Proyecto: Base de Datos TaconMadre

Alumno: Abner Soberon Martinez

Profesor: Ricardo Monrroy

Clase: G4-25 Introduccion a bases de datos

Este documento evalúa el proyecto de base de datos MySQL **TaconMadre** con base en cinco criterios clave requeridos para un sistema de gestión de pedidos en un restaurante.

---

1. Requisitos del sistema e identificación de entidades

El proyecto define e identifica correctamente todas las entidades esenciales para gestionar las operaciones de un restaurante, incluyendo usuarios, empleados, productos, pedidos, proveedores, ventas y compras. Cada entidad cuenta con atributos adecuados, relaciones bien establecidas mediante claves foráneas, y restricciones claras como `NOT NULL`, campos auto-incrementales y tipos enumerados (`ENUM`). La estructura refleja de manera fiel los procesos del mundo real y permite manejar tanto pedidos físicos como digitales de forma eficiente.

---

2. Coherencia, normalización y estructura organizativa

La base de datos está altamente normalizada y organizada de forma lógica, lo cual garantiza coherencia, integridad de los datos y facilidad de mantenimiento. Los módulos funcionales están claramente separados (usuarios, inventario, ventas, compras, logística, recursos humanos, etc.), y se integran componentes avanzados como gestión de almacenes, movimientos de inventario, bitácora de actividades del personal, y diversos métodos y canales de pago. Esta estructura facilita el crecimiento futuro del sistema y un desempeño óptimo en operaciones reales.

---

3. Datos de muestra y validación práctica

El archivo `consultas.sql` incluye ejemplos básicos de datos de muestra, como inserciones en las tablas `locations` y `orders`, lo cual permite validar parcialmente el funcionamiento del sistema. Sin embargo, sería recomendable complementar con más registros de prueba que incluyan productos, empleados, clientes y ventas completas para realizar simulaciones más amplias y comprobar a fondo la funcionalidad e integridad de la base de datos.

---

4. Uso de consultas SQL avanzadas

El proyecto incorpora consultas SQL avanzadas que permiten extraer información valiosa y generar reportes estratégicos. Se utilizan correctamente estructuras como `JOIN`, `GROUP BY`, funciones de agregación (`SUM`, `COUNT`), filtros de fechas (`INTERVAL`, `DATE_FORMAT`), y condiciones lógicas. Las consultas presentadas permiten responder preguntas clave del negocio como: ventas por mes, productos más vendidos, análisis por canal de venta y pedidos pendientes de envío, lo cual demuestra un dominio sólido del lenguaje SQL orientado al análisis de datos.

---

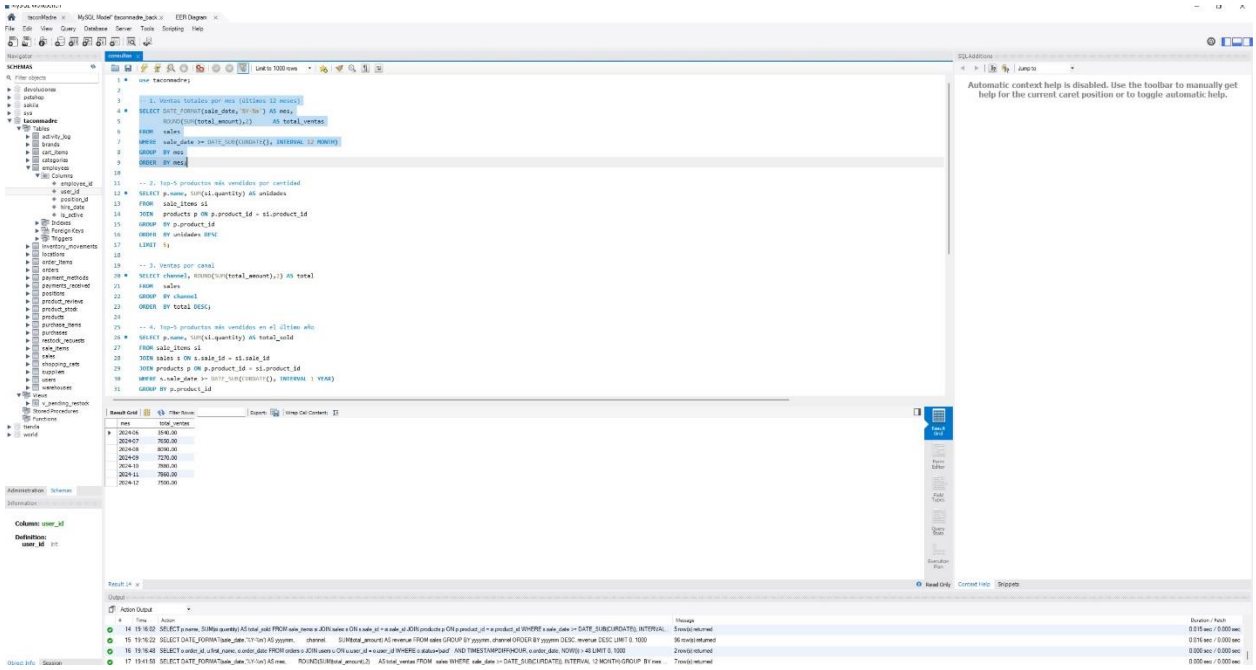
5. Documentación técnica y claridad

La documentación técnica es clara y bien estructurada. El archivo principal contiene un encabezado con el nombre del proyecto, autor, además de comentarios explicativos en cada tabla. Las claves foráneas están bien nombradas y se sigue una convención consistente. El archivo de consultas también cuenta con comentarios que explican cada bloque de código. Para una documentación

---

Consultas:

```
-- 1. Ventas totales por mes (últimos 12 meses)
SELECT DATE_FORMAT(sale_date,'%Y-%m') AS mes,
     ROUND(SUM(total_amount),2)     AS total_ventas
FROM   sales
WHERE  sale_date >= DATE_SUB(CURDATE(), INTERVAL 12 MONTH)
GROUP  BY mes
ORDER  BY mes;
```



-- 2. Top-5 productos más vendidos por cantidad

SELECT p.name, SUM(si.quantity) AS unidades

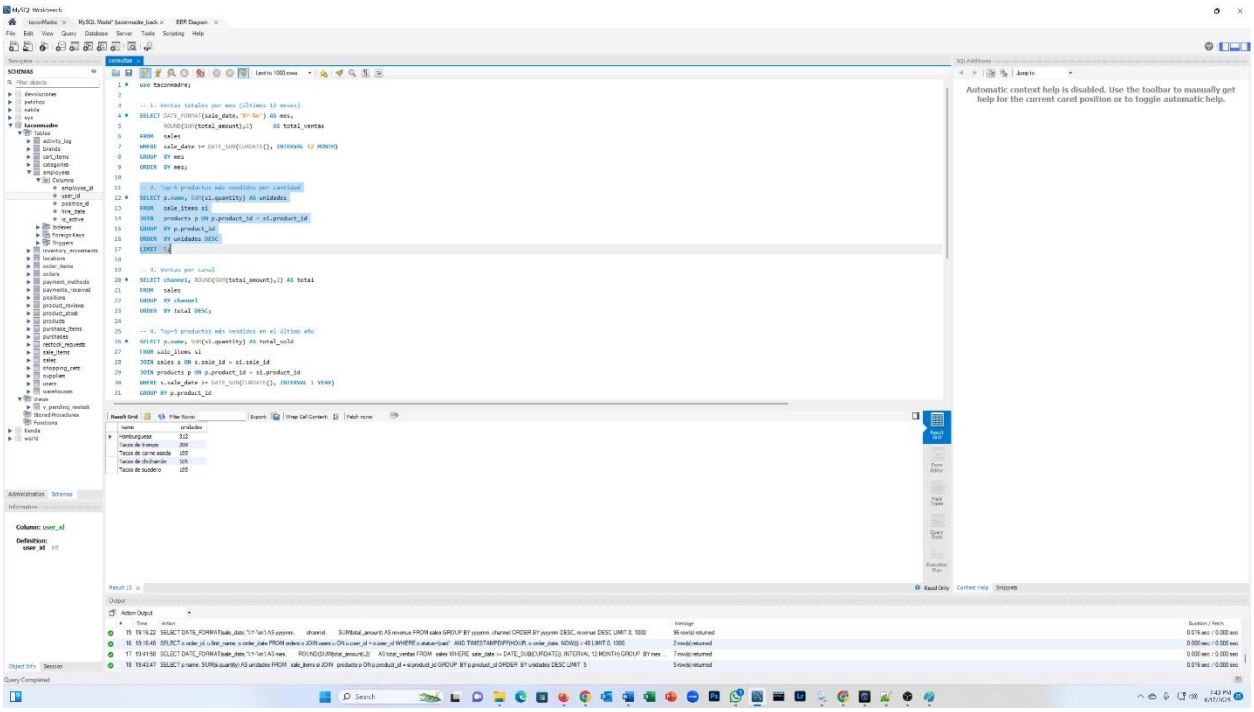
FROM   sale_items si

JOIN   products p ON p.product_id = si.product_id

GROUP BY p.product_id

ORDER BY unidades DESC

LIMIT 5;



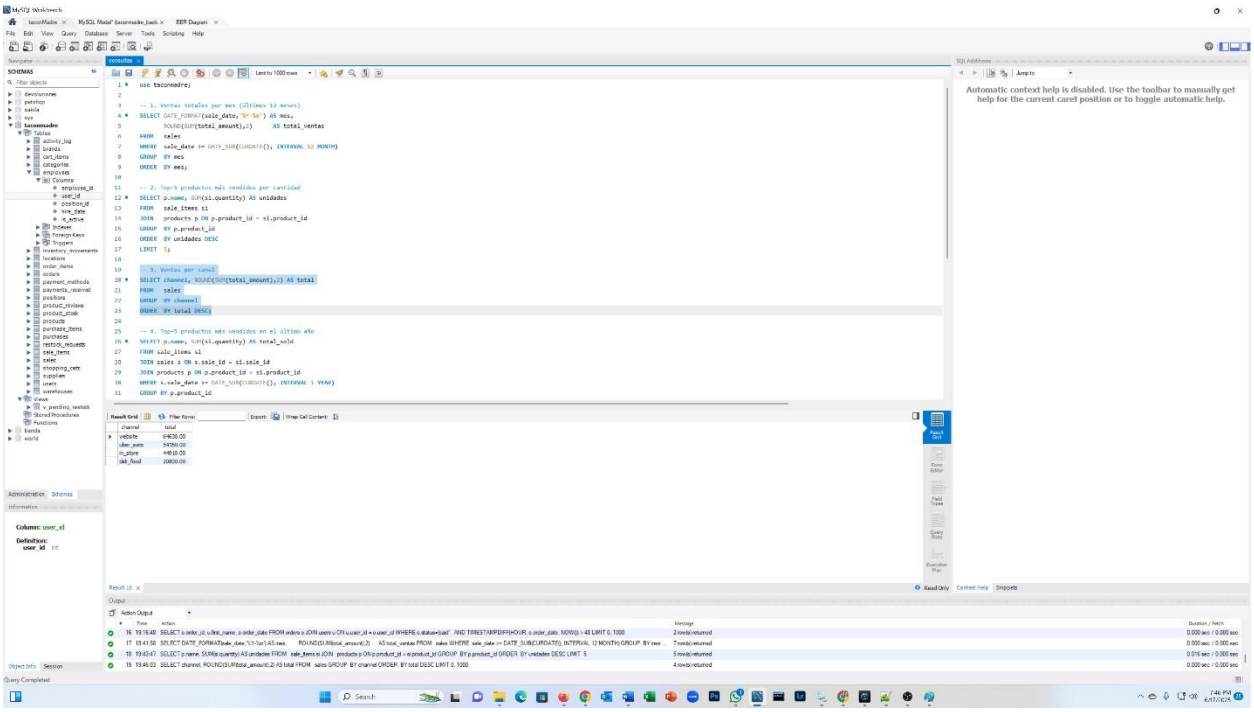
-- 3. Ventas por canal

SELECT channel, ROUND(SUM(total_amount),2) AS total

FROM sales

GROUP BY channel

ORDER BY total DESC;



-- 4. Top–5 productos más vendidos en el último año

SELECT p.name, SUM(si.quantity) AS total_sold

FROM sale_items si

JOIN sales s ON s.sale_id = si.sale_id

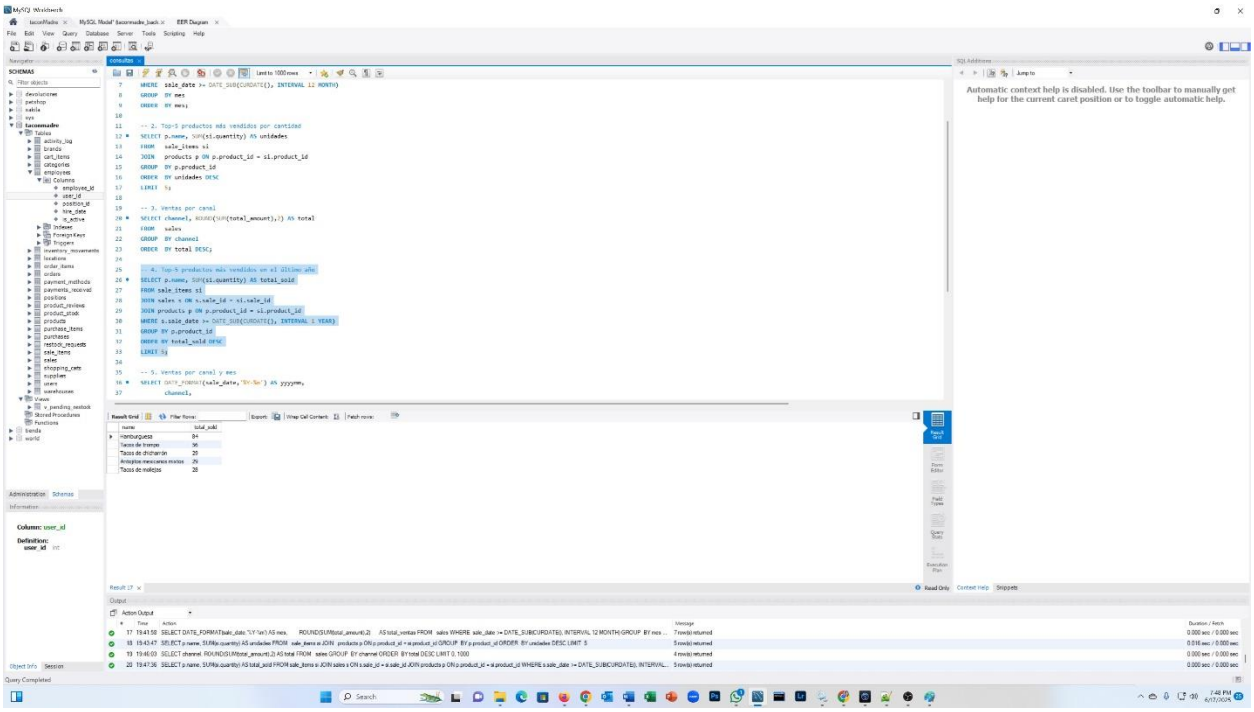
JOIN products p ON p.product_id = si.product_id

WHERE s.sale_date >= DATE_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY p.product_id

ORDER BY total_sold DESC

LIMIT 5;



```
-- 5. Ventas por canal y mes
SELECT DATE_FORMAT(sale_date,'%Y-%m') AS yyyymm,
    channel,
    SUM(total_amount) AS revenue
FROM sales
GROUP BY yyyymm, channel
ORDER BY yyyymm DESC, revenue DESC;
```