



UA

Unidad 2: Procesamiento y Optimización de Consultas

*Bases de Datos Avanzadas, Sesión 6:
Procesamiento y Optimización de Consultas*

*Iván González Diego
Dept. Ciencias de la Computación
Universidad de Alcalá*



UA

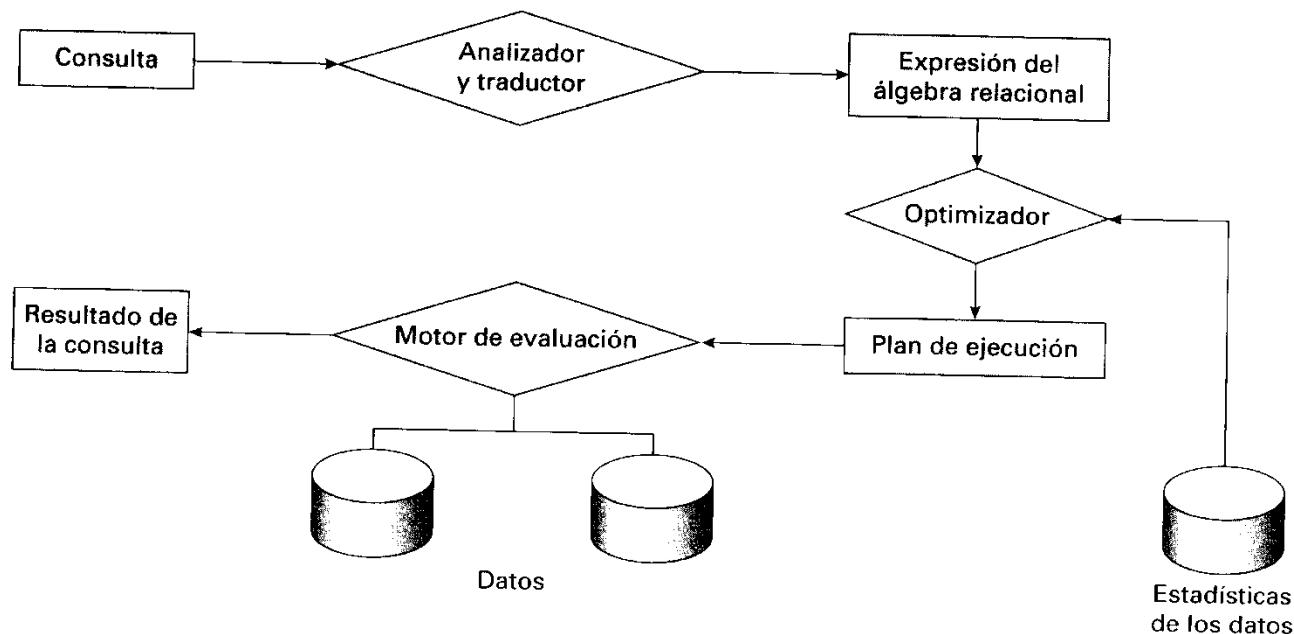
INDICE

- *Introducción.*
- *Álgebra Relacional – SQL*
- *Transformación de Expresiones Equivalentes*
- *Plan de Evaluación (o Ejecución)*
- *Optimización basada en costes*
- *Optimización Heurística*

Referencias: *Silberschatz 4^a Ed.*
Elmasri, 3^a Ed.

Pasos básicos en el procesamiento de consultas

- 1. Análisis y traducción.
- 2. Optimización.
- 3. Evaluación.





Pasos básicos en el procesamiento de consultas

- *Análisis y traducción*
 - Traducción de la consulta a su formato interno.
 - Se transforma en el álgebra relacional extendida.
 - Se verifica la sintaxis y se verifican las relaciones.
- *Evaluación*
 - El motor de ejecución de la consulta toma un plan de evaluación, ejecuta el plan y devuelve el resultado de la consulta.



Pasos básicos en el procesamiento de consultas: Optimización

- *Un expresión del álgebra relacional puede tener muchas expresiones equivalentes:*
 - *Ejemplo:* $\sigma_{\text{saldo} < 2500}(\Pi_{\text{saldo}}(\text{cuenta}))$ es equivalente a
 $\Pi_{\text{saldo}}(\sigma_{\text{saldo} < 2500}(\text{cuenta}))$
- *Cada operación del álgebra relacional puede ser evaluada usando uno de los diferentes algoritmos \Rightarrow Primitivas*
 - *Una expresión del álgebra relacional puede ser evaluada de muchas maneras*
- *Una secuencia de operaciones que se pueden utilizar para evaluar una consulta \Rightarrow plan de evaluación*
 - *Ejemplo:* poder usar un índice en saldo para encontrar $\text{saldo} < 2500$
 - *O realizar una búsqueda completa y descartar los saldos ≥ 2500*



Pasos básicos en el procesamiento de consultas: Optimización

- *Optimización de consultas* ⇒ entre todos los planes de evaluación de consultas, elegir la de menor coste
 - El coste se estima usando información estadística del catálogo ⇒ nº de tuplas de cada relación, tamaño de las tuplas, etc
- *Medida de los costes de las consultas.*
- *Algoritmos para evaluar operaciones del álgebra relacional*
- *Combinación de algoritmos para evaluar expresiones*
- *Optimizar consultas* ⇒ encontrar un plan de evaluación con el menor coste estimado.



UA

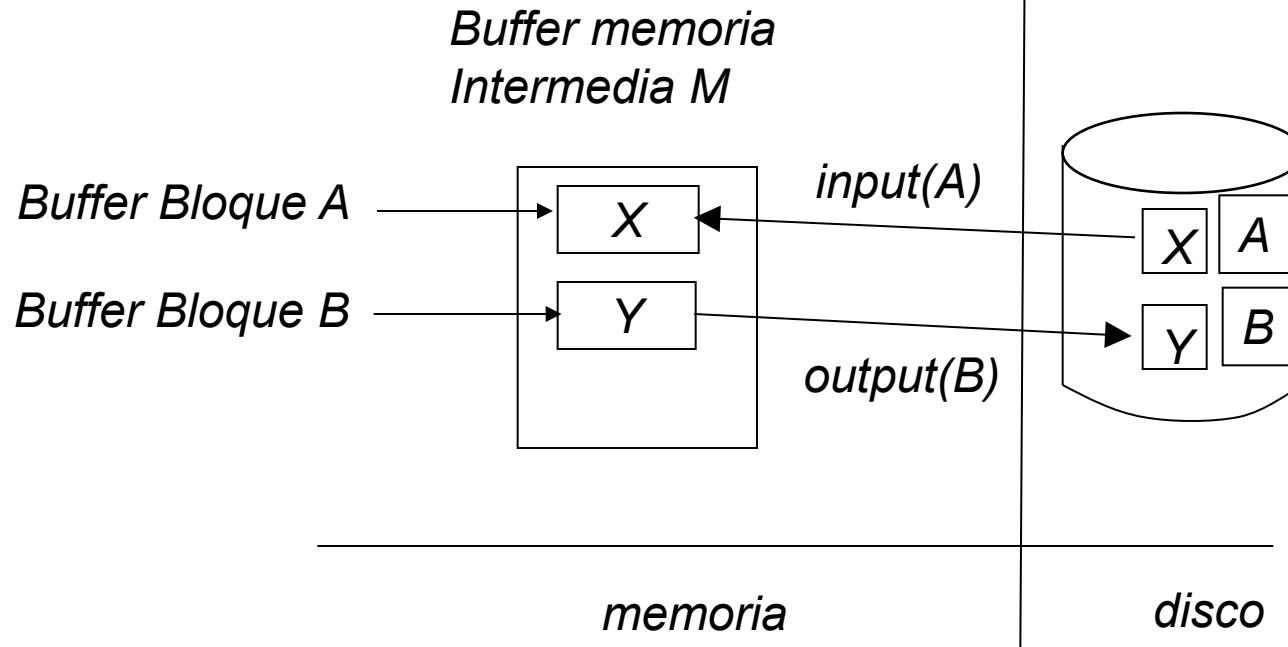
Medidas del coste de una consulta

- *Se mide por el tiempo utilizado para responder a la consulta*
 - Factores: acceso al disco, CPU o red de comunicación.
- *Coste más predominante: acceso disco*
 - Fácil de estimar, teniendo en cuenta
 - Número de búsquedas \Rightarrow coste medio de búsqueda
 - Número de bloques leídos \Rightarrow coste medio de lectura de bloque
 - Número de bloques escritos \Rightarrow coste medio de escritura de bloque
- *Se usará el número de bloques transferidos del disco como una medida del coste.*
- *Tamaño de la memoria disponible M (bloques/páginas en memoria)*
- *Lo que no cabe en RAM \rightarrow se graba temporalmente en disco*



UA

Medidas del coste de una consulta





UA

Algebra Relacional - SQL

- Proyección $\Pi_A(r)$ → **SELECT DISTINCT A from r;**
- Selección $\sigma_{\text{condición}}(r)$ → **select * from r WHERE condición;**
- Producto cartesiano $r \times s$ → **select * from r,s;**
- Unión $r \cup s$ → **select * from r UNION select * from s;**
- Diferencia $r - s$ → **select * from r EXCEPT select * from s;**
- Intersección $r \cap s$ → **select * from r INTERSECT select * from s;**
- Natural Join $r \bowtie s$ → **select * from r NATURAL JOIN s;**
- Join $r \bowtie_{\text{condición}} s$ → **select * from r INNER JOIN s ON condición;**
- Outer join $r \bowtie s$ → **select * from r FULL OUTER JOIN s;**
- Agregado ${}^A g_{\text{count}(B)}(r)$ → **select A,count(B) from r GROUP BY A;**
- Select * from r **ORDER BY A** → no existe en Algebra Relacional



Transformación de Expresiones Relacionales

- *Dos expresiones del álgebra relacional son **equivalentes** si para cada instancia de la base de datos legal, las dos expresiones generan el mismo conjunto de tuplas*
- *En SQL \Rightarrow entradas y salidas son un multiconjunto de tuplas*
- *Una **regla de equivalencia** afirma que las expresiones de las dos formas son equivalentes*
 - *Se puede reemplazar la expresión de la primera forma por la segunda ó viceversa.*



UA

Reglas de Equivalencia

1. Operaciones de selección conjuntiva se pueden dividir en una secuencia de selecciones individuales.

$$\sigma_{\theta_1 \wedge \theta_2}(E) = \sigma_{\theta_1}(\sigma_{\theta_2}(E))$$

2. Operaciones de selección son conmutativas.

$$\sigma_{\theta_1}(\sigma_{\theta_2}(E)) = \sigma_{\theta_2}(\sigma_{\theta_1}(E))$$

3. Sólo la última secuencia de operaciones de proyección se necesitan, las otras se pueden omitir:

$$\Pi_{t_1}(\Pi_{t_2}(\kappa(\Pi_{tn}(E))\kappa)) = \Pi_{t_1}(E)$$

4. Las selecciones se pueden combinar con productos cartesianos y reuniones zeta

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

b. $\sigma_{\theta_1}(E_1 \bowtie_{\theta_2} E_2) = E_1 \bowtie_{\theta_1 \wedge \theta_2} E_2$



UA

Reglas de Equivalencia

5. Las operaciones de reunión zeta son conmutativas

$$E_1 \bowtie_{\theta} E_2 = E_2 \bowtie_{\theta} E_1$$

6. (a) Operaciones de reunión natural son asociativas:

$$(E_1 \bowtie E_2) \bowtie E_3 = E_1 \bowtie (E_2 \bowtie E_3)$$

(b) Reuniones zeta son asociativas de la siguiente manera:

$$(E_1 \bowtie_{\theta_1} E_2) \bowtie_{\theta_2 \wedge \theta_3} E_3 = E_1 \bowtie_{\theta_1 \wedge \theta_3} (E_2 \bowtie_{\theta_2} E_3)$$

donde θ_2 involucra sólo atributos de E_2 y E_3 .



UA

Reglas de Equivalencia

7. La operación de selección se distribuye por la operación de reunión zeta bajo las dos condiciones siguientes:
- (a) cuando todos los atributos θ_0 implican sólo los atributos de una de las expresiones (E_1) que están reuniendo.

$$\sigma_{\theta 0}(E_1 \bowtie_{\theta} E_2) = (\sigma_{\theta 0}(E_1)) \bowtie_{\theta} E_2$$

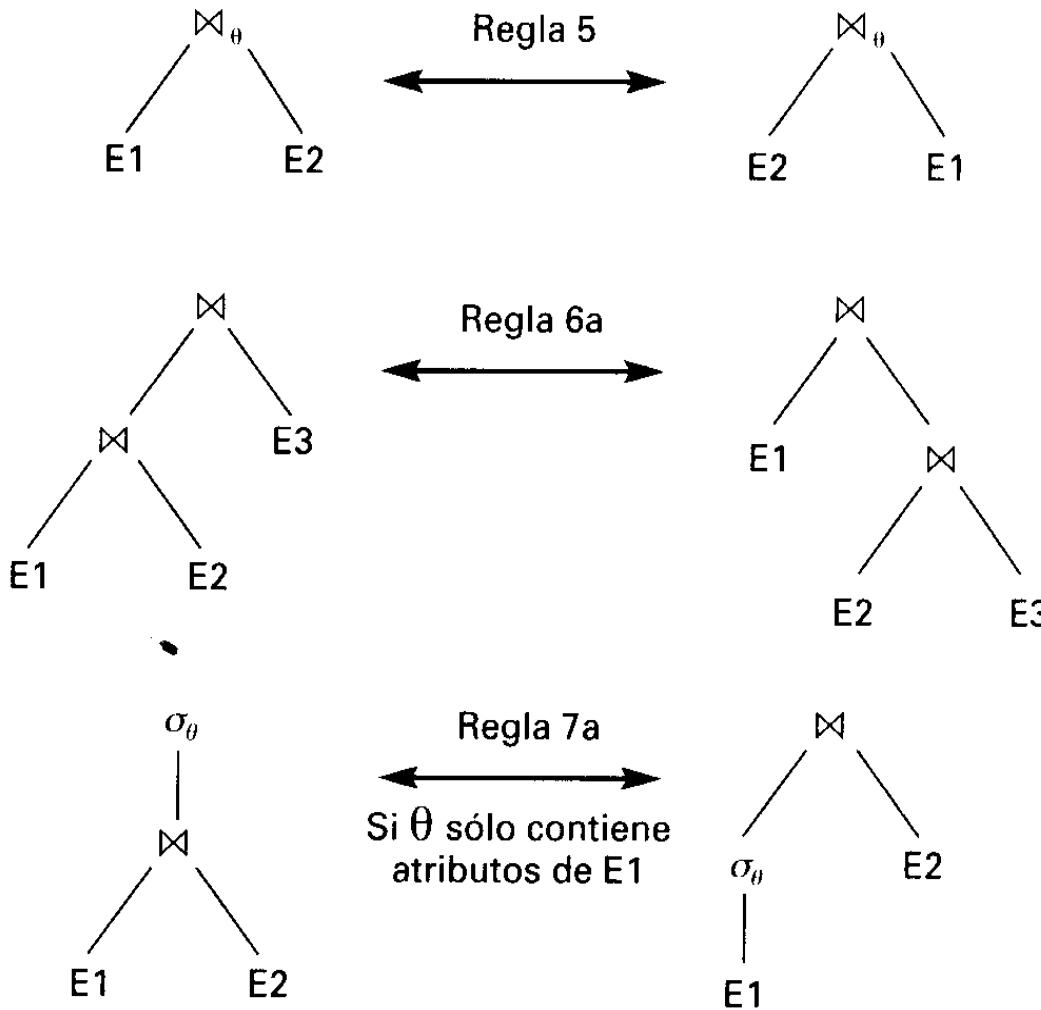
- (b) Cuando θ_1 implica sólo los atributos de E_1 y θ_2 implica sólo los atributos de E_2 .

$$\sigma_{\theta 1} \wedge_{\theta 2} (E_1 \bowtie_{\theta} E_2) = (\sigma_{\theta 1}(E_1)) \bowtie_{\theta} (\sigma_{\theta 2}(E_2))$$



UA

Reglas de Equivalencia





UA

Reglas de Equivalencia

8. Las operaciones de proyección se distribuyen por la operación de reunión zeta bajo las condiciones:

(a) Si θ implica sólo los atributos de $L_1 \cup L_2$:

$$\Pi_{L_1 \cup L_2}(E_1 \bowtie_{\theta} E_2) = (\Pi_{L_1}(E_1)) \bowtie_{\theta} (\Pi_{L_2}(E_2))$$

(b) Considerar una reunión $E_1 \bowtie_{\theta} E_2$.

- L_1 y L_2 conjunto de atributos de E_1 y E_2 , respectivamente.
- L_3 atributos de E_1 , que están implicados en la condición de reunión θ , pero no están incluidos en $L_1 \cup L_2$
- L_4 atributos de E_2 que están implicados en la condición de reunión θ , pero no están incluidos en $L_1 \cup L_2$.

$$\Pi_{L_1 \cup L_2}(E_1 \bowtie_{\theta} E_2) = \Pi_{L_1 \cup L_2}((\Pi_{L_1 \cup L_3}(E_1)) \bowtie_{\theta} (\Pi_{L_2 \cup L_4}(E_2)))$$



UA

Reglas de Equivalencia

9. Operaciones de conjuntos de unión e intersección son conmutativas

$$E_1 \cup E_2 = E_2 \cup E_1$$

$$E_1 \cap E_2 = E_2 \cap E_1$$

10. Unión e intersección de conjuntos son asociativas.

$$(E_1 \cup E_2) \cup E_3 = E_1 \cup (E_2 \cup E_3)$$

$$(E_1 \cap E_2) \cap E_3 = E_1 \cap (E_2 \cap E_3)$$

11. La operación de selección se distribuye sobre \cup , \cap y $-$.

$$\sigma_\theta(E_1 - E_2) = \sigma_\theta(E_1) - \sigma_\theta(E_2)$$

Similarmente para \cup y \cap en lugar de $-$

También: $\sigma_\theta(E_1 - E_2) = \sigma_\theta(E_1) - E_2$

y similarmente para \cap , pero no para \cup

12. Operación de proyección se distribuye sobre la operación de unión

$$\Pi_L(E_1 \cup E_2) = (\Pi_L(E_1)) \cup (\Pi_L(E_2))$$



Enumeración de expresiones Equivalentes

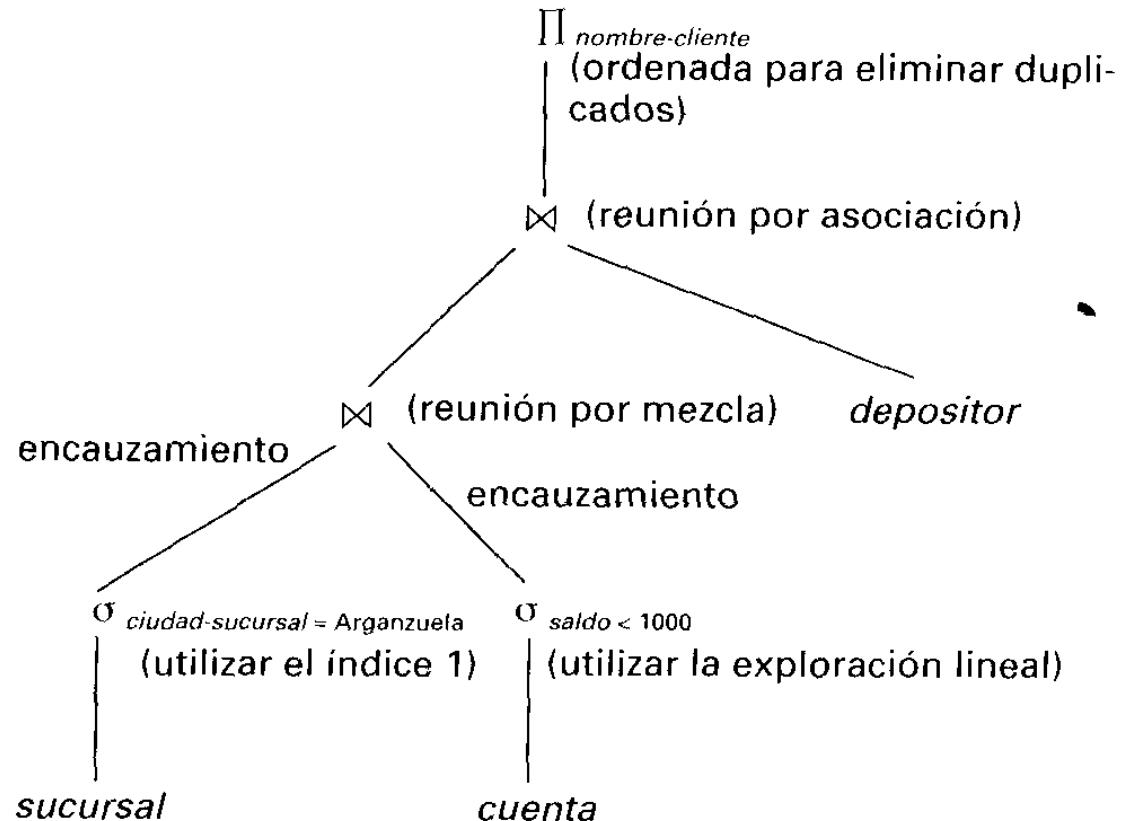
- *Optimizadores de consultas usan reglas de equivalencia para generar sistemáticamente expresiones equivalentes a una expresión dada.*
- *Genera todas las expresiones equivalentes hasta que no se pueden encontrar más expresiones:*
 - *Para cada expresión encontrada, usar todas las reglas que se pueden encontrar y añadir las nuevas expresiones a las encontradas.*
- *Muy costosa en espacio y tiempo*
- *Requerimiento de espacio se puede reducir compartiendo subexpresiones comunes*
 - *Cuando E1 se genera de E2 \Rightarrow usualmente sólo el nivel alto de las dos son diferentes, los subárboles por debajo son el mismo y se pueden compartir*
- *Requerimiento de tiempo se puede reducir no generando todas las expresiones*



Plan de Evaluación

UA

- Un plan de evaluación define exactamente qué algoritmo se usa para cada operación y cómo se coordinan la ejecución de las operaciones.





UA

Elección de Planes de Evaluación

- *Se debe de considerar la interacción de las técnicas de evaluación cuando se eligen planes de evaluación ⇒ elegir el algoritmo menos costoso para cada operación individual puede que no sea la mejor elección.*
- *Los optimizadores incorporan elementos de las dos aproximaciones siguientes:*
 - *Buscar todos los planes de evaluación y elegir el mejor plan según el coste*
 - *Utilizar la heurística para elegir un plan.*



UA

Optimización basada en el coste

- Considerar encontrar la mejor reunión para $r_1 \bowtie r_2 \bowtie \dots r_n$.
- Hay $(2(n - 1))!/(n - 1)!$ diferentes planes
 - Con $n = 7$, el número es 665280
 - Con $n = 10$, el número es mayor que 17.600 millones!
- No es necesario generar todos los planes
- Usando programación dinámica \Rightarrow el orden menos costoso para $\{r_1, r_2, \dots r_n\}$ se analiza una vez y se almacena para su uso futuro.



UA

Programación dinámica en Optimización

- *Para encontrar la mejor reunión para un conjunto S de n relaciones*
 - Considerar todos los posibles planes de la forma: $S_1 \bowtie (S - S_1)$
 - Recursivamente analizar los costes para reunir subconjuntos de S para encontrar el coste de cada plan
 - Elegir el menos costoso de las $2^n - 1$ alternativas.
 - Cuando el plan para cualquier subconjunto se analiza \Rightarrow se almacena y se reutiliza en vez de volver a calcularlo
- *Coste temporal $O(3^n)$*
- *Coste espacial $O(2^n)$*



UA

Optimización Heurística

- *Optimización por coste es costosa*
- *Utilizar la heurística para reducir el número de elecciones de una manera basada en costes*
- *Optimización heurística transforma el árbol de consultas usando un conjunto de reglas que generalmente mejoran el rendimiento:*
 - *Realizar la selección cuanto antes ⇒ reduce el número de tuplas*
 - *Realizar la proyección cuanto antes ⇒ reduce el número de atributos*
 - *Realizar las operaciones de selección y reunión más restrictivas antes que otras operaciones similares*
 - *Algunos sistemas usan sólo heurística , otros combinan heurística con optimizaciones parciales basadas en coste.*



Pasos en una Optimización Heurística

1. *Descomponer las selecciones conjuntivas en una secuencia de operaciones de selección sencillas.* \Rightarrow Regla equivalencia 1
2. *Desplazar las operaciones de selección hacia la parte baja del árbol* \Rightarrow ejecutarlas lo antes posible (Reglas 2, 7a, 7b, 11).
3. *Ejecutar primero las selecciones y reuniones que produzcan relaciones más pequeñas* (Regla 6).
4. *Reemplazar operaciones producto cartesiano seguidas de una selección por una operación reunión.* (Regla 4a).
5. *Dividir y desplazar hacia abajo del árbol como sea posible las listas de atributos de proyección, creando nuevas proyecciones si se necesitan* (Reglas 3, 8a, 8b, 12).
6. *Identificar aquellos subárboles cuyas operaciones pueden ser encauzadas y ejecutarlos usando encauzamiento*



Estructura de los Optimizadores de consultas

- *Optimizador System R considera sólo órdenes de reunión en profundidad por la izquierda ⇒ reduce la complejidad y genera planes que se pueden evaluar por encauzamiento.*
Utiliza la heurística para poner selecciones y proyecciones abajo en el árbol.
- *Optimización heurística usada en algunas versiones de Oracle:*
 - *Repetidamente tomar la mejor relación para la siguiente reunión.*
 - *Comenzando desde n planes de evaluación. Tomar el mejor de estos*
- *Para exploraciones usando índices secundarios, algunos optimizadores tienen en cuenta la probabilidad de que la página que contiene la tupla se encuentre en la memoria*
- *Complejidad en los optimizadores de consultas SQL*
 - *Ejemplo: subconsultas anidadas*



Estructura de los Optimizadores de consultas

- *Algunos optimizadores integran selección heurística y generación de planes de acceso alternativos*
- *Incluso con el uso de la heurística, optimización basada en coste impone una sobrecarga adicional*
- *Pero este gasto se ve recompensando por el ahorro del tiempo de ejecución de la consulta, que queda dominado por los accesos al disco.*



Ejemplo

Sucursal(nombre_sucursal,ciudad_sucursal,activo)

Cuenta(numero_cuenta,nombre_sucursal,saldo)

IMPOSITOR(nombre_cliente,numero_cuenta)

Optimizar:

```
SELECT nombre_cliente FROM cuenta NATURAL JOIN  
    IMPOSITOR NATURAL JOIN sucursal WHERE  
    ciudad='Arganzuela';
```

```
SELECT nombre_cliente FROM cuenta NATURAL JOIN  
    IMPOSITOR NATURAL JOIN sucursal WHERE  
    ciudad='Arganzuela' AND saldo>1000;
```