Documentación Base de Datos

**🧑‍💼 TABLA: users**

**Propósito**

Almacena todos los usuarios del sistema: administradores, líderes, vendedores y clientes. Incluye sistema completo de referidos y jerarquías comerciales.

**Campos Principales**

**Información Personal**

* **id**: Identificador único del usuario
* **name**: Nombre del usuario
* **apellidos**: Apellidos completos
* **cedula**: Cédula única (CLAVE para autenticación)
* **email**: Email único para login
* **telefono**: Teléfono de contacto
* **direccion**: Dirección física completa
* **ciudad**: Ciudad de residencia
* **departamento**: Departamento/estado
* **fecha\_nacimiento**: Para validaciones de edad

**Sistema de Roles**

* **rol**: 4 niveles jerárquicos
  + cliente: Usuarios que compran
  + vendedor: Gestionan pedidos y venden
  + lider: Supervisan vendedores
  + administrador: Control total del sistema

**Sistema de Referidos**

* **referido\_por**: ID del usuario que lo refirió (NULL si no fue referido)
* **codigo\_referido**: Código único para referir otros (ej: VEND001, CLIENTE002)
* **total\_referidos**: Contador de personas que ha referido
* **comisiones\_ganadas**: Total histórico de comisiones
* **comisiones\_disponibles**: Comisiones pendientes de pago

**Información Comercial**

* **meta\_mensual**: Meta de ventas (solo vendedores/líderes)
* **ventas\_mes\_actual**: Ventas del mes en curso
* **nivel\_vendedor**: Del 1-5 basado en rendimiento
* **zonas\_asignadas**: JSON con zonas que puede atender

**Cómo Funciona el Sistema de Referidos**

**Flujo de Referidos:**

1. **Vendedor/Líder** obtiene código único (ej: VEND001)
2. **Comparte su código** con posibles clientes
3. **Nuevo cliente se registra** usando el código
4. **Se crea relación** en tabla referidos
5. **Se generan comisiones** automáticamente

**Jerarquía de Roles:**

Administrador

↓

Líder (puede tener vendedores referidos)

↓

Vendedor (puede referir clientes)

↓

Cliente (puede ser referido)

**Ejemplos de Uso**

**Cliente Referido:**

-- Ana fue referida por María (vendedor ID 3)

INSERT INTO users (..., referido\_por) VALUES (..., 3);

-- Automáticamente se actualiza total\_referidos de María

**Consulta de Red de Referidos:**

-- Ver todos los referidos de un vendedor

SELECT \* FROM users WHERE referido\_por = 3;

-- Ver el referidor de un cliente

SELECT r.name FROM users u

JOIN users r ON u.referido\_por = r.id

WHERE u.id = 5;

**Campos de Control**

* **activo**: Para suspender usuarios sin eliminarlos
* **ultimo\_acceso**: Para estadísticas de uso
* **created\_at/updated\_at**: Auditoría de cambios

**🏷️ TABLA: categorias**

**Propósito**

Organizar productos en grupos lógicos para facilitar navegación, reportes y gestión del catálogo.

**Campos**

* **id**: Identificador único
* **nombre**: Nombre de la categoría (ej: "Arepas Tradicionales")
* **descripcion**: Descripción detallada de la categoría
* **imagen**: URL de imagen representativa (opcional)
* **activo**: Para ocultar/mostrar categorías sin eliminarlas

**Categorías Actuales del Sistema**

1. **Arepas Tradicionales**: Arepas básicas (sencilla, queso)
2. **Arepas Rellenas**: Con rellenos elaborados (pollo, carne)
3. **Arepas Especiales**: Gourmet o con preparación especial
4. **Bebidas**: Líquidos (chichas, jugos)
5. **Postres**: Dulces tradicionales

**Escalabilidad Futura**

El sistema está preparado para agregar fácilmente:

* **Empanadas**
* **Tamales**
* **Platos Principales**
* **Salsas y Aderezos**
* **Combos y Promociones**

**Cómo se Usa**

**En el Catálogo:**

-- Mostrar productos por categoría

SELECT p.\* FROM productos p

JOIN categorias c ON p.categoria\_id = c.id

WHERE c.activo = TRUE

ORDER BY c.nombre, p.orden\_mostrar;

**En Reportes:**

-- Ventas por categoría

SELECT c.nombre, SUM(dp.subtotal) as total\_ventas

FROM categorias c

JOIN productos p ON c.id = p.categoria\_id

JOIN detalle\_pedidos dp ON p.id = dp.producto\_id

GROUP BY c.id;

**Beneficios**

* **Navegación intuitiva** en el catálogo
* **Reportes segmentados** por tipo de producto
* **Gestión de inventario** por categorías
* **Promociones específicas** por grupo de productos

**🥞 TABLA: productos**

**Propósito**

Catálogo maestro de productos con información completa, control de inventario, precios diferenciados y métricas de ventas.

**Identificación y Básicos**

* **id**: Identificador único interno
* **codigo**: Código comercial único (ARE-001, BEB-001)
* **nombre**: Nombre comercial del producto
* **descripcion**: Descripción detallada para clientes
* **categoria\_id**: Relación con tabla categorias
* **slug**: URL amigable (arepa-sencilla)

**Sistema de Precios (3 Niveles)**

* **precio**: Precio público final
* **precio\_costo**: Costo de producción (para calcular rentabilidad)
* **precio\_mayorista**: Precio especial para vendedores

**Ejemplo de Precios:**

Arepa de Queso:

- Costo: $2,200 (lo que cuesta producirla)

- Mayorista: $4,400 (precio para vendedores)

- Público: $5,500 (precio final al cliente)

- Margen: 150% sobre costo

**Control de Inventario**

* **stock**: Cantidad actual disponible
* **stock\_minimo**: Alerta cuando llegue a este nivel
* **stock\_maximo**: Límite máximo (para no sobreproducir)
* **unidad\_medida**: unidad, gramos, litros, etc.

**Flujo de Inventario:**

1. **Producción**: Se aumenta stock
2. **Venta**: Se disminuye automáticamente
3. **Alerta**: Sistema avisa cuando stock ≤ stock\_minimo
4. **Reporte**: Vista de productos críticos

**Información del Producto (Específica para Arepas)**

* **ingredientes**: JSON ["harina de maíz", "agua", "sal"]
* **alergenos**: JSON ["gluten", "lactosa"] (para alertas)
* **informacion\_nutricional**: Texto libre con datos nutricionales
* **tiempo\_preparacion**: Minutos que toma preparar
* **peso**: Gramos (para costos de envío)
* **calorias**: Información nutricional

**Multimedia y Marketing**

* **imagenes**: JSON con array de URLs de fotos
* **imagen\_principal**: URL de la foto principal
* **tags**: JSON ["tradicional", "sin-gluten", "vegetariana"]
* **orden\_mostrar**: Para ordenar en catálogo
* **destacado**: Para mostrar en sección especial

**Estados del Producto**

* **disponible**: Si se puede vender (true/false)
* **requiere\_preparacion**: Si hay que cocinarlo al momento
* **permite\_personalizacion**: Si acepta modificaciones (sin cebolla, extra queso)

**Métricas (Desnormalización para Performance)**

* **veces\_vendido**: Contador histórico total
* **rating\_promedio**: Promedio de calificaciones (futuro)
* **total\_reviews**: Cantidad de reseñas (futuro)

**Ejemplos de Consultas Importantes**

**Productos con Stock Crítico:**

SELECT codigo, nombre, stock, stock\_minimo

FROM productos

WHERE stock <= stock\_minimo AND disponible = TRUE;

**Productos Más Vendidos:**

SELECT nombre, veces\_vendido, precio

FROM productos

ORDER BY veces\_vendido DESC

LIMIT 10;

**Cálculo de Rentabilidad:**

SELECT nombre, precio, precio\_costo,

ROUND(((precio - precio\_costo) / precio\_costo) \* 100, 2) as margen\_porcentaje

FROM productos

WHERE precio\_costo IS NOT NULL;

**Búsqueda por Tags:**

SELECT \* FROM productos

WHERE JSON\_CONTAINS(tags, '"vegetariana"')

AND disponible = TRUE;

**Escalabilidad**

Esta estructura permite agregar fácilmente:

* **Variaciones** (tallas, sabores)
* **Promociones** temporales
* **Descuentos** por volumen
* **Nuevas categorías** sin cambiar estructura

**🚚 TABLA: zonas\_entrega**

**Propósito**

Definir áreas geográficas de cobertura con costos y tiempos de entrega diferenciados, optimizando la logística de distribución.

**Información Básica**

* **id**: Identificador único
* **nombre**: Nombre de la zona (Centro, Norte, Sur, Suba)
* **descripcion**: Descripción detallada del área
* **activo**: Para habilitar/deshabilitar zonas

**Configuración de Entrega**

* **costo\_envio**: Precio fijo por entrega en esta zona
* **tiempo\_entrega\_min**: Tiempo mínimo en minutos
* **tiempo\_entrega\_max**: Tiempo máximo en minutos
* **pedido\_minimo**: Valor mínimo para atender la zona

**Definición Geográfica**

* **barrios**: JSON con array de barrios cubiertos
* **coordenadas**: JSON con polígono de cobertura (futuro GPS)
* **referencias**: Puntos de referencia importantes

**Configuración Operativa**

* **horarios**: JSON con horarios de atención por día
* **vendedores\_asignados**: JSON con IDs de vendedores de la zona

**Ejemplo de Configuración**

**Zona Centro:**

{

"nombre": "Centro",

"costo\_envio": 3000,

"tiempo\_entrega\_min": 25,

"tiempo\_entrega\_max": 45,

"pedido\_minimo": 10000,

"barrios": ["La Candelaria", "Centro", "Santa Bárbara"],

"horarios": {

"lunes": {"inicio": "08:00", "fin": "18:00", "activo": true},

"sabado": {"inicio": "08:00", "fin": "16:00", "activo": true},

"domingo": {"inicio": "09:00", "fin": "15:00", "activo": false}

},

"vendedores\_asignados": [3, 6]

}

**Cómo se Usa en el Sistema**

**Al Crear Pedido:**

1. **Cliente ingresa dirección**
2. **Sistema determina zona** (manual o automático)
3. **Calcula costo y tiempo** según zona
4. **Valida pedido mínimo**
5. **Asigna vendedor** de la zona

**Consulta de Disponibilidad:**

-- Verificar si una zona está activa en horario actual

SELECT \* FROM zonas\_entrega

WHERE activo = TRUE

AND JSON\_EXTRACT(horarios, '$.lunes.activo') = true

AND CURTIME() BETWEEN

JSON\_UNQUOTE(JSON\_EXTRACT(horarios, '$.lunes.inicio'))

AND JSON\_UNQUOTE(JSON\_EXTRACT(horarios, '$.lunes.fin'));

**Asignación de Vendedores:**

-- Vendedores disponibles para una zona

SELECT u.\* FROM users u

JOIN zonas\_entrega z ON JSON\_CONTAINS(z.vendedores\_asignados, CAST(u.id AS JSON))

WHERE z.id = 1 AND u.activo = TRUE;

**Beneficios del Sistema**

* **Costos justos** según distancia y dificultad
* **Tiempos realistas** de entrega
* **Optimización de rutas** por vendedor
* **Control de cobertura** geográfica
* **Escalabilidad** para nuevas zonas

**Futuras Mejoras**

* **Integración GPS** para detección automática
* **Rutas optimizadas** con algoritmos
* **Zonas dinámicas** según demanda
* **Precios variables** por horario (ej: noche más caro)

**📋 TABLA: pedidos**

**Propósito**

Gestionar el ciclo completo de órdenes desde la creación hasta la entrega, incluyendo cálculos financieros y asignación de vendedores.

**Identificación y Cliente**

* **id**: Identificador único interno
* **numero\_pedido**: Código visible para cliente (ARE-2024-0001)
* **user\_id**: Cliente que hizo el pedido
* **vendedor\_id**: Vendedor asignado para gestionar
* **fecha\_asignacion**: Cuándo se asignó al vendedor

**Estados del Pedido (Flujo Completo)**

pendiente → confirmado → en\_preparacion → listo → en\_camino → entregado

↓

cancelado (en cualquier momento)

**Descripción de Estados:**

* **pendiente**: Recién creado, esperando confirmación
* **confirmado**: Validado, se descuenta inventario
* **en\_preparacion**: Cocinándose en cocina
* **listo**: Terminado, esperando entrega
* **en\_camino**: Vendedor en ruta al cliente
* **entregado**: Completado exitosamente
* **cancelado**: Anulado (se restaura inventario)

**Información Financiera**

* **subtotal**: Suma de productos sin impuestos ni envío
* **impuestos**: IVA calculado (19% en Colombia)
* **costo\_envio**: Según zona de entrega
* **total**: subtotal + impuestos + costo\_envio

**Ejemplo de Cálculo:**

Productos: $16,000

IVA (19%): $3,040

Envío: $5,000

TOTAL: $24,040

**Información de Entrega**

* **tipo\_entrega**: 'domicilio' o 'retiro'
* **direccion\_entrega**: Dirección completa del cliente
* **telefono\_entrega**: Teléfono para coordinación
* **zona\_entrega\_id**: Relación con tabla zonas\_entrega
* **observaciones**: Notas especiales del cliente

**Control de Tiempos**

* **fecha\_estimada\_entrega**: Cuándo debe llegar
* **fecha\_entrega\_real**: Cuándo realmente llegó
* **created\_at**: Cuándo se creó el pedido

**Flujo Típico de un Pedido**

**1. Creación (Estado: pendiente)**

INSERT INTO pedidos (numero\_pedido, user\_id, estado, subtotal, total, ...)

VALUES ('ARE-2024-0001', 4, 'pendiente', 16000, 24040, ...);

**2. Confirmación (Estado: confirmado)**

* Se valida disponibilidad de productos
* Se descuenta inventario automáticamente
* Se asigna vendedor de la zona
* Se calculan comisiones

**3. Preparación (Estado: en\_preparacion)**

* Cocina recibe la orden
* Se estima tiempo real de preparación

**4. Entrega (Estado: listo → en\_camino → entregado)**

* Vendedor recoge el pedido
* Inicia ruta al cliente
* Confirma entrega

**Consultas Importantes**

**Pedidos Pendientes por Vendedor:**

SELECT p.numero\_pedido, u.name as cliente, p.total, p.estado

FROM pedidos p

JOIN users u ON p.user\_id = u.id

WHERE p.vendedor\_id = 3

AND p.estado IN ('confirmado', 'en\_preparacion', 'listo');

**Métricas de Tiempo de Entrega:**

SELECT

AVG(TIMESTAMPDIFF(MINUTE, created\_at, fecha\_entrega\_real)) as tiempo\_promedio,

zona\_entrega\_id

FROM pedidos

WHERE estado = 'entregado'

AND fecha\_entrega\_real IS NOT NULL

GROUP BY zona\_entrega\_id;

**Ventas del Día:**

SELECT COUNT(\*) as total\_pedidos, SUM(total) as total\_ventas

FROM pedidos

WHERE DATE(created\_at) = CURDATE()

AND estado IN ('entregado', 'confirmado');

**Automatizaciones del Sistema**

**Al Confirmar Pedido:**

1. **Descuenta inventario** automáticamente
2. **Calcula comisiones** para vendedor y referidores
3. **Envía notificaciones** a vendedor asignado
4. **Actualiza métricas** de productos vendidos

**Al Entregar:**

1. **Registra fecha real** de entrega
2. **Actualiza estadísticas** del vendedor
3. **Procesa comisiones** pendientes
4. **Envía confirmación** al cliente

**Integración con Otras Tablas**

* **detalle\_pedidos**: Productos específicos del pedido
* **comisiones**: Ganancias generadas por este pedido
* **movimientos\_inventario**: Registro de cambios de stock
* **notificaciones**: Alertas relacionadas al pedido

**🛒 TABLA: detalle\_pedidos**

**Propósito**

Almacenar cada producto individual dentro de un pedido, con cantidades, precios al momento de la venta y personalizaciones.

**Campos Principales**

* **id**: Identificador único del detalle
* **pedido\_id**: Relación con tabla pedidos
* **producto\_id**: Relación con tabla productos
* **cantidad**: Cuántas unidades del producto
* **precio\_unitario**: Precio al momento de la venta (histórico)
* **subtotal**: cantidad × precio\_unitario
* **observaciones\_producto**: Personalizaciones del cliente

**¿Por Qué Guardar Precio Histórico?**

**Problema sin Precio Histórico:**

Cliente compró arepa a $5,000 en enero

En marzo el precio subió a $6,000

Si consulto el pedido de enero, ¿qué precio muestro?

**Solución con Precio Histórico:**

-- Al crear el pedido, guardamos el precio actual

INSERT INTO detalle\_pedidos (producto\_id, precio\_unitario, ...)

VALUES (1, (SELECT precio FROM productos WHERE id = 1), ...);

**Ejemplo de Pedido Completo**

**Pedido ARE-2024-0001:**

-- Pedido principal

INSERT INTO pedidos (...) VALUES ('ARE-2024-0001', 4, ...);

-- Detalles del pedido

INSERT INTO detalle\_pedidos VALUES

(1, 1, 3, 1, 12000.00, 12000.00, 'Sin cebolla'), -- Arepa Rellena Pollo

(1, 1, 1, 1, 3500.00, 3500.00, NULL), -- Arepa Sencilla

(1, 1, 6, 1, 4500.00, 4500.00, NULL); -- Chicha de Maíz

-- Total del pedido: 12000 + 3500 + 4500 = $20,000

**Personalización de Productos**

**Observaciones Típicas:**

* **"Sin cebolla"** - Para arepas rellenas
* **"Extra queso"** - Agregado especial
* **"Sin sal"** - Restricciones dietéticas
* **"Bien asada"** - Preferencia de cocción
* **"Para llevar"** - Empaque especial

**Consulta de Personalizaciones:**

SELECT p.nombre, dp.cantidad, dp.observaciones\_producto

FROM detalle\_pedidos dp

JOIN productos p ON dp.producto\_id = p.id

WHERE dp.pedido\_id = 1

AND dp.observaciones\_producto IS NOT NULL;

**Consultas Importantes**

**Productos Más Vendidos:**

SELECT

p.nombre,

SUM(dp.cantidad) as total\_vendido,

SUM(dp.subtotal) as total\_ingresos

FROM detalle\_pedidos dp

JOIN productos p ON dp.producto\_id = p.id

JOIN pedidos ped ON dp.pedido\_id = ped.id

WHERE ped.estado IN ('entregado', 'confirmado')

GROUP BY p.id

ORDER BY total\_vendido DESC;

**Análisis de Precios Históricos:**

SELECT

p.nombre,

dp.precio\_unitario as precio\_vendido,

p.precio as precio\_actual,

dp.created\_at as fecha\_venta

FROM detalle\_pedidos dp

JOIN productos p ON dp.producto\_id = p.id

WHERE p.id = 1

ORDER BY dp.created\_at DESC;

**Ventas por Período:**

SELECT

DATE(ped.created\_at) as fecha,

COUNT(dp.id) as items\_vendidos,

SUM(dp.subtotal) as total\_dia

FROM detalle\_pedidos dp

JOIN pedidos ped ON dp.pedido\_id = ped.id

WHERE ped.estado = 'entregado'

AND ped.created\_at >= DATE\_SUB(NOW(), INTERVAL 7 DAY)

GROUP BY DATE(ped.created\_at)

ORDER BY fecha DESC;

**Integridad de Datos**

**Validaciones Importantes:**

1. **Cantidad > 0**: No se pueden vender cantidades negativas
2. **Precio > 0**: Todos los productos deben tener precio
3. **Subtotal = cantidad × precio**: Coherencia matemática
4. **Producto disponible**: Solo productos activos

**Triggers Automáticos:**

* **Al insertar**: Actualiza stock del producto
* **Al eliminar**: Restaura stock (si pedido se cancela)
* **Al modificar**: Recalcula totales del pedido

**Relación con Inventario**

**Flujo de Stock:**

-- Al confirmar pedido, se ejecuta automáticamente:

UPDATE productos

SET stock = stock - (SELECT SUM(cantidad) FROM detalle\_pedidos WHERE producto\_id = productos.id AND pedido\_id = NEW.pedido\_id)

WHERE id IN (SELECT producto\_id FROM detalle\_pedidos WHERE pedido\_id = NEW.pedido\_id);

**Control de Disponibilidad:**

-- Verificar stock antes de confirmar

SELECT p.nombre, p.stock, dp.cantidad,

CASE WHEN p.stock >= dp.cantidad THEN 'OK' ELSE 'INSUFICIENTE' END as estado

FROM detalle\_pedidos dp

JOIN productos p ON dp.producto\_id = p.id

WHERE dp.pedido\_id = 1;

**🤝 TABLA: referidos**

**Propósito**

**Registrar y hacer seguimiento a la red de referidos, permitiendo calcular comisiones y medir el crecimiento de la red comercial.**

**Relación Principal**

* **referidor\_id: Usuario que refiere (vendedor/líder)**
* **referido\_id: Usuario que fue referido (nuevo cliente)**
* **Clave única: (referidor\_id, referido\_id) - Una persona solo puede ser referida una vez por la misma persona**

**Estados del Referido**

**1. pendiente:**

* **Persona se registró con código de referido**
* **Aún no ha hecho ninguna compra**
* **No genera comisiones aún**

**2. activo:**

* **Ya hizo su primera compra**
* **Se activan las comisiones**
* **Se actualiza fecha\_activacion**

**3. completado:**

* **Ha cumplido condiciones especiales (ej: 5 compras, $100K en compras)**
* **Genera bonos adicionales**
* **Estado para referidos "premium"**

**Métricas (Desnormalización)**

**¿Por qué Desnormalizar?**

**En lugar de calcular cada vez las métricas, las guardamos para optimizar performance:**

* **total\_pedidos: Cuántos pedidos ha hecho este referido**
* **total\_compras: Valor total comprado por este referido**
* **comision\_generada: Total de comisiones que ha generado**

**Ejemplo de Flujo:**

**1. Ana se registra con código de María → estado: 'pendiente'**

**2. Ana hace primera compra de $25,000 → estado: 'activo'**

**3. Se actualiza: total\_pedidos=1, total\_compras=25000, comision\_generada=2500**

**Información del Contexto**

* **fecha\_referido: Cuándo se registró**
* **fecha\_activacion: Cuándo hizo primera compra**
* **fecha\_completado: Cuándo alcanzó estado premium**
* **canal\_referido: 'web', 'whatsapp', 'presencial'**
* **metadata: JSON con información adicional**

**Consultas Principales**

**Red de Referidos de un Vendedor:**

**SELECT**

**r.name as referidor,**

**rf.name as referido,**

**ref.estado,**

**ref.total\_compras,**

**ref.comision\_generada**

**FROM referidos ref**

**JOIN users r ON ref.referidor\_id = r.id**

**JOIN users rf ON ref.referido\_id = rf.id**

**WHERE ref.referidor\_id = 3**

**ORDER BY ref.fecha\_referido DESC;**

**Top Referidores:**

**SELECT**

**u.name,**

**COUNT(r.id) as total\_referidos,**

**SUM(r.total\_compras) as ventas\_generadas,**

**SUM(r.comision\_generada) as comisiones\_generadas**

**FROM users u**

**JOIN referidos r ON u.id = r.referidor\_id**

**WHERE r.estado = 'activo'**

**GROUP BY u.id**

**ORDER BY total\_referidos DESC;**

**Conversión de Referidos:**

**SELECT**

**COUNT(\*) as total\_referidos,**

**SUM(CASE WHEN estado = 'activo' THEN 1 ELSE 0 END) as activos,**

**SUM(CASE WHEN estado = 'completado' THEN 1 ELSE 0 END) as completados,**

**ROUND((SUM(CASE WHEN estado = 'activo' THEN 1 ELSE 0 END) / COUNT(\*)) \* 100, 2) as tasa\_conversion**

**FROM referidos**

**WHERE referidor\_id = 3;**

**Análisis por Canal:**

**SELECT**

**canal\_referido,**

**COUNT(\*) as total,**

**AVG(total\_compras) as compra\_promedio**

**FROM referidos**

**WHERE estado = 'activo'**

**GROUP BY canal\_referido;**

**Automatizaciones del Sistema**

**Al Registrar Nuevo Usuario con Código:**

**-- Se ejecuta automáticamente**

**CALL ProcesarReferido(referidor\_id, nuevo\_user\_id, 'web');**

**-- Esto crea el registro en referidos y genera comisión inicial**

**Al Primera Compra del Referido:**

**-- Se actualiza automáticamente el estado**

**UPDATE referidos**

**SET estado = 'activo',**

**fecha\_activacion = NOW(),**

**total\_pedidos = 1,**

**total\_compras = nuevo\_total,**

**comision\_generada = nuevo\_total \* 0.10**

**WHERE referido\_id = user\_id;**

**Reglas de Negocio**

**Comisiones por Referido:**

1. **Registro: $5,000 fijos al registrarse**
2. **Primera compra: 10% del valor de la compra**
3. **Compras posteriores: 5% del valor de cada compra**
4. **Bono completado: $20,000 al llegar a 5 compras**

**Límites y Validaciones:**

* **Una persona solo puede ser referida una vez**
* **No se puede auto-referir**
* **Vendedores pueden referir clientes y otros vendedores**
* **Clientes solo pueden referir otros clientes**

**Escalabilidad del Sistema**

**Niveles Futuros:**

**Nivel 1: Referido directo (10% comisión)**

**Nivel 2: Referido del referido (3% comisión)**

**Nivel 3: Referido del nivel 2 (1% comisión)**

**Implementación Multi-Nivel:**

**-- Consulta recursiva para ver la red completa**

**WITH RECURSIVE red\_referidos AS (**

**SELECT referidor\_id, referido\_id, 1 as nivel**

**FROM referidos WHERE referidor\_id = 3**

**UNION ALL**

**SELECT r.referidor\_id, r.referido\_id, rr.nivel + 1**

**FROM referidos r**

**JOIN red\_referidos rr ON r.referidor\_id = rr.referido\_id**

**WHERE rr.nivel < 3**

**)**

**SELECT \* FROM red\_referidos;**

**KPIs Importantes**

**Para Vendedores:**

* **Total de referidos registrados**
* **Tasa de conversión (% que hace primera compra)**
* **Valor promedio por referido**
* **Retención (% que hace segunda compra)**

**Para la Empresa:**

* **Crecimiento de la red (nuevos referidos por mes)**
* **Costo de adquisición por cliente**
* **Lifetime value de clientes referidos vs. directos**
* **ROI de comisiones pagadas**

**Reportes Automáticos**

**Dashboard de Referidor:**

* **Mi red de referidos**
* **Comisiones ganadas este mes**
* **Referidos activos vs. pendientes**
* **Proyección de ganancias**

**Admin Dashboard:**

* **Top 10 referidores**
* **Crecimiento de red por mes**
* **Análisis de canales más efectivos**
* **Alertas de comportamiento inusual**

**💰 TABLA: comisiones**

**Propósito**

Registrar, calcular y gestionar todas las comisiones generadas en el sistema, desde ventas directas hasta bonos por referidos.

**Identificación y Relaciones**

* **id**: Identificador único de la comisión
* **user\_id**: Usuario que recibe la comisión
* **pedido\_id**: Pedido que genera la comisión (opcional)
* **referido\_id**: Usuario referido relacionado (opcional)

**Tipos de Comisiones**

**1. venta\_directa (15%)**

* Comisión por venta propia del vendedor
* Se calcula sobre el total del pedido
* Se genera al confirmar el pedido

**2. referido\_nuevo ($5,000 fijos)**

* Bono por registrar un nuevo referido
* Se paga al momento del registro
* Independiente de si el referido compra

**3. referido\_compra (10%)**

* Comisión por compra de referido
* Se calcula sobre cada pedido del referido
* Solo para la primera compra es 10%, luego 5%

**4. liderazgo (5%)**

* Comisión para líderes por ventas de su equipo
* Se calcula sobre las ventas de vendedores bajo su mando
* Estructura jerárquica

**5. bono\_meta (variable)**

* Bono especial por cumplir metas mensuales
* Calculado según nivel alcanzado
* Se paga al final del mes

**6. ajuste (manual)**

* Correcciones manuales por admin
* Puede ser positivo o negativo
* Requiere aprobación especial

**Cálculos Financieros**

* **monto\_base**: Valor sobre el que se calcula (ej: total del pedido)
* **porcentaje**: Porcentaje aplicado (15%, 10%, etc.)
* **monto\_comision**: Resultado final del cálculo

**Ejemplo de Cálculo:**

Venta Directa:

- Pedido: $24,040

- Porcentaje: 15%

- Comisión: $24,040 × 0.15 = $3,606

**Estados de las Comisiones**

**pendiente:**

* Recién calculada automáticamente
* Esperando validaciones del sistema
* No disponible para el vendedor aún

**aprobada:**

* Validada por el sistema o admin
* Disponible para pago
* Se suma a comisiones\_disponibles del usuario

**pagada:**

* Ya fue pagada al vendedor
* Se registra fecha\_pago
* Se transfiere a comisiones\_ganadas (histórico)

**cancelada:**

* Anulada por algún motivo (pedido cancelado, error, etc.)
* No se paga al vendedor
* Se mantiene registro para auditoría

**Ejemplo de Flujo Completo**

**Pedido con Referido:**

-- 1. María (vendedor) vende $25,000 a Ana (referida por Juan)

INSERT INTO pedidos (...) VALUES (...);

-- 2. Se generan automáticamente 3 comisiones:

-- Comisión venta directa para María

INSERT INTO comisiones VALUES (

3, -- María (vendedor)

1, -- Pedido ID

NULL, -- No es por referido

'venta\_directa',

25000, -- Monto base

15, -- 15%

3750, -- $3,750 comisión

'pendiente'

);

-- Comisión referido para Juan

INSERT INTO comisiones VALUES (

2, -- Juan (referidor)

1, -- Mismo pedido

5, -- Ana (referida)

'referido\_compra',

25000, -- Monto base

10, -- 10%

2500, -- $2,500 comisión

'pendiente'

);

-- Comisión liderazgo para Líder de María

INSERT INTO comisiones VALUES (

1, -- Líder

1, -- Mismo pedido

NULL,

'liderazgo',

25000, -- Monto base

5, -- 5%

1250, -- $1,250 comisión

'pendiente'

);

**Consultas Importantes**

**Comisiones Pendientes por Usuario:**

SELECT

tipo,

COUNT(\*) as cantidad,

SUM(monto\_comision) as total\_pendiente

FROM comisiones

WHERE user\_id = 3 AND estado = 'pendiente'

GROUP BY tipo;

**Ranking de Comisiones del Mes:**

SELECT

u.name,

SUM(c.monto\_comision) as total\_comisiones,

COUNT(c.id) as total\_transacciones

FROM comisiones c

JOIN users u ON c.user\_id = u.id

WHERE c.estado = 'aprobada'

AND MONTH(c.fecha\_calculo) = MONTH(NOW())

GROUP BY u.id

ORDER BY total\_comisiones DESC;

**Análisis por Tipo de Comisión:**

SELECT

tipo,

COUNT(\*) as cantidad,

SUM(monto\_comision) as total,

AVG(monto\_comision) as promedio

FROM comisiones

WHERE estado = 'aprobada'

AND DATE(fecha\_calculo) >= DATE\_SUB(NOW(), INTERVAL 30 DAY)

GROUP BY tipo;

**Automatizaciones**

**Cálculo Automático:**

-- Trigger que se ejecuta al confirmar pedido

CALL CalcularComisionesPedido(pedido\_id);

**Aprobación Automática:**

-- Comisiones menores a $10K se aprueban automáticamente

UPDATE comisiones

SET estado = 'aprobada', fecha\_aprobacion = NOW()

WHERE estado = 'pendiente'

AND monto\_comision < 10000

AND tipo IN ('venta\_directa', 'referido\_compra');

**Proceso de Pago Masivo:**

-- Marcar como pagadas las comisiones del período

UPDATE comisiones

SET estado = 'pagada', fecha\_pago = NOW()

WHERE estado = 'aprobada'

AND user\_id IN (lista\_vendedores\_a\_pagar);

**Información Adicional**

* **descripcion**: Texto explicativo de la comisión
* **detalles**: JSON con cálculos detallados
* **aprobado\_por**: Usuario admin que aprobó (si es manual)

**Ejemplo de Detalles JSON:**

{

"pedido": "ARE-2024-0001",

"cliente": "Ana María",

"productos": ["Arepa Pollo", "Chicha"],

"calculo": "24040 \* 0.15 = 3606",

"notas": "Primera venta del mes"

}

**KPIs y Métricas**

**Para Vendedores:**

* Total comisiones del mes
* Promedio por venta
* Efectividad de referidos
* Proyección mensual

**Para Administradores:**

* Costo total en comisiones
* ROI de programa de referidos
* Vendedores más productivos
* Tendencias por tipo de comisión

**Integridad y Auditoría**

* Todas las comisiones son **inmutables** una vez creadas
* Cambios requieren registro en tabla **auditoria**
* Balances deben cuadrar con totales de usuarios
* Reportes automáticos de inconsistencias

**📦 TABLA: movimientos\_inventario**

**Propósito**

Registrar TODOS los cambios de stock para mantener trazabilidad completa, auditoría de inventario y análisis de patrones de consumo.

**Identificación y Relaciones**

* **id**: Identificador único del movimiento
* **producto\_id**: Producto afectado
* **user\_id**: Usuario responsable del movimiento
* **pedido\_id**: Pedido relacionado (si aplica)

**Tipos de Movimiento**

**entrada (Aumenta Stock)**

* Compra de materia prima
* Producción de nuevos productos
* Devolución de clientes
* Ajustes por inventario físico

**salida (Disminuye Stock)**

* Venta a cliente
* Merma por deterioro
* Regalo promocional
* Pérdida o robo

**ajuste (Corrección)**

* Diferencias en inventario físico
* Correcciones por errores
* Cambios por auditoría

**transferencia (Cambio de ubicación)**

* Entre sucursales
* Entre almacenes
* Cambio de presentación

**Motivos Específicos**

**Para entradas:**

* compra: Compra a proveedores
* produccion: Fabricación propia
* devolucion: Cliente devuelve producto

**Para salidas:**

* venta: Venta normal
* merma: Deterioro natural
* regalo: Promociones
* vencimiento: Productos vencidos
* perdida: Robo o extravío

**Control de Cantidades**

* **cantidad\_anterior**: Stock antes del movimiento
* **cantidad\_movimiento**: Cantidad del cambio (+/-)
* **cantidad\_actual**: Stock después del movimiento

**Ejemplo de Movimiento:**

Venta de 3 Arepas de Queso:

- cantidad\_anterior: 40

- cantidad\_movimiento: -3

- cantidad\_actual: 37

**Información de Costos**

* **costo\_unitario**: Costo por unidad en este movimiento
* **costo\_total**: costo\_unitario × cantidad\_movimiento

**Importancia de Costos:**

1. **Valorización del inventario** en diferentes momentos
2. **Cálculo de rentabilidad** por lote de producción
3. **Análisis de inflación** de materias primas
4. **Toma de decisiones** de precios

**Información Adicional**

* **observaciones**: Notas del usuario
* **lote**: Código de producción o compra
* **fecha\_vencimiento**: Para productos perecederos
* **metadata**: JSON con información extra

**Ejemplos de Movimientos Típicos**

**1. Producción de Arepas:**

INSERT INTO movimientos\_inventario VALUES (

NULL, -- ID auto

1, -- Arepa Sencilla

1, -- Usuario Admin

NULL, -- No es por pedido

'entrada', -- Tipo

'produccion', -- Motivo

20, -- Había 20

30, -- Se produjeron 30

50, -- Ahora hay 50

1200, -- Costo $1,200 c/u

36000, -- Costo total $36,000

'Producción matutina',

'LOTE-2024-001',

'2024-03-15', -- Vence en 15 días

NULL

);

**2. Venta a Cliente:**

INSERT INTO movimientos\_inventario VALUES (

NULL,

1, -- Arepa Sencilla

3, -- Vendedor María

1, -- Pedido ARE-2024-0001

'salida',

'venta',

50, -- Había 50

-2, -- Se vendieron 2

48, -- Quedan 48

1200, -- Costo original

2400, -- Costo de lo vendido

'Venta a Carlos Rodríguez',

NULL,

NULL,

'{"cliente": "Carlos", "canal": "domicilio"}'

);

**3. Merma por Vencimiento:**

INSERT INTO movimientos\_inventario VALUES (

NULL,

8, -- Quesillo Llanero

1, -- Admin

NULL,

'salida',

'vencimiento',

15, -- Había 15

-3, -- Se perdieron 3

12, -- Quedan 12

2800, -- Costo unitario

8400, -- Pérdida $8,400

'Producto vencido, lote LOTE-2024-001',

'LOTE-2024-001',

NULL,

NULL

);

**Consultas de Análisis**

**Historial de un Producto:**

SELECT

created\_at,

tipo,

motivo,

cantidad\_anterior,

cantidad\_movimiento,

cantidad\_actual,

observaciones

FROM movimientos\_inventario

WHERE producto\_id = 1

ORDER BY created\_at DESC;

**Análisis de Mermas:**

SELECT

p.nombre,

SUM(CASE WHEN mi.motivo = 'merma' THEN ABS(mi.cantidad\_movimiento) ELSE 0 END) as total\_merma,

SUM(CASE WHEN mi.motivo = 'venta' THEN ABS(mi.cantidad\_movimiento) ELSE 0 END) as total\_vendido,

ROUND((SUM(CASE WHEN mi.motivo = 'merma' THEN ABS(mi.cantidad\_movimiento) ELSE 0 END) /

NULLIF(SUM(CASE WHEN mi.motivo IN ('merma', 'venta') THEN ABS(mi.cantidad\_movimiento) ELSE 0 END), 0)) \* 100, 2) as porcentaje\_merma

FROM productos p

LEFT JOIN movimientos\_inventario mi ON p.id = mi.producto\_id

WHERE mi.created\_at >= DATE\_SUB(NOW(), INTERVAL 30 DAY)

GROUP BY p.id

ORDER BY porcentaje\_merma DESC;

**Valorización de Inventario:**

SELECT

p.nombre,

p.stock,

AVG(mi.costo\_unitario) as costo\_promedio,

p.stock \* AVG(mi.costo\_unitario) as valor\_inventario

FROM productos p

LEFT JOIN movimientos\_inventario mi ON p.id = mi.producto\_id

WHERE mi.tipo = 'entrada'

AND mi.created\_at >= DATE\_SUB(NOW(), INTERVAL 60 DAY)

GROUP BY p.id

HAVING p.stock > 0;

**Rotación de Inventario:**

SELECT

p.nombre,

SUM(CASE WHEN mi.tipo = 'salida' AND mi.motivo = 'venta' THEN ABS(mi.cantidad\_movimiento) ELSE 0 END) as vendido\_mes,

AVG(mi.cantidad\_actual) as stock\_promedio,

CASE

WHEN AVG(mi.cantidad\_actual) > 0 THEN

ROUND(SUM(CASE WHEN mi.tipo = 'salida' AND mi.motivo = 'venta' THEN ABS(mi.cantidad\_movimiento) ELSE 0 END) / AVG(mi.cantidad\_actual), 2)

ELSE 0

END as rotacion\_mensual

FROM productos p

LEFT JOIN movimientos\_inventario mi ON p.id = mi.producto\_id

WHERE mi.created\_at >= DATE\_SUB(NOW(), INTERVAL 30 DAY)

GROUP BY p.id

ORDER BY rotacion\_mensual DESC;

**Automatizaciones**

**Trigger Automático en Ventas:**

-- Al confirmar pedido, se registra automáticamente

TRIGGER actualizar\_stock\_pedido

AFTER UPDATE ON pedidos

FOR EACH ROW

WHEN NEW.estado = 'confirmado' AND OLD.estado = 'pendiente'

**Alertas de Stock Crítico:**

-- Verificar después de cada movimiento

IF NEW.cantidad\_actual <= (SELECT stock\_minimo FROM productos WHERE id = NEW.producto\_id) THEN

INSERT INTO notificaciones (user\_id, titulo, mensaje, tipo)

VALUES (1, 'Stock Crítico', CONCAT('Producto ', producto\_nombre, ' con stock bajo'), 'warning');

END IF;

**KPIs de Inventario**

**Eficiencia:**

* **Rotación por producto** (veces que se vende el stock promedio)
* **Días de inventario** (cuántos días dura el stock actual)
* **Porcentaje de merma** (pérdidas vs. ventas)

**Costos:**

* **Costo promedio por producto**
* **Valor total del inventario**
* **Pérdidas por vencimiento**

**Operativa:**

* **Frecuencia de movimientos** por usuario
* **Productos más/menos dinámicos**
* **Efectividad de compras/producción**

**Beneficios del Sistema**

1. **Auditoría completa** - Saber exactamente qué pasó con cada unidad
2. **Control de pérdidas** - Identificar patrones de merma
3. **Optimización de compras** - Basada en rotación real
4. **Cumplimiento normativo** - Trazabilidad requerida por ley
5. **Análisis de rentabilidad** - Costos reales por producto

**⚙️ TABLA: configuraciones**

**Propósito**

**Almacenar todas las configuraciones del sistema de forma dinámica, evitando valores hardcodeados y permitiendo cambios sin modificar código.**

**Estructura de Configuración**

* **clave: Identificador único de la configuración**
* **valor: El valor actual (siempre como texto)**
* **tipo: Tipo de dato ('string', 'number', 'boolean', 'json')**
* **descripcion: Explicación de qué hace esta configuración**
* **categoria: Agrupación lógica**
* **editable: Si puede modificarse desde admin**

**Categorías de Configuración**

**empresa - Información de la Empresa**

**nombre\_empresa = 'Arepa la Llanerita'**

**nit\_empresa = '900123456-1'**

**direccion\_empresa = 'Calle 123 #45-67, Bogotá'**

**telefono\_empresa = '3001234567'**

**ventas - Parámetros de Ventas**

**iva\_porcentaje = '19' -- 19% IVA Colombia**

**pedido\_minimo = '15000' -- $15,000 mínimo**

**costo\_envio\_base = '5000' -- $5,000 envío base**

**tiempo\_preparacion\_base = '30' -- 30 minutos base**

**referidos - Sistema de Referidos**

**comision\_referido\_registro = '5000' -- $5K por registro**

**comision\_referido\_primera\_compra = '10' -- 10% primera compra**

**comisiones - Porcentajes de Comisión**

**comision\_venta\_directa = '15' -- 15% ventas propias**

**comision\_lider\_equipo = '5' -- 5% ventas del equipo**

**metas - Metas de Vendedores**

**meta\_vendedor\_bronce = '500000' -- $500K nivel bronce**

**meta\_vendedor\_plata = '1000000' -- $1M nivel plata**

**meta\_vendedor\_oro = '2000000' -- $2M nivel oro**

**meta\_lider = '5000000' -- $5M líder**

**sistema - Configuraciones Técnicas**

**moneda = 'COP'**

**formato\_pedido = 'ARE-{YYYY}-{NNNN}'**

**stock\_minimo\_alerta = '5'**

**Cómo se Usa en el Código**

**Función Helper:**

**-- Función para obtener configuración**

**SELECT ObtenerConfiguracion('iva\_porcentaje'); -- Retorna '19'**

**En Laravel (Ejemplo):**

**// Modelo Configuration**

**public static function get($key, $default = null) {**

**$config = self::where('clave', $key)->first();**

**return $config ? $config->valor : $default;**

**}**

**// Uso en el código**

**$iva = Configuration::get('iva\_porcentaje', 19);**

**$pedidoMinimo = Configuration::get('pedido\_minimo', 15000);**

**Tipos de Datos**

**string - Texto Simple**

**nombre\_empresa = 'Arepa la Llanerita'**

**moneda = 'COP'**

**number - Números (se convierten al usar)**

**iva\_porcentaje = '19' -- Se usa como 19**

**pedido\_minimo = '15000' -- Se usa como 15000**

**boolean - Verdadero/Falso**

**sistema\_mantenimiento = 'true' -- Se convierte a boolean**

**notificaciones\_email = 'false'**

**json - Objetos Complejos**

**horario\_operacion = '{**

**"lunes": {"inicio": "08:00", "fin": "18:00", "activo": true},**

**"domingo": {"inicio": "09:00", "fin": "15:00", "activo": false}**

**}'**

**Configuraciones Críticas del Sistema**

**Comisiones Automáticas:**

**-- Estos valores se usan en cálculos automáticos**

**SELECT valor FROM configuraciones WHERE clave = 'comision\_venta\_directa';**

**-- Se usa en: monto\_comision = total \* (porcentaje / 100)**

**Validaciones de Pedido:**

**-- Validar pedido mínimo**

**IF total\_pedido < ObtenerConfiguracion('pedido\_minimo') THEN**

**SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Pedido no cumple mínimo requerido';**

**END IF;**

**Cálculo de Impuestos:**

**-- IVA dinámico**

**SET impuestos = subtotal \* (ObtenerConfiguracion('iva\_porcentaje') / 100);**

**Ejemplos de Uso Práctico**

**Panel de Administración:**

**// Mostrar configuraciones editables**

**$configs = Configuration::where('editable', true)**

**->orderBy('categoria', 'clave')**

**->get()**

**->groupBy('categoria');**

**foreach($configs as $categoria => $items) {**

**echo "<h3>$categoria</h3>";**

**foreach($items as $config) {**

**echo "<input name='{$config->clave}' value='{$config->valor}' />";**

**}**

**}**

**Cálculo de Comisiones:**

**public function calcularComision($tipoComision, $montoBase) {**

**$porcentaje = Configuration::get("comision\_{$tipoComision}", 0);**

**return $montoBase \* ($porcentaje / 100);**

**}**

**Validación de Metas:**

**public function evaluarNivel($ventasMes) {**

**$bronce = Configuration::get('meta\_vendedor\_bronce');**

**$plata = Configuration::get('meta\_vendedor\_plata');**

**$oro = Configuration::get('meta\_vendedor\_oro');**

**if ($ventasMes >= $oro) return 5;**

**if ($ventasMes >= $plata) return 4;**

**if ($ventasMes >= $bronce) return 3;**

**return 1;**

**}**

**Ventajas del Sistema**

**1. Flexibilidad**

* **Cambiar comisiones sin tocar código**
* **Ajustar precios en tiempo real**
* **Modificar reglas de negocio fácilmente**

**2. Mantenibilidad**

* **Centralizadas en un solo lugar**
* **Documentadas con descripción**
* **Categorizadas para organización**

**3. Auditoría**

* **Registro de cambios con timestamps**
* **Control de quién puede editar qué**
* **Historial de modificaciones**

**4. Escalabilidad**

* **Agregar nuevas configuraciones sin migración**
* **Soporte para tipos de datos complejos**
* **Configuraciones por tenant (futuro)**

**Configuraciones Específicas para Arepas**

**Preparación:**

**tiempo\_arepa\_sencilla = '15' -- 15 minutos**

**tiempo\_arepa\_rellena = '25' -- 25 minutos**

**temperatura\_plancha = '180' -- 180°C**

**Inventario:**

**vida\_util\_arepa = '2' -- 2 días**

**stock\_critico\_arepas = '10' -- 10 unidades**

**alerta\_vencimiento = '24' -- 24 horas antes**

**Promociones:**

**descuento\_combo = '15' -- 15% en combos**

**precio\_delivery\_nocturno = '7000' -- $7K después 8pm**

**promocion\_activa = 'true' -- Promociones habilitadas**

**Mejores Prácticas**

**Naming Convention:**

* **categoria\_concepto\_especificacion**
* **Ejemplos: comision\_venta\_directa, meta\_vendedor\_oro, tiempo\_preparacion\_base**

**Valores por Defecto:**

* **Siempre incluir valores sensatos por defecto**
* **Documentar el impacto de cada configuración**
* **Validar rangos (ej: porcentajes entre 0-100)**

**Seguridad:**

* **Configuraciones críticas solo editables por admin**
* **Validar tipos de datos al guardar**
* **Auditar cambios importantes**

**Tablas Relacionadas**

**auditoria - Registro de Cambios:**

**-- Al cambiar configuración crítica**

**INSERT INTO auditoria (tabla, registro\_id, accion, datos\_anteriores, datos\_nuevos)**

**VALUES ('configuraciones', config\_id, 'actualizar', old\_values, new\_values);**

**notificaciones - Alertas de Cambios:**

**-- Notificar cambios críticos**

**IF configuracion\_clave IN ('comision\_venta\_directa', 'iva\_porcentaje') THEN**

**INSERT INTO notificaciones (user\_id, titulo, mensaje)**

**VALUES (1, 'Configuración Crítica Modificada',**

**CONCAT('Se modificó: ', configuracion\_clave));**

**END IF;**

**🔔 TABLA: notificaciones**

**Propósito**

Sistema de comunicación interna para alertar a usuarios sobre eventos importantes: nuevos pedidos, stock bajo, comisiones, etc.

**Estructura Principal**

* **id**: Identificador único
* **user\_id**: Usuario que recibe la notificación
* **titulo**: Asunto de la notificación
* **mensaje**: Contenido descriptivo
* **tipo**: Categoría visual ('info', 'success', 'warning', 'error')
* **leida**: Si el usuario ya la vio
* **url**: Enlace opcional para acción
* **metadata**: JSON con información adicional

**Tipos de Notificación**

**info (Azul) - Información General**

* Nuevos pedidos asignados
* Actualizaciones del sistema
* Recordatorios generales

**success (Verde) - Eventos Positivos**

* Comisiones aprobadas
* Metas cumplidas
* Pedidos entregados exitosamente

**warning (Amarillo) - Alertas Importantes**

* Stock bajo o crítico
* Pedidos retrasados
* Pagos pendientes

**error (Rojo) - Problemas Urgentes**

* Errores en pedidos
* Fallos del sistema
* Cancelaciones

**Ejemplos de Notificaciones**

**Stock Crítico:**

INSERT INTO notificaciones VALUES (

NULL,

1, -- Admin

'Alerta de Stock Bajo',

'El producto "Arepa de Queso" tiene stock crítico: 3 unidades',

'warning',

FALSE,

'/admin/productos/2', -- Link al producto

'{"producto\_id": 2, "stock\_actual": 3, "stock\_minimo": 8}',

NOW(),

NOW()

);

**Nuevo Pedido para Vendedor:**

INSERT INTO notificaciones VALUES (

NULL,

3, -- Vendedor María

'Nuevo Pedido Asignado',

'Se te ha asignado el pedido ARE-2024-0002 por valor de $28,585',

'info',

FALSE,

'/vendedor/pedidos/2', -- Ver pedido

'{"pedido\_id": 2, "total": 28585, "cliente": "Ana María"}',

NOW(),

NOW()

);

**Comisión Aprobada:**

INSERT INTO notificaciones VALUES (

NULL,

3, -- Vendedor

'Comisión Aprobada',

'Tu comisión de $3,606 por el pedido ARE-2024-0001 ha sido aprobada',

'success',

FALSE,

'/vendedor/comisiones', -- Ver comisiones

'{"comision\_id": 1, "monto": 3606, "tipo": "venta\_directa"}',

NOW(),

NOW()

);

**Nuevo Referido:**

INSERT INTO notificaciones VALUES (

NULL,

2, -- Juan (referidor)

'Nuevo Referido',

'Ana María se registró usando tu código de referido LIDER001. ¡Felicitaciones!',

'success',

FALSE,

'/referidos', -- Ver red de referidos

'{"referido\_id": 5, "codigo\_usado": "LIDER001", "comision\_generada": 5000}',

NOW(),

NOW()

);

**Metadata JSON - Información Adicional**

**Para Stock:**

{

"producto\_id": 2,

"producto\_codigo": "ARE-002",

"stock\_actual": 3,

"stock\_minimo": 8,

"categoria": "Arepas Tradicionales"

}

**Para Pedidos:**

{

"pedido\_id": 2,

"numero\_pedido": "ARE-2024-0002",

"cliente\_id": 5,

"cliente\_nombre": "Ana María",

"total": 28585,

"zona": "Centro"

}

**Para Comisiones:**

{

"comision\_id": 1,

"monto": 3606,

"tipo": "venta\_directa",

"pedido\_relacionado": "ARE-2024-0001",

"porcentaje\_aplicado": 15

}

**Consultas Importantes**

**Notificaciones No Leídas por Usuario:**

SELECT id, titulo, mensaje, tipo, created\_at

FROM notificaciones

WHERE user\_id = 3

AND leida = FALSE

ORDER BY created\_at DESC;

**Marcar como Leída:**

UPDATE notificaciones

SET leida = TRUE, updated\_at = NOW()

WHERE id = 1 AND user\_id = 3;

**Dashboard de Alertas (Admin):**

SELECT

tipo,

COUNT(\*) as total,

COUNT(CASE WHEN leida = FALSE THEN 1 END) as no\_leidas

FROM notificaciones

WHERE created\_at >= DATE\_SUB(NOW(), INTERVAL 24 HOUR)

GROUP BY tipo;

**Notificaciones por Categoría:**

SELECT

CASE

WHEN titulo LIKE '%Stock%' THEN 'Inventario'

WHEN titulo LIKE '%Pedido%' THEN 'Ventas'

WHEN titulo LIKE '%Comisión%' THEN 'Finanzas'

WHEN titulo LIKE '%Referido%' THEN 'Red'

ELSE 'General'

END as categoria,

COUNT(\*) as total

FROM notificaciones

WHERE created\_at >= DATE\_SUB(NOW(), INTERVAL 7 DAY)

GROUP BY categoria;

**Automatización de Notificaciones**

**Trigger en Stock:**

-- Al actualizar productos, verificar stock crítico

AFTER UPDATE ON productos

FOR EACH ROW

WHEN NEW.stock <= NEW.stock\_minimo AND OLD.stock > OLD.stock\_minimo

BEGIN

INSERT INTO notificaciones (user\_id, titulo, mensaje, tipo, metadata)

VALUES (

1, -- Admin

'Alerta de Stock Bajo',

CONCAT('El producto "', NEW.nombre, '" tiene stock crítico: ', NEW.stock, ' unidades'),

'warning',

JSON\_OBJECT('producto\_id', NEW.id, 'stock\_actual', NEW.stock, 'stock\_minimo', NEW.stock\_minimo)

);

END;

**Evento Automático:**

-- Limpieza automática de notificaciones antiguas ya leídas

CREATE EVENT limpiar\_notificaciones\_antiguas

ON SCHEDULE EVERY 1 DAY

DO

DELETE FROM notificaciones

WHERE created\_at < DATE\_SUB(NOW(), INTERVAL 30 DAY)

AND leida = TRUE;

**Sistema de Prioridades**

**Notificaciones Críticas (Inmediatas):**

* Stock agotado
* Errores en pedidos
* Problemas de pago

**Notificaciones Importantes (Diarias):**

* Stock bajo
* Nuevos pedidos
* Comisiones pendientes

**Notificaciones Informativas (Semanales):**

* Resúmenes de ventas
* Actualizaciones del sistema
* Tips y consejos

**Integración con Frontend**

**Badge de Notificaciones:**

// Contar no leídas

$notificacionesNoLeidas = Notificacion::where('user\_id', auth()->id())

->where('leida', false)

->count();

**Modal de Notificaciones:**

// Últimas 10 notificaciones

$notificaciones = Notificacion::where('user\_id', auth()->id())

->orderBy('created\_at', 'desc')

->limit(10)

->get();

**Notificación en Tiempo Real (Livewire):**

// Listener para nuevas notificaciones

protected $listeners = ['nuevaNotificacion' => 'cargarNotificaciones'];

public function cargarNotificaciones() {

$this->notificaciones = $this->obtenerNotificaciones();

$this->emit('notificacionRecibida'); // Sonido, animación, etc.

}

**Expansiones Futuras**

**Push Notifications:**

* Integración con Firebase
* Notificaciones móviles
* Configuración por usuario

**Email Notifications:**

* Resúmenes diarios por email
* Alertas críticas por correo
* Configuración de preferencias

**Whatsapp Integration:**

* Alertas por WhatsApp
* Estados de pedidos
* Confirmaciones automáticas

**KPIs de Notificaciones**

**Efectividad:**

* Tasa de apertura (% de notificaciones leídas)
* Tiempo promedio hasta leer
* Acciones tomadas después de notificación

**Volumen:**

* Notificaciones por usuario por día
* Distribución por tipo
* Picos de actividad

**Optimización:**

* Reducir notificaciones spam
* Mejorar relevancia
* Personalización por rol

**🎫 TABLA: cupones**

**Propósito**

Sistema de descuentos y promociones para incentivar ventas y fidelizar clientes.

**Estructura Principal**

* **codigo**: Código único del cupón (BIENVENIDO10, PRIMERACOMPRA)
* **nombre**: Nombre descriptivo interno
* **descripcion**: Descripción para el cliente
* **tipo**: 'porcentaje' o 'monto\_fijo'
* **valor**: Cantidad del descuento
* **monto\_minimo**: Valor mínimo del pedido para aplicar
* **usos\_maximos**: Límite de usos (NULL = ilimitado)
* **usos\_realizados**: Contador de veces usado
* **fecha\_inicio/fecha\_fin**: Período de validez
* **activo**: Para habilitar/deshabilitar

**Tipos de Cupones**

**Porcentaje:**

-- 10% de descuento

codigo = 'BIENVENIDO10'

tipo = 'porcentaje'

valor = 10.00 -- 10%

monto\_minimo = 20000 -- En pedidos > $20K

**Monto Fijo:**

-- $5,000 de descuento

codigo = 'PRIMERACOMPRA'

tipo = 'monto\_fijo'

valor = 5000.00 -- $5,000

monto\_minimo = 25000 -- En pedidos > $25K

**Segmentación Avanzada**

* **categorias\_aplicables**: JSON con IDs de categorías válidas
* **productos\_aplicables**: JSON con IDs de productos específicos

**Ejemplo:**

// Solo para arepas rellenas

"categorias\_aplicables": [2]

// Solo para productos específicos

"productos\_aplicables": [1, 3, 5]

**Validación de Cupones**

**Proceso de Validación:**

1. **Existe el código** y está activo
2. **Fecha vigente** (entre fecha\_inicio y fecha\_fin)
3. **Usos disponibles** (usos\_realizados < usos\_maximos)
4. **Monto mínimo** cumplido
5. **Productos aplicables** válidos

**Query de Validación:**

SELECT \* FROM cupones

WHERE codigo = 'BIENVENIDO10'

AND activo = TRUE

AND CURDATE() BETWEEN fecha\_inicio AND fecha\_fin

AND (usos\_maximos IS NULL OR usos\_realizados < usos\_maximos)

AND 25000 >= monto\_minimo; -- Monto del pedido

**🔍 TABLA: auditoria**

**Propósito**

Registro completo de todas las acciones críticas del sistema para seguridad, cumplimiento y debugging.

**Información Registrada**

* **user\_id**: Usuario que realizó la acción
* **tabla**: Tabla afectada
* **registro\_id**: ID del registro modificado
* **accion**: 'crear', 'actualizar', 'eliminar'
* **datos\_anteriores**: JSON con valores antes del cambio
* **datos\_nuevos**: JSON con valores después del cambio
* **ip**: Dirección IP del usuario
* **user\_agent**: Navegador/dispositivo usado

**Eventos Auditados**

**Cambios en Usuarios:**

-- Al cambiar rol de usuario

INSERT INTO auditoria VALUES (

NULL, 1, 'users', 3, 'actualizar',

'{"rol": "cliente", "activo": true}', -- Antes

'{"rol": "vendedor", "activo": true}', -- Después

'192.168.1.100', 'Mozilla/5.0...', NOW()

);

**Modificaciones de Precios:**

-- Al cambiar precio de producto

INSERT INTO auditoria VALUES (

NULL, 1, 'productos', 2, 'actualizar',

'{"precio": 5000, "precio\_mayorista": 4000}', -- Antes

'{"precio": 5500, "precio\_mayorista": 4400}', -- Después

'192.168.1.100', 'Mozilla/5.0...', NOW()

);

**Cancelación de Pedidos:**

-- Al cancelar pedido

INSERT INTO auditoria VALUES (

NULL, 3, 'pedidos', 5, 'actualizar',

'{"estado": "confirmado", "total": 24000}', -- Antes

'{"estado": "cancelado", "total": 24000}', -- Después

'192.168.1.50', 'Mobile App...', NOW()

);

**Consultas de Auditoría**

**Actividad de Usuario:**

SELECT

tabla,

accion,

created\_at,

SUBSTRING(datos\_nuevos, 1, 100) as cambios

FROM auditoria

WHERE user\_id = 1

ORDER BY created\_at DESC

LIMIT 20;

**Cambios en Producto Específico:**

SELECT

u.name as usuario,

a.accion,

a.datos\_anteriores,

a.datos\_nuevos,

a.created\_at

FROM auditoria a

LEFT JOIN users u ON a.user\_id = u.id

WHERE a.tabla = 'productos'

AND a.registro\_id = 2

ORDER BY a.created\_at DESC;

**Actividad Sospechosa:**

-- Muchos cambios desde la misma IP

SELECT

ip,

COUNT(\*) as total\_acciones,

COUNT(DISTINCT user\_id) as usuarios\_diferentes

FROM auditoria

WHERE created\_at >= DATE\_SUB(NOW(), INTERVAL 1 HOUR)

GROUP BY ip

HAVING total\_acciones > 50

ORDER BY total\_acciones DESC;

**Triggers Automáticos**

**Usuario Modificado:**

AFTER UPDATE ON users

FOR EACH ROW

BEGIN

INSERT INTO auditoria (user\_id, tabla, registro\_id, accion, datos\_anteriores, datos\_nuevos, ip)

VALUES (NEW.id, 'users', NEW.id, 'actualizar',

JSON\_OBJECT('rol', OLD.rol, 'activo', OLD.activo),

JSON\_OBJECT('rol', NEW.rol, 'activo', NEW.activo),

@user\_ip);

END;

**Producto Eliminado:**

AFTER DELETE ON productos

FOR EACH ROW

BEGIN

INSERT INTO auditoria (user\_id, tabla, registro\_id, accion, datos\_anteriores, ip)

VALUES (@current\_user\_id, 'productos', OLD.id, 'eliminar',

JSON\_OBJECT('codigo', OLD.codigo, 'nombre', OLD.nombre, 'precio', OLD.precio),

@user\_ip);

END;

**📊 RESUMEN DEL SISTEMA COMPLETO**

**Flujo de Datos Principal**

**1. Registro de Usuario**

users → referidos → comisiones → notificaciones

**2. Creación de Pedido**

users → pedidos → detalle\_pedidos → movimientos\_inventario → comisiones

**3. Gestión de Inventario**

productos → movimientos\_inventario → notificaciones (stock bajo)

**4. Sistema de Referidos**

users (referidor) → referidos → comisiones → notificaciones

**Métricas Clave del Sistema**

**Performance:**

* 13 tablas principales interconectadas
* 15+ índices optimizados para consultas frecuentes
* 4 vistas materializadas para reportes
* Triggers automáticos para consistencia

**Escalabilidad:**

* Estructura preparada para multi-tenant
* Sistema de configuraciones dinámico
* Auditoría completa para compliance
* Diseño modular para nuevas funcionalidades

**Seguridad:**

* Auditoría de todas las acciones críticas
* Validaciones de integridad referencial
* Control de acceso por roles
* Trazabilidad completa de transacciones

1. **users** - Sistema de roles y referidos multinivel
2. **categorias** - Organización escalable de productos
3. **productos** - Catálogo completo con inventario inteligente
4. **zonas\_entrega** - Logística optimizada por área
5. **pedidos** - Ciclo completo de órdenes
6. **detalle\_pedidos** - Productos específicos con historial de precios
7. **referidos** - Red comercial con seguimiento detallado
8. **comisiones** - Cálculo automático de ganancias
9. **movimientos\_inventario** - Trazabilidad total de stock
10. **configuraciones** - Sistema dinámico sin hardcodeo
11. **notificaciones** - Comunicación interna inteligente
12. **cupones** - Sistema de descuentos avanzado
13. **auditoria** - Registro completo para seguridad

**🔄 Flujos Principales que Dominas:**

**🤝 Sistema de Referidos:**

Vendedor obtiene código → Cliente se registra → Se crea relación →

Primera compra → Se activan comisiones → Seguimiento continuo

**🛒 Proceso de Ventas:**

Cliente hace pedido → Se asigna vendedor → Se descuenta inventario →

Se calculan comisiones → Se notifica a stakeholders → Se entrega

**📦 Control de Inventario:**

Entrada/Salida → Registro automático → Alertas de stock →

Análisis de rotación → Optimización de compras

**💡 Conceptos Clave que Manejas:**

* **Desnormalización inteligente** para performance
* **Triggers automáticos** para consistencia
* **Sistema de configuraciones** dinámico
* **Auditoría completa** para seguridad
* **Métricas en tiempo real** para decisiones