OPTIMIZACIÓN DE CONSULTAS - Base de datos: Jardinería

1. Consulta cuáles son los índices que hay en las tablas de la BD de Jardinería, utilizando las instrucciones SQL que nos permiten obtener esta información de la tabla.

SHOW INDEX FROM cliente;

SHOW INDEX FROM producto;

SHOW INDEX FROM pedido;

SHOW INDEX FROM pago;

SHOW INDEX FROM detalle_pedido;

2. 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?.

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

1 comparacion.

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

comparaciones en todas las filas.

3. Supongamos que estamos trabajando con la base de datos jardinería 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.

```
SELECT AVG(total)
FROM pago
```

```
WHERE YEAR(fecha_pago) = 2008;

SELECT AVG(total)
FROM pago
WHERE fecha_pago >= '2008-01-01' AND fecha_pago <= '2008-12-31';
```

Nota: Lectura recomendada sobre la función YEAR y el uso de índices.

La segunda es más eficiente

4. Optimiza la siguiente consulta creando índices cuando sea necesario.

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

CREATE INDEX idx_nombre_cliente ON cliente(nombre_cliente);

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

5. ¿Por qué no es posible optimizar el tiempo de ejecución de las siguientes consultas, incluso haciendo uso de índices?

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

No se pueden usar índices cuando el patrón comienza con %

6. Crea un índice de tipo FULLTEXT sobre las columnas nombre y descripción de la tabla producto.

ALTER TABLE producto ADD FULLTEXT(nombre, descripcion);

7. Una vez creado el índice del ejercicio anterior realiza las siguientes consultas haciendo uso de la función MATCH, para buscar todos los productos que:
   - Contienen la palabra planta en el nombre o en la descripción. Realice una consulta para cada uno de los modos de búsqueda full-text que existen en MySQL (IN NATURAL LANGUAGE MODE, IN BOOLEAN MODE y WITH QUERY EXPANSION) y compare los resultados que ha obtenido en cada caso.

```
SELECT * FROM producto
WHERE MATCH(nombre, descripcion) AGAINST ('planta' IN NATURAL LANGUAGE MODE);

SELECT * FROM producto
WHERE MATCH(nombre, descripcion) AGAINST ('+planta' IN BOOLEAN MODE);

SELECT * FROM producto
WHERE MATCH(nombre, descripcion) AGAINST ('planta' WITH QUERY EXPANSION);
```

   - Contienen la palabra planta seguida de cualquier carácter o conjunto de caracteres, en el nombre o en la descripción.

```
SELECT * FROM producto WHERE MATCH(nombre, descripcion) AGAINST ('planta*' IN BOOLEAN MODE);
```

   - Empiezan con la palabra planta en el nombre o en la descripción.

```
SELECT * FROM producto WHERE nombre LIKE 'planta%' OR descripcion LIKE 'planta%';
```

● Contienen la palabra tronco o la palabra árbol en el nombre o en la descripción.

SELECT * FROM producto WHERE MATCH(nombre, descripcion) AGAINST ('tronco árbol' IN BOOLEAN MODE);

● Contienen la palabra tronco y la palabra árbol en el nombre o en la descripción.

SELECT * FROM producto WHERE MATCH(nombre, descripcion) AGAINST ('+tronco +árbol' IN BOOLEAN MODE);

● Contienen la palabra tronco pero no contienen la palabra árbol en el nombre o en la descripción.

SELECT * FROM producto WHERE MATCH(nombre, descripcion) AGAINST ('+tronco -árbol' IN BOOLEAN MODE);

● Contiene la frase proviene de las costas en el nombre o en la descripción.

SELECT * FROM producto WHERE MATCH(nombre, descripcion) AGAINST ('"proviene de las costas"' IN BOOLEAN MODE);

8. Crea un índice de tipo INDEX compuesto por las columnas apellido_contacto y nombre_contacto de la tabla cliente.

CREATE INDEX idx_apellido_nombre ON cliente(apellido_contacto, nombre_contacto);

9. Una vez creado el índice del ejercicio anterior realice las siguientes consultas haciendo uso de EXPLAIN:

● Busca al cliente Javier Villar. ¿Cuántas filas se han examinado hasta encontrar el

resultado?

```
EXPLAIN SELECT * FROM cliente
WHERE apellido_contacto = 'Villar' AND nombre_contacto = 'Javier';
```

- Busca el cliente anterior utilizando solamente el apellido Villar. ¿Cuántas filas se han examinado hasta encontrar el resultado?

```
EXPLAIN SELECT * FROM cliente WHERE apellido_contacto = 'Villar';
```

- Busca el cliente anterior utilizando solamente el nombre Javier. ¿Cuántas filas se han examinado hasta encontrar el resultado? ¿Qué ha ocurrido en este caso?

```
EXPLAIN SELECT * FROM cliente WHERE nombre_contacto = 'Javier';
```

10. Calcula cuál podría ser un buen valor para crear un índice sobre un prefijo de la columna nombre_cliente de la tabla cliente. Tenga en cuenta que un buen valor será aquel que nos permita utilizar el menor número de caracteres para diferenciar todos los valores que existen en la columna sobre la que estamos creando el índice.
    - En primer lugar calculamos cuántos valores distintos existen en la columna nombre_cliente. Necesitarás utilizar la función COUNT y DISTINCT.

```
SELECT COUNT(DISTINCT nombre_cliente) FROM cliente;
```

    - Haciendo uso de la función LEFT ve calculando el número de caracteres que necesitas utilizar como prefijo para diferenciar todos los valores de la columna. Necesitarás la función COUNT, DISTINCT y LEFT.

```
SELECT COUNT(DISTINCT LEFT(nombre_cliente, 1)) FROM cliente;
SELECT COUNT(DISTINCT LEFT(nombre_cliente, 2)) FROM cliente;
```

    - Una vez que hayas encontrado el valor adecuado para el prefijo, crea el índice sobre la columna nombre_cliente de la tabla cliente.

```
CREATE INDEX idx_nombre_cliente_prefix ON cliente(nombre_cliente(4));
```

    - Ejecuta algunas consultas de prueba sobre el índice que acabas de crear.

```
EXPLAIN SELECT * FROM cliente WHERE nombre_cliente LIKE 'Gold%';
```