Práctica: Diseño e Implementación de Consultas en Base de Datos

Juan Diego Alvarado Moreno

24 de octubre de 2024

1. Introducción

Resumen del Sistema

Esta práctica se centra en el diseño e implementación de consultas en una base de datos para un **sistema de inventario de herramientas**. El sistema gestiona información sobre:

- **Proveedores**
- **Compras**
- **Herramientas**
- **Bodegas**
- **Inventario**

La implementación utiliza **MySQL** como sistema gestor de base de datos y emplea conceptos fundamentales de bases de datos relacionales, incluyendo el uso de *vistas*, *índices* y diferentes tipos de *fragmentación* para optimizar el rendimiento y la organización de los datos.

2. Marco Teórico

2.1. Conceptos Fundamentales

2.1.1. Grado de la Relación

El **grado de la relación** se refiere al número de atributos (columnas) que contiene una tabla en una base de datos relacional. Por ejemplo, la tabla `supplier` tiene un grado de **9**, con atributos: *idSupplier, name, street, number, city, state, phone, email* y *contact*.

2.1.2. Claves y Relaciones

- **Clave Candidata**: Atributo o conjunto de atributos que identifican de manera única cada registro.
- **Clave Primaria**: Clave candidata seleccionada como identificador principal.
- **Superclave**: Conjunto de atributos que incluye una clave candidata.
- **Clave Foránea**: Atributo que referencia a la clave primaria de otra tabla.

2.2. Cardinalidad del Sistema

Tabla	Registros
supplier	5
warehouse	5
tool	5
purchase	10
inventory	25
buyTool	50

2.3. Tipos de Fragmentación

2.3.1. Fragmentación Horizontal

```sql
CREATE VIEW inventory_warehouse1 AS
SELECT * FROM inventory
WHERE idWarehouse = 1;
```

Listing 1: Ejemplo de Fragmentación Horizontal

2.3.2. Fragmentación Vertical

```sql
CREATE VIEW supplier_contact_info AS
SELECT idSupplier, name, phone, email, contact
FROM supplier;
```

Listing 2: Ejemplo de Fragmentación Vertical

2.3.3. Index

Un índice es una estructura de datos que mejora la velocidad de recuperación de registros. Ejemplo de creación de índice:

2.3.4. Create View

Una vista es una tabla virtual basada en el resultado de una consulta SQL. Las vistas pueden simplificar consultas complejas y proporcionar seguridad adicional. Ejemplo:

3. Consultas Implementadas

3.1. Reporte de Compras del Mes de Enero

Álgebra Relacional

$$\pi_{\text{proveedor.name, tool.name, buyTool.amount, buyTool.unitPrice, total}}(\sigma_{\text{MONTH(purchase.date)=1}}(\text{purchase} \bowtie \text{buyTool}))$$

```sql
CREATE VIEW january_purchases AS
SELECT
    s.name AS proveedor,
    t.name AS herramienta,
    bt.amount AS cantidad,
    bt.unitPrice AS precio_unitario,
    (bt.amount * bt.unitPrice) AS precio_total
FROM buyTool bt
JOIN purchase p ON bt.idPurchase = p.idPurchase
JOIN supplier s ON p.idSupplier = s.idSupplier
JOIN inventory i ON bt.idInventory = i.idInventory
JOIN tool t ON i.idTool = t.idTool
WHERE MONTH(p.date) = 1;

\subsection{Listado de responsables de las bodegas}

\textbf{ lgebra  Relacional:}
\[
\pi_{\text{warehouse.manager, warehouse.street, warehouse.number, warehouse.
    phone}} (\text{warehouse})
\]

\textbf{SQL:}
\begin{lstlisting}[language=SQL]
CREATE VIEW warehouse_managers AS
SELECT
    manager AS responsable,
    street AS calle,
    number AS numero,
    phone AS telefono
FROM warehouse;
```

Listing 3: Implementación en SQL

3.2. Listado de contactos con los proveedores

Álgebra Relacional:

$$\pi_{\text{supplier.contact, supplier.name, supplier.phone, supplier.email}}(\text{supplier})$$

SQL:

```sql
CREATE VIEW supplier_contacts AS
SELECT
    contact AS nombre_contacto,
```

```
    name AS nombre_proveedor,
    phone AS telefono,
    email AS correo_electronico
FROM supplier;
```

3.3. Reporte de herramientas compradas cuyo precio unitario es menor o igual a $250

Álgebra Relacional:

$\pi_{\text{tool.name, purchase.date, buyTool.amount}}(\sigma_{\text{buyTool.unitPrice} \leq 250{,}00}(\text{buyTool} \bowtie \text{purchase} \bowtie \text{inventory} \bowtie \text{tool}))\tau_{\text{purchase.dat}}$

SQL:

```
CREATE VIEW tools_under_250 AS
SELECT
    t.name AS herramienta,
    p.date AS fecha_compra,
    bt.amount AS cantidad_comprada
FROM buyTool bt
JOIN purchase p ON bt.idPurchase = p.idPurchase
JOIN inventory i ON bt.idInventory = i.idInventory
JOIN tool t ON i.idTool = t.idTool
WHERE bt.unitPrice <= 250.00
ORDER BY p.date DESC;
```

3.4. Reporte de herramientas en inventario con stock entre 5 y 20 piezas

Álgebra Relacional:

$\pi_{\text{warehouse.street, warehouse.number, tool.name, inventory.location, inventory.amount}}(\sigma_{\text{inventory.amount BETWEEN 5 AND 20}}(\text{invento}$

SQL:

```
CREATE VIEW stock_5_to_20 AS
SELECT
    w.street AS calle,
    w.number AS numero,
    t.name AS herramienta,
    i.location AS ubicacion,
    i.amount AS cantidad
FROM inventory i
JOIN warehouse w ON i.idWarehouse = w.idWarehouse
JOIN tool t ON i.idTool = t.idTool
WHERE i.amount BETWEEN 5 AND 20;
```

3.5. Reporte del stock total por bodega

Álgebra Relacional:

$\gamma_{\text{warehouse.street, warehouse.number, warehouse.manager, warehouse.phone;}\sum\text{(inventory.amount) as total\_herramientas}}(\text{warehouse} \bowtie \text{in}$

SQL:

```
CREATE VIEW warehouse_total_stock AS
SELECT
    w.street AS calle,
    w.number AS numero,
    w.manager AS responsable,
    w.phone AS telefono,
    SUM(i.amount) AS total_herramientas
FROM warehouse w
JOIN inventory i ON w.idWarehouse = i.idWarehouse
GROUP BY w.street, w.number, w.manager, w.phone;
```

3.6. Reporte del valor total de inventario por bodega

Álgebra Relacional:

$$\gamma_{\text{warehouse.street, warehouse.state;}\sum(\text{inventory.amount}\times\text{inventory.storeSalePrice})\text{ as valor\_total}}(\text{warehouse} \bowtie \text{inventory})$$

SQL:

```sql
CREATE VIEW warehouse_inventory_value AS
SELECT
    w.street AS calle,
    w.state AS estado,
    SUM(i.amount * i.storeSalePrice) AS monto_total
FROM warehouse w
JOIN inventory i ON w.idWarehouse = i.idWarehouse
GROUP BY w.street, w.state;
```

4. Conclusiones

Aspectos Clave

La implementación de esta base de datos y sus consultas demuestra:

- La importancia de un **diseño bien estructurado**
- La utilidad de conceptos como **vistas**, **índices** y **fragmentación**
- La capacidad de obtener **información valiosa** para la gestión de inventario
- La optimización del **rendimiento** en consultas complejas