

Entrega 2: Restaurant Franquicia Almendra

Alumno: Diego Emanuel Silva

Profesor: Maximiliano Torreblanca

Tutor: Matías Cantora

Comisión: 81840

Fecha: 4 de Noviembre de 2025

Link GitHUB: https://github.com/diesilva87/franquicia_almendra

Contenido

Entrega 2: Restaurant Franquicia Almendra	1
1. ENTREGA 1	1
Introducción	1
Problemática	2
Modelo de negocio	2
Diagrama Entidad-Relación	3
Listado de tablas.....	4
2. ENTREGA 2	6
Vistas	6
Stored Procedures.....	7
Funciones	8
Triggers.....	8
Script	10

1. ENTREGA 1

Introducción

El proyecto desarrolla una base de datos para una franquicia gastronómica que abrirá en Quilmes, con horario potencial de 8:00 a 24:00. Las sucursales actuales de zona norte ya cuentan con datos de ventas y costos de reposición, pero sin detalle por franjas horarias ni días de la semana.

Si bien el negocio resulta rentable, los altos costos fijos de personal representan un desafío, ya que mantener el local abierto todo el día requiere unos 10 empleados. Por ello, se busca analizar distintas alternativas de horario y dotación para optimizar la eficiencia operativa.

El objetivo es disponer de información detallada sobre ventas y contribución marginal por franja horaria y categoría de producto, lo que permitirá entender el comportamiento de los consumidores de la zona sur y tomar decisiones basadas en datos sobre dotación, horarios de apertura y cierre, e incluso ajustes en el menú según la rentabilidad de cada ítem.

Los objetivos del proyecto son:

- Registrar ventas a nivel de ítem con su precio de venta y costo de reposición.
- Calcular contribución marginal por franja horaria y categoría de producto.
- Identificar franjas horarias de mayor y menor facturación para ajustar la dotación de empleados.
- Determinar la franja horaria ideal de operación del local, equilibrando ingresos y gastos fijos de sueldos.
- Proveer información confiable durante un período de tres meses para generar reportes y simulaciones.

Problemática

Actualmente, las franquicias de zona norte cuentan con información de ventas y costos, pero sin discriminación por franjas horarias ni días de la semana, lo que limita la capacidad de análisis y dificulta la toma de decisiones operativas.

Como resultado, las decisiones sobre la dotación de personal y los horarios de atención se basan en datos agregados, lo que provoca:

- Sobre-dotación en horas de baja contribución.
- Sub-dotación en horarios de alta demanda.
- Horarios de apertura y cierre poco eficientes, que incrementan los gastos fijos sin un retorno proporcional en ingresos.

Estas ineficiencias reducen la rentabilidad total del negocio.

La implementación de una base de datos relacional permitirá vincular ventas, costos y horarios, facilitando el cálculo de la contribución marginal por franja horaria y respaldando decisiones estratégicas como la definición del horario óptimo de operación.

Modelo de negocio

La franquicia comercializa productos organizados en distintas categorías (por ejemplo: Cafetería, Panificados, Platos y Bebidas).

Cada producto tiene un costo de reposición unitario, definido por la franquicia, y un precio de venta que puede variar según la lista vigente.

El modelo de negocio contempla:

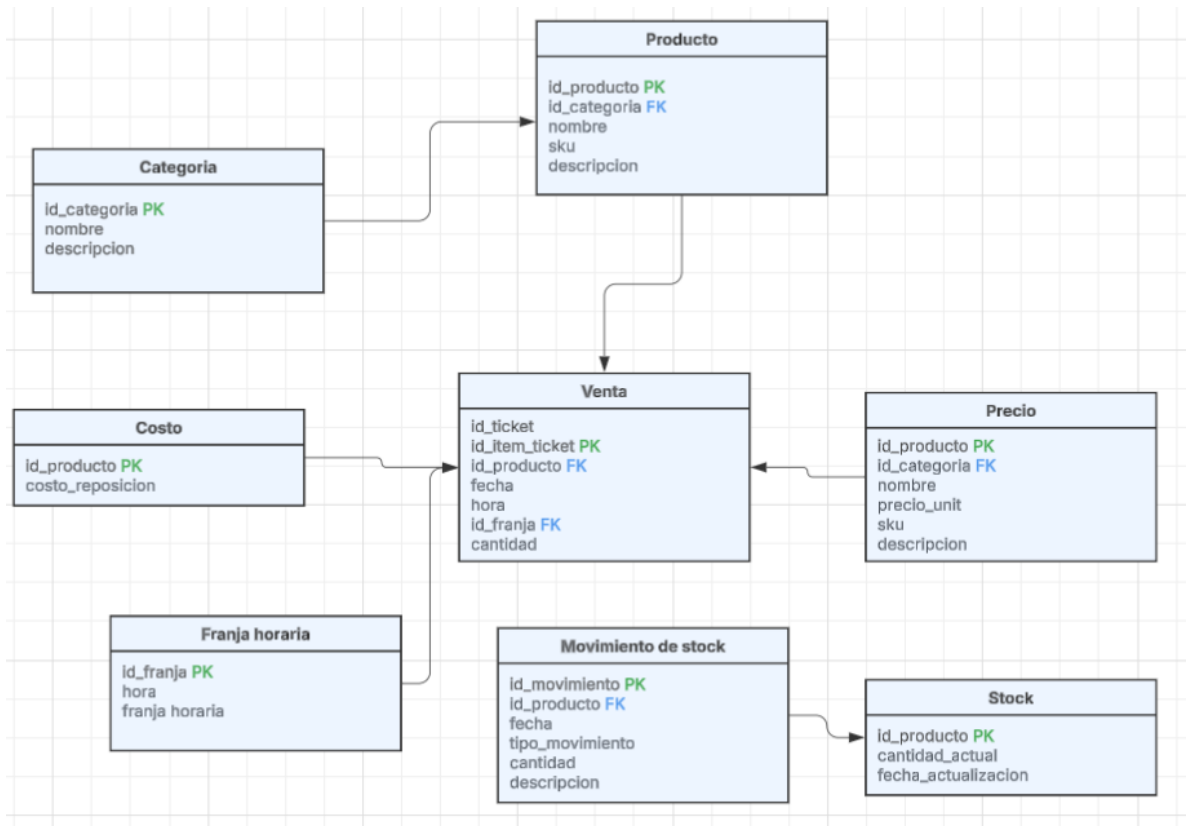
- El cliente realiza un pedido (Venta).
- Cada pedido incluye uno o más ítems (tabla Ventas), con su precio unitario y cantidad.
- Cada ítem se asocia a un Producto (tabla Productos), que pertenece a una Categoría (tabla Categoría) y tiene asignado un costo de reposición unitario (tabla Costos).
- Cada venta se registra con fecha y hora, lo que permite asignarla a una Franja Horaria predefinida (por ejemplo: 8–9, 9–11, 11–12, 13–14, etc.).
- La contribución marginal se calcula como:
(precio de venta – costo de reposición unitario) × cantidad.
- Los reportes se pueden consolidar por Franja Horaria y por Categoría.

De esta manera, el modelo permite optimizar la dotación de personal y determinar el horario de operación más rentable para el local.

Diagrama Entidad-Relación

Relaciones principales:

- **Categoría (1–N) Producto**
- **Producto (1–1) Costos**
- **Producto (1–1) Precios**
- **Producto (1–N) Ventas**
- **Franja horaria (1–N) hora**



Listado de tablas

A continuación, se detalla las tablas utilizadas:

Tabla Producto:

Campo	Tipo de dato	PK	FK	NULL
id_producto	INT	X		
id_categoria	INT		X	
nombre	VARCHAR(100)			
sku	VARCHAR(100)			x
descripcion	VARCHAR(150)			x

Tabla Categoría:

Campo	Tipo de dato	PK	FK	NULL
id_categoria	INT	X		
categoria	VARCHAR(100)			
descripcion	VARCHAR(100)			x

Tabla Costo:

Campo	Tipo de dato	PK	FK	NULL
id_producto	INT	X		
costo_reposicion	DECIMAL(10,2)			

Tabla Precio:

Campo	Tipo de dato	PK	FK	NULL
id_producto	INT	X		
id_categoria	INT		X	
nombre	VARCHAR(100)			
precio_unit	DECIMAL(10,2)			
sku	VARCHAR(100)			x
descripcion	VARCHAR(150)			x

Tabla Venta:

Campo	Tipo de dato	PK	FK	NULL
Id_ticket	INT			
id_item_ticket	INT	X		
id_producto	INT		X	
fecha	DATE			
Hora	TIME			
id_franja	INT		X	
cantidad	DECIMAL(10,0)			

Tabla Franja Horaria:

Campo	Tipo de dato	PK	FK	NULL
id_franja	INT	X		
Hora	TIME			
Franja_horaria	VARCHAR(100)			

Tabla Stock:

Campo	Tipo de dato	PK	FK	NULL
id_producto	INT	X		
cantidad_actual	DECIMAL(10,2)			
fecha_actualizacion	DATETIME			

Tabla Movimiento de stock:

Campo	Tipo de dato	PK	FK	NULL
id_movimiento	INT	X		
id_producto	DECIMAL(10,2)		X	
fecha	DATETIME			
tipo_movimiento	ENUM			
cantidad	DECIMAL(10,2)			
descripcion	VARCHAR(200)			X

2. ENTREGA 2

Este documento incluye las correcciones correspondientes de la ENTREGA 1 y detalla las *Vistas*, *Funciones*, *Stored Procedures* y *Triggers* creados, junto con su objetivo y tablas involucradas.

Al final de este documento se adjuntan 2 scripts. Uno con la creación de todos los objetos y el otro con la inserción de los datos para cada una de las tablas.

Vistas

1) vw_stock_actual_detalle

Objetivo: Ver el stock actual por producto con su categoría, ordenado de menor a mayor.

Devuelve: id_producto, producto, categoria, cantidad_actual, fecha_actualizacion.

Tablas: stock, product y categoria.

2) vw_movimientos_recientes

Objetivo: Listar movimientos de stock de los últimos 15 días con detalle de producto y tipo.

Devuelve: id_movimiento, fecha, tipo_movimiento, cantidad, descripcion, id_producto, producto, categoria.

Tablas: movimiento_stock, product y categoria.

3) vw_ventas_por_franja

Objetivo: Resumen de volumen de ventas por franja horaria.

Devuelve: franja_horaria, tickets_emitidos, total_unidades_vendidas.

Tablas: venta, franja_horaria

4) vw_facturacion_por_franja

Objetivo: facturación, costo, contribución marginal y % de margen.

Devuelve: franja_horaria, tickets_emitidos, facturacion_bruta, costo_total, contribucion_marginal, margen_porcentual.

Tablas: venta, franja_horaria, precio, costo.

5) vw_margen_por_categoria

Objetivo: Contribución marginal por categoría (sin franja).

Devuelve: categoria, tickets, unidades, facturacion_bruta, costo_total, contribucion_marginal, margen_porcentual.

Tablas: venta, producto, categoria, precio, costo.

Stored Procedures

1) Sp_registrar_mov_stock (p_id_producto, p_tipo_movimiento, p_cantidad, p_descripcion)

Objetivo: Registrar un movimiento de stock (INGRESO/EGRESO/AJUSTE). Valida cantidad > 0.

Efecto: INSERT en movimiento_stock. La actualización de stock la realiza el trigger trg_actualizacion_stock.

Tablas: movimiento_stock. Indirectamente: stock (vía trigger).

2) sp_resumen_por_franja_simple(p_id_franja)

Objetivo: Resumen simple para una franja: tickets, unidades, facturación, costo, contribución y % margen.

Devuelve (Select): id_franja, franja_horaria, tickets, unidades, facturacion_bruta, costo_total, contribucion_marginal, margen_porcentual.

Tablas: venta, franja_horaria, precio, costo.

Funciones

1) fn_contribucion_marginal(precio_unitario, costo_unitario)

Objetivo: Retornar la contribución marginal unitaria: precio – costo.

Tablas: Precio y Costo

2) fn_margen_total_producto(precio_unitario, costo_unitario, cantidad)

Objetivo: Retornar la contribución marginal total: (precio – costo) × cantidad.

Tablas: Precio, Costo y venta.

3) fn_margen_producto(o_id_producto)

Objetivo: Calcular % de margen unitario de un producto: $((\text{precio} - \text{costo}) / \text{precio}) \times 100$.

Tablas: Precio, Costo.

Triggers

1) trg_venta_before_insert

Objetivo: Validar que el producto tenga precio y costo vigentes antes de registrar la venta.

Efecto: Si falta precio/costo → SIGNAL error. No escribe en otras tablas.

Tablas: venta, precio, costo.

2) trg_ms_before_insert

Objetivo: Evitar stock negativo en egresos.

Efecto: Si cantidad_actual del stock < NEW.cantidad → SIGNAL 'Stock insuficiente'.

Tablas: movimiento_stock, stock.

3) trg_actualizacion_stock

Objetivo: Actualizar la tabla stock en función del movimiento recién insertado.

Efecto:

- ✓ INGRESO: suma cantidad.
- ✓ EGRESO: resta cantidad.
- ✓ AJUSTE: setea cantidad exacta.
- ✓ Si no había registro en stock: crea en INGRESO/AJUSTE; en EGRESO señala error

Tablas: movimiento_stock, stock.

4) tr_venta_set_id_franja

Objetivo: Asignar automáticamente id_franja según la hora de la venta.

Efecto: SET NEW.id_franja con el id_franja cuya hora coincide.

Tablas: venta, franja_horaria.

5) tr_precio_set_categoria

Objetivo: Completar id_categoria del precio a partir del producto.

Efecto: SET NEW.id_categoria.

Tablas: precio, producto.

6) tr_precio_set_nombre

Objetivo: Completar nombre del producto en la fila de precio.

Tablas: precio, producto.

7) tr_precio_set_sku

Objetivo: Completar sku en precio desde producto.

Tablas: precio, producto.

8) tr_precio_set_descripcion

Objetivo: Completar descripcion en precio desde producto.

Tablas: precio, producto.

Script

Se dejan los scripts de objetos y datos adjuntos a continuación y también en la carpeta correspondiente de Github: https://github.com/diesilva87/franquicia_almendra



script entrega
2.objetos.sql



script entrega
2.datos.sql