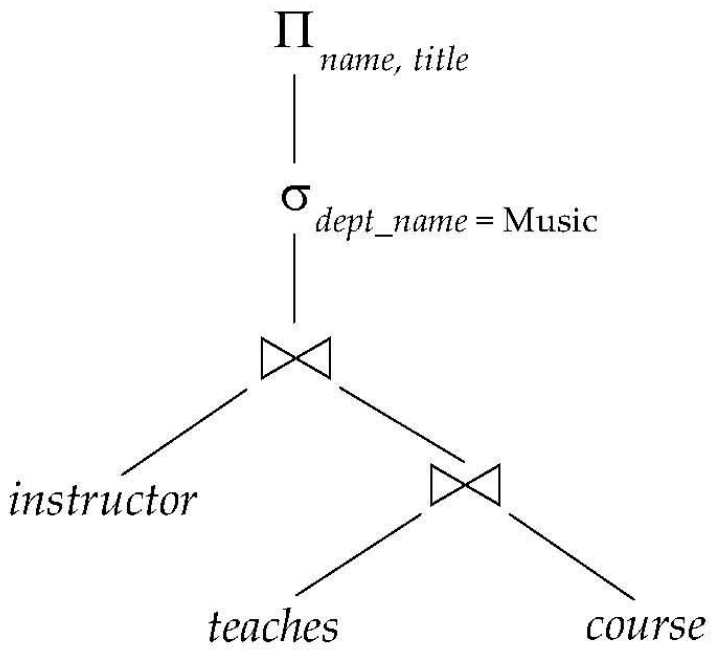
Capítulo 4

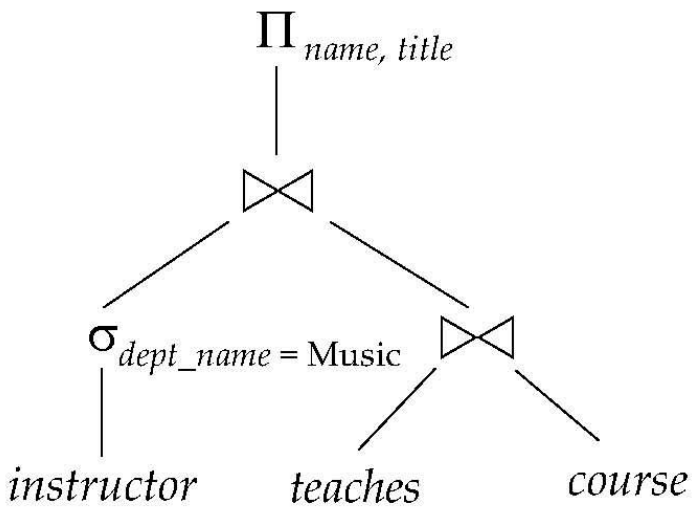
Procesamiento de consultas

Optimización de Consultas

Repaso

- Hay formas alternativas de evaluar una expresión de consulta:
  - Usando expresiones equivalentes
  - Usando diferentes algoritmos para cada operación





Repaso

- Un **plan de evaluación** define qué algoritmo es usado para cada operación y cómo se coordina la ejecución de las operaciones.
- La diferencia de costo entre planes de evaluación puede ser enorme.
  - Por ejemplo, de segundos a días en algunos casos.

Introducción a la optimización de consultas

- **Pasos en optimización de consultas basada en costo:**
  1. Generar expresiones lógicamente equivalentes usando **reglas de equivalencia.**
  2. Anotar expresiones resultantes con operadores físicos para obtener **planes de evaluación alternativos**.
  3. Elegir el **plan más económico** basado en el **costo estimado**.

Repaso

- **El costo estimado de un plan se basa en:**
  - Información estadística acerca de tablas.
    - P.ej: número de tuplas, número de valores distintos de un atributo.
  - Estimación estadística para resultados intermedios.
    - Para computar el costo de expresiones complejas.
  - Fórmula de costo para algoritmos de operadores físicos, computados usando estadísticas.

Introducción a la optimización de consultas

- Las estadísticas son computadas **periódicamente** porque
  - tienden a no cambiar radicalmente en un corto tiempo;
  - estadísticas **algo imprecisas** son útiles siempre que sean aplicadas consistentemente a todos los planes.

Repaso

- **El número de transferencias de bloques es influenciado por:**
  1. Los operadores lógicos de la consulta.
  2. El tamaño de los resultados intermedios.
  3. Los operadores físicos usados para implementar los operadores lógicos.
  4. El método para pasar argumentos de un operador físico al siguiente.
  5. **El orden de aplicación de operaciones similares.**
     - Por ejemplo, varias reuniones o varias selecciones.
- El último ítem es menos conocido y va a quedar más claro durante este capítulo.

Transformación de expresiones de consulta

- Dos expresiones del álgebra de tablas son **equivalentes por igualdad** si las dos tablas del resultado son equivalentes en esquema e información.
- Dos expresiones del álgebra de tablas son **equivalentes módulo ordenamiento de registros** si las dos expresiones generan el mismo multiconjunto de tuplas (i.e. las mismas tuplas en la misma cantidad para cada tupla).
  - O sea, dos expresiones equivalentes a lo más alteran el orden de las tuplas.
  - $r =_O s \Leftrightarrow O(r) = O(s)$, donde O es la operación de ordenación del álgebra de tablas en base a todas las columnas.
- Una **regla de equivalencia** dice que las expresiones de dos formas son equivalentes usando uno de los tipos de equivalencia de arriba.
  - Podemos reemplazar una expresión de la primera forma por la segunda forma o vice-versa.

Reglas de equivalencia

1. La selección conjuntiva de operaciones puede ser deconstruida en una secuencia de selecciones individuales:

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

2. Las operaciones de selección son conmutativas:

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

3. Solo la última en una secuencia de operaciones proyección es necesitada, las otras pueden ser omitidas.

$$\Pi_{L_1}(\Pi_{L_2}(\ldots(\Pi_{Ln}(E))\ldots)) = \Pi_{L_1}(E)$$

Reglas de equivalencia

- **Nomenclatura**:
  - Usaremos θ_r^N para indicar una lista de N columnas de la tabla r.
  - Usaremos $\theta_r^N = \theta_s^N$ para indicar la igualación de cada columna de θ_r^N con las columnas de θ_s^N . Omitiremos la tabla y/o cantidad de columnas si no hay ambigüedad.
  - Indicaremos con p_r a una proposición que sólo utilice columnas de r.

Reglas de equivalencia

4. La reunión natural es asociativa:

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

5. La siguiente regla es parecida a la definición de reunión

$$\Pi_\theta(\sigma_{\theta_r^N = \theta_s^N}(r \times s)) = r \; {}_{\theta_r^N}\!\bowtie_{\theta_s^N} s \quad \text{con } \theta \text{ las columnas de } \bowtie$$

6. La siguiente regla muestra cómo podemos expresar una selección de una reunión selectiva por medio de una reunión selectiva.

$$\sigma_{\theta_r^N = \theta_s^N}\left(r \; {}_{\theta_r^M}\!\bowtie_{\theta_s^M} s\right) = r \; {}_{\theta_r^N, \theta_r^M}\!\bowtie_{\theta_s^N, \theta_s^M} s$$

Reglas de equivalencia

7. La siguiente regla nos permite aplicar selección antes de reunión selectiva:

$$\sigma_{p_r}(r \ {}_{\theta_r}\!\bowtie_{\theta_s} s) = \sigma_{p_r}(r) \ {}_{\theta_r}\!\bowtie_{\theta_s} s$$

8. La siguiente regla se deduce de reglas previas:

$$\sigma_{p_r \wedge p_s}(r \ {}_{\theta_r}\!\bowtie_{\theta_s} s) = \sigma_{p_r}(r) \ {}_{\theta_r}\!\bowtie_{\theta_s} \sigma_{p_s}(s)$$

9. La siguiente regla muestra cómo se comporta la proyección cuando se usa junto con la reunión selectiva.

$$\Pi_{\theta_r^N, \theta_s^O}(r \ {}_{\theta_r^M}\!\bowtie_{\theta_s^M} s) = \Pi_{\theta_r^N}(r) \ {}_{\theta_r^M}\!\bowtie_{\theta_s^M} \Pi_{\theta_s^O}(s) \quad \text{con } \theta_r^M \subseteq \theta_r^N \text{ y } \theta_s^M \subseteq \theta_s^O$$

Reglas de equivalencia

10. La concatenación “conmuta” alterando solo el orden de las tuplas:

    r ++ s $=_o$ s ++ r

11. La concatenación es asociativa.
12. Selección distribuye con concatenación, intersección y resta

$$\sigma_p(r \star s) = \sigma_p(r) \star \sigma_p(s) \quad \text{con} \star \in \{++, \cap, \setminus\}$$

13. La siguiente propiedad es más débil que la anterior.

$$\sigma_p(r \star s) = \sigma_p(r) \star s \quad \text{con} \star \in \{\cap, \setminus\}$$

14. La proyección distribuye con la concatenación:

$$\Pi_\theta(r \mathbin{++} s) = \Pi_\theta(r) \mathbin{++} \Pi_\theta(s)$$

Reglas de equivalencia

15. Eliminación de duplicados distribuye con reunión natural:
    $\nu(r \bowtie s) = \nu(r) \bowtie \nu(s)$
16. Eliminación de duplicados conmuta con selección:
    $\nu(\sigma_P(r)) = \sigma_P(\nu(r))$

Empujando selecciones

- **Consulta**: encontrar los nombres de todos los instructores en el departamento de música, junto con los títulos de los cursos que enseñan.

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”}} (instructor \bowtie (teaches \bowtie \Pi_{course\_id,\ title}\ (course))))$$

- **¿Qué regla de equivalencia se puede usar?**

Empujando selecciones

- **Consulta**: encontrar los nombres de todos los instructores en el departamento de música, junto con los títulos de los cursos que enseñan.

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”}} (instructor \bowtie (teaches \bowtie \Pi_{course\_id,\ title}\ (course))))$$

- Transformación usando la regla 7:

$$\Pi_{name,\ title}((\sigma_{dept\_name=\ \text{“Music”}}\ (instructor)) \bowtie (teaches \bowtie \Pi_{course\_id,\ title}\ (course)))$$

- **Conclusión**: realizar la selección tan temprano como sea posible reduce el tamaño de la tabla a ser combinada.

Ejemplo con múltiples transformaciones

- **Consulta**: Encontrar los nombres de todos los instructores en el departamento de música que han enseñado un curso en 2009, junto con los títulos de los cursos que han enseñado.

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”} \wedge year\ =\ 2009} (instructor \bowtie (teaches \bowtie \Pi_{course\_id,\ title}\ (course))))$$

- **¿Qué reglas de equivalencia se pueden usar?¿En qué orden?**

Ejemplo con múltiples transformaciones

- **Consulta**: Encontrar los nombres de todos los instructores en el departamento de música que han enseñado un curso en 2009, junto con los títulos de los cursos que han enseñado.

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”} \wedge year\ =\ 2009} (instructor \bowtie (teaches \bowtie \Pi_{course\_id,\ title}\ (course))))$$

- Por asociatividad de la reunión natural (regla 4):

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”} \wedge gear\ =\ 2009} ((instructor \bowtie teaches) \bowtie \Pi_{course\_id,\ title}\ (course)))$$

- Usamos regla 7 de hacer selecciones temprano:

$$\sigma_{dept\_name\ =\ \text{“Music”}}\ (instructor) \bowtie \sigma_{\ year\ =\ 2009}\ (teaches)$$

Ordenamiento de reuniones naturales

- Considere la expresión:

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”}}\ (instructor) \bowtie teaches) \bowtie \Pi_{course\_id,\ title}\ (course))))$$

- **¿Como la reunión natural es asociativa, qué reunión conviene hacer primero?**

Ordenamiento de reuniones naturales

- Considere la expresión:

$$\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”}}(instructor) \bowtie teaches) \bowtie \Pi_{course\_id,\ title}(course))))$$

- Podemos computar primero $teaches \bowtie \Pi_{course\_id,\ title}(course)$ y luego combinamos el resultado con $\sigma_{dept\_name=\ \text{“Music”}}(instructor)$ pero el resultado de la primera reunión es probable que sea una tabla grande.
- Solo una pequeña fracción de los instructores es probable que sean del departamento de música. Es mejor computar primero:

$$\sigma_{dept\_name=\ \text{“Music”}}(instructor) \bowtie teaches$$

Optimización Heurística

- **Problema**: Optimización basada en costo es cara.
  - Aunque el costo de optimización de consultas puede ser reducido por algoritmos inteligentes, el número de diferentes planes de evaluación para una consulta puede ser muy grande y encontrar el plan óptimo para ese conjunto requiere mucho esfuerzo computacional.
- Una **heurística** es una técnica que aplica reglas generales o atajos para transformar una consulta en una consulta más eficiente.
  - También en lugar de heurística se usa la frase **estrategia de regla de pulgar**.

Optimización Heurística

- **Solución**: Los sistemas pueden usar heurísticas para reducir el número de elecciones que deben ser hechas cuando se trabaja con costos.
  - La **optimización heurística** transforma el árbol de consulta usando un conjunto de reglas que típicamente pero no siempre mejoran el desempeño de ejecución.
- La mayoría de los optimizadores incluye **heurísticas** para reducir el costo de la optimización de consultas,
  - con el riesgo potencial de no encontrar un plan óptimo.

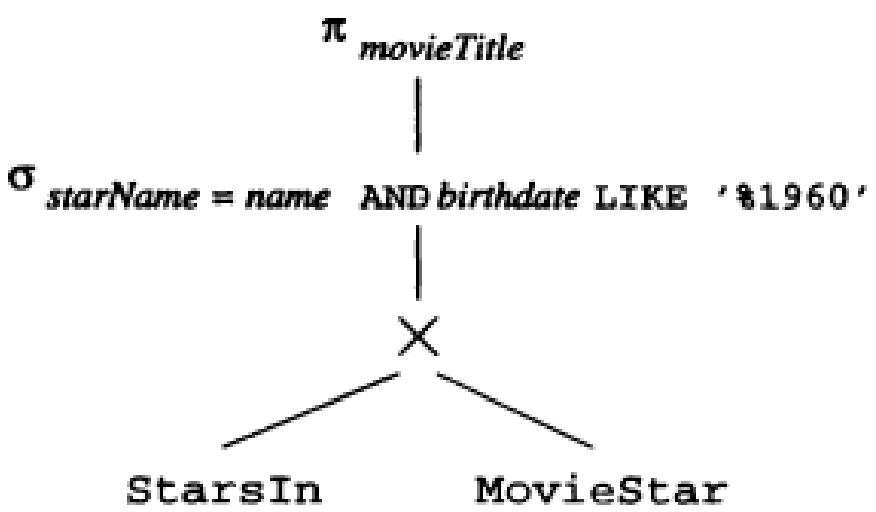
Optimización Heurística

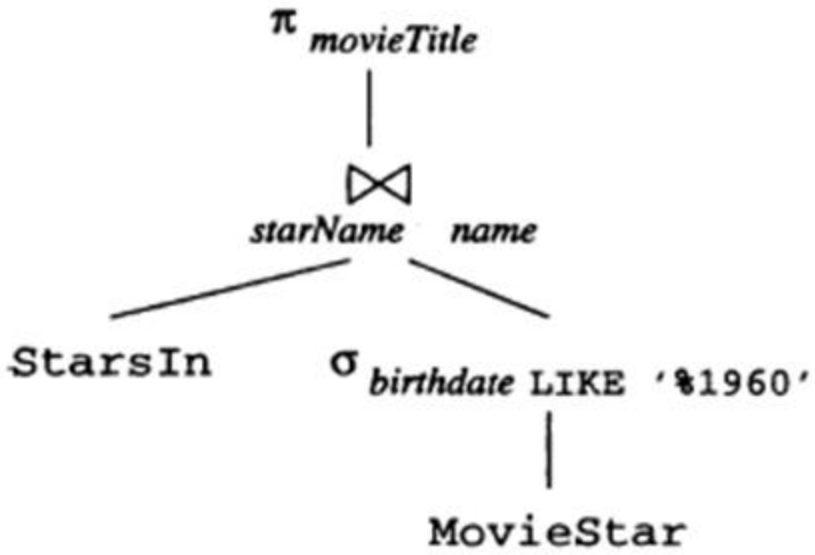
- **Ejemplo de optimización heurística:**
  - Realizar selección tempranamente (reduce el número de tuplas)
    - Empujar selección abajo en el árbol de ejecución tanto como sea posible.
    - Si una condición de selección es un AND de varias condiciones, podemos dividir la condición y empujar cada pieza abajo en el árbol de ejecución separadamente.
  - Realizar proyección temprano (reduce el número de atributos y por ende el tamaño de un resultado intermedio)
    - Proyecciones pueden ser empujadas abajo en el árbol.
  - Hacer la selección más restrictiva (i.e. con tamaño de resultado menor) antes de hacer las otras selecciones.
    - P.ej. Si una condición de selección es un AND de varias condiciones, podemos aplicar selecciones sobre esas condiciones separadamente en algún orden buscando hacer la selección lo más restrictiva posible.

Optimización Heurística

- **Ejemplo de optimización heurística (cont):**
  - Hacer operaciones de reunión más restrictivas:
    - Restringir a tipos particulares de ordenes de reuniones (p.ej. Ordenes de reunión profunda a la izquierda)
  - Ciertas selecciones pueden ser combinadas con producto cartesiano abajo para tornar las operaciones en una reunión (natural o selectiva),
    - la cual es generalmente más eficiente de evaluar esas operaciones separadamente.
    - Para esto suele ser conveniente introducir una proyección también.

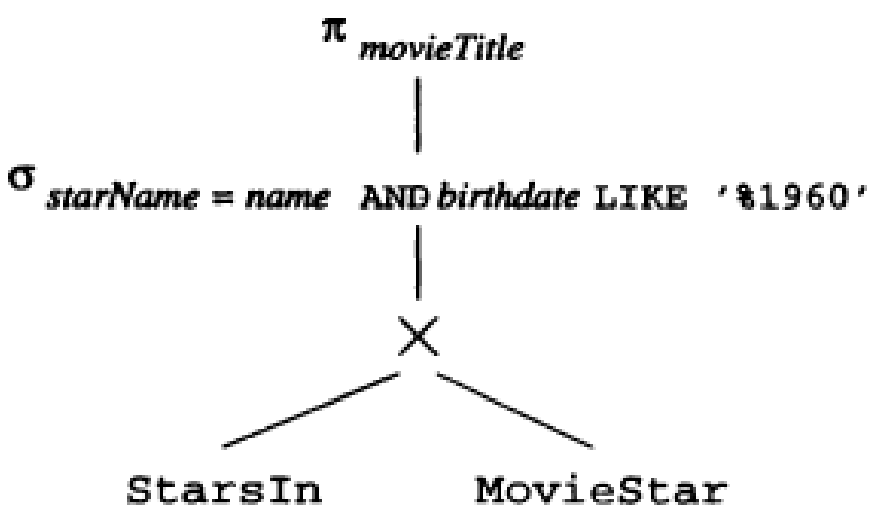
Optimización Heurística

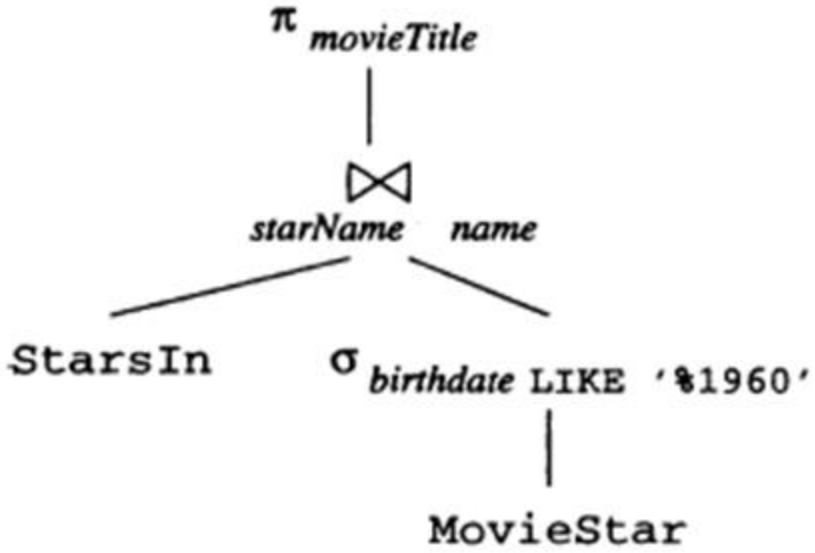




Sugerir como aplicar la optimización heurística anterior para pasar de la expresión de la izquierda a la de la derecha.

Optimización Heurística





Ejemplo de uso de optimización heurística anterior: primero dividimos las dos partes de la selección en $\sigma_{\text{starName = name}}$ y $\sigma_{\text{birthdate LIKE '%1960'}}$. La última puede ser empujada abajo en el árbol debido a que el único atributo involucrado *birthdate* es de tabla *MovieStar*. La primera condición involucra atributos de ambos lados del producto y se puede introducir proyección de todos los atributos menos *name* pues la proyección de la consulta es mucho más restrictiva. Así, podemos aplicar definición de reunión selectiva según *starName* y *name*. Haciendo todo esto obtenemos el árbol de la derecha.

Optimización Heurística

- También una optimización heurística puede referirse a operadores físicos a usar.
- **Ejemplo de optimización heurística**:
  1. Para hacer reunión de varias tablas, comenzar juntando el par de tablas cuyo resultado tiene el tamaño estimado menor; luego repetir el proceso para el resultado de esa reunión y las otras tablas en el conjunto a ser combinado.
  2. Si el plan lógico usa una selección $\sigma_{A=C}(r)$ y r tiene un índice en el atributo A, realizar escaneo de ese índice en r (vimos 2 algoritmos para esto).

Optimización Heurística

- **Ejemplo de optimización heurística (cont)**:
  3. Si un argumento de una reunión tiene un índice en atributos de la reunión para la tabla de la derecha, usar reunión de loop anidado indexada.
  4. Si los argumentos de la reunión están ordenado en los atributos de la reunión, es preferible usar reunión por mezcla.
  5. Cuando se computa la unión o intersección de 3 o más relaciones, agrupar las más pequeñas primero

Selección de orden de reunión basado en costo

- Si tengo reunión natural/selectiva de varias tablas, la optimización basada en costo es demasiado costosa, la razón es la gran cantidad de casos a examinar.
  - Para n = 3 hay 12 órdenes
  - Para n = 5 hay 1680 órdenes.
  - Para n = 7 hay 665280 órdenes.
  - Con n = 10 hay 17,6 mil millones de órdenes.
  - Con *n* tablas hay
    - (2(n-1))!/(n-1)! diferentes ordenaciones.
- **Observación**: los distintos ordenes pueden cambiar también el orden de las columnas. Pero podemos proyectar según los atributos de la expresión original de la consulta para que eso no suceda.

Selección de orden de reunión basado en costo

- Para la selección de orden de reunión natural basado en costo hay **algoritmos de programación dinámica**.
  - En lugar de generar todas las expresiones de reunión posibles, se consideran conjuntos de relaciones a reunir y optimizar.
  - Estos conjuntos se usan en tablas (como índices en ellas).
  - Estas tablas tienen muchos menos elementos que la cantidad de reuniones posibles.
  - Usando **programación dinámica** el orden de reunión de menor costo para cada subconjunto de $\{r_1, \ldots, r_n\}$ es computado solo una vez y almacenado para uso futuro.

Selección de orden de reunión basado en costo

- **Ejemplo**: si hay 4 tablas, tenemos en total:
  - (2*3)!/(3)! = 6!/3! = 720/6 = 120 casos
  - Sin embargo, la cantidad de subconjuntos de 4 elementos es 16.
  - Programación dinámica trabaja con tablas de 16 subconjuntos.
- **Ejemplo**: si hay 5 tablas hay 1680 órdenes. Sin embargo, programación dinámica trabaja con tabla de 32 subconjuntos.

Selección de orden de reunión basado en costo

- **Solución usando programación dinámica**:
  - Construimos una **tabla** con una entrada para cada subconjunto de una o más de las tablas de la reunión.
  - **En la tabla ponemos:**
    - Un índice es un conjunto de tablas de la base de datos.
    - Ponemos en una celda:
      1. El tamaño estimado de la reunión de esas tablas (en cantidad de registros).
         - Ya vimos cómo se hace esto usando factor de selectividad.
      2. El menor costo de computar la reunión de esas tablas (en cantidad de tuplas).
      3. La expresión que da lugar al menor costo.
  - La construcción de esa tabla se hace por **inducción** en el **tamaño del subconjunto.**

Selección de orden de reunión basado en costo

- **Solución usando programación dinámica (cont)**:
  - **Caso base:**
    - la entrada para una sola tabla r consiste del tamaño de r, un costo de 0 (no se computa la reunión) y la expresión que es r.
    - La entrada para un par de tablas {r_i, r_j}:
      - tiene estimación de tamaño que es el producto de tamaños de r_i y r_j multiplicado por el factor de selectividad;
      - tiene costo 0 porque no hay tablas intermedias involucradas (es como asumir que las tablas de la BD están en memoria),
      - Además, tomamos la menor de r_i y r_j como el argumento izquierdo de la expresión de la reunión natural.

Selección de orden de reunión basado en costo

- **Solución usando programación dinámica (cont)**:
  - **Inducción**:
    - Consideramos todas las maneras de particionar el conjunto actual de tablas S en dos subconjuntos disjuntos S1 y S2.
    - Para cada una de estas maneras consideramos la suma de:
      1. los mejores costos de S1 y S2.
      2. los tamaños para los resultados para S1 y S2.
    - Sea cual sea la partición que da el mejor costo, usamos esta suma como el costo de S.
    - La expresión de S es la reunión natural de las mejores expresiones para S1 y S2.

Selección de orden de reunión basado en costo

- **Solución alternativa**: modificar el algoritmo de programación dinámica anterior para tomar en cuenta algoritmos de reunión natural.
  - Cuando se computa el costo de $S1 \bowtie S2$ sumamos el costo de S1, el costo de S2 y el menor costo de juntar los dos resultados usando el mejor algoritmo disponible.
- **Complejidad**
  - La complejidad en tiempo del procedimiento puede probarse que es $O(3^n)$.
  - Con n = 10 este número es 59000 en lugar de 176 mil millones de casos a evaluar.
  - La complexidad en espacio es $O(2^n)$ – cantidad de subconjuntos de n elementos..

Selección de orden de reunión basado en costo

- **Ejemplo**: considerar la reunión natural de 4 tablas R, S, T y U. Por simplicidad asumimos que cada una tiene 1000 tuplas. Además asumimos:

$R(a,b)$	$S(b,c)$	$T(c,d)$	$U(d,a)$
$V(R,a) = 100$			$V(U,a) = 50$
$V(R,b) = 200$	$V(S,b) = 100$		
	$V(S,c) = 500$	$V(T,c) = 20$	
		$V(T,d) = 50$	$V(U,d) = 1000$

- Para los conjuntos de una tabla los tamaños, costos y mejores planes son:

	$\{R\}$	$\{S\}$	$\{T\}$	$\{U\}$
Size	1000	1000	1000	1000
Cost	0	0	0	0
Best plan	R	S	T	U

Selección de orden de reunión basado en costo

- **Ejemplo (cont.):** Ahora consideramos pares de tablas.
- El costo para cada uno es 0 porque no hay tablas intermedias en la reunión de las dos tablas (las 2 tablas son chicas y están en memoria principal).
- Hay dos posibles planes. Como todas las tablas tienen igual tamaño tomamos la primera en orden alfabético para ser el argumento izquierdo (independientemente del orden el tamaño es el mismo).
- Los tamaños de las tablas resultantes se calculan como dijimos. El resultado es:

	$\{R,S\}$	$\{R,T\}$	$\{R,U\}$	$\{S,T\}$	$\{S,U\}$	$\{T,U\}$
Size	5000	1,000,000	10,000	2000	1,000,000	1000
Cost	0	0	0	0	0	0
Best plan	$R \bowtie S$	$R \bowtie T$	$R \bowtie U$	$S \bowtie T$	$S \bowtie U$	$T \bowtie U$

Selección de orden de reunión basado en costo

Ejercicio (cont.): ahora consideramos los conjuntos de 3 tablas.

- Vemos como calcular la celda de un conjunto S de 3 tablas.
- Las particiones de un conjunto S tienen dos tablas por un lado y una tabla por el otro.
- **Ejemplo**: Para el conjunto {R, S, T} las particiones son:
  - {{R,S}, {T}}, {{R,T},{S}}, {{S, T},{R}}
- El costo asociado a una partición {S1, S2} de S con S1 de dos tablas y S2 de una tabla es el costo de la reunión de la expresión para S1 y la expresión para S2. El mismo se calcula:
  - Mejor costo de S1 + mejor costo de S2 + tamaño de S1 = 0 + 0 + tamaño de S1 = tamaño S1.
  - No contamos tamaño de S2, porque al ser tabla de la base de datos la asumimos que esta cargada en memoria.
- **Ejemplo**: volviendo a {R, S, T}, los costos de sus particiones son:
  - Costo {{R,S}, {T}} = costo $(R \bowtie S) \bowtie T$ = tamaño $R \bowtie S = 5000$
  - Costo {{R,T},{S}} = costo $(R \bowtie T) \bowtie S$ = tamaño $R \bowtie T = 1000000$
  - Costo {{S, T},{R}} = costo $(S \bowtie T) \bowtie R$ = tamaño $S \bowtie T = 2000$

Selección de orden de reunión basado en costo

Ejercicio (cont.):

- El costo de un conjunto de tres tablas S va a ser el costo más bajo para las expresiones de las particiones de S.
- **Ejemplo**: En el ejemplo anterior la partición de {R, S, T} que da el menor costo es elegida: {{S, T},{R}}, que tiene costo 2000.
- La expresión de S va a ser la reunión de las expresiones de las partes de la partición con costo mas bajo.
- **Ejemplo**: En el ejemplo anterior, para {R, S, T} la expresión va a ser:
  - Expresión {S, T} ⋈ Expresión R = (S ⋈ T) ⋈ R
  - Ya vimos que el costo de esta expresión es 2000.

Selección de orden de reunión basado en costo

Ejercicio (cont.):

- La estimación del tamaño del resultado de para un conjunto de tres tablas S es como siempre, usando factor de selectividad.
- **Ejemplo**: volviendo al ejemplo anterior, necesitamos calcular el tamaño de (S ⋈ T) ⋈ R.
  - Tamaño $|(S \bowtie T) \bowtie R| = |(S \bowtie T)| * |R| * fs((S \bowtie T).b = R.b, S \bowtie T, R)$
    $= 2000 * 1000 / \max\{100, 200\}$
    $= 2000000 / 200$
    $= 10000$
- Al final repitiendo el proceso para cada conjunto de 3 tablas obtenemos (en la primera columna esta el ejemplo hecho):

	$\{R,S,T\}$	$\{R,S,U\}$	$\{R,T,U\}$	$\{S,T,U\}$
Size	10,000	50,000	10,000	2,000
Cost	2,000	5,000	1,000	1,000
Best plan	$(S \bowtie T) \bowtie R$	$(R \bowtie S) \bowtie U$	$(T \bowtie U) \bowtie R$	$(T \bowtie U) \bowtie S$

Selección de orden de reunión basado en costo

- **Ejercicio (cont.):**
  - Ahora consideramos conjuntos de 4 tablas.
  - Para las particiones de un conjunto de 4 tablas tenemos los siguientes casos:
    - Dos conjuntos con dos tablas cada uno.
    - Un conjunto de tres tablas y un conjunto de una tabla.
  - **Ejemplo**: para el conjunto {R, S, T, U} tenemos las particiones:
    - Dos conjuntos de dos tablas: {{R, S}, {T, U}}, {{R, T}, {S, U}}, {{R, U}, {S, T}}.
    - Las expresiones de esas particiones son: (T ⋈ U) ⋈ (R ⋈ S), (R ⋈ T) ⋈ (S ⋈ U), y (S ⋈ T) ⋈ (R ⋈ U).
    - Uno de los conjuntos de tres tablas: {{R, S, T},{U}}, {{R, S, U}, {T}}, {{R, T, U}, {S}}, y finalmente {{S, T, U}, {R}}.
    - Las expresiones de esas particiones se sacan de la tabla de la filmina 43 y son: ((S ⋈ T) ⋈ R) ⋈ U, ((R ⋈ S)⋈ U) ⋈ T, ((T ⋈ U) ⋈ R) ⋈ S, y ((T ⋈ U) ⋈ S) ⋈ R.
  - Se calculan los costos asociados a las expresiones de las particiones. Son los siguientes:

Grouping	Cost
$((S \bowtie T) \bowtie R) \bowtie U$	12,000
$((R \bowtie S) \bowtie U) \bowtie T$	55,000
$((T \bowtie U) \bowtie R) \bowtie S$	11,000
$((T \bowtie U) \bowtie S) \bowtie R$	3,000
$(T \bowtie U) \bowtie (R \bowtie S)$	6,000
$(R \bowtie T) \bowtie (S \bowtie U)$	2,000,000
$(S \bowtie T) \bowtie (R \bowtie U)$	12,000

Selección de orden de reunión basado en costo

- **Ejercicio (cont.):**
  - **Ejemplo**: Calculamos el costo de {{S,T}, {R,U}}
    - Es el costo de (S ⋈ T)⋈ (R ⋈ U) porque elegimos la tabla mas pequeña como argumento izquierdo.
    - Es la suma de los tamaños y los costos de (S ⋈ T) y (R ⋈ U).
    - Los costos de los pares son 0 y los tamaños son 2000 y 10000 respectivamente.
    - O sea, que da como resultado 12000.
  - **Ejemplo**: calculamos el costo de {{R, S, U}, {T}}
    - Es el costo de reunión de expresión de {R, S, U} y T, o sea: ((R ⋈ S)⋈ U) ⋈ T
    - Es la suma del tamaño y el costo de ((R ⋈ S)⋈ U) - El tamaño de T no se cuenta, porque se asume que T ya está en memoria y el costo de T es cero.
    - Usando la tabla de filmina 43 obtenemos: 5000 + 50000 = 55000.

Selección de orden de reunión basado en costo

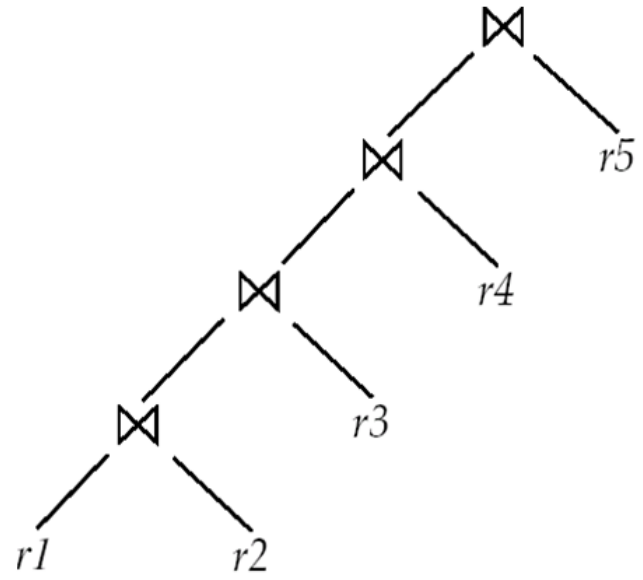
Ejercicio – Cont:

- En la tabla, el menor de todos los costos es el asociado con:
  - ((T ⋈ U) ⋈ S) ⋈R.
  - Esta expresión es la que se elige para computar la reunión natural;
  - su costo es 3000.

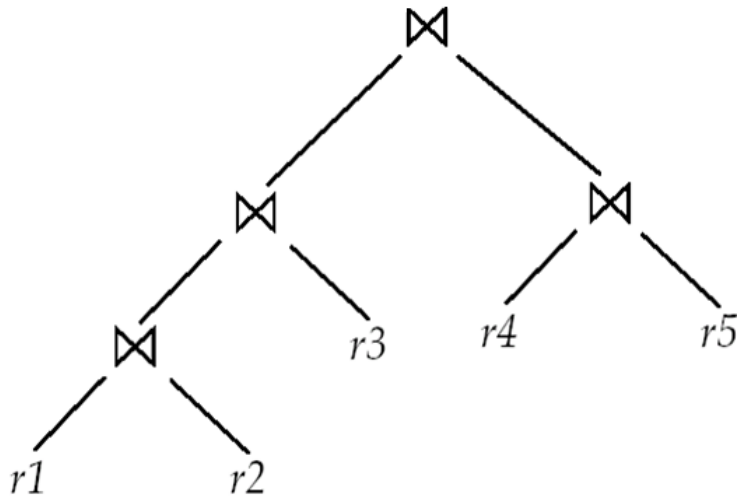
Grouping	Cost
$((S \bowtie T) \bowtie R) \bowtie U$	12,000
$((R \bowtie S) \bowtie U) \bowtie T$	55,000
$((T \bowtie U) \bowtie R) \bowtie S$	11,000
$((T \bowtie U) \bowtie S) \bowtie R$	3,000
$(T \bowtie U) \bowtie (R \bowtie S)$	6,000
$(R \bowtie T) \bowtie (S \bowtie U)$	2,000,000
$(S \bowtie T) \bowtie (R \bowtie U)$	12,000

Árboles de reunión profunda a la izquierda

- En **árboles de reunión profunda a la izquierda** el lado derecho de cada reunión natural es una tabla, no el resultado de una reunión intermedia.



(a) Left-deep join tree



(b) Non-left-deep join tree

Árboles de reunión profunda a la izquierda

- **Para encontrar el mejor árbol de reunión profunda a la izquierda para un conjunto de n tablas**:
  - **Paso inductivo:**
    - Si S tiene k tablas, para cada r en S, primero computamos la reunión de S – {r} y luego hacemos la reunión natural con r.
    - La expresión de reunión para S tiene la mejor expresión reunión para S – {r} como argumento izquierdo de la reunión final y r como el argumento derecho.
    - El costo de la reunión de S es el costo de S – {r} más el tamaño del resultado para S - {r}. Tomamos el r que da el menor costo.
    - El tamaño de S se calcula por la fórmula que usa factor de selectividad.
- **Complejidad**:
  - Si solo árboles de reunión profunda a la izquierda son considerados, el tiempo de complejidad para encontrar el mejor orden de reunión es: $O(n\ 2^n)$.
  - La complejidad en espacio permanece en $O(2^n)$.

Árboles de reunión profunda a la izquierda

- **Ejercicio**: considerar la reunión natural de 4 tablas R, S, T y U. Por simplicidad asumimos que cada una tiene 1000 tuplas. Además asumimos:

$R(a,b)$	$S(b,c)$	$T(c,d)$	$U(d,a)$
$V(R,a) = 100$			$V(U,a) = 50$
$V(R,b) = 200$	$V(S,b) = 100$		
	$V(S,c) = 500$	$V(T,c) = 20$	
		$V(T,d) = 50$	$V(U,d) = 1000$

- Aplicar el procedimiento anterior. Para los casos de una y dos tablas es igual que en el ejercicio anterior.

	$\{R\}$	$\{S\}$	$\{T\}$	$\{U\}$
Size	1000	1000	1000	1000
Cost	0	0	0	0
Best plan	R	S	T	U

	$\{R,S\}$	$\{R,T\}$	$\{R,U\}$	$\{S,T\}$	$\{S,U\}$	$\{T,U\}$
Size	5000	1,000,000	10,000	2000	1,000,000	1000
Cost	0	0	0	0	0	0
Best plan	$R \bowtie S$	$R \bowtie T$	$R \bowtie U$	$S \bowtie T$	$S \bowtie U$	$T \bowtie U$

Árboles de reunión profunda a la izquierda

- **Ejercicio – cont:**
  - Para el caso de conjuntos de tres tablas, el resultado coincide con el ejercicio anterior (pues son todos casos de reunión profunda a la izquierda).

	$\{R, S, T\}$	$\{R, S, U\}$	$\{R, T, U\}$	$\{S, T, U\}$
Size	10,000	50,000	10,000	2,000
Cost	2,000	5,000	1,000	1,000
Best plan	$(S \bowtie T) \bowtie R$	$(R \bowtie S) \bowtie U$	$(T \bowtie U) \bowtie R$	$(T \bowtie U) \bowtie S$

  - Para el conjunto de 4 tablas {R, S, T, U} tenemos las particiones:
    - {{R, S, T},{U}}, {{R, S, U}, {T}}, {{R, T, U}, {S}}, y finalmente {{S, T, U}, {R}}.
    - Usando la figura anterior las expresiones para estas particiones son: ((S ⋈ T) ⋈ R) ⋈ U, ((R ⋈ S)⋈ U) ⋈ T, ((T ⋈ U) ⋈ R) ⋈ S, y ((T ⋈ U) ⋈ S) ⋈ R.

Árboles de reunión profunda a la izquierda

- **Ejercicio – cont:**
  - Calculamos de las 4 expresiones anteriores:

Grouping	Cost
$((S \bowtie T) \bowtie R) \bowtie U$	12,000
$((R \bowtie S) \bowtie U) \bowtie T$	55,000
$((T \bowtie U) \bowtie R) \bowtie S$	11,000
$((T \bowtie U) \bowtie S) \bowtie R$	3,000

  - el menor de todos los costos es el asociado con ((T ⋈ U) ⋈ S) ⋈R, o sea 3000.
  - Cond i = ((T ⋈ U) ⋈ S). Ai == R. Ai, con Ai en {a, b}.
  - Asumimos independencia de las condiciones. Entonces:
  - |((T ⋈ U) ⋈ S) ⋈R|

    = |(T ⋈ U) ⋈ S)| * |R| * fs (cond 1 ∧ cond 2, T ⋈ U) ⋈ S, R)

    = 2000 * 1000 * fs (cond 1, (T ⋈ U) ⋈ S, R) * fs (cond 2, (T ⋈ U) ⋈ S, R)

    = 2000000 * 1/100 * 1/ 200 = 2000000 / 20000 = 100

Enfoques híbridos

- **Enfoques híbridos**: Algunos sistemas gestores de BD combinan heurísticas con optimización parcial basada en costo.
  - Enfoques de optimización de consultas que aplican elecciones heurísticas de plan para algunas partes de la consulta con elección basada en costo basada en la generación de planes alternativos para otras partes de la consulta han sido adoptados en varios SGBD.
- **Ejemplo**: Muchos optimizadores siguen un enfoque basado en usar transformaciones heurísticas para manejar construcciones diferentes de reuniones y aplican el algoritmo de orden de reunión optimizado por costo para subexpresiones que involucran solo reuniones y selecciones.
  - Los detalles de tales heurísticas son específicos a optimizadores individuales.

Enfoques híbridos

- **Ejemplo**: Varios optimizadores consideran solo reunión profunda a la izquierda más heurísticas que empujan selecciones y proyecciones hacia abajo en el árbol de consulta.
  - Esto reduce la complejidad de la optimización y genera planes que permiten evaluación usando tubería.