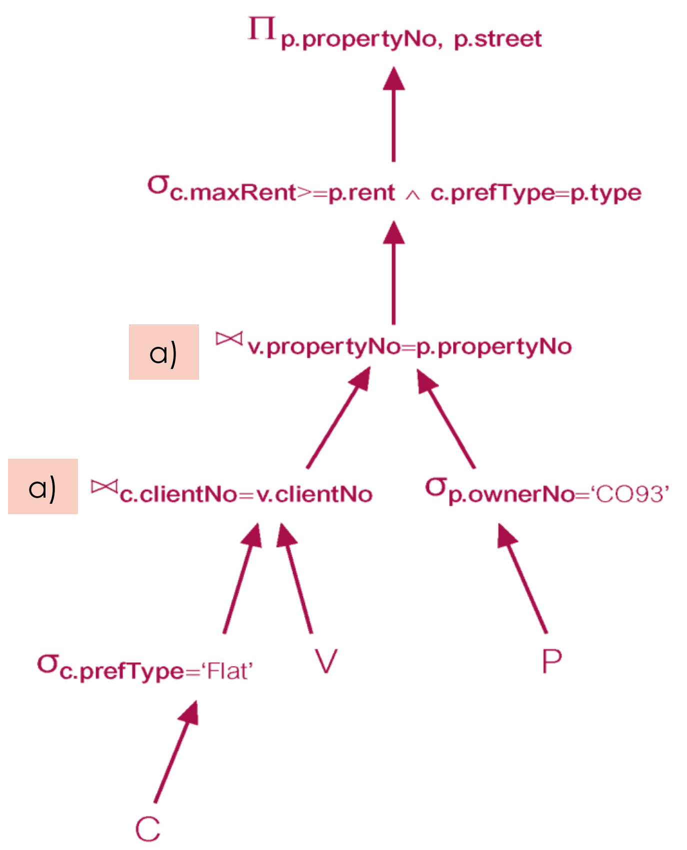
a) CASCADA DE σ => $\sigma_{\theta1 \wedge \theta2}(E) = \sigma_{\theta1}(\sigma_{\theta2}(E))$

b) CONMUTATIVIDAD DE σ => $\sigma_{\theta1}(\sigma_{\theta2}(E)) = \sigma_{\theta2}(\sigma_{\theta1}(E))$

c) $\sigma_{\theta0}(E1 X E2) = \sigma_{\theta0}(E1) X E2$ donde θ0 implica solo atributos de E1

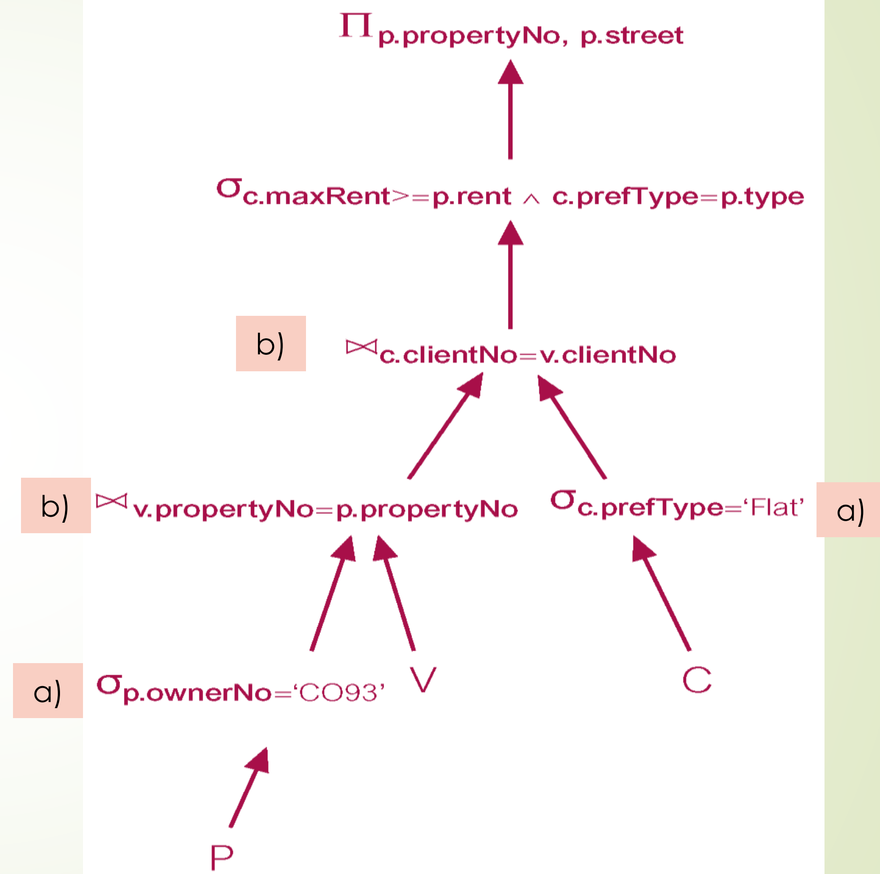
d) $\sigma_{\theta1 \wedge \theta2}(E1 X E2) = \sigma_{\theta1}(E1) X \sigma_{\theta2}(E2)$ donde θ1 implica solo atributos de E1 y θ2 de E2

Ejemplo 3 uso reglas de equivalencia



2. Sustituir el producto cartesiano seguido de σ por ⋈

a) $\sigma_\theta(E_1 \times E_2) = E_1 \bowtie_\theta E_2$



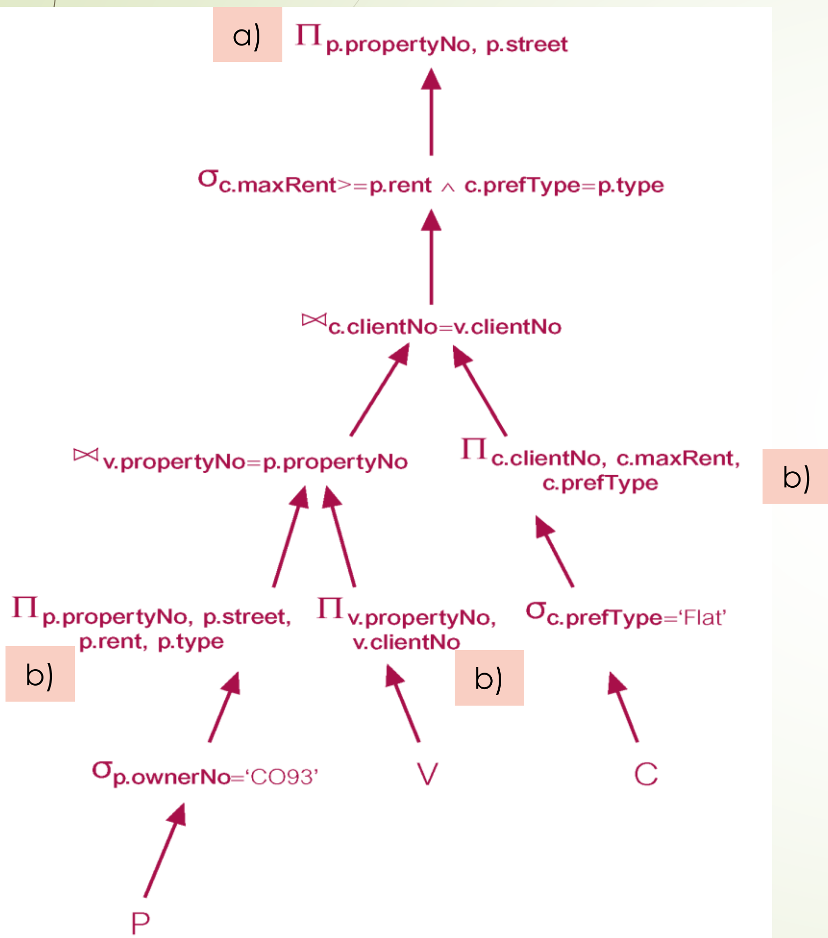
3. Determinar las operaciones σ y ⋈ que producen menos tuplas

(en este caso, el nº propiedades es inferior al nº clientes)

a) CONMUTATIVIDAD DE σ => $\sigma_{\theta 1}(\sigma_{\theta 2}(E)) = \sigma_{\theta 2}(\sigma_{\theta 1}(E))$

b) ASOCIATIVIDAD DE ⋈ => $(E1 \bowtie E2) \bowtie E3 = E1 \bowtie (E2 \bowtie E3)$

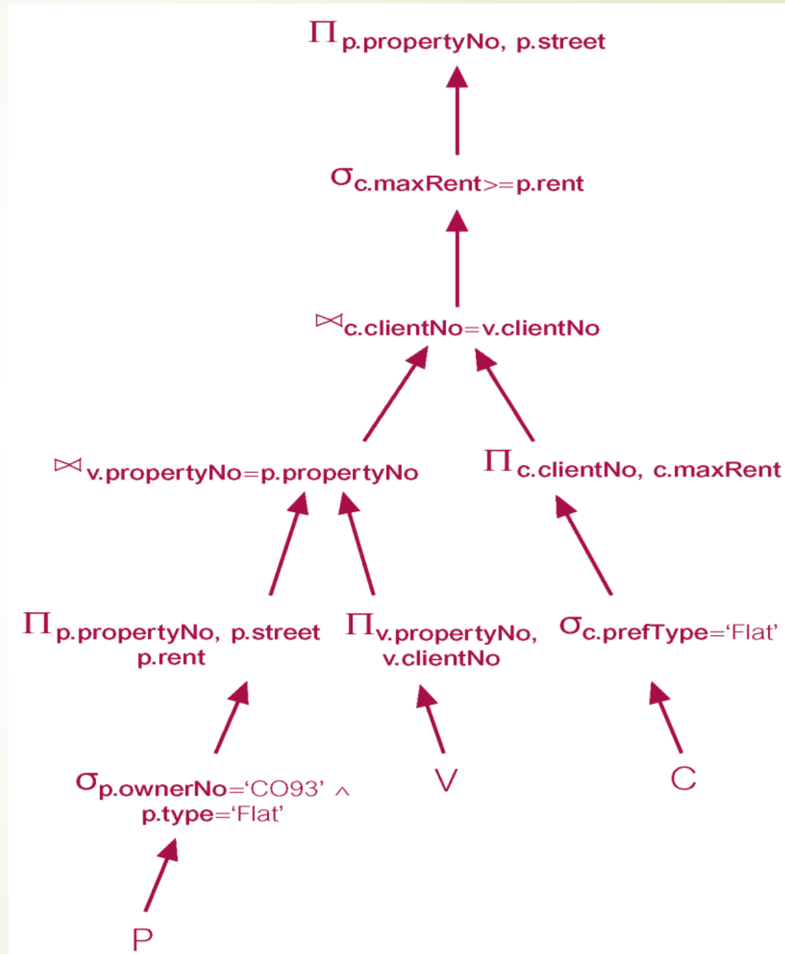
Ejemplo 3 uso reglas de equivalencia



4. Desplazar $\prod$ hacia las hojas del árbol

a) CASCADA DE $\prod$ => $\prod_{L1}(\prod_{L2}(\ldots\prod_{Li}(E))\ldots)) = \prod_{L1}(E)$

b) $\prod_{L1\cup L2}(E1 \bowtie_{\theta} E2) = \prod_{L1\cup L2}(\prod_{L1\cup L3}(E1) \bowtie_{\theta} \prod_{L2\cup L4}(E2))$



5. Sustitución de c.prefType=p.type por (p.type='Flat')

Tipos de optimización: basada en coste

- Se genera una gama de planes de evaluación empleando reglas de equivalencia
- Se realiza una estimación estadística de los resultados de las expresiones
  - Ver apartado 14.3 Silberstchatz, A.; Korth, H.; Sudarshan, S. Fundamentos de Bases de Datos. Mc Graw-Hill (5ª edición)
- Se escoge el plan de evaluación de coste mínimo

Estimación estadística de tamaños

- Información del catálogo:
  - **n_r** (*nº registros de r*)
  - **V(A, r)** (*nº valores <u>distintos</u> para el (conjunto de) atributo(s) A en la relación r*)

Estimación estadística de tamaños

- Tamaño del filtro de atributos (proyección)

$\prod_A$ **(r)** => **V(A, r)** porque elimina duplicados

Ej: $\prod_{edad}$ *(empleado) donde el rango de edad es [20, 60)*

=> V(edad, empleado) = 40 valores diferentes

- Tamaño de la selección de tuplas (igualdad)

$$\boldsymbol{\sigma}_{A\,=\,v}\,\mathbf{(r)} \Rightarrow \boldsymbol{n_r}\,\frac{\boldsymbol{casos\ favorables}}{\boldsymbol{casos\ posibles}} = \boldsymbol{n_r}\,\frac{\mathbf{1}}{\mathbf{V}(\boldsymbol{A},\boldsymbol{r})}$$

Ej: $\sigma_{ciudad\_suc\ =\ \text{“Madrid”}}$ *(sucursal)*

donde $\boldsymbol{n_{sucursal}}$*= 1000 tuplas y* $\mathrm{V}(ciudad\_suc, sucursal) = 5$ *equiprobables*

Probabilidad de que 1 tupla satisfaga la condición: $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{5}$

Nº de tuplas que satisfacen la condición: $1000\,\frac{1}{5} = \boldsymbol{200\ tuplas}$

Estimación estadística de tamaños

- Tamaño de la selección de tuplas (rango)

$\sigma_{A<v}(r) \Rightarrow \boldsymbol{n_r}\left(\frac{\boldsymbol{v-min(A,r)}}{\mathbf{max}(\boldsymbol{A,r})-\mathbf{min}(\boldsymbol{A,r})}\right)$ *si se conoce v y hay distribución uniforme de valores*

Ej: $\sigma_{edad < 30}$ *(empleado)* *edad [20, 60),* $n_{empleado} = 2000$

Probabilidad de que 1 tupla satisfaga la condición: $\frac{casos\ favorables}{casos\ posibles} = \frac{30-20}{60-20} = \frac{1}{4}$

Nº de tuplas que satisfacen la condición: $2000\ \frac{1}{4} = \boldsymbol{500\ tuplas}$

$\sigma_{A<v}(r) \Rightarrow \frac{\boldsymbol{n_r}}{\mathbf{2}}$ *si NO se conoce v*

Ej: $\sigma_{edad < 'X'}$ *(empleado)* *edad [20, 60),* $n_{empleado} = 2000$

Probabilidad de que 1 tupla satisfaga la condición: $\frac{1}{2}$

Nº de tuplas que satisfacen la condición: $2000\ \frac{1}{2} = \boldsymbol{1000\ tuplas}$

Estimación estadística de tamaños

- Tamaño de la selección de tuplas (rango)

$$\sigma_{A \geq v}(r) \Rightarrow n_r \left(\frac{\max(A,r)-v}{\max(A,r)-\min(A,r)}\right)$$

Ej: $\sigma_{edad \geq 30}$ *(empleado)* *edad [20, 60],* $n_{empleado}$ *= 2000*

Probabilidad de que 1 tupla satisfaga la condición: $\frac{casos\ favorables}{casos\ posibles} = \frac{60-30}{60-20} = \frac{3}{4}$

Nº de tuplas que satisfacen la condición: $2000\ \frac{3}{4} = \mathbf{1500\ tuplas}$

Estimación estadística de tamaños

- Tamaño de la selección de tuplas (conjunción de condiciones)

$\sigma_{\theta 1 \wedge \theta 2 \wedge \ldots \wedge \theta m}$ (r)

Probabilidad $P_{\theta t}$ de que 1 tupla *t* satisfaga la condición θ1: $\frac{casos\ favorables}{casos\ posibles}$

Si las condiciones son <u>independientes</u>, la probabilidad de que 1 tupla cumpla todas las condiciones es el producto de las probabilidades de las condiciones: $P_{\theta 1} * P_{\theta 2} * \ldots * P_{\theta m}$

Nº tuplas que satisface la selección completa: $\boldsymbol{n_r}(P_{\theta 1} * P_{\theta 2} * \ldots * P_{\theta m})$

Ej: $\sigma_{(edad < 30) \wedge (salario = 120)}$ *(empleado) donde edad [20, 60), $n_{empleado}$ = 2000, salario [50, 150)*

Probabilidad de que 1 tupla satisfaga la condición (edad < 30): $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{4}$

Probabilidad de que 1 tupla satisfaga la condición (salario = 120): $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{150-50} = \frac{1}{100}$

Probabilidad de que 1 tupla satisfaga (edad < 30) ^ (salario = 120): $\frac{1}{4}\frac{1}{100} = \frac{1}{400}$

Nº de tuplas que satisface (edad < 30) ^ (salario = 120): $2000\ \frac{1}{400} = \boldsymbol{5\ tuplas}$

Estimación estadística de tamaños

- Tamaño de la selección de tuplas (disyunción de condiciones)

$\sigma_{\theta 1 \vee \theta 2 \vee \dots \vee \theta m}$ (r)

Probabilidad $P_{\theta t}$ de que 1 tupla *t* satisfaga la condición θ1: $\frac{casos\ favorables}{casos\ posibles}$

Si las condiciones son independientes, la probabilidad de que 1 tupla satisfaga la disyunción es:

(1 menos la probabilidad de que no satisfaga ninguna) $1-(1-P_{\theta 1}) * (1-P_{\theta 2}) * \dots * (1-P_{\theta m})$

Nº tuplas que satisface la selección completa: $\boldsymbol{n_r}\,[1-(1-P_{\theta 1}) * (1-P_{\theta 2}) * \dots * (1-P_{\theta m})]$

Ej: σ *(edad < 30) ∨ (salario = 120)* *(empleado) donde edad [20, 60),* $n_{empleado}$ *= 2000, salario [50, 150)*

Probabilidad de que 1 tupla satisfaga la condición (edad < 30): $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{4}$

Probabilidad de que 1 tupla satisfaga la condición (salario = 120): $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{150-50} = \frac{1}{100}$

Probabilidad de que 1 tupla satisfaga (edad < 30) o (salario = 120)

es (1 menos la probabilidad de que no satisfaga ***ninguna****) :* $1-\left(1-\frac{1}{4}\right)*\left(1-\frac{1}{100}\right) = \frac{103}{400}$

Nº de tuplas que satisface (edad < 30) o (salario = 120): $2000\,\frac{103}{400} = \mathbf{515}\ \boldsymbol{tuplas}$

Estimación estadística de tamaños

- Join de relaciones r ∞ s, siendo **R** el esquema de **r** y **S** el esquema de **s**

a) Si R ∩ S = 0 => **no tienen atributos en común** => $n_r * n_s$

b) Si R ∩ S es **clave de R** => cada tupla de s se combina como máximo con 1 tupla de r => el nº de tuplas del join NO es mayor que las tuplas de s

No Titulados/as

c_emp	…
e2	…
e3	…
e6	…

Titulados/as

c_emp	…
e1	…
e4	…
e5	…

Proyectos

n_proy	jefe
p1	e2
p2	e1
p3	e4
p4	e1

Titulados/as ∞ Proyectos

c_emp	…	n_proy	jefe
e1	…	p2	e1
e4	…	p3	e4
e1	…	p4	e1

Estimación estadística de tamaños

- Join de relaciones r ∞ s, siendo **R** el esquema de **r** y **S** el esquema de **s**

a) Si R ∩ S = 0 => **no tienen atributos en común** => $n_r * n_s$

b) Si R ∩ S es clave de R => cada tupla de s se combina como máximo con 1 tupla de r => el nº de tuplas del join NO es mayor que las tuplas de s

Si, además, los atributos comunes son **clave foránea de S**, entonces el nº de tuplas del join es EXACTAMENTE el nº de tuplas de s

Empleados/as

c_emp	...
e1	...
e2	...
e3	...
e4	...

Proyectos

n_proy	jefe
p1	e2
p2	e1
p3	e1

Empleados/as ∞ Proyectos

c_emp	...	n_proy	jefe
e2	...	p1	e2
e1	...	p2	e1
e1	...	p3	e1

Estimación estadística de tamaños

- Join de relaciones r ∞ s, siendo **R** el esquema de **r** y **S** el esquema de **s**

c) Si R ∩ S **NO es clave de R ni de S**, suponiendo que todos los valores son equiprobables, entonces:

Cli

n_cli	edad
c1	10
c2	10
c3	20
c4	20
c5	25
c6	25
c7	30
c8	30

V(edad,clientes) = 4
$n_{clientes}$ = 8

Prov

n_prov	edad
p1	15
p2	15
p3	15
p4	20
p5	20
p6	20

V(edad,proveedores) = 2
$n_{proveedores}$ = 6

1 tupla de Cli produce como máximo: $\frac{n_{Prov}}{V(edad,Prov)} = \frac{6}{2}$ = 3 tuplas

Todas las tuplas de Cli producen como máximo $n_{Cli} \frac{n_{Prov}}{V(edad,Prov)}$ = 8 x 3 = 24 tuplas

1 tupla de Prov produce como máximo: $\frac{n_{Cli}}{V(edad,Cli)} = \frac{8}{4}$ = 2 tuplas

Todas las tuplas de Prov producen como máximo $n_{Prov} \frac{n_{Cli}}{V(edad,Cli)}$ = 6 x 2 = 12 tuplas

$n_{Cli \infty Prov} = \min(24{,}12)$ = 12 tuplas

Estimación estadística de tamaños

- Join de relaciones r ∞ s, siendo **R** el esquema de **r** y **S** el esquema de **s**

c) Si R ∩ S NO es clave de R ni de S, suponiendo que todos los valores son equiprobables, entonces:

a) cada tupla de R produce $\frac{\boldsymbol{n_s}}{\boldsymbol{V(A,s)}}$ tuplas en el join, donde A son los atributos comunes

=> todas las tuplas de R producen $\boldsymbol{n_r}\,\frac{\boldsymbol{n_s}}{\boldsymbol{V(A,s)}}$ tuplas

b) cada tupla de S produce $\frac{\boldsymbol{n_r}}{\boldsymbol{V(A,r)}}$ tuplas en el join, donde A son los atributos comunes

=> todas las tuplas de S producen $\boldsymbol{n_s}\,\frac{\boldsymbol{n_r}}{\boldsymbol{V(A,r)}}$ tuplas

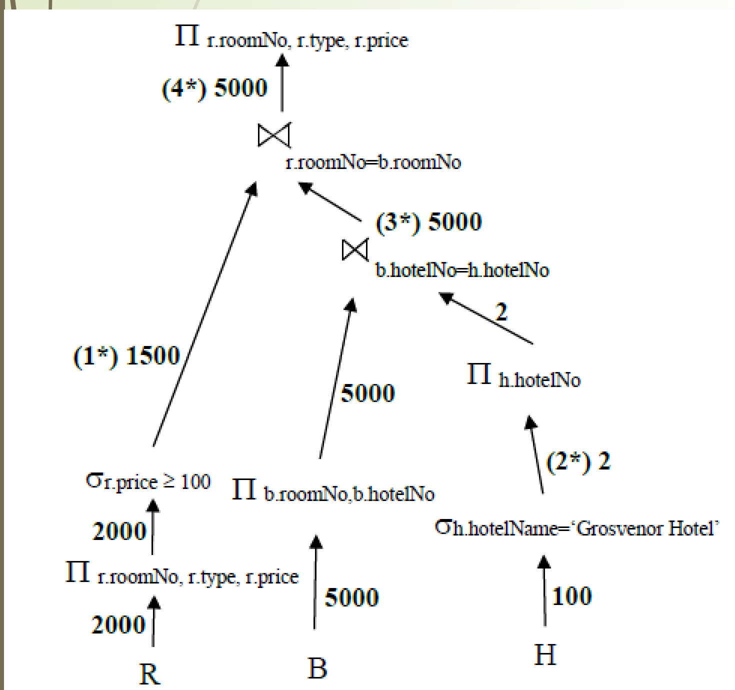
c) si $V(A,r) \neq V(A,s)$, el nº de tuplas del join es $\boldsymbol{min}\,(\frac{\boldsymbol{n_r * n_s}}{\boldsymbol{V(A,r)}}, \frac{\boldsymbol{n_r * ns}}{\boldsymbol{V(A,s)}})$

Ejemplo 1 cálculo de estadísticas

SELECT r.roomNo, r.type, r.price
FROM Room r, Booking b, Hotel h
WHERE r.roomNo = b.roomNo AND b.hotelNo = h.hotelNo
AND h.hotelName = "Grosvenor Hotel" AND r.price ≥ 100;

Estimar el tamaño de las operaciones que aparecen en el árbol final, teniendo en cuenta lo siguiente:

nRoom = 2000, nHotel = 100, nBooking = 5000,
V(hotelName, Hotel) = 50, min(r.price, Room) = 50, max(r.price, Room) = 250



(1*) $\sigma_{r.price \geq 100}$, dado que se conoce a priori el valor de comparación se calcula como:

2000 (250-100)/(250-50) = 1500 tuplas

(2*) $\sigma_{h.hotelName = \text{'Grosvernor Hotel'}}$ si se supone una distribución uniforme de los valores del atributo h.hotelName, dado que V(hotelName, Hotel) = 50, la probabilidad de que una tupla cumpla la selección es $\frac{\textit{casos favorables}}{\textit{casos posibles}} = \frac{1}{50}$

Por tanto, el nº de tuplas que satisfacen la condición: 100 X 1/50=2 tuplas

(3*) B ⋈ H, donde B ∩ H = hotelNo, que es clave de H y foránea de B, entonces el número de tuplas no es mayor que las de B = 5000.

(4*) (B ⋈ H) ⋈ R, donde B ⋈ H ∩ R = roomNo, que es clave de R y foránea de B. Entonces el número de tuplas no es mayor que las de B ⋈ H = 5000.

Ejemplo 2 cálculo de estadísticas

Estimar el tamaño de las operaciones que aparecen en el árbol, teniendo en cuenta lo siguiente:

n_{Client} = 1000, $n_{Viewing}$ = 3000, $n_{PropertyForRent}$ = 500,

V(preftype, Client) = 5, V(maxRent, Client) = 50

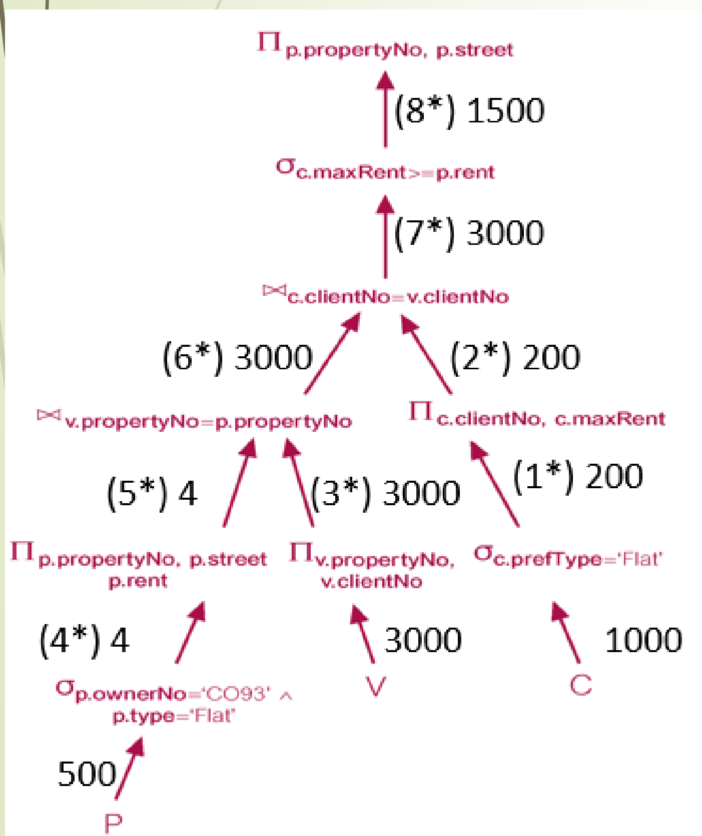
min(maxRent, Client) = 100, max(maxRent, Client) = 3000,

min (rent, PropertyForRent) = 350, max (rent, PropertyForRent) = 1000,

V(rent, PropertyForRent) = 20 V(ownerNo, PropertyForRent) = 30,

V(clientNo, Viewing) = 250, V(propertyNo, Viewing) = 400

V(type, PropertyForRent) = 5



(1*) $\sigma_{c.prefType = 'Flat'}$ si se supone una distribución uniforme de los valores del atributo c.prefType, la probabilidad de que 1 tupla cumpla la selección es $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{5}$

Por tanto, el nº de tuplas que satisfacen la condición: $1000\ X \frac{1}{5} = 200\ tuplas$

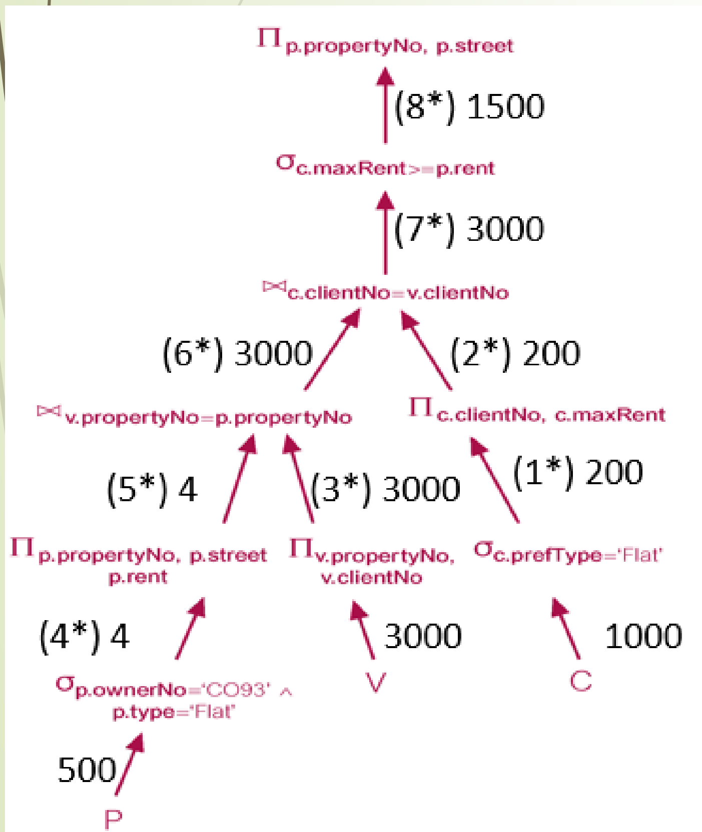
(2*) $\pi_{c.clientNo,\ c.maxRent}$ su tamaño estimado es V(c.clientNo c.maxRent, C). Como c.clientNo es clave en C, no hay posibilidad de la existencia de duplicados, por lo que el número de tuplas se mantiene.

(3*) $\pi_{v.propertyNo,\ v.clientNo}$ su tamaño estimado es V(v.propertyNo v.clientNo, V). Como v.propertyNo y v.clientNo forman la clave en V, no hay posibilidad de la existencia de duplicados, por lo que el número de tuplas se mantiene.

Ejemplo 2 cálculo de estadísticas

Estimar el tamaño de las operaciones que aparecen en el árbol final, teniendo en cuenta lo siguiente:

n_{Client} = 1000, $n_{Viewing}$ = 3000, $n_{PropertyForRent}$ = 500,
V(preftype, Client) = 5, V(maxRent, Client) = 50
min(maxRent, Client) = 100, max(maxRent, Client) = 3000,
min (rent, PropertyForRent) = 350, max (rent, PropertyForRent) = 1000,
V(rent, PropertyForRent) = 20 V(ownerNo, PropertyForRent) = 30,
V(clientNo, Viewing) = 250, V(propertyNo, Viewing) = 400
V(type, PropertyForRent) = 5



(4*) $\sigma_{p.ownerNo='C093' \wedge p.type='Flat'}$ Debe estimarse el tamaño de cada parte de la conjunción

$\sigma_{p.ownerNo}$ = 'C093' como se conoce el valor de *p.ownerNo*, y se supone una distribución uniforme de los valores del atributo, la probabilidad de que 1 tupla cumpla la selección es $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{30}$

$\sigma_{p.type}$= 'Flat' como se conoce el valor de *p.type*, y se supone una distribución uniforme de los valores del atributo, la probabilidad de que 1 tupla cumpla la selección es $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{5}$

Dado que las condiciones son independientes, la probabilidad de que 1 tupla satisfaga ($\sigma_{p.ownerNo}$ = 'C093') ^ ($\sigma_{p.type}$= 'Flat') es $\frac{1}{30} X \frac{1}{5} = \frac{1}{150}$

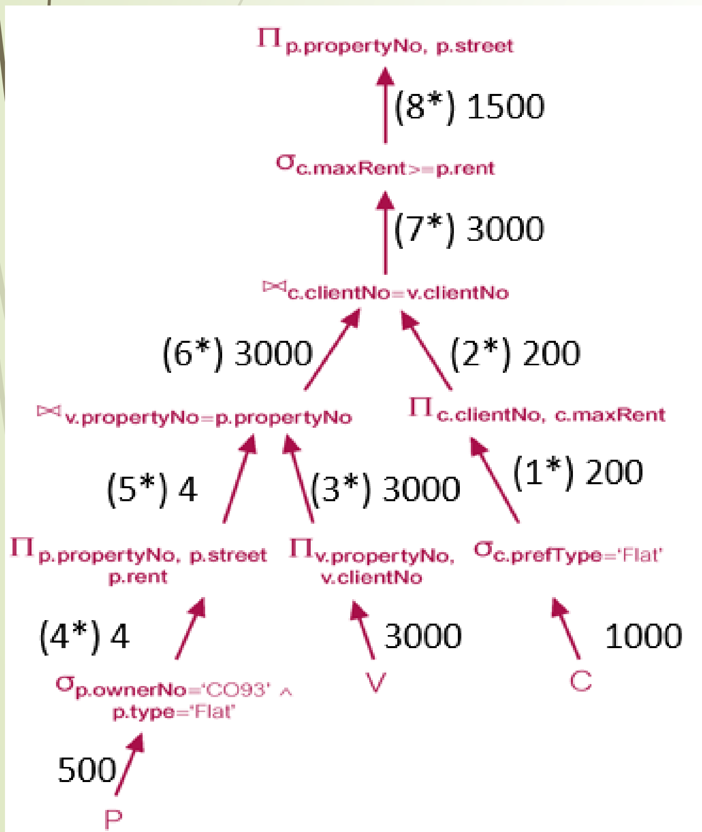
De este modo, el nº de tuplas que satisface ambas condiciones es 500 X $\frac{1}{150} \simeq 4\ tuplas$

(5*) $\pi_{p.propertyNo,\ p.street,\ p.rent}$ su tamaño estimado es V(*p.propertyNo p.street p.rent, P*). Como *p.propertyNo* es clave en P, no hay posibilidad de la existencia de duplicados, por lo que el número de tuplas se mantiene.

Ejemplo 2 cálculo de estadísticas

Estimar el tamaño de las operaciones que aparecen en el árbol final, teniendo en cuenta lo siguiente:

n_{Client} = 1000, $n_{Viewing}$ = 3000, $n_{PropertyForRent}$ = 500,
V(preftype, Client) = 5, V(maxRent, Client) = 50
min(maxRent, Client) = 100, max(maxRent, Client) = 3000,
min (rent, PropertyForRent) = 350, max (rent, PropertyForRent) = 1000,
V(rent, PropertyForRent) = 20 V(ownerNo, PropertyForRent) = 30,
V(clientNo, Viewing) = 250, V(propertyNo, Viewing) = 400
V(type, PropertyForRent) = 5



(6*) $\bowtie_{v.propertyNo\,=\,p.propertyNo}$, como *propertyNo* es clave de *PropertyForRent*, cada tupla de V se combina como máximo con una tupla de P. Por lo tanto, $n_{V \bowtie P} <= n_V$

Como *propertyNo* es foránea en P, entonces $n_{V \bowtie P} = n_V$ = *3000 tuplas*

(7*) $\bowtie_{c.clientNo=v.\,clientNo}$, como *c.clientNo* es clave de *Client*, cada tupla de V se combina como máximo con una tupla de C. Por lo tanto, $n_{C \bowtie V} <= n_V$

Como *clientNo* es foránea en V, entonces $n_{C \bowtie V} = n_V$ = *3000 tuplas*

(8*) $\sigma_{c.maxRent >= p.rent}$ dado que no se conoce a priori el valor de *p.rent*, la probabilidad de que 1 tupla cumpla la selección es $\frac{casos\ favorables}{casos\ posibles} = \frac{1}{2}$

Por tanto, el nº de tuplas que satisfacen la condición: $3000\ X \frac{1}{2} = 1500\ tuplas$