

## Práctica 06

Sobre optimización de consultas aplicando indexación, desnormalización y buenas prácticas

### PLANTILLA DE RESPUESTAS

#### PARTE 1: PROCESAMIENTO DE CONSULTAS

##### Ejercicio 1

El error en la consulta es que al momento de hacer el **where** con la variable e.apellido es imposible que la misma empiece con A – B – C.

**Sintácticamente** es correcta.

**Semánticamente** no es correcta.

```
SELECT *
FROM empleados e
WHERE e.apellidos like 'A%'
      AND e.apellidos like 'B%'
      AND e.apellidos like 'C%';
```

```
SELECT *
FROM empleados e
WHERE e.apellidos like 'A%'
      OR e.apellidos like 'B%'
      OR e.apellidos like 'C%';
```

##### Ejercicio 2

Después de ver la consulta puedo entender lo que se desea consultar es sacar de la tabla empleados todos los que su fecha sea mayor a 01-ENE-1998 y 15-MAY-2001 y seguramente que su género sea M, en esta cláusula está el error en donde se le pone que fecha sea mayor a dos fechas distintas pero basta con la primera fecha ya que abarca ambas y también hacer el cambio de OR por el AND para que se cumpla obligatoriamente.

**Sintácticamente** es correcta.

**Semánticamente** no es correcta.

```
SELECT *
FROM empleados
WHERE fecha_ing > '01-ENE-1998'
      AND fecha_ing > '15-MAY-2001'
      OR genero = 'M';
```

```
SELECT *
FROM empleados
WHERE fecha_ing > '15-MAY-2001'
      AND genero = 'M';
```

##### Ejercicio 3

Esta consulta tiene discordancia ya que se está contando las cédulas por cada correo existente y esta consulta va a tener una salida, pero es una salida incoherente ya que cuando se definió el esquema de la base se puso como llave única a email y eso dice que cada uno de ellos tiene una sola cédula.

**Sintácticamente** es correcta.

**Semánticamente** no es correcta.

```
SELECT email, count(cedula)
FROM empleados
WHERE genero = 'M'
GROUP BY email
HAVING count(cedula) > 1;
```

## Ejercicio 4

### Consulta SQL

```
SELECT e.*  
FROM empleados e, cargos c  
WHERE e.idcargo = c.idcargo  
AND c.cargo = 'FISCALIZADOR';
```

$\pi$  e.cedula, e.apellidos, e.nombres, e.direccion, e.genero, e.fecha\_ing, e.email, e.idcargo  
1 row

$\sigma$  e.idcargo = c.idcargo and c.cargo = 'FISCALIZADOR'

1 row

$\times$

30 rows

$\rho_e$

6 rows

$\rho_c$

5 rows

empleados

6 rows

cargos

5 rows

### Algebra Relacional

$\pi$  e.cedula, e.apellidos, e.nombres, e.direccion, e.genero, e.fecha\_ing, e.email, e.idcargo ( $\sigma$  e.idcargo = c.idcargo  $\wedge$  c.cargo = 'FISCALIZADOR'(( $\rho_e$  empleados)  $\times$  ( $\rho_c$  cargos)))

$\pi$  e.cedula, e.apellidos, e.nombres, e.direccion, e.genero, e.fecha\_ing, e.email, e.idcargo

1 row

$\sigma$  e.idcargo = c.idcargo and c.cargo = 'FISCALIZADOR'

1 row

$(\times)$

30 rows

$\rho_e$

6 rows

$\rho_c$

5 rows

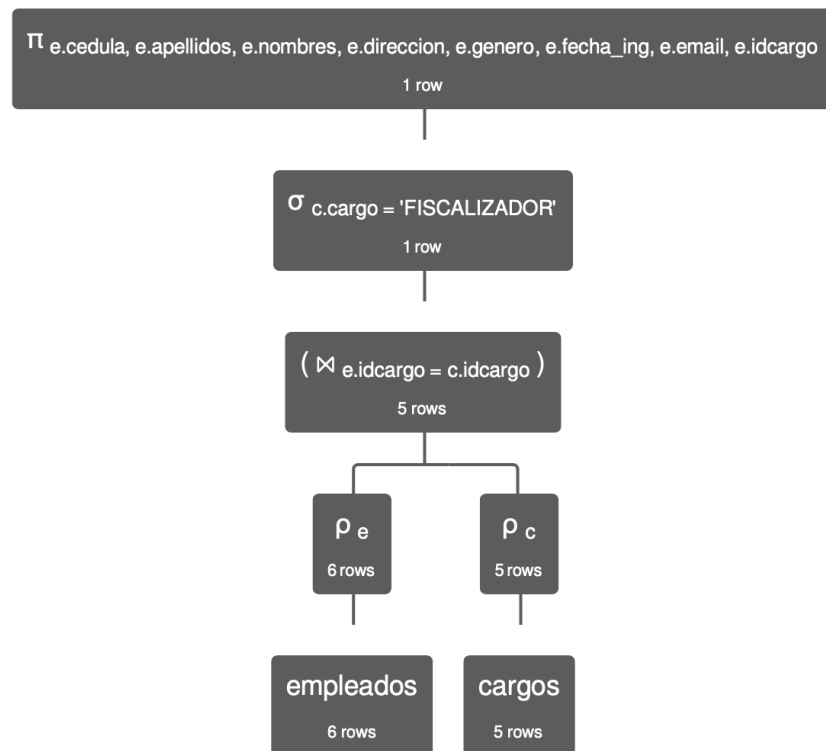
empleados

6 rows

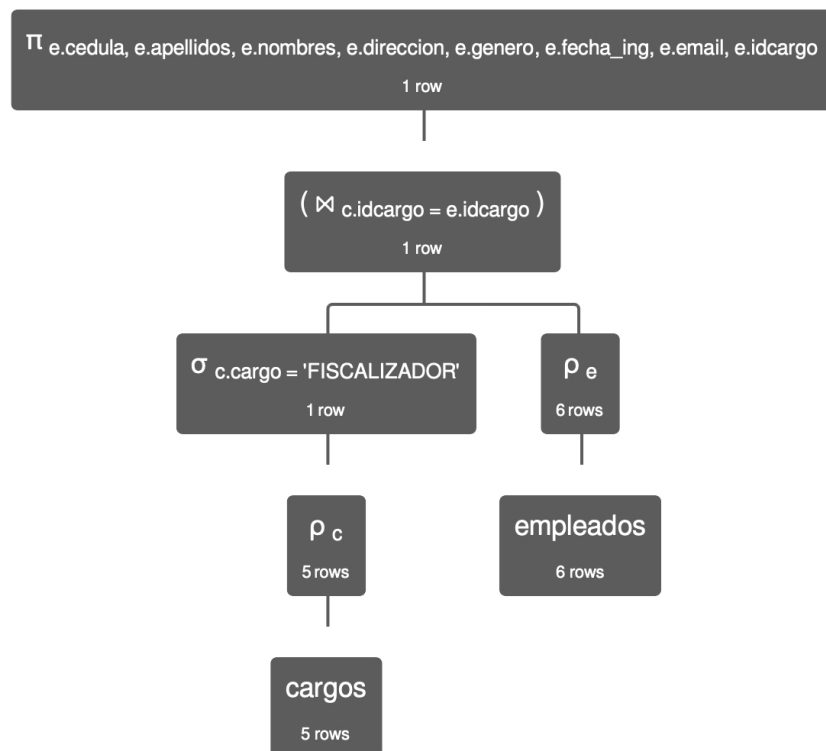
cargos

5 rows

$\pi$  e.cedula, e.apellidos, e.nombres, e.direccion, e.genero, e.fecha\_ing, e.email, e.idcargo  $(\sigma$  c.cargo = 'FISCALIZADOR'  $((\rho$  e empleados)  $\bowtie$  e.idcargo = c.idcargo  $(\rho$  c cargos)))



$\pi$  e.cedula, e.apellidos, e.nombres, e.direccion, e.genero, e.fecha\_ing, e.email, e.idcargo  $((\sigma$  c.cargo = 'FISCALIZADOR'  $(\rho$  c cargos))  $\bowtie$  c.idcargo = e.idcargo  $(\rho$  e empleados))



## Ejercicio 5

<b>PLAN A</b>				
$\Pi$ l.titulo ( $\sigma$ l.idedt = e.idedt and l.anio = 2016 and l.numpaginas > 500 and e.pais = 'Argentina' (( $\rho$ l libros) $\times$ ( $\rho$ e editoriales)))				
<b>OPERACION</b>	<b>LECTURA</b>	<b>ESCRITURA</b>		
Realizar producto cartesiano entre libros y editoriales.	10200	2000000		
Aplicar filtros	2000000	20		
Proyectar columnas	20	0		
<b>TOTAL</b>	<b>2010220</b>	<b>2000020</b>	4010240	oper l/O
<b>PLAN B</b>				
$\Pi$ l.titulo ( $\sigma$ e.pais = 'Argentina' (( $\sigma$ l.anio = 2016 and l.numpaginas > 500 ( $\rho$ l libros)) $\bowtie$ l.idedt = e.idedt ( $\rho$ e editoriales)))				
<b>OPERACION</b>	<b>LECTURA</b>	<b>ESCRITURA</b>		
Filtrar de la tabla libros.	10000	250		
Join, combinar entre el resultado del filtro de libros y editoriales.	450	50000		
Aplicar el filtro al resultado del JOIN	50000	20		
Proyectar columnas	20	0		
<b>TOTAL</b>	<b>60470</b>	<b>50270</b>	110740	oper l/O
<b>PLAN C</b>				
$\Pi$ l.titulo ( $\sigma$ l.anio = 2016 and l.numpaginas > 500 and e.pais = 'Argentina' (( $\rho$ l libros) $\bowtie$ l.idedt = e.idedt ( $\rho$ e editoriales)))				
<b>OPERACION</b>	<b>LECTURA</b>	<b>ESCRITURA</b>		
Join, combinar libros y editoriales	10200	2000000		
Aplicar Filtros	2000000	20		
Proyectar columnas	20	0		
<b>TOTAL</b>	<b>2010220</b>	<b>2000020</b>	4010240	oper l/O

Segun el costo de operaciones de cada una el plan mas eficiente es el **PLAN B**.

## PARTE 2: BUENAS PRÁCTICAS SQL

### Ejercicio 6

a) `NOT (NUM_PAG < 80 OR EDITORIAL = 'LIMUSA')`

La formulación podría mejorarse eliminando la doble comparación en la cláusula **not** reemplazando el signo del **menor** por el **mayor** y reemplazando por **diferente** para ya no poner el **not** en la sección editorial, quedaria asi:  
`NUM_PAG >= 80 OR EDITORIAL <> 'LIMUSA'`

b) `NOT (SUELDO < 800) AND COD_POSTAL = '110401'`

La formulación podría mejorarse eliminando la doble comparación en la cláusula **not** y reemplazando el signo de **menor** por el **mayor**, quedaria asi:  
`SUELDO > 800 AND COD_POSTAL = '110401'`

## PARTE 3: DESNORMALIZACIÓN

### Ejercicio 7

PROYECTOS(codigo, denominacion, anio)  
INVESTIGADORES(id, nombre, empresa)  
INTEGRANTES(proyecto, integrante, rol)

1. ¿Cuál técnica de desnormalización aplicaría para simplificar esta consulta, de manera que sea mas eficiente?

La técnica que aplicaría es la de duplicidad de atributos en asociaciones de M:N.

2. ¿Cómo quedarían las tablas desnormalizadas (solo estructura)?

PROYECTOS(codigo, denominacion, anio)  
INVESTIGADORES(id, nombre, empresa)  
INTEGRANTES(proyecto, integrante, rol, **denominacion**, **nombre**)

3. ¿Como quedaría la consulta luego de desnormalizar?

```
SELECT i.denominacion, i.nombre, i.rol
FROM INTEGRANTES i;
```

### Ejercicio 8

MATRICULA(idMatricula, fecha, estudiante, periodoAcademico, costoCurso, tasaAdministrativa, descuento)

```
SELECT idMatricula, fecha, estudiante, periodoAcademico,
costoCurso+tasaAdministrativa-descuento AS "ImporteTotal"
FROM matriculas;
```

1. ¿Cuál es la técnica de desnormalización que se podría aplicar en este caso para que la ejecución de la consulta sea más eficiente?

La tecnica que aplicaria es la de inclusion dea tributos derivados.

2. ¿Cómo quedaría la tabla desnormalizada?

- importeTotal seria un atributo derivado que se calcularia con el costoCurso + tasaAdministrativa – descuento

MATRICULA(idMatricula, fecha, estudiante, periodoAcademico, costoCurso, tasaAdministrativa, descuento, **importeTotal**)

3. ¿Cómo quedaría la consulta luego de desnormalizar?

```
SELECT idMatricula, fecha, estudiante, periodoAcademico, importeTotal
FROM matriculas;
```

## Ejercicio 9

**PROVINCIAS**(codprov, nomprov, extension, region, ptelef, codcap)

**CAPITALES**(codcap, nomcap, poblacion, altura)

```
SELECT p.nomprov "Provincia",
       p.region "Region" ,
       c.nomcap "Capital",
       c.altura "Altura Capital (msnm) "
FROM provincias p, capitales c
WHERE p.codcap = c. codcap;
```

1. ¿Cuál técnica de desnormalización aplicaría para simplificar esta consulta, de manera que sea más eficiente?

La técnica que aplicaría es la combinación de tablas con asociación 1:1

2. ¿Cómo quedarían las tablas desnormalizadas (solo estructura)?

**PROVINCIAS**(codprov, nomprov, extension, region, ptelef, codcap, **nomcap**, **altura**)

**CAPITALES**(codcap, nomcap, poblacion, altura)

3. ¿Cómo quedaría la consulta luego de desnormalizar?

```
SELECT p.nomprov "Provincia",
       p.region "Region" ,
       p.nomcap "Capital",
       p.altura "Altura Capital (msnm) "
FROM provincias p;
```

## Ejercicio 10

**PAISES**(id\_pais, nombre\_pais, capital, id\_continente)

**CONTINENTES**(id\_continente, nombre\_continente)

1. ¿Cuál técnica de desnormalización aplicaría para simplificar esta consulta, de manera que sea más eficiente?

La técnica que aplicaría es la de duplicidad de atributos que no forman parte de la clave en asociaciones 1:N.

2. ¿Cómo quedarían las tablas desnormalizadas (solo estructura)?

**PAISES**(id\_pais, nombre\_pais, capital, id\_continente, **nombre\_continente**)

**CONTINENTES**(id\_continente, nombre\_continente)

3. ¿Cómo quedaría la consulta luego de desnormalizar?

```
SELECT
    PAI. ID PAIS,
    PAI. NOMBRE_PAIS,
    PAI. CAPITAL,
    PAI. NOMBRE CONTINENTE
FROM
    PAISES PAI;
```

## PARTE 4: PREGUNTAS REPASO

### Pregunta 1

Los índices pueden mejorar la eficiencia del sistema de la base de datos cuando existe un gran volumen de datos y necesitamos hacer alguna consulta específica. Al indexar las columnas relevantes el sistema nos encuentra dicho valor más rápido que si lo hubiéramos hechos sin índices.

### Pregunta 2

- **Views**

No usemos la palabra 'View' en el nombre de la vista.

- ViewEmployeesDepartments
- EmployeeDeparmentsView

Bien:

- EmployeeDeparments

No incluyamos las condiciones OrderBy y Where en la vista.

MALO	BUENO
<pre>SELECT  EmployeeId,         Name,         LastName,         Address,         State FROM    Employees WHERE   EmployeeId &gt; 0 ORDER BY Name</pre>	<pre>SELECT  Name,         LastName,         Address,         State,         City,         Zip FROM    Employees;</pre>

- **Looping**

Evitemos ejecutar consultas mediante un bucle. La codificación de consultas SQL en bucles ralentiza toda la ejecución. Supongamos que queremos insertar números del 1 al 10 en una tabla llamada Numeros:

MALO	BUENO
<pre>DECLARE @Numero INT = 1  WHILE @Numero &lt;= 10 BEGIN     INSERT INTO Numeros (Numero)     VALUES (@Numero)      SET @Numero = @Numero + 1 END</pre>	<pre>INSERT INTO Numeros (Numero) VALUES (1), (2), (3), (4), (5), (6), (7), (8), (9), (10);</pre>

- **Between**

Utilizar en vez de la cláusula IN la cláusula BETWEEN, cuando sea posible en la instrucción WHERE.

MALO	BUENO
<pre>SELECT * FROM Ventas WHERE YEAR(fechaVenta) IN (2021, 2022, 2023);</pre>	<pre>SELECT v.cod, v.nombre, v.fechaVenta FROM Ventas v WHERE YEAR(fechaVenta) BETWEEN 2021 AND 2023;</pre>

### Pregunta 3

Las circunstancias en la que utilizaría la desnormalización es por ejemplo en una empresa en donde piden reportes constantemente y para no estar consultando en varias tablas constantemente para sacar la información, es preferiblemente tener una sola tabla en donde abarque todo lo necesario.

La estructura normalizada podría incluir las siguientes tablas:

- **EMPLEADO**(id\_empleado, nombre, id\_departamento)
- **DEPARTAMENTO**(id\_departamento, nombre\_departamento)
- **PROYECTO**(id\_proyecto, nombre\_proyecto, id\_empleado)

En esta empresa constantemente quieren saber que empleado y de qué departamento trabajan en cada proyecto así que aplicaremos desnormalización:

- **EMPLEADO**(id\_empleado, nombre, id\_departamento)
- **DEPARTAMENTO**(id\_departamento, nombre\_departamento)
- **PROYECTO**(id\_proyecto, nombre\_proyecto, id\_empleado, nombre\_empleado, nombre\_departamento)

Ahora ya aplicada la desnormalización se nos hará más fácil ya que solo debemos de consultar a una sola tabla para obtener los resultados necesarios y que el motor nos dé una respuesta en un tiempo menor.

.-