TEMA 6 – Optimización de consultas en bases de datos

1. Introducción

Consulta en BD relacionales:

- ✓ Expresiones complejas en SQL (**SELECT**), es un lenguaje muy declarativo
- ✓ Es posible reescribir una consulta SQL en otra equivalente más económica
- ✓ Es posible usar distintas **estrategias de ejecución** de la consulta.

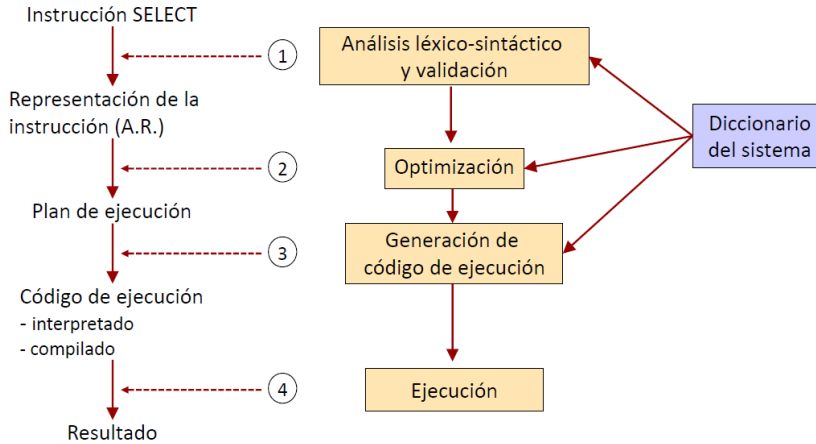
Optimización de consultas, dos estrategias:

1) **Basadas en la aplicación de reglas heurísticas:** transformación de la consulta en otra equivalente, que (heurísticamente) es más económica, utilizando reglas de transformación sintácticas.
   - No se usa información física (estadística) sobre la BD
   - Las reglas pueden ser elegidas o alteradas por el administrador de la BD

2) **Basadas en la estimación de costes:** elección de la estrategia de ejecución más económica, utilizando información física sobre la BD.
   - Se usa información física (estadística) sobre la BD
   - No hay intervención del administrador de la BD

Etapas en la ejecución de una consulta SQL (SELECT)

1. Análisis léxico-sintáctico de la instrucción SELECT
2. Planificación de la secuencia de operaciones (optimizada) que resuelve la instrucción
3. Generación del código de ejecución de la instrucción
4. Ejecución del código.



2. Traducción de SQL a Álgebra Relacional (AR)

1. Descomposición de la consulta en bloques de consulta.

"Obtener el nombre de las asignaturas que superan en créditos de teoría a todas las asignaturas del departamento de código DSIC."

```
SELECT nombre
FROM Asignatura
WHERE teoría > (SELECT MAX(teoría)
                FROM Asignatura
                WHERE cod_dep='DSIC')
```

C =
```
SELECT MAX(teoría)
FROM Asignatura
WHERE cod_dep='DSIC'
```
Bloque interno

```
SELECT nombre
FROM Asignatura
WHERE teoría > C
```
Bloque externo

2. Traducción de cada bloque a una expresión de Álgebra Relacional (AR)
3. Composición de las expresiones obtenidas.

C =
```
SELECT MAX(teoría)
FROM Asignatura
WHERE cod_dep='DSIC'
```
$$\Im_{MAX(teoría)}\left(\sigma_{cod\_dep='DSIC'}(Asignatura)\right)$$

```
SELECT nombre
FROM Asignatura
WHERE teoría > C
```
$$\pi_{nombre}\left(\sigma_{teoría>C}(Asignatura)\right)$$

3. Algoritmos de ejecución de las operaciones básicas de consulta

Operadores del Álgebra Relacional:

- Ordenación externa
- Selección (σ_L)
- Proyección (π_L) (eliminando duplicados)
- Concatenación interna ($\bowtie_{A=B}$) (**INNER JOIN**)
- Producto cartesiano ($\times$)
- Operaciones conjuntistas ($\cup$ $\cap$ $-$)
- Operaciones de agregación
- Concatenación externa (**OUTER JOIN**)

Ordenación externa (en memoria secundaria)

Algoritmo para realizar ordenaciones (de filas) en ejecución de algunas operaciones de AR: **ORDER BY**, **GROUP BY**, concatenación, unión e intersección, proyección con **DISTINCT**.

Algoritmo ordenar-fusionar (sort-merge)

Ordenación de un fichero de tamaño n_P bloques. Espacio en MP disponible: n_B bloques.

- **División:** se divide el fichero en subficheros temporales (***runs***) cada uno con n_B bloques.
  - Número de *runs* iniciales: $\boldsymbol{n_R = \lceil n_P / n_B \rceil}$
- **Ordenación:** se ordena cada *run* internamente en MP y se graba en disco como run ordenado.
- **Fusión:** los *runs* ordenados se fusionan en MP en una o más iteraciones.
  - **Grado de fusión $\boldsymbol{d_M}$:** número de *runs* que se pueden fusionar en una iteración

    $$\boldsymbol{d_M = \min(n_B - 1, n_R)}$$
  - **Número de iteraciones:** $\lceil \log_{d_M} n_R \rceil$

Número de accesos a disco: $2 \cdot n_P \cdot \left(1 + \lceil \log_{d_M} n_R \rceil\right)$

Selección

Condición simple: atr = valor, atr ≠ valor, atr *op_com* valor, donde *op_com* ∈ {<, >, ≤, ≥}

- S1: búsqueda lineal (cualquier organización del fichero), condiciones 1, 2 y 3.
- S2: búsqueda binaria (fichero ordenado), condiciones 1 y 3.
- S3: búsqueda con direccionamiento calculado (fichero disperso), condición 1.
- S4: búsqueda con índice (cualquier organización del fichero), condiciones 1 y 3.

Condición compleja conjuntiva: $(\text{atr}_1\ op_1\ \text{valor}_1) \wedge (\text{atr}_2\ op_2\ \text{valor}_2) \wedge \ldots \wedge (\text{atr}_n\ op_n\ \text{valor}_n)$

- S5: uso de un índice simple: si existe un índice definido sobre un atributo de la condición. *Sobre los registros seleccionados se comprueba el resto de condiciones*.
- S6: uso de un índice compuesto: si existe un índice definido sobre dos o más atributos. *Sobre los registros seleccionados se comprueba el resto de condiciones*.
- S7: uso de varios índices: si existen varios índices definidos sobre atributos de la condición, con punteros a registros, se usan para recuperar varios conjuntos de punteros a registros, y se obtiene su intersección. *Sobre los registros seleccionados se comprueba el resto de condiciones*.

Condición compleja disyuntiva: $(\text{atr}_1\ op_1\ \text{valor}_1) \vee (\text{atr}_2\ op_2\ \text{valor}_2) \vee \ldots \vee (\text{atr}_n\ op_n\ \text{valor}_n)$

- S8: si algún atributo que participa en la condición no tiene índice, se hace búsqueda lineal.
- S9: si todos los atributos tienen índices definidos, se usan los índices para recuperar conjuntos de punteros a registros, y se obtiene su unión.

Concatenación natural: $R \bowtie_{R.A=S.B} S$

Métodos de concatenación:

1) **Concatenación por bucle anidado:** para cada registro r de R (bucle externo) recorrer todos los registros s de S (bucle interno) y comprobar si r.A = s.B.

2) **Concatenación por bucle único:** si existe un índice definido sobre uno de los atributos de la concatenación (ej. B), para cada registro r de R (bucle único), usar el índice sobre B para obtener todos los registros s de S coincidentes r.A = s.B.

    - **Si una tabla está dispersa sobre un atributo de la concatenación (ej.** S sobre B), para cada registro r de R (bucle único), usar la función de dispersión con valor r.A para obtener los registros s de S coincidentes r.A = s.B

3) **Concatenación por ordenación:** si R y S son ficheros ordenados según A y B: recorrer R y S en paralelo (bucle único), buscando pares de registros coincidentes (si R y S no están ordenados primero ordenar).

4) **Concatenación por direccionamiento calculado:**

    - Crear una copia, implementada con direccionamiento calculado, de la tabla más pequeña (ej. S), con una función de dispersión f sobre B.

    - Recorrer R en un bucle, y para cada registro r de R aplicar f sobre A para acceder a un bloque de la copia de S donde buscar los registros coincidentes (r.A = s.B).

Proyección: $\pi_{<lista\ de\ atributos>} R$

Casos:

- **<*lista de atributos*> incluye una clave de R:** no aparecen filas duplicadas en el resultado
  Número de registros de la proyección = número de registros de R

- **<*lista de atributos*> no incluye una clave de R:** pueden aparecer filas duplicadas
  Eliminación de duplicados:

    a) Ordenación del resultado de la operación y eliminación de los duplicados.

    b) Direccionamiento calculado:

        - Crear un fichero con direccionamiento calculado sobre todos los atributos de la lista para almacenar el resultado.

        - Antes de insertar un nuevo registro en el fichero se compara con los que ya existen en el cubo direccionado, para evitar los duplicados.

Operaciones conjuntistas: ∪ ∩ −

- **Ordenación:** ordenación de las dos tablas. Recorrido en paralelo de las tablas ordenadas para ejecutar la operación.
- **Direccionamiento calculado:** crear una copia, implementada con direccionamiento calculado, de la tabla más pequeña (p.e. S) con función de direccionamiento f sobre sus atributos. Recorrer R, aplicar f a cada registro para buscar registros idénticos en S, y ejecutar la operación.

Operaciones conjuntistas: ×

Operación muy costosa: doble bucle. Generar todos los registros combinados posibles:

- *registro(R × S) = registro(R) + registro(S)*
- *card(R × S) = card(R) × card(S)*

Evitar esta operación mediante la situación por otras operaciones equivalentes.

Funciones agregadas: MIN, MAX, AVG, SUM, COUNT

Consulta de tabla completa:

- Recorrido lineal sobre la tabla y se calcula la función agregada sobre el atributo.
- Uso de un índice definido sobre el atributo de agregación:
  - MAX o MIN: búsqueda de la entrada mayor o menor en el índice (*en un índice B^+, se busca la hoja más a la derecha o más a la izquierda*).
  - COUNT, AVG y SUM: si el índice es denso se puede calcular con un recorrido completo del índice (*en un índice B^+, recorrido de todas las hojas*)
  - COUNT, AVG y SUM con DISTINCT: si el índice es disperso con una entrada por valor se puede calcular con un recorrido completo del índice (*en un índice B^+, recorrido de todas las hojas*).

Consulta agrupada:

- La tabla se ordena por los atributos de agrupación.
- La tabla se divide en subconjuntos de filas con el mismo valor en los atributos de agrupación.
- Se calcula la función agregada en cada subconjunto.

Concatenación externa (OUTER JOIN)

Tipos: **`LEFT`**, **`RIGHT`** y **`FULL`**. Similar al **`INNER JOIN`**, pero las filas que no combinan aparecen en el resultado final combinándose con una fila de valores nulos.

→ Se pueden usar los **métodos para el `INNER JOIN`** con alguna variación.

Ejemplo: `R LEFT JOIN S`

- R se coloca en el bucle externo o en el bucle único.
- Si el registro de R se combina con registros de S, en el resultado aparecen las combinaciones.
- Si no, el registro de R también aparece en el resultado combinado con una fila de nulos.

4. Optimización basada en reglas heurísticas

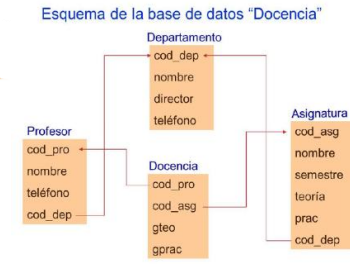
Árbol de consulta

Árbol de consulta: estructura de datos (árbol) que representa una expresión de Álgebra Relacional. Las tablas de la consulta de representan como nodos hojas y los operadores de AR se representan como nodos internos.

Ejecución de un árbol de consulta:

1) Empezar con los nodos hoja.
2) Ejecutar un nodo interno (operador) cuando sus operandos (nodos-hijo) están disponibles (tablas).
3) Reemplazar el nodo interno por el resultado de la operación.
4) La ejecución termina al llegar al nodo raíz y realizar el último reemplazamiento.

"Código y nombre de las asignaturas que imparten los profesores del departamento DSIC"



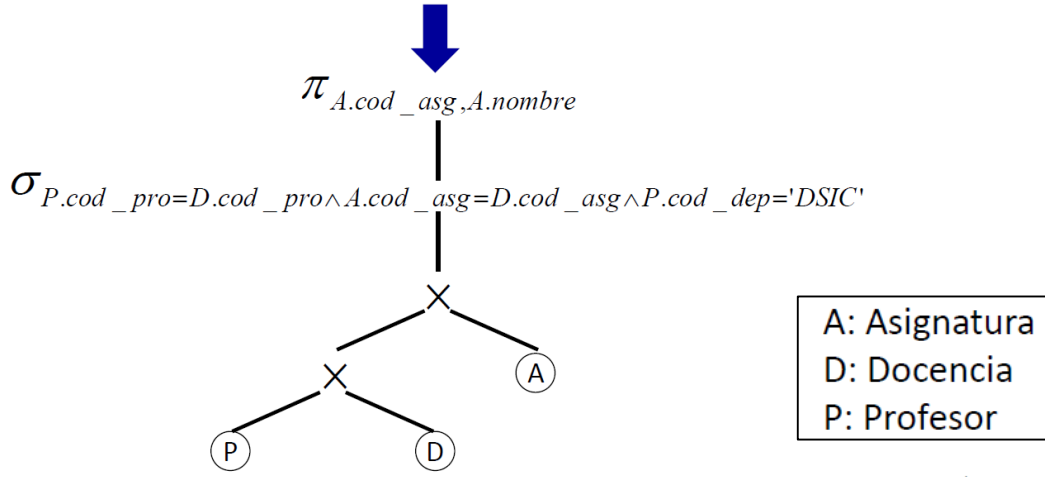

```
SELECT DISTINCT A.cod_asg, A.nombre
FROM Profesor P, Docencia D, Asignatura A
WHERE P.cod_pro=D.cod_pro AND A.cod_asg=D.cod_asg AND
      P.cod_dep='DSIC';
```

$$\pi_{A.cod\_asg,A.nombre}(\sigma_{P.cod\_pro=D.cod\_pro \wedge A.cod\_asg=D.cod\_asg \wedge P.cod\_dep='DSIC'}((P \times D) \times A))$$

A: Asignatura
D: Docencia
P: Profesor

Árbol de consulta correspondiente a la consulta de ejemplo:

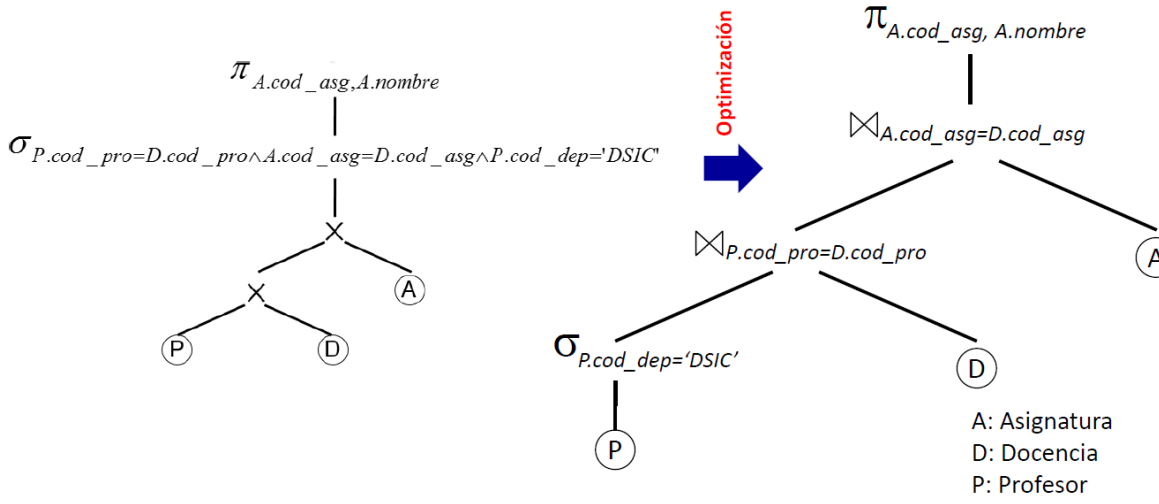
$$\pi_{A.cod\_asg,A.nombre}(\sigma_{P.cod\_pro=D.cod\_pro \wedge A.cod\_asg=D.cod\_asg \wedge P.cod\_dep='DSIC'}((P \times D) \times A))$$



Optimización heurística de árbol de consulta

1) El analizador léxico-sintáctico **genera el árbol de consulta inicial** sin optimización (imagen anterior).

2) El optimizador aplica **reglas sintácticas** y transforma el árbol de consulta inicial en un árbol de consulta final (optimizado).



Reglas de transformación sintácticas para AR

1) **Cascada de σ:** una selección con condición compleja conjuntiva se puede descomponer en una secuencia (cascada) de selecciones con condiciones simples:

$$\sigma_{c1 \wedge c2 \wedge \ldots \wedge cN}(R) \equiv \sigma_{c1}\left(\sigma_{c2}\left(\ldots\left(\sigma_{cN}(R)\right)\right)\right)$$

2) **Conmutatividad de σ con σ:**

$$\sigma_{c1}(\sigma_{c2}(R)) \equiv \sigma_{c2}(\sigma_{c1}(R))$$

3) **Cascada de π:** en una secuencia (cascada) de proyecciones todas las proyecciones menos la última se pueden ignorar:

$$\pi_{lista1}(\pi_{lista2}(\ldots(\pi_{listaN}(R))..)) \equiv \pi_{lista1}(R)$$

4) **Conmutatividad de σ con π:** si la condición c sólo afecta a atributos de la lista de proyección, las dos operaciones pueden conmutarse:

$$\pi_{a1,a2,..,aN}(\sigma_c(R)) \equiv \sigma_c(\pi_{a1,a2,\ldots,aN}(R))$$

5) **Conmutatividad de los operadores $\bowtie$ y $\times$:**

$$R \bowtie_c S \equiv S \bowtie_c R$$
$$R \times S \equiv S \times R$$

6) **Conmutatividad de σ con $\bowtie$ y $\times$:**

   - Si todos los atributos en la condición de la selección son de una de las tablas operando (ej. R) las dos operaciones se pueden conmutar:

$$\sigma_c(R \bowtie S) \equiv (\sigma_c(R)) \bowtie S$$
$$\sigma_c(R \times S) \equiv (\sigma_c(R)) \times S$$

   - Si la condición de la selección es $c = c1 \wedge c2$, donde en $c1$ sólo aparecen atributos de R y en $c2$ sólo aparecen atributos de S:

$$\sigma_c(R \bowtie S) \equiv (\sigma_{c1}(R)) \bowtie (\sigma_{c2}(S))$$
$$\sigma_c(R \times S) \equiv (\sigma_{c1}(R)) \times (\sigma_{c2}(S))$$

7) **Conmutatividad de π con $\bowtie$ y $\times$:**

- Lista de proyección $L = \{a1, a2, \dots, aN, b1, b2, \dots, bM\}$
- Si los atributos de la condición de concatenación c están en L:

$$\pi_L(R \bowtie_c S) \equiv (\pi_{a1,\dots,aN}(R)) \bowtie_c (\pi_{b1,\dots,bM}(S))$$
$$\pi_L(R \times S) \equiv (\pi_{a1,\dots,aN}(R)) \times (\pi_{b1,\dots,bM}(S))$$

8) **Conmutatividad de $\cup$ y de $\cap$:**

$$R \cap S \equiv S \cap R$$
$$R \cup S \equiv S \cup R$$

9) **Asociatividad de $\bowtie$, $\times$ y de $\cap$:**

$$R \cap (S \cap T) \equiv (R \cap S) \cap T$$
$$R \cup (S \cup T) \equiv (R \cup S) \cup T$$
$$R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T$$

10) **Conmutatividad de σ con las operaciones conjuntistas $\cup$ $\cap$ y $-$:**

$$\sigma_c(R \cap S) \equiv (\sigma_c(R)) \cap (\sigma_C(S))$$
$$\sigma_c(R \cup S) \equiv (\sigma_c(R)) \cup (\sigma_c(S))$$
$$\sigma_c(R - S) \equiv \big(\sigma_c(R)\big) - (\sigma_c(S))$$

11) **Conmutatividad de π con $\cup$:**

$$\pi_L(R \cup S) \equiv \pi_L(R)) \cup (\pi_L(S))$$

12) **Convertir la secuencia de operaciones $\times$ seguida de σ en $\bowtie$:** si la condición c de σ es una condición de concatenación.

$$\sigma_c(R \times S) \equiv R \bowtie_c S$$

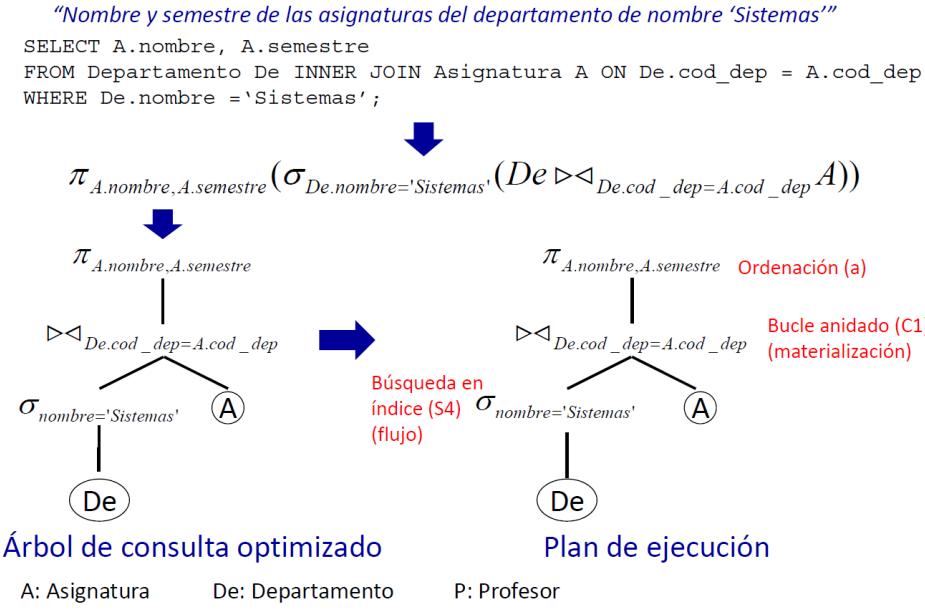
Guía de un algoritmo de optimización heurístico

1. **Usando la regla 1:** descomponer las selecciones con condiciones conjuntivas en una secuencia de selecciones simples.
2. **Usando las reglas 2, 4, 6, y 10:** mover las selecciones hacia abajo en las ramas del árbol.
3. **Usando las reglas 5 y 9:** recolocar los nodos hojas siguiendo el siguiente criterio:
   - Primero, mover hacia la izquierda en los nodos hoja, las tablas con valor más pequeño de selectividad de la condición de selección.
   - Segundo, asegurarse de que el orden de las hojas no causa un producto cartesiano.
4. **Usando la regla 12:** combinar un producto cartesiano con una selección para obtener una concatenación.
5. **Usando reglas 3, 4, 7 y 11:** romper y distribuir proyecciones y moverlas hacia abajo en las ramas del árbol (sólo los atributos necesarios par el resultado final y para operaciones posteriores deben aparecer).
6. Identificar subárboles que representen grupos de operaciones que pueden ser ejecutadas en un único algoritmo.

Convertir árboles de consulta en planes de ejecución

Métodos para computar los operadores en el árbol:

- **Evaluación materializada:** el resultado de la ejecución de la operación se materializa, se almacena temporalmente, para usarse en la ejecución del siguiente operador en el árbol.
- **Evaluación en flujo:** a medida que se obtienen las filas resultantes de ejecutar el operador, se van utilizando para la ejecución del siguiente operador en el árbol.



A: Asignatura De: Departamento P: Profesor

5. Optimización basada en estimación de costes

Consiste en estimar y comparar el **coste** (tiempo de ejecución) de distintos planes de ejecución para la consulta, y elegir el más económico.

Componentes del coste en la ejecución de consultas

1. **Coste de acceso a disco**. Coste de leer y escribir bloques en disco.
2. **Coste de almacenamiento** de ficheros intermedios generados durante la ejecución de la consulta.
3. **Coste computacional:** coste de las operaciones sobre datos realizadas en MP.
4. **Coste de uso de memoria:** número de buffers de memoria necesarios durante la ejecución de la consulta.
5. **Coste de comunicaciones:** coste de mandar la consulta y sus resultados al terminal donde se originó la consulta.

Información del catálogo (diccionario) usada en la estimación del coste

FICHEROS QUE IMPLEMENTAN LAS TABLAS:

- Número de registros (r)
- Tamaño medio de los registros (R)
- Número de bloques (b)
- Factor de bloque (fb)

ÍNDICES:

- Atributos (campos) de indexación
- Número de niveles del índice (x)

ATRIBUTOS (CAMPOS):

- Número de valores distintos (d)
- Selectividad de una condición (sl)
- Cardinalidad de una condición (s): $s = sl \cdot r$
  - Condición igualdad sobre atributo clave: $d = r, sl = 1/r, s = 1$
  - Condición igualdad sobre atributo no clave (*con distribución uniforme*): $sl = 1/d, s = r/d$

La información puede no estar actualizada al instante, el sistema la recoge periódicamente.

Estudio de coste para la selección (σ)

S1. Búsqueda lineal (*condición de igualdad*)

- Sobre atributo no clave: $C_{S1} = b$
- Sobre atributo clave: $C_{S1} = b/2$ (*si se encuentra*), b (*si no se encuentra*)

S2. Búsqueda binaria (*condición de igualdad*):

- Sobre atributo no clave: $C_{S2} = \log_2 b + \lceil (s/fb) \rceil - 1$
- Sobre atributo clave: $C_{S2} = \log_2 b$

S3. Búsqueda con direccionamiento calculado (*condición de igualdad*): $C_{S3} = 1$

S4. Búsqueda con índices

- Índice multinivel sobre atributo clave (*condición de igualdad*): $C_{S4} = x + 1$
- Índice multinivel sobre atributo no clave (*condición de igualdad*):
  - En fichero ordenado: $C_{S4} = x + \lceil (s/fb) \rceil$
  - En fichero desordenado: $C_{S4} = x + s$
- Índice multinivel para condición de rango en fichero ordenado: $C_{S4} = x + b/2$ (promedio)

Estudio de coste para la concatenación (R ⋈ S)

Selectividad de la concatenación (js): cociente entre el número de pares de filas (de ambas tablas) que cumplen la condición de concatenación y la cardinalidad del producto cartesiano.

$$0 \leq js \leq 1 \qquad js = |R \bowtie_{R.A=S.B} S| / |R \times S| = |R \bowtie_{R.A=S.B} S| / (|R| \cdot |S|)$$

Se estima js para las condiciones de concatenación más comunes:

- Cardinalidad de la concatenación: $|R \bowtie_{R.A=S.B} S| = js \cdot |R| \cdot |S|$
- Bloques de R: b_R
- Bloques de S: b_S

*$|R|$ *es la cardinalidad de* R *(cantidad de filas de* R*).*

C1. Método de <u>bucle anidado</u>:

- R se recorre en el bucle externo y S en el interno.
- Se asumen tres buffers ($B1, B2, B3$) en MP.
- Factor de bloque del resultado de la concatenación: fb_{RS}

$$C_{C1} = b_R + (b_R \cdot b_S) + (\lceil js \cdot |R| \cdot |S| \,/\, fb_{RS} \rceil)$$

Lectura (en B1) de bloques de R

Lectura (en B2) de bloques de S

Escritura (desde B3) de bloques de R ⋈ S

C2. Método de <u>bucle único</u>:

- Existe un índice sobre el atributo (clave) de B de S con x_B niveles:

$$C_{C2} = b_R + \big(|R| \cdot (x_B + 1)\big) + (\lceil js \cdot |R| \cdot |S| \,/\, fb_{RS} \rceil)$$

- S está implementada como un fichero disperso sobre B:

$$C_{C2} = b_R + (|R| \cdot h) + (\lceil js \cdot |R| \cdot |S| \,/\, fb_{RS} \rceil)$$

**h es el número medio de bloques de S, accedidos para recuperar un registro, a partir de un valor del campo de direccionamiento. En el caso de un fichero disperso donde sus cubos son de un bloque y no hay desbordamiento, h sería igual a 1.*

C3. Método de <u>ordenación-fusión</u>:

- Si los ficheros están ordenados:

$$C_{C3} = b_R + b_S + (\lceil js \cdot |R| \cdot |S| \,/\, fb_{RS} \rceil)$$

- Si los ficheros no están ordenados (sumar el coste de ordenar):

$$C_{C3} = \big(2 \cdot b_R \cdot (1 + \log_2 b_R)\big) + \big(2 \cdot b_S \cdot (1 + \log_2 b_S)\big) + b_R + b_S + (\lceil js \cdot |R| \cdot |S| \,/\, fb_{RS} \rceil)$$

C4. Método de <u>direccionamiento calculado</u>:

$$C_{C4} = b_R + b_S + (\lceil js \cdot |R| \cdot |S| \,/\, fb_{RS} \rceil)$$