



# Proyecto final


## Desafíos

Alejandra Morales Daza

# **Contenido**

## **Proyecto**

### **Final.**



1. Descripción modelo de negocio.
2. Objetivo del proyecto.
3. Modelo entidad relación.
4. Descripción tablas.
5. Listado de Vistas.
6. Listado de Funciones.
7. Listado de Storeprocedures.
8. Listado de Triggers.



# Descripción modelo de negocio



CODERHOUSE

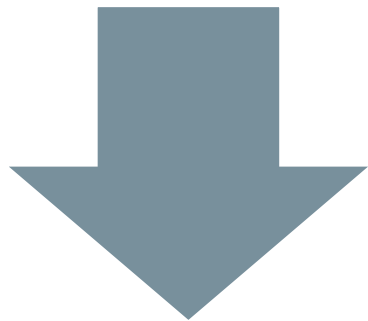
Proyecto final

# Plataforma de domicilios.

Empresa dedicada al comercio electrónico a domicilio, conecta comercios y establecimientos con usuarios acercándoles los productos que ellos desean sin la necesidad de dirigirse directamente, solo por medio de la aplicación o página web.

The logo for Rappi, featuring the word "Rappi" in a bold, orange, cursive script font.

# Modelo de negocio:



El modelo de negocio de Rappi es un **modelo de plataforma multilateral de economía colaborativa P2P (Peer to peer)**.



La plataforma actúa de intermediaria para poner en contacto a consumidores con otras empresas que ofrecen productos o servicios.



*Rappi*

# Servicios:



## **Mercado.**

- Compra de cualquier producto de mercado.



## **Restaurantes**

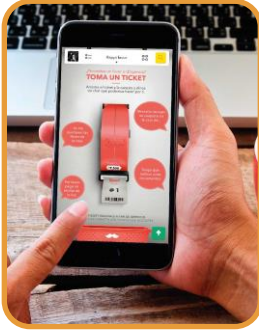
- Pedido a domicilio en restaurantes de la zona del usuario.



## **Rappicash, un cajero a tu mano**

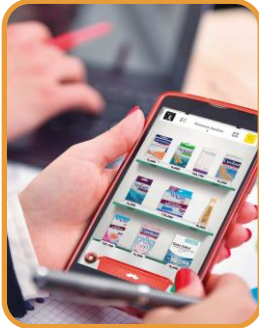
- Llevan efectivo al lugar donde requiera el usuario

# Servicios:



## Favores, antojos y deseos.

- Servicios por hora del **Rappitendero** para realizar cualquier favor que necesite el usuario.



## Farmacia y Bienestar

- Compra de medicamentos, artículos de aseo personal, perfumería, entre otros.





# Objetivo del proyecto



CODERHOUSE

Proyecto final



# Objetivo del proyecto.



Crear una base de datos relacional en MYSQL Workbench para la empresa Rappi en el segmento de servicio de domicilios de restaurantes y establecimientos.



Elaborar el modelo de entidad relación que enuncie las entidades y atributos necesarios para la creación de la base de datos.



Desarrollar las queries para creación de schema, tablas , inserción de datos, vistas, funciones, SP y triggers.



# Modelo Entidad-Relación

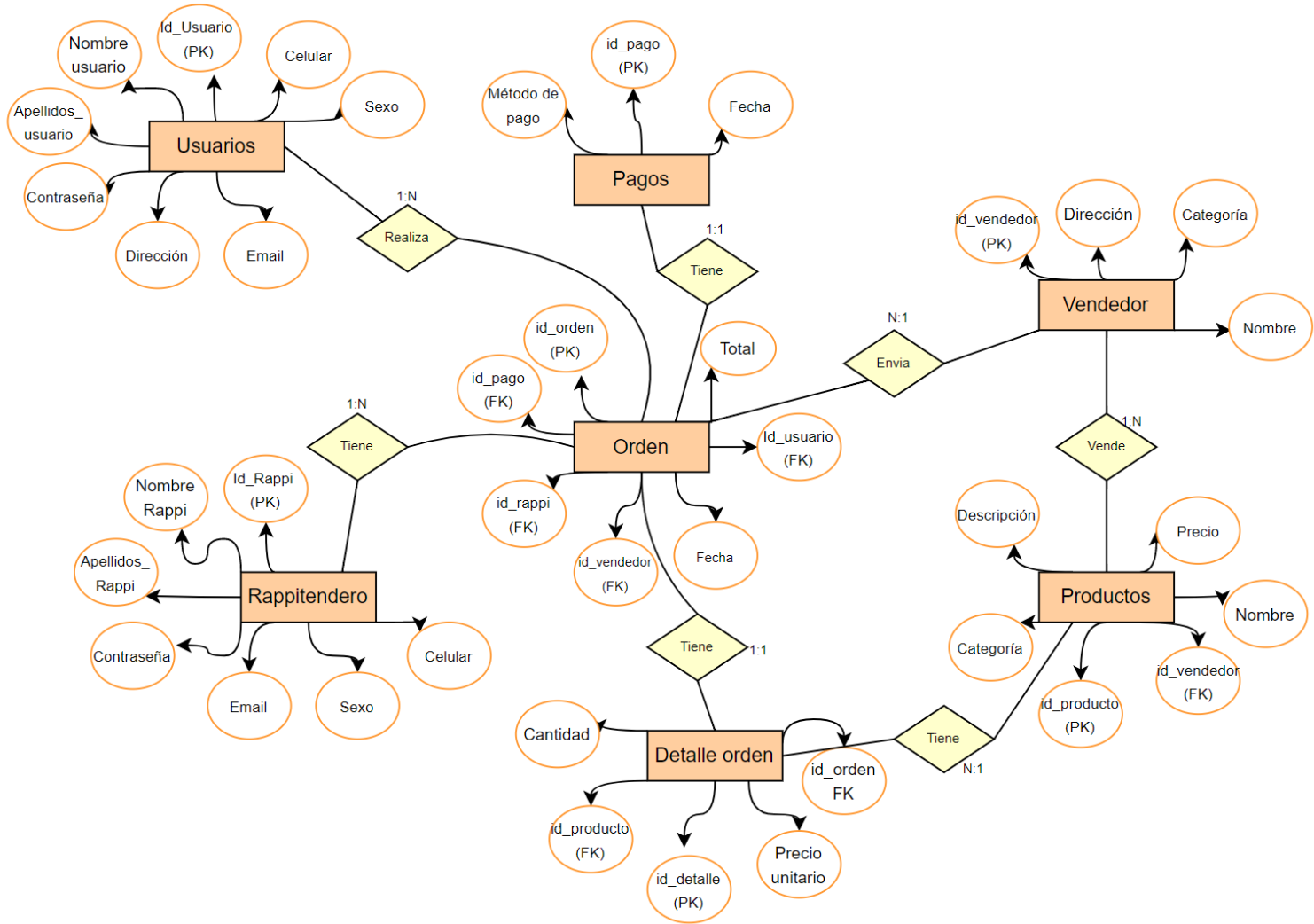


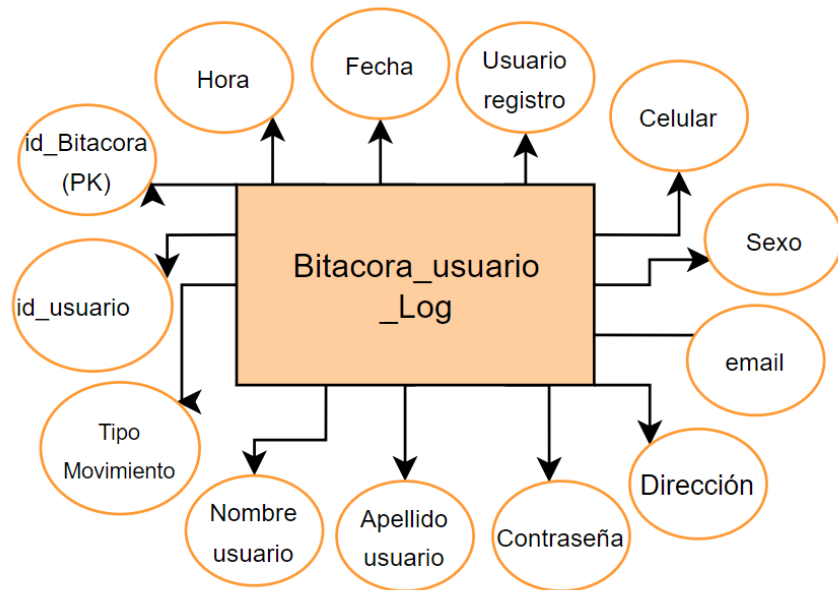
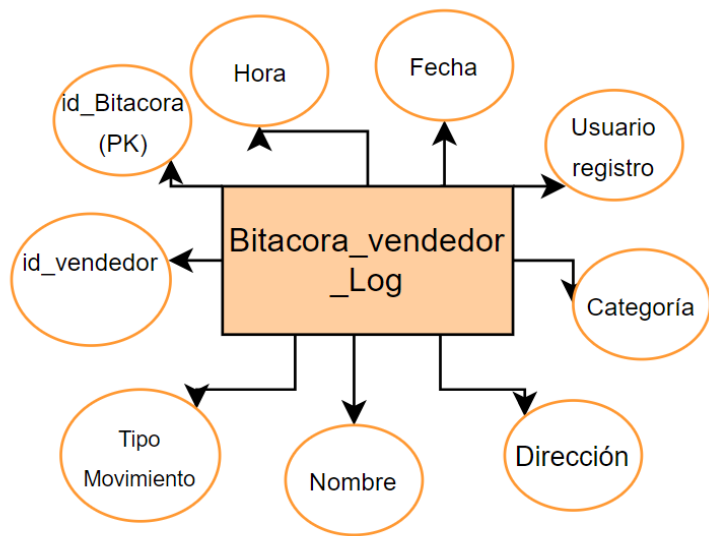
CODERHOUSE

Proyecto final

# Modelo Entidad Relación

# Rappi

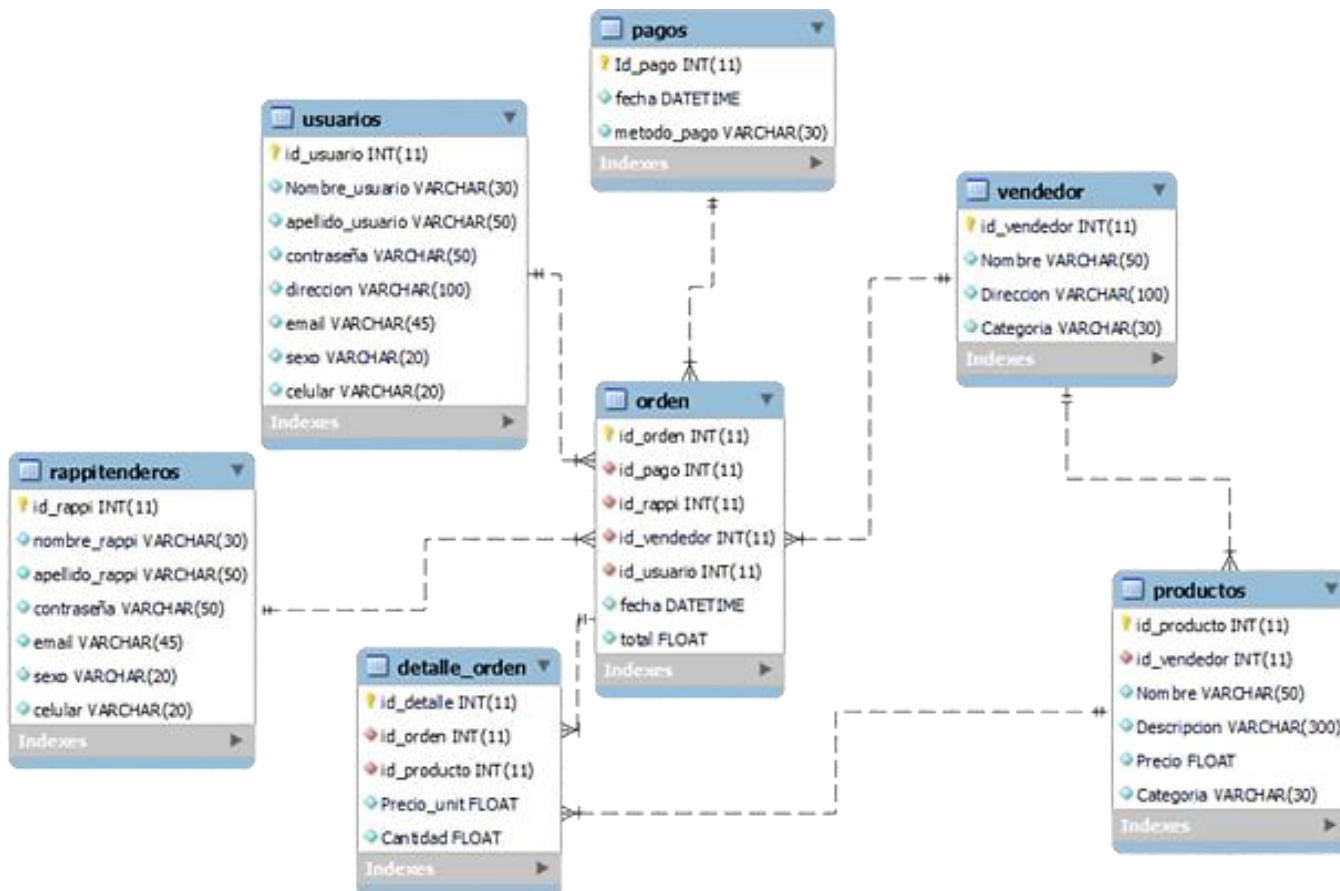




# Modelo Entidad Relación de MYSQL



A continuación se presenta el modelo de entidad relación generado en MYSQL WORKBENCH



# Modelo Entidad Relación de MYSQL



A continuación se presenta el modelo de entidad relación generado en MYSQL WORKBENCH

bitacora_vendedor_log	
id_bitacora	INT(11)
id_vendedor	INT(11)
Tipo_movimiento	VARCHAR(30)
Nombre	VARCHAR(50)
Direccion	VARCHAR(100)
Categoria	VARCHAR(30)
Usuario_registro	VARCHAR(50)
Fecha	DATE
Hora	TIME
Indexes	

bitacora_usuario_log	
id_bitacora	INT(11)
id_usuario	INT(11)
Tipo_movimiento	VARCHAR(30)
nombre_usuario	VARCHAR(50)
apellido_usuario	VARCHAR(50)
contraseña	VARCHAR(50)
direccion	VARCHAR(100)
email	VARCHAR(45)
sexo	VARCHAR(20)
Celular	VARCHAR(20)
Usuario_registro	VARCHAR(50)
Fecha	DATE
Hora	TIME
Indexes	



# Descripción tablas



CODERHOUSE

Proyecto final



# Descripción tabla Vendedor y pagos.



**Tabla Vendedor**

Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_vendedor	INT	NOT NULL AUTO INCREMENT	Almacena datos de los proveedores de los productos en venta en la plataforma.
	Nombre	VARCHAR(50)	NOT NULL	
	Dirección	VARCHAR(100)	NOT NULL	
	Categoría	VARCHAR(30)	NOT NULL	

**Tabla Pagos**

Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_pago	INT	AUTO INCREMENT	Almacena la trazabilidad de cada uno de los pagos realizados en la plataforma
	Método_pago	VARCHAR(30)	NOT NULL	
	Fecha	Datetime	NOT NULL	

**Nota:** Se realizó comentario de cada columna de las tablas en MYSQL Workbench por motivos prácticos en esta presentación se da una descripción general de cada tabla.

# Descripción tabla usuarios.



Tabla Usuarios				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_usuario	INT	AUTO INCREMENT	Almacenamiento de datos personales del usuario que hace uso de la plataforma.
	Nombre_usuario	VARCHAR(30)	NOT NULL	
	Apellido_usuario	VARCHAR(50)	NOT NULL	
	Contraseña	VARCHAR(50)	NOT NULL	
	Dirección	VARCHAR(100)	NOT NULL	
	Email	VARCHAR(45)	NOT NULL	
	Sexo	VARCHAR (20)	NOT NULL	
	Celular	VARCHAR(20)	NOT NULL	

# Descripción tabla Rappitenderos.



Tabla Rappitendero				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_Rappi	INT	AUTO INCREMENT	Almacenamiento de datos personales del rappitendero que presta servicios en la plataforma.
	Nombre_Rappi	VARCHAR(30)	NOT NULL	
	Apellido_Rappi	VARCHAR(50)	NOT NULL	
	Contraseña	VARCHAR(50)	NOT NULL	
	Email	VARCHAR(45)	NOT NULL	
	Sexo	VARCHAR (20)	NOT NULL	
	Celular	VARCHAR(20)	NOT NULL	

# Descripción tabla productos.



Tabla Productos				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_producto	INT	AUTO INCREMENT	Almacena los datos correspondientes a los productos dispuestos en la plataforma para su compra y distribución.
FK	Id_vendedor	INT	NOT NULL	
	Nombre	VARCHAR(50)	NOT NULL	
	Descripción	VARCHAR (300)	NOT NULL	
	Precio	FLOAT	NOT NULL	
	Categoría	VARCHAR(30)	NOT NULL	

# Descripción tabla orden.



Tabla Orden				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_orden	INT	AUTO INCREMENT	Almacenamiento de datos correspondientes a las órdenes de pedido creadas a través de la plataforma.
FK	Id_pago	INT	NOT NULL	
FK	Id_rappi	INT	NOT NULL	
FK	Id_usuario	INT	NOT NULL	
FK	Id_vendedor	INT	NOT NULL	
	Fecha	DATETIME	NOT NULL	
	Total	FLOAT	DEFAULT 0	

# Descripción tabla detalle orden.



Detalle Orden				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_Detalle	INT	AUTO INCREMENT NOT NULL	Especifica el detalle de los productos que conlleva la orden
FK	Id_orden	INT	NOT NULL	
FK	Id_producto	INT	NOT NULL	
	Precio_unit	FLOAT	NOT NULL	
	Cantidad	FLOAT	NOT NULL	



# Descripción tabla Bitácora Vendedor

Bitacora_vendedor_log				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_Bitacora	INT	AUTO INCREMENT NOT NULL	En esta tabla se guardaran los log de los triggers asociados a las transacciones en la tabla vendedor.
	Id_vendedor	INT	NOT NULL	
	Tipo_movimiento	VARCHAR(30)		
	Nombre	VARCHAR(50)	NOT NULL	
	Dirección	VARCHAR(100)	NOT NULL	
	Categoría	VARCHAR(30)	NOT NULL	
	Usuario_registro	VARCHAR(50)		
	Fecha	DATE		
	Hora	TIME		





# Descripción tabla Bitácora Usuario

Bitacora_usuario_log				
Tipo de clave	Nombre del campo	Tipo de campo	Características	Descripción
PK	Id_Bitacora	INT	AUTO INCREMENT NOT NULL	En esta tabla se guardaran los log de los triggers asociados a las transacciones en la tabla usuarios.
	Id_usuario	INT	NOT NULL	
	Tipo_movimiento	VARCHAR(30)		
	Nombre_usuario	VARCHAR(50)		
	Apellido_usuario	VARCHAR(50)		
	Contraseña	VARCHAR(50)		
	Dirección	VARCHAR(100)		
	email	VARCHAR(45)		
	Sexo	VARCHAR(20)		
	Celular	VARCHAR(20)		
	Usuario_registro	VARCHAR(50)		
	Fecha	DATE		
	Hora	TIME		



# Listado de vistas



CODERHOUSE

Proyecto final



# Descripción Vistas

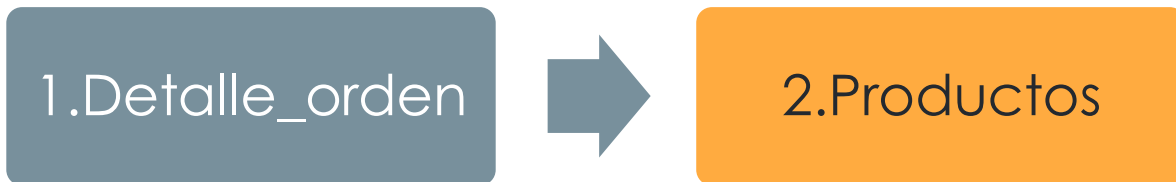


## 1. Creación vista de top de productos más vendidos

**Nombre: Vw\_productos\_vendidos**

Arroja un listado de los productos de mayor a menor según las unidades vendidas registradas en la plataforma.

**Tablas que relacionan:**





## Descripción Vistas



### 2. Creación vista top de rappitenderos con mayores ordenes

**Nombre: Vw\_top\_rappitenderos**

Esta vista genera el top de los rappitenderos con mayor numero de domicilios realizados, a través de un INNER JOIN.

**Tablas que relacionan:**

1.Rappitenderos



2.Orden



## Descripción Vistas



### 3. Métodos de pago más usados

**Nombre:** `Vw_top_metodo_pago`

Esta vista ordena de mayor a menor los métodos de pago que tienen mayor uso en la plataforma y el monto total efectuado, a través de la unión de la tabla de pagos y orden según el número de veces usado de forma descendiente.

**Tablas que relacionan:**



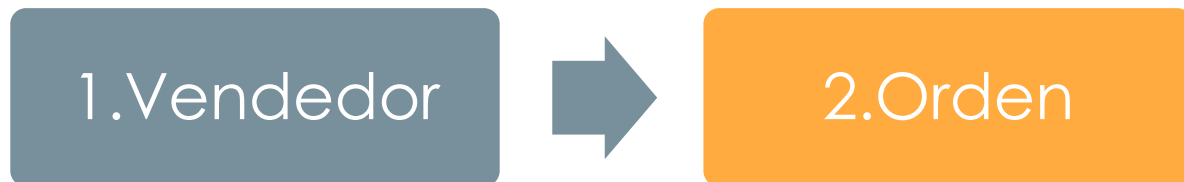
## Descripción Vistas

### 4. Top de vendedores con mayor número de ordenes

**Nombre:** `Vw_top_vendedores`

Esta vista genera un listado de los vendedores con mayor número de ordenes en la plataforma a través de la unión de la tabla de vendedor y orden.

**Tablas que relacionan:**



## Descripción Vistas

### 5. Top de Usuarios con más numero de ordenes en rappi

**Nombre:** `Vw_top_usuarios`

Esta vista genera un listado de los usuarios ordenados de mayor a menor según el numero de ordenes realizadas en la plataforma a través de la unión de la tabla de usuarios y orden.

**Tablas que relacionan:**







# Listado de funciones



CODERHOUSE

Proyecto final



# Descripción Funciones



## 1. Nombre: fnnum\_compras

Esta función calcula el numero de compras que ha realizado un usuario ingresando el id.

### Tablas que relacionan:

- 1.Usuarios
- 2.Orden

## 2. Nombre: fncat\_prod\_vendidos

Esta función ingresando el id del producto regresa la categoría a la cual pertenece dicho producto.

### Tablas que relacionan:

- 1.Productos



## Descripción Funciones



### 3. Nombre: fnund\_vendidas\_cat

Esta función ingresando la categoría a evaluar devuelve el numero de productos vendidos que pertenecen a dicha categoría.

#### Tablas que relacionan:

- 1.Productos
- 2.Detalle orden



# Listado Storedprocedures



CODERHOUSE

Proyecto final

## 1. Nombre: `sp_nueva_order`

Crea un procedimiento almacenado para ingresar datos de usuarios nuevos en la tabla de usuarios según los valores de entrada. Lo que permite agilizar la inserción de nuevos datos a la tabla.

### Tablas que relacionan:

1. Orden

## 2. Nombre: `sp_detalle_order`

Diligencia la tabla detalle de orden especificando los productos y la cantidad de cada uno de ellos, además de asociarlo con el precio establecido en la tabla productos, realiza el calculo del valor de la orden, multiplicando la cantidad por el precio y actualizándolo en el campo "total" de la tabla orden.

### Tablas que relacionan:

1. Orden  
2. Detalle orden

### 3. Nombre: **sp\_vendedores**

Crea un procedimiento almacenado para consultar todas las filas de la tabla de vendedores según un campo de ordenación ingresado ya sea de manera ascendente o descendente.

#### **Tablas que relacionan:**

1.Vendedor

### 4. Nombre: **sp\_usuario\_nuevo**

Crea un procedimiento almacenado para ingresar datos de usuarios nuevos en la tabla de usuarios según los valores de entrada. Lo que permite agilizar la inserción de nuevos datos a la tabla.

#### **Tablas que relacionan:**

1.Usuarios



# Listado Triggers



CODERHOUSE

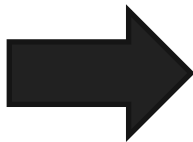
Proyecto final



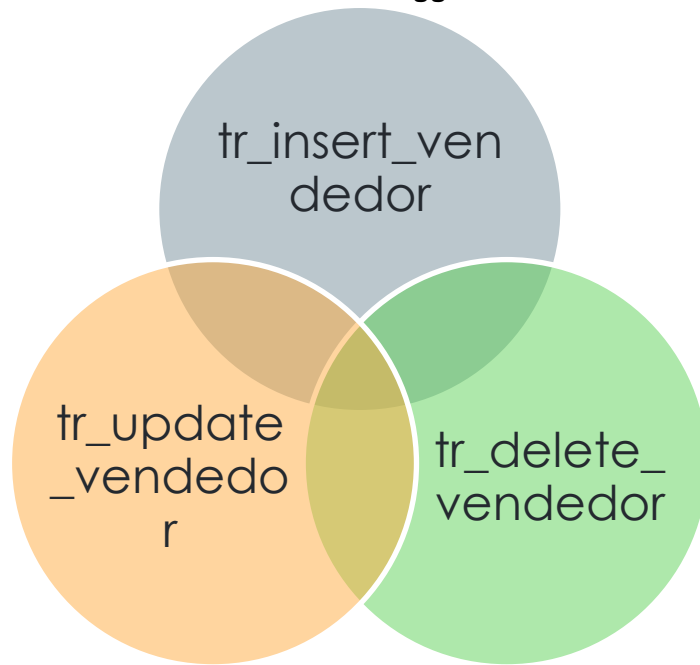
# Triggers en tabla vendedor



Se crearon 3 triggers, uno para insert, update y delete para la tabla vendedor con el fin de almacenar en una tabla denominada **Bitacora\_vendedor\_log** el tipo de movimiento realizado, el usuario, fecha y hora en la cual se registro la novedad.



Listado de Triggers.

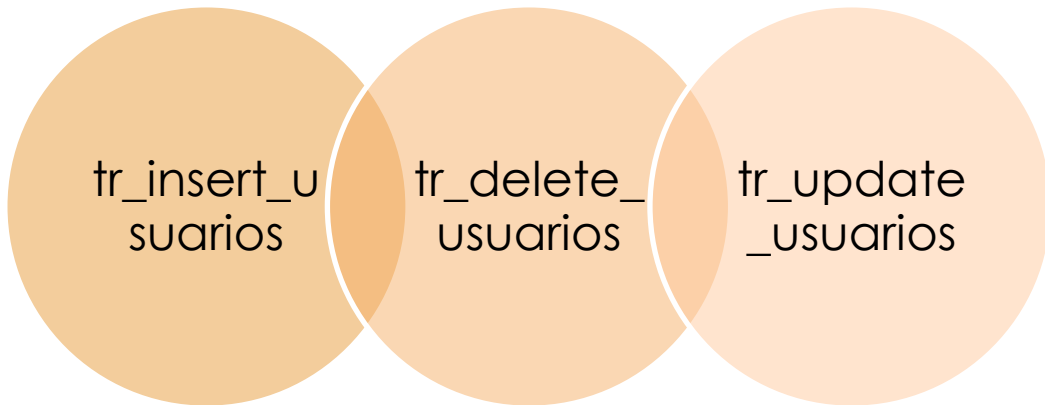


# Triggers en tabla Usuarios



Se crearon 3 triggers para la tabla usuarios la cual almacena en una tabla denominada **Bitacora\_Usuario\_log**, los movimientos realizados según corresponda.

Listado de Triggers usuarios.



**GRACIAS**

**Entrega proyecto final**

**Alejandra Morales Daza**