Teórico 7:

Optimización de Consultas

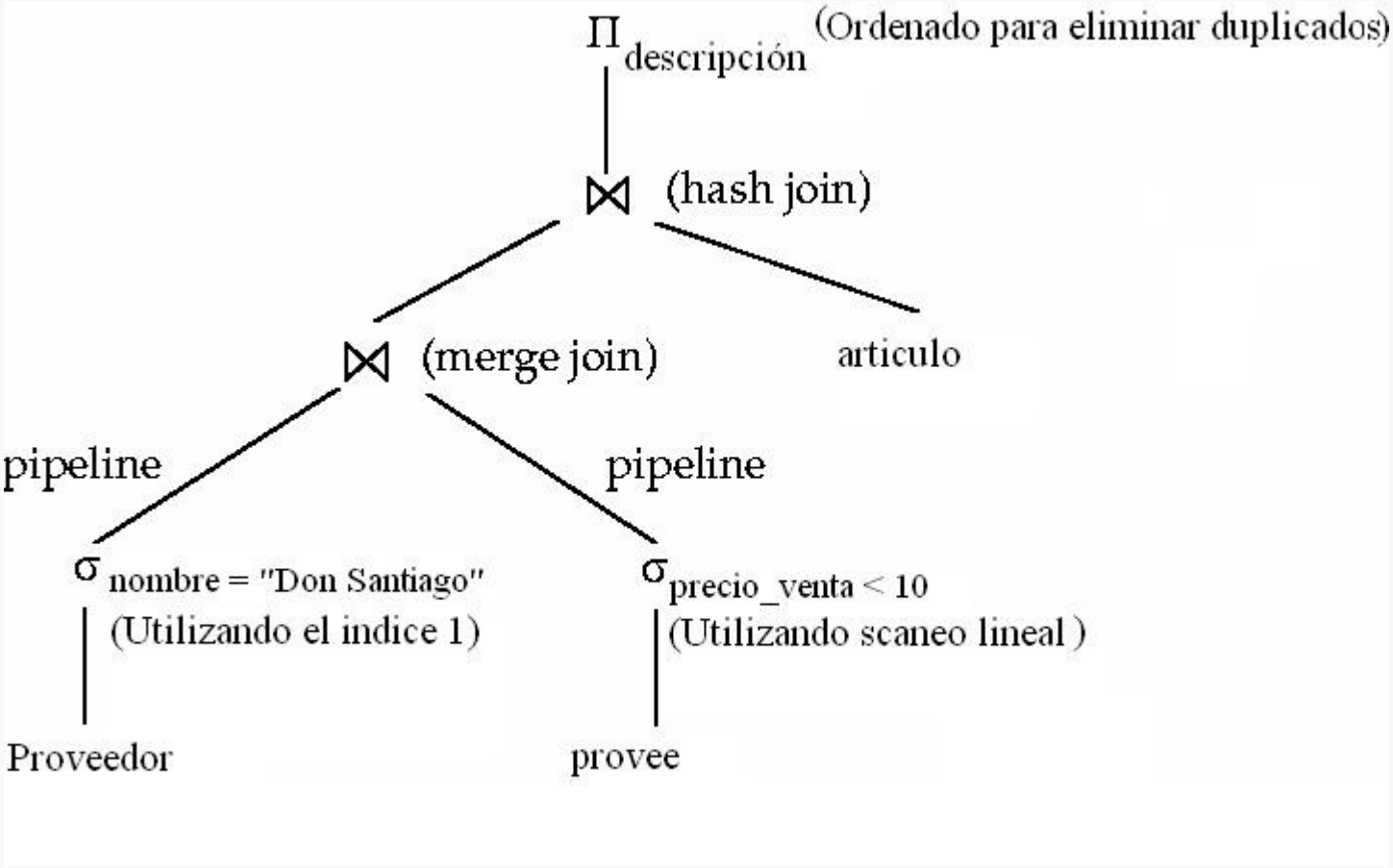
Introducción

- Hay diferentes alternativas en la evaluación de un query.
  - Existen expresiones equivalentes.
  - Existen diferentes algoritmos para cada operación.

CC BY NC SA

Introducción (Cont.)

- Un plan de evaluación define cual será el algoritmo a usar en cada operación y como se coordina la ejecución de las operaciones.



Introducción (Cont.)

- La diferencia de los costos entre dos planes de evaluación puede ser enorme
- La generación de planes de evaluación en la optimización basada en costos tiene tres pasos:
  1. Generar expresiones que sean lógicamente equivalentes a la expresión dada aplicando las reglas de equivalencia
  2. Estimar el costo de cada plan
  3. Elegir el plan más barato basado en los costos.
- La estimación del costo del plan se basa en:
  - Información estadística sobre las relaciones.
    - Por ejemplo, numero de tuplas, número de valores distintos para un atributo.
  - Estimación de estadísticas para resultados intermedios

cc BY NC SA

Generación de Expresiones Equivalentes

Transformación de Expresiones Relacionales

- Dos expresiones del álgebra se dicen *equivalentes* si las dos expresiones generan el mismo conjunto de tuplas para cualquier instancia de la base de datos.
- Tener en cuenta que en SQL se manejan multisets.
  - Dos expresiones del álgebra se dicen *equivalentes* si las dos expresiones generan el mismo multiset de tuplas para cualquier instancia de la base de datos.
- Si dos expresiones son equivalentes pueden ser reemplazadas una por otra en cualquier expresión que contenga una de ellas.

Reglas de Equivalencia

1. Las operaciones de selección conjuntivas se pueden dividir en una secuencia de selecciones individuales.

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

2. Las operaciones de selección son conmutativas.

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

3. Sólo es necesaria la última operación de una secuencia de proyecciones, las demás pueden omitirse.

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

7. Las selecciones pueden combinarse con los productos cartesianos y con theta joins.

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

   β. $\sigma_{\theta 1}(E_1 \bowtie_{\theta 2} E_2) = E_1 \bowtie_{\theta 1 \wedge \theta 2} E_2$

Reglas de Equivalencia (Cont.)

5. La operaciones de Theta-join y join natural son conmutativas.

$$E_1 \bowtie_\theta E_2 = E_2 \bowtie_\theta E_1$$

6. (a) Los join naturales son asociativos:

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

aplicación

(b) Los Theta joins son asociativos en 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 atributos sólo de E_2 y E_3.

Reglas de Equivalencia (Cont.)

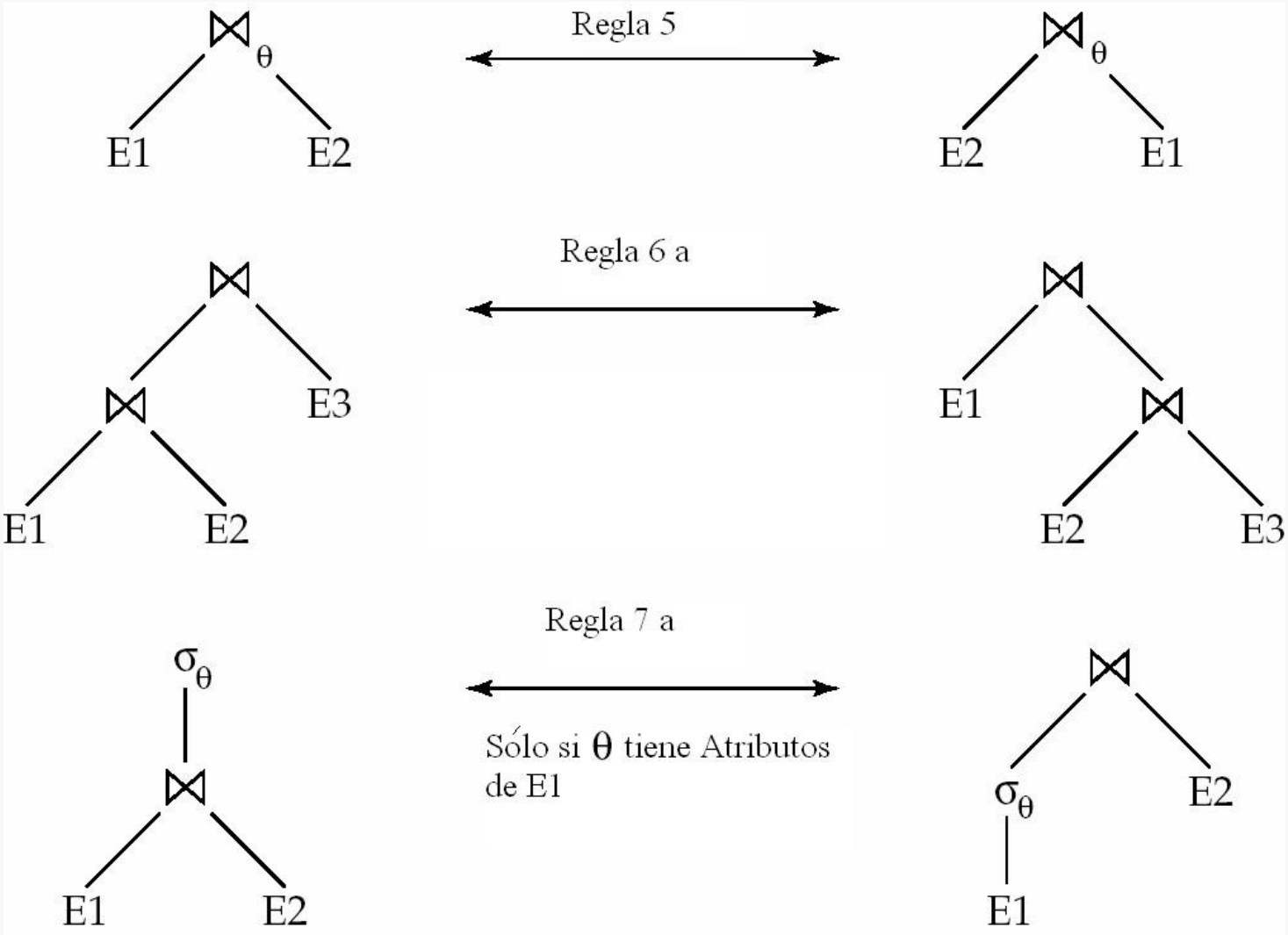
7. La operación de selección se distribuye sobre un theta join bajo las siguientes condiciones:
   (a) Cuando todos los atributos en θ_0 Involucran sólo los atributos de una de las expresiones (E_1) que están involucradas en el Join.

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

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

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

Reglas de Equivalencia vistas gráficamente



Reglas de Equivalencia (Cont.)

8. La Operación de proyección distribuye sobre la operación de theta join de la siguiente manera:
   - Sean L_1 y L_2 atributos de E_1 y E_2, respectivamente.

   (a) Si θ involucra 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 un join $E_1 \bowtie_\theta E_2$.
   - Sea L_3 los atributos de E_1 que están involucrados en la condición θ del join, pero no están en $L_1 \cup L_2$, y
   - Sea L_4 los atributos de E_2 que están involucrados en la condición θ del join, pero no están en $L_1 \cup L_2$.

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

Reglas de Equivalencia (Cont.)

9. Las operaciones de unión e intersección son conmutativas

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

- La operación de resta de conjuntos no es conmutativa.

19. Las operaciones de unión e intersección son asociativas.

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

21. La operación de selección distribuye sobre $\cup$, $\cap$ y –.

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

y de manera similar para $\cup$ y $\cap$ en lugar de –

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

y de manera similar para $\cap$ en lugar de –, pero no para $\cup$

12. La operación de proyección distribuye sobre la unión

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

Ejemplos

De ahora en más los ejemplos de consultas SQL estarán hechos sobre el siguiente esquema de base de datos.

articulos(Nart, desc, precio, cant, stock_min, stock_max)

Provee(Nprov,Nart, precio_venta)

Proveedor(Nprov, nombre, direccion)

Cliente(Ncli, nombre, direccion)

Compran(Ncli, Nart)

cc BY NC SA

Ejemplos de transformaciones

- Query: Encontrar la descripción de los artículos que provee el proveedor con nombre “Super Vea”.

$$\Pi_{descripcion}(\sigma_{proveedor.nombre = \text{“Super Vea”}}(proveedor \bowtie (provee \bowtie articulo)))$$

- Aplicando la Regla 7a

$$\Pi_{descripcion}((\sigma_{proveedor.nombre = \text{“Super Vea”}}(proveedor)) \bowtie (provee \bowtie articulo))$$

- De esta forma se generan tablas intermedias más pequeñas.

Ejemplos (Cont.)

- Query: Encontrar la descripción de los artículos con precio_venta < $10 que provee el proveedor con nombre “Super Vea”.

  $\Pi_{descriocion}(\sigma_{proveedor.nombre = \text{“Super Vea”} \wedge \text{precio\_venta} < 10}(proveedor \bowtie (provee \bowtie articulo)))$

- Transformación utilizando la Regla 6 a

  $\Pi_{descripcion}((\sigma_{proveedor.nombre = \text{“Super Vea”} \wedge \text{precio\_venta} < 10}(proveedor \bowtie provee)) \bowtie articulo)$

- La subExpresion de la selección se puede transformar Regla 1 en :

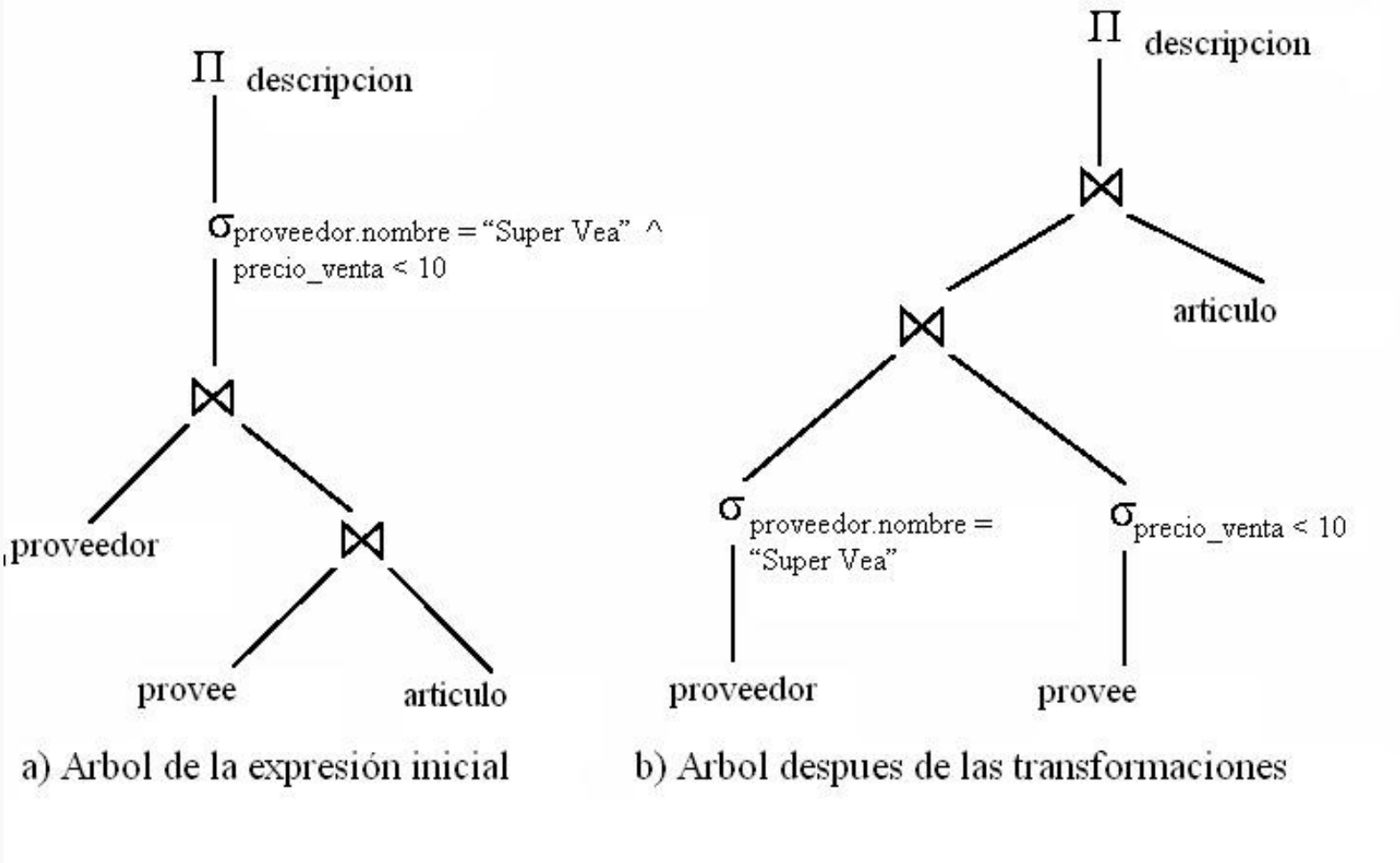
  $\sigma_{proveedor.nombre = \text{“Super Vea”}}(\sigma_{\text{precio\_venta} < 10}(proveedor \bowtie provee))$

Aplicando 2 veces la regla 7.a a la subexpresión anterior, tenemos:

$\sigma_{proveedor.nombre = \text{“Super Vea”}}(proveedor) \bowtie \sigma_{\text{precio\_venta} < 10}(provee)$

cc BY NC SA

Gráficamente



Orden de JOIN

- Para toda instancia de relación $r_1, r_2, y\, r_3$,

$$(r_1 \bowtie r_2) \bowtie r_3 = r_1 \bowtie (r_2 \bowtie r_3)$$

(Asociatividad)

- Si $r_2 \bowtie r_3$ es más grande que $r_1 \bowtie r_2$, se elige

$$(r_1 \bowtie r_2) \bowtie r_3$$

de esta forma la relación temporal es más pequeña.

CC BY NC SA

Ejemplo de Orden del Join(Cont.)

- Considere la siguiente expresión

$$\Pi_{descripcion}\ ((\sigma_{nombre = \text{“Don Santiago”}}(\text{proveedor}))\bowtie(\text{provee} \bowtie \text{articulo}))$$

- Se podría computar *provee* $\bowtie$ *articulo* primero, y luego hacer el join con

$$\sigma_{nombre = \text{“Don Santiago”}}(proveedor)$$

pero *provee* $\bowtie$ *articulo* es una relación muy grande.

- Sólo algunos artículos son provistos por el proveedor “Don Santiago”
  - Es mejor primero computar

$$\sigma_{nombre = \text{“Don Santiago”}}(proveedor)\bowtie provee$$

.

cc BY NC SA

Enumeración de Expresiones Equivalente

- Los optimizadores de Query utilizan reglas de equivalencia para generar expresiones equivalentes de manera sistemática.
- Se pueden encontrar una cantidad muy grande de expresiones equivalentes.
- Encontrar todas la expresiones equivalentes es muy costoso en tiempo y espacio.

cc BY NC SA

Estimación de Costos

- Los costos de alguno de los operadores se vieron anteriormente
  - Se necesitan estadísticas de las relaciones de entrada, por ejemplo número de tuplas, tamaño de las tuplas, etc.
  - Los motores de base de datos llevan estas estadísticas en sus catálogos.

- Las estadísticas no se actualizan con cada modificación de una relación, ya que sería una sobrecarga grande en el sistema, las actualizaciones se hacen cunado el motor de base de datos tiene poca carga.

CC BY NC SA

Elección de los planes de evaluación

- Para definir un plan de evaluación se debe tener en cuenta la interacción de diferentes algoritmos de evaluación
  - La elección del algoritmo más barato para una subexpresion, no garantiza que el costo total de evaluación de la expresión completa. Por ejemplo:
    - Merge-join puede ser más costoso que un hash-join, pero provee una salida ordenada que puede hacer más eficiente operaciones que utilizan su resultado.
- Los optimizadores de query se basan en las siguientes dos visiones:
  1. Elección del plan basados en costos.
  2. Elección del plan basados en heurísticas.

CC BY NC SA

Optimización basada en costo

- Los optimizadores basados en costos generan una gama de planes de evaluación a partir de una consulta dada empleando reglas de equivalencia y eligen el de costo mínimo.

- Consideremos encontrar el menor costo para $r_1 \bowtie r_2 \bowtie \ldots r_n$.

- Hay $(2(n-1))!/(n-1)!$ Diferentes ordenes para ejecutar el join de la expresión. Con $n = 5$, el número es 1680, con $n = 10$, el número es 17.600 millones

- No es necesario generar todos los ordenes de los join. Se puede utilizar un algoritmo de programación dinámica que computa el menor costo para $\{r_1, r_2, \ldots r_n\}$ sólo una vez, lo almacena y lo reutiliza luego.

CC BY NC SA

Optimización Heurística

- La optimización basada en costos es cara, incluso con técnicas de programación dinámica.
- Los RDBMS pueden usar heurísticas para reducir el número de posibilidades a evaluar basada en costos.
- La optimización heurística transforma el árbol del Query utilizando un conjunto de reglas que generalmente (pero no en todos los casos) mejoran la velocidad de ejecución, algunas de estas reglas son:
  - Realizar las selecciones tan pronto como sea posible (reduce el número de tuplas)
  - Realizar las proyecciones tan pronto como sea posible (reduce el número de atributos)
  - Realizar tan pronto como sea posible las selecciones y joins más restrictivos (es decir los que den resultados más chicos) antes de otras operaciones similares.

cc BY NC SA

Estadísticas para Estimación de Costos

Información Estadística para Estimación de Costos

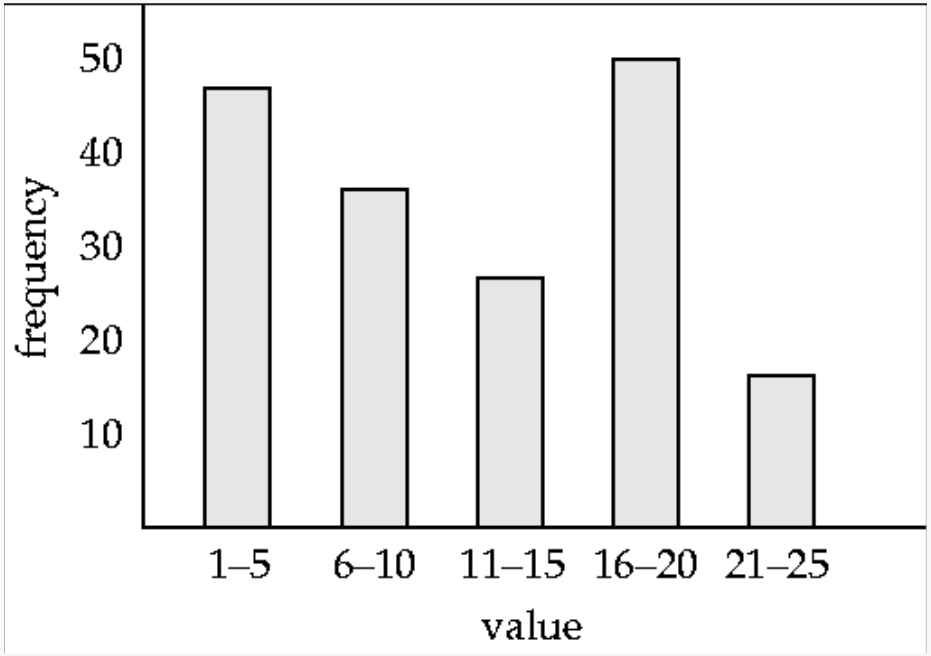
- n_r: número de tuplas en una relación *r.*
- b_r: número de bloques conteniendo tuplas de *r.*
- t_r: tamaño de una tupla de *r.*
- f_r: factor de bloque de *r* — es decir, el número de tuplas de r que entran en un bloque.
- *V(A, r):* número de valores distintos de un atributo A en una relación *r; es* igual a tamaño de $\prod_A(r)$.
- Suponiendo que las tuplas del archivo se almacenan juntas se cumple:

$$b_r = \left\lceil \frac{n_r}{f_r} \right\rceil$$

BY NC SA

Histogramas

- Histograma para el atributo edad de la tabla persona



- Histogramas equi-width: divide el rango de valores en rangos de igual tamaño
- Histogramas equi-depth, ajustan los rangos de para que cada rango tenga la misma cantidad de valores

Estimación del tamaño de la Selección

- $\sigma_{A=V}(r)$
  - n_r / *V(A,r)* : número de registros que satisfacen la selección.
  - Condición de igualdad en un atributo clave: el tamaño estimado es 1.
- $\sigma_{A \leq V}(r)$ (razonamiento similar para >)
  - Dado c que denota el número estimado de tuplas que satisfacen la condición.
  - Si min(A,r) y max(A,r) están disponible en el catalogo
    - c = 0 if v < min(A,r)
    - c = n_r if v >= max(A,r)
    - c = $n_r . \frac{v - \min(A,r)}{\max(A,r) - \min(A,r)}$
  - Si hay histogramas disponibles se puede refinar la estimación
  - Si no hay información estadística se asume $c = n_r / 2$.

Estimación de tamaño de Selecciones Complejas

- Sea s_i el tamaño estimado de la selección $\sigma\,\theta_i$ (r)
- La probabilidad que una tupla en la relación *r* satisfaga θ_i esta dado por s_i/n_r .Esta probabilidad se denomina ***selectividad*** de la selección $\sigma\,\theta_i$ (r) .
- **Conjunción:** $\sigma_{\theta 1 \wedge \theta 2 \wedge \ldots \wedge \theta n}(r)$. Asumiendo independencia, la estimación de las tuplas en el resultado:

$$n_r * \frac{s_1 * s_2 * \ldots * s_n}{n_r^n}$$

Disyunción: $\sigma_{\theta 1 \vee \theta 2 \vee \ldots \vee \theta n}(r)$. Estimación del número de tuplas:

$$n_r * \left(1 - (1 - \frac{s_1}{n_r}) * (1 - \frac{s_2}{n_r}) * \ldots * (1 - \frac{s_n}{n_r})\right)$$

- **Negación:** $\sigma_{\neg\theta}(r)$. Estimación del número de tuplas:

n_r – *tamaño*($\sigma_\theta(r)$)

Estimación del tamaño de los Joins

- El producto cartesiano r x s contiene $n_r * n_s$ tuplas; cada tupla ocupa $t_r + t_s$ bytes.

- Si $R \cap S = \varnothing$, entonces $r \bowtie s$ es lo mismo que r x s.

- Si $R \cap S$ es una clave de R, entonces cada tupla de s se mezclará con a lo sumo una tupla de r
  - Por lo tanto, el número de tuplas en $r \bowtie s$ será menor o igual al número de tuplas en s.

Estimación del Tamaño de Joins (Cont.)

- Si $R \cap S = \{A\}$ y A no es una clave de R o S.
  Si asumimos que todas las tuplas t en r producen tuplas en $R \bowtie S$, el número de tuplas en $R \bowtie S$ se estima de la siguiente manera:

$$\frac{n_r * n_s}{V(A,s)}$$

  Si se invierten r y s la estimación será:

$$\frac{n_r * n_s}{V(A,r)}$$

  La menor estimación es probablemente la mejor estimación.

- Las estimación se puede mejorar si hay histogramas disponible, en ese caso se debe usar la misma formula para cada celda del histograma.

CC BY NC SA

Ejemplo

Ejemplo del join:

$proveedor \bowtie provee$

Información de catalogo:

- n_{provee} = 10,000.
- f_{provee} = *25,* implica que

  b_{provee} =10000/25 = 400.
- $n_{proveedor}$ = 500.
- $f_{proveedor}$ = 50, implica que

  $b_{proveedor}$ = 500/50 = 10.
- Variabilidad de los atributos del join:
  - *V(NProv, provee) = 400,* nprov foreing key a proveedor

    hay 100 proveedores que no proveen ningún artículo
  - *V(NProv, proveedor)* = 500, esto implica que, en promedio, cada proveedor provee 20 artículos.

Ejemplo (Cont.)

- Utilizando la información que proveedor $\bowtie$ provee es clave primaria de proveedor el tamaño del join es el número de tuplas de provee, o sea 10000.

- Si no se utiliza la información de claves las estimaciones serian:
  - Una 500 * 10000/ 400 = 12500
  - La segunda 500 * 10000/ 500 = 10000

- Como se puede la menor estimación es igual a la calculada anteriormente.

cc BY NC SA

Estimación de otras Operaciones

- Proyección: tamaño estimado para $\prod_A(r)$ = $V(A,r)$
- Operaciones de conjuntos:
  - Para uniones e intersecciones de selecciones sobre la misma relación se reescribe y se usan las estimaciones de las selecciones:
    - Ejemplo $\sigma_{\theta 1}(r) \cup \sigma_{\theta 2}(r)$ puede ser reescrito $\sigma_{\theta 1 \vee \theta 2}(r)$
  - Para operaciones sobre diferentes relaciones:
    - Tamaño de $r \cup s$ = tamaño(r) + tamaño(s).
    - Tamaño de $r \cap s$ = min(tamaño(r) + tamaño(s)).
    - Tamaño de $r - s$ = tamaño(r).
    - La tres estimaciones anteriores son imprecisas, pero proveen cotas superiores .

Técnicas Adicionales de Optimización

- Subqueries Anidados
- Vistas Materializadas

Optimización de Queries Anidados

- Ejemplo de queries anidados:
  select *descripcion* **from** *articulo*
  where exists (**select** *
  from *provee*
  where *provee.nArt = articulo.nArt*)

- SQL conceptualmente trata a los subqueries anidados en la cláusula **where** como funciones que toman parámetros y devuelven un valor o conjunto de valores
  - Los parámetros son variables del query del nivel externo que son utilizadas en el subquery anidado; estas variables se llaman variables de correlación. Para el ejemplo anterior articulo.nArt
- Conceptualmente, el query anidado es ejecutado una vez por cada tupla en el producto cartesiano del nivel externo en la cláusula **from.**
  - Esta evaluación se denomina **evaluación correlacionada**
  - Note: Las otras condiciones del where puede ser usadas para computar un join , en lugar del producto cartesiano) antes de ejecutar el subquery anidado.

Optimización de Subqueries Anidados (Cont.)

- La evaluación correlacionada puede ser muy ineficiente
  - Un gran número de llamadas pueden ser hechas a los queries anidados.

- Los optimizadores SQL intentan transformar los subqueries anidados a joins.

CC BY NC SA

Vistas Materializadas

- Una **vista materializada** es una vista cuyo contenido es computado y almacenado.

- Considere la vista

  create view *precioMenor* (*nArt, precio_venta*) **as**
  select *nArt*, **min**(*precio_venta*)
  from *provee* **group by** *nArt*

CC BY NC SA

Mantenimiento de Vistas Materializadas

- El trabajo de mantener actualizadas las vistas materializadas se denomina **mantenimiento de vistas materializadas.**
- Las vistas materializadas pueden ser mantenidos por recomputaciones en cada actualización de las tablas involucradas en la vista.
- La mejor opción es usar el **mantenimiento incremental de vistas**
  - Los cambios en las relaciones de la base de datos son usados para computar los cambios en las vistas materializadas y así mantenerlas actualizadas
- El mantenimiento de vistas puede ser hecho por:
  - Manualmente definiendo triggers en insert, delete, y update de las relaciones involucradas en la vista.
  - Recomputando periódicamente (Por ejemplo cada noche)
  - Estos mecanismos son implementados por los motores de base de datos, sin que el usuario tenga que hacer ninguna acción

cc BY NC SA

Ejemplo en Oracle

El siguiente ejemplo permite crear una vista materializada en Oracle que se actualiza cada vez que se hace el commit de una transacción:

- CREATE MATERIALIZED VIEW LOG ON *cliente*;

- CREATE MATERIALIZED VIEW vista_dni_cliente
  BUILD IMMEDIATE
  REFRESH FAST ON COMMIT
  AS
  SELECT dni FROM *cliente* ;

CC BY NC SA