IIC2413

AYUDANTÍA 10

Evaluación de Consultas

INTERFAZ DE OPERADORES FÍSICOS

Importante para componer y ejecutar el plan físico

`open()`

- Inicializa el estado del operador.
- Ajusta parámetros (ej. condición de selección).

`next()`

- Procesa el input y responde la siguiente tupla.

`close()`

- Limpia elementos temporales.

Operaciones

SELECCIÓN SIN ÍNDICES

Necesariamente recorremos todo R

Leer cada tupla, una a una, y retornarla si cumple con el predicado.

```
open()
  R.open()
next() // siguiente seleccionado
  t := R.next()
  while t != null do
    if t satisface condición then
      return t
    t := R.next()
  return null
close()
  R.close()
```

SELECCIÓN CON ÍNDICE Y CONSULTA DE IGUALDAD

- Solo se leen las páginas que satisfacen la condición (más I/O si muchas tuplas la satisfacen).
- Si el atributo es llave primaria, la operación tiene prácticamente un costo de I/O ≈ 1.

```
open()
  I.open()
  I.search(Atributo = valor)
next()
  t := I.next()
  while t != null do
    return t
  return null
close()
  I.close()
```

PROYECCIÓN

Necesariamente recorremos todo R

Leer cada tupla, una a una, y proyectar en los atributos L.

```
open()
  R.open()
next()
  t := R.next()
  while t != null do
    return project(t, atributos)
  return null
close()
  R.close()
```

Joins

NESTED LOOP JOIN

Implementación directa de join basada en loops

Toma cada fila de una tabla externa (bucle principal) y recorre toda la tabla interna (bucle anidado)

```
next()
  while r != null do
    s := S.next()
    if s == null then
      S.close()
      r := R.next()
      S.open()
    else if (r, s) satisfacen p then
      return (r, s)
  return null
```

Costo en I/O:

$$Costo(R) + Tuplas(R) \cdot Costo(S)$$

BLOCK NESTED LOOP JOIN

Optimización del Nested Loop Join

Lee bloques enteros de la tabla externa y los almacena en un búfer de memoria (join buffer) para compararlos de una sola pasada con la tabla interna

```
next()
  while Buff != empty do
    while s != null do
      r := Buffer.next()
      if r == null then
        Buffer.reset()
        s := S.next()
      else if (r, s) satisfacen p then
        return (r, s)
    fillBuffer()
  return null
```

Costo en I/O:

$$Costo(R) + (Páginas(R) / (Buffer - 2)) \cdot Costo(S)$$

Conviene que la relación más pequeña vaya al exterior

EXTERNAL MERGE SORT

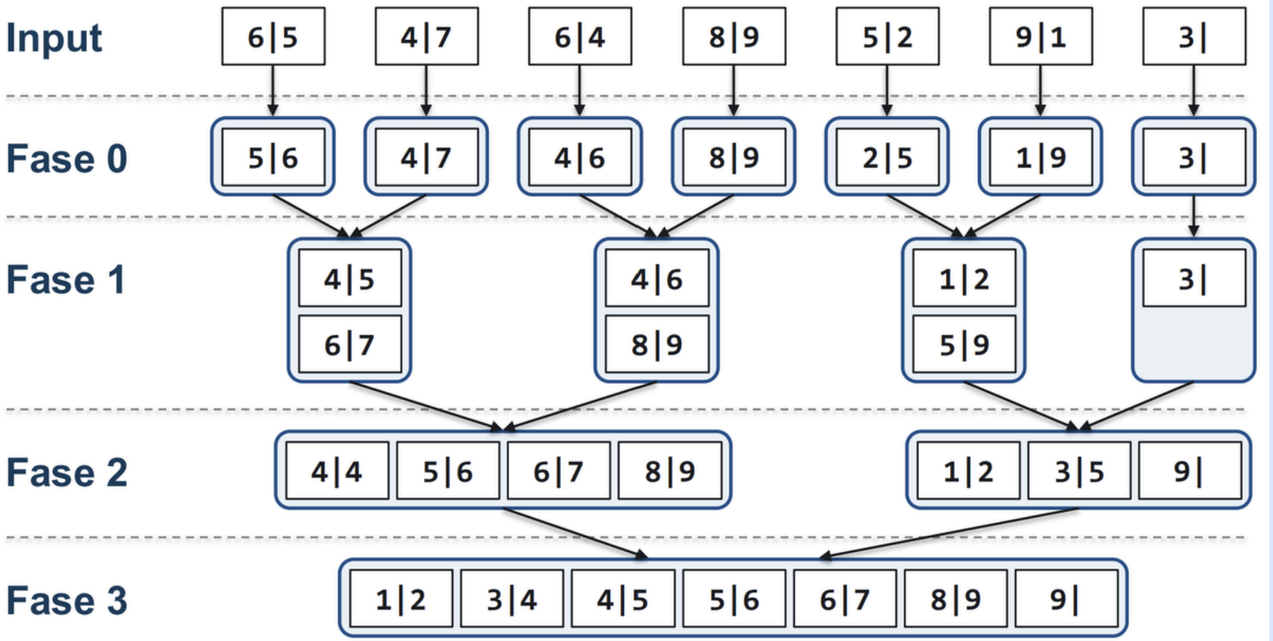
Fase o

Se crean los runs iniciales.

Fase i

1. Se traen los runs a memoria.
2. Se realiza el merge de cada par de runs.
3. Se almacena el nuevo run en disco (es decir, se materializan los resultados intermedios).

Ejemplo de ejecución



Cada página contiene dos tuplas; cada recuadro azul es un run.

EJERCICIOS

EJERCICIOS I2 2026-1

a) Calcule el costo de I/O del Block Nested Loop Join para las siguientes relaciones, asumiendo M = 10 páginas de buffer disponibles.

   i) R tiene 100 páginas y 1.000 tuplas, S tiene 400 páginas y 2.000 tuplas (3 pts)

Fórmula: $\text{Costo}(R) + \left\lceil \frac{\text{Páginas}(R)}{\text{Buffer} - 2} \right\rceil \cdot \text{Costo}(S)$

EJERCICIOS I2 2026-1

a) Calcule el costo de I/O del Block Nested Loop Join para las siguientes relaciones, asumiendo M = 10 páginas de buffer disponibles.

i) R tiene 100 páginas y 1.000 tuplas, S tiene 400 páginas y 2.000 tuplas (3 pts)

Fórmula: $\text{Costo}(R) + \left\lceil \frac{\text{Páginas}(R)}{\text{Buffer} - 2} \right\rceil \cdot \text{Costo}(S)$

i) Usando R como outer:

$$\text{Costo} = p_R + \left\lceil \frac{p_R}{M-2} \right\rceil \cdot p_S = 100 + \left\lceil \frac{100}{8} \right\rceil \cdot 400 = 100 + 13 \cdot 400 = \boxed{5{,}300 \text{ I/Os}}$$

Usando S como outer (para comparar):

$$\text{Costo} = p_S + \left\lceil \frac{p_S}{M-2} \right\rceil \cdot p_R = 400 + \left\lceil \frac{400}{8} \right\rceil \cdot 100 = 400 + 50 \cdot 100 = 5{,}400 \text{ I/Os}$$

EJERCICIOS I2 2026-1

a) Calcule el costo de I/O del Block Nested Loop Join para las siguientes relaciones, asumiendo M = 10 páginas de buffer disponibles.

   i) R tiene 100 páginas y 1.000 tuplas, S tiene 400 páginas y 2.000 tuplas (3 pts)

   ii) ¿Conviene usar R como outer o S como outer? ¿Por qué? (3 pts)

EJERCICIOS I2 2026-1

a) Calcule el costo de I/O del Block Nested Loop Join para las siguientes relaciones, asumiendo M = 10 páginas de buffer disponibles.

   i) R tiene 100 páginas y 1.000 tuplas, S tiene 400 páginas y 2.000 tuplas (3 pts)

   ii) ¿Conviene usar R como outer o S como outer? ¿Por qué? (3 pts)

Conviene usar R (la relación más pequeña) como outer: 5.300 I/Os $<$ 5.400 I/Os. El costo está dominado por el término $\lceil p_{\text{outer}}/(M-2)\rceil \cdot p_{\text{inner}}$. Al poner la relación más pequeña como outer se necesitan **menos pasadas completas** sobre la relación inner, ya que $\lceil p_R/(M-2)\rceil = 13 < \lceil p_S/(M-2)\rceil = 50$. Minimizar el número de vueltas sobre la tabla más grande reduce el costo total.

EJERCICIOS I2 2026-1

c) Responda las siguientes preguntas.

   1) Define el algoritmo Nested Loop Join e indica su costo en I/O. (2 pts)

EJERCICIOS I2 2026-1

c) Responda las siguientes preguntas.

i) Define el algoritmo Nested Loop Join e indica su costo en I/O. (2 pts)

El Nested Loop Join es la implementación más directa: para cada tupla de R se recorre la relación S completa buscando coincidencias con el predicado, además de leer R entera una vez.

$$\text{Costo en I/O} = \text{Costo}(R) + \text{Tuplas}(R) \cdot \text{Costo}(S) = p_R + n_R \cdot p_S$$

EJERCICIOS I2 2026-1

c) Responda las siguientes preguntas.

   I) Define el algoritmo Nested Loop Join e indica su costo en I/O. (2 pts)

   II) Explica Nested Loop Join con el siguiente ejemplo (Páginas R = 500, Páginas S = 800, N° Tuplas R = 10.000) (2 pts)

EJERCICIOS I2 2026-1

c) Responda las siguientes preguntas.

i) Define el algoritmo Nested Loop Join e indica su costo en I/O. (2 pts)

ii) Explica Nested Loop Join con el siguiente ejemplo (Páginas R = 500, Páginas S = 800, N° Tuplas R = 10.000) (2 pts)

Ejemplo con $p_R = 500$, $p_S = 800$, $n_R = 10{,}000$ (2 pts)

$$\text{Costo} = 500 + 10{,}000 \cdot 800 = 500 + 8{,}000{,}000 = \boxed{8{,}000{,}500 \text{ I/Os}}$$

EJERCICIOS I2 2026-1

c) Responda las siguientes preguntas.

i) Define el algoritmo Nested Loop Join e indica su costo en I/O. (2 pts)

ii) Explica Nested Loop Join con el siguiente ejemplo (Páginas R = 500, Páginas S = 800, N° Tuplas R = 10.000) (2 pts)

iii) Por qué este algoritmo puede ser muy costoso. (2 pts)

EJERCICIOS I2 2026-1

c) Responda las siguientes preguntas.

i) Define el algoritmo Nested Loop Join e indica su costo en I/O. (2 pts)

ii) Explica Nested Loop Join con el siguiente ejemplo (Páginas R = 500, Páginas S = 800, N° Tuplas R = 10.000) (2 pts)

iii) Por qué este algoritmo puede ser muy costoso. (2 pts)

Porque por **cada tupla individual** de R (no por cada página) se vuelve a leer la relación S completa desde disco. Se desperdicia el buffer: solo se usa para mantener una tupla de R y recorrer S, en vez de aprovechar el espacio disponible en memoria para cargar muchas tuplas de R a la vez. El término $n_R \cdot p_S$ domina el costo total y crece muy rápido cuando ambas relaciones son grandes, haciendo el algoritmo impracticable para tablas de tamaño real (más de 8 millones de I/Os en el ejemplo).

Conexión al servidor

¿CÓMO SE CONECTA?

- PDO (PHP Data Objects) es una librería que permite a PHP hablar con la base de datos. Le mandas un query, lo ejecuta en PostgreSQL, y te devuelve los resultados como un array de PHP.
- Se crea un objeto PDO pasándole host, dbname, usuario y clave. Se usa try/catch para manejar errores de conexión.
- Esta función se reutiliza en todos los archivos PHP del proyecto.

```php
<?php
function conectarBD() {
    $host = 'localhost'; // Cambiar al servidor stonebraker.ing.uc.cl si se
    $dbname = ''; // usuariouc.e3
    $usuario = ''; // usuariouc.e3
    $clave = ''; // Número de alumno

    try {
        $db = new PDO("pgsql:host=$host;dbname=$dbname", $usuario, $clave);
        $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        return $db;
    } catch (PDOException $e) {
        echo "Error de conexión: " . $e->getMessage();
        exit();
    }
}
?>
```

Proyecto/E3/utils.php