

Ejercicios de Optimización de consultas sobre la base de datos Jardinería.

1. Consulte cuáles son los índices que hay en la tabla producto utilizando las dos instrucciones SQL que nos permiten obtener esta información de la tabla.

```
SELECT index_name
FROM information_schema.statistics
WHERE table_schema = 'jardineria'
AND table_name = 'producto';
```

✓ Mostrando filas 0 - 1 (total de 2, La consulta tardó 0,0003 segundos.)

```
SELECT index_name FROM information_schema.statistics WHERE table_schema = 'jardineria' AND
table_name = 'producto';
```

☐ Perfilando [[Editar en línea](#)] [[Editar](#)] [[Explicar SQL](#)] [[Crear código PHP](#)] [[Actualizar](#)]

☐ Mostrar todo | Número de filas: 25 ▼ Filtrar filas:

Opciones extra

index_name

PRIMARY

gama

SHOW INDEX FROM producto;

Su consulta se ejecutó con éxito.

```
SHOW INDEX FROM producto;
```

☐ Perfilando [[Editar en línea](#)] [[Editar](#)] [[Crear código PHP](#)] [[Actualizar](#)]

Opciones extra

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
producto	0	PRIMARY	1	codigo_producto	A	276	NULL	NULL		BTREE		
producto	1	gama	1	gama	A	8	NULL	NULL		BTREE		

2. Haga uso de EXPLAIN para obtener información sobre cómo se están realizando las consultas y diga cuál de las dos consultas realizará menos comparaciones para encontrar el producto que estamos buscando. ¿Cuántas comparaciones se realizan en cada caso? ¿Por qué?

EXPLAIN SELECT * FROM producto WHERE codigo_producto = 'OR-114';

Su consulta se ejecutó con éxito.

```
EXPLAIN SELECT * FROM producto WHERE codigo_producto = 'OR-114';
```

[[Editar en línea](#)] [[Editar](#)] [[Omitir la explicación del SQL](#)] [[Crear código PHP](#)]

Opciones extra

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	producto	const	PRIMARY	PRIMARY	62	const	1	

Operaciones sobre los resultados de la consulta

EXPLAIN SELECT * FROM producto WHERE nombre= 'Evonimus Pulchellus';

Su consulta se ejecutó con éxito.




```
EXPLAIN SELECT * FROM producto WHERE nombre = 'Evonimus Pulchellus';
```

[[Editar en línea](#)] [[Editar](#)] [[Omitir la explicación del SQL](#)] [[Crear código PHP](#)]

Opciones extra

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	producto	ALL	NULL	NULL	NULL	NULL	276	Using where

Operaciones sobre los resultados de la consulta

 Imprimir  Copiar al portapapeles  Crear vista

En general, se puede decir que la consulta que utiliza la columna "codigo_producto" será más eficiente y realizará menos comparaciones, ya que se puede utilizar un índice para buscar rápidamente el valor deseado en la tabla. Por otro lado, la consulta que utiliza la columna "nombre" requerirá una búsqueda lineal a través de todas las filas de la tabla y, por lo tanto, puede requerir muchas comparaciones.

3. Suponga que estamos trabajando con la base de datos jardineria y queremos saber optimizar las siguientes consultas. ¿Cuál de las dos sería más eficiente?. Se recomienda hacer uso de EXPLAIN para obtener información sobre cómo se están realizando las consultas

EXPLAIN SELECT AVG(total) FROM pago WHERE YEAR(fecha_pago) = 2008;

Su consulta se ejecutó con éxito.

```
EXPLAIN SELECT AVG(total) FROM pago WHERE YEAR(fecha_pago) = 2008;
```

[Editar en línea] [Editar] [Omitir la explicación del SQL] [Crear código PHP]

Opciones extra

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	pago	ALL	NULL	NULL	NULL	NULL	26	Using where

EXPLAIN SELECT AVG(total) FROM pago WHERE fecha_pago >= '2010-01-01' AND fecha_pago <= '2010-12-31';

Su consulta se ejecutó con éxito.

```
EXPLAIN SELECT AVG(total) FROM pago WHERE fecha_pago >= '2010-01-01' AND fecha_pago <= '2010-12-31';
```

[Editar en línea] [Editar] [Omitir la explicación del SQL] [Crear código PHP]

Opciones extra

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	pago	ALL	NULL	NULL	NULL	NULL	26	Using where

Se espera que la segunda consulta sea más eficiente y utilice menos recursos que la primera pero eso todo dependerá de varios factores por ejemplo como la cantidad de filas que cumplan la condición.

4. Optimiza las siguientes consultas creando índices cuando sea necesario. Se recomienda hacer uso de EXPLAIN para obtener información sobre cómo se están realizando las consultas.

Para crear un índice sobre la columna cliente hacemos uso de CREATE INDEX :

```
CREATE INDEX index_nombre_cliente ON cliente(nombre_cliente);
```

```
EXPLAIN SELECT * FROM cliente INNER JOIN pedido ON cliente.codigo_cliente =  
pedido.codigo_cliente WHERE cliente.nombre_cliente LIKE 'A%';
```

Su consulta se ejecutó con éxito.

```
EXPLAIN SELECT * FROM cliente INNER JOIN pedido ON cliente.codigo_cliente = pedido.codigo_cliente WHERE cliente.nombre_cliente LIKE 'A%';
```

[Editar en línea] [Editar] [Omitir la explicación del SQL] [Crear código PHP]

Opciones extra

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	cliente	range	PRIMARY,index_nombre_cliente	index_nombre_cliente	202	NULL	3	Using index condition
1	SIMPLE	pedido	ref	codigo_cliente	codigo_cliente	4	jardineria.cliente.codigo_cliente	3	

5. ¿Por qué no es posible optimizar el tiempo de ejecución de las siguientes consultas, incluso haciendo uso de índices? Prueba a optimizar dichas consultas con el índice adecuado.

```
CREATE INDEX index_codigo_cliente ON pedido(codigo_cliente);
```

✓ MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0,0004 segundos.)

```
CREATE INDEX index_codigo_cliente ON pedido(codigo_cliente);
```

[Editar en línea] [Editar] [Crear código PHP]

```
CREATE INDEX index_nombre ON cliente(nombre_cliente);
```

✓ MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0,0004 segundos.)

```
CREATE INDEX index_nombre ON cliente(nombre_cliente);
```

Si analizamos la consulta con el INDEX, podemos ver que se puede seguir optimizando dichas consultas.