

# PROYECTO SQL – CODERHOUSE

## MUSIC STORE DATABASE

- Alumno: **Santiago Olais**
- Comisión: **34970**
- Tema de proyecto: **ECOMMERCE VENTA DE DISCOS (MUSICSTORE)**

## INTRODUCCIÓN

Se desarrolla un sistema de base de datos en MySQL para organizar la información de negocio de un emprendimiento, lo cual que permite tener un mejor y mayor control sobre el mismo así como tomar buenas decisiones.

## OBJETIVO

El objetivo del proyecto es organizar y obtener información de relevancia para facilitar la toma de decisiones. Se almacenará la data (ej: usuarios, compras, ventas, productos) en una base de datos y a partir de allí se obtendrá información que sirva al negocio (ej: ventas por país, clientes que más compran, productos más vendidos, costos/ventas mensuales y más)

## SITUACION PROBLEMÁTICA

Un emprendimiento comercial administra, entre otras cosas, la compra de stock, venta de productos, gestión de usuarios, gestión de empleados. Por tanto, es necesario organizar la información para así generar reportes de calidad, proyecciones e informes sobre el estado de negocio.

## MODELO DE NEGOCIO

El emprendimiento ficticio es una tienda de música online que vende discos en formato físico. Permite a los clientes tener usuarios en la plataforma y realizar pedidos o mandar mensajes.

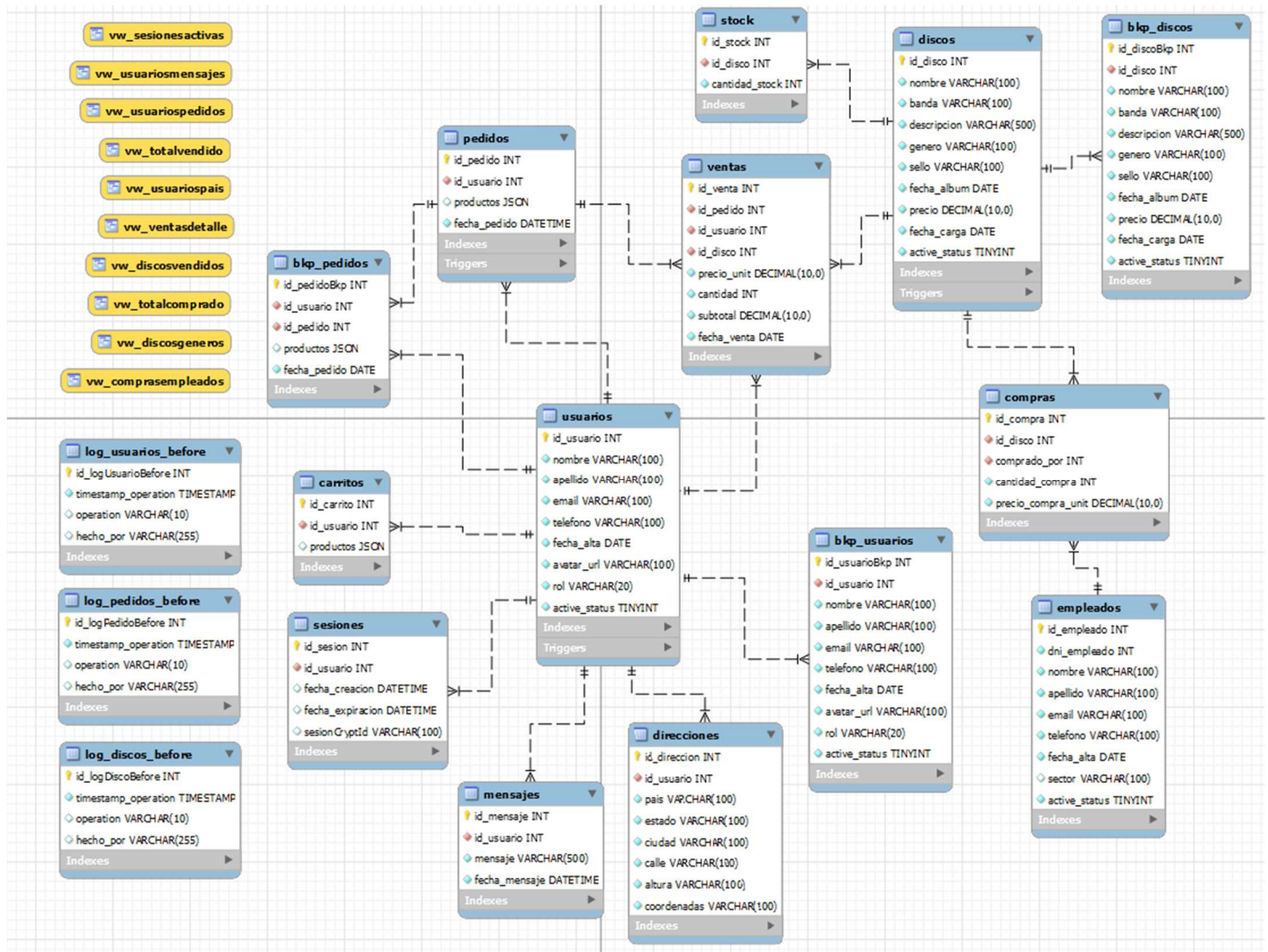
Se organiza y analiza la información con el fin de analizar tendencias y plantear un plan de negocio futuro.

## BASE DE DATOS - MUSICSTORE

- DER
- TABLAS
- DISPARADORES
- VISTAS
- FUNCIONES
- PROCEDIMIENTOS
- USUARIOS
- BACKUP
- HERRAMIENTAS

## DER (DIAGRAMA ENTIDAD-RELACIÓN)

Se observa en el DER de la Base de Datos cómo se relacionan las distintas tablas.



## DESCRIPCIÓN DE TABLAS (TABLES)

- Tabla **empleados**: quienes gestionan compras y cargan productos a la plataforma online.

### EMPLEADOS

**id\_empleado** INT NOT NULL AUTO\_INCREMENT: id único a cada empleado (PK)

**dni\_empleado** INT NOT NULL: número de dni de cada empleado

**nombre** VARCHAR (100) NOT NULL: nombre del empleado

**apellido** VARCHAR (100) NOT NULL: apellido del empleado

**email** VARCHAR (100) NOT NULL: e-mail del empleado

**telefono** VARCHAR (100) NOT NULL: nro. de teléfono del empleado

**fecha\_alta** DATE NOT NULL: fecha en que se da de alta al empleado

**sector** VARCHAR (100): sector del empleado en la empresa

**active\_status** TINYINT NOT NULL: estado del empleado (1- activo, 0- no activo)

**PRIMARY KEY (id\_empleado)**: la llave primaria es id\_empleado

- Tabla **usuarios**: quienes utilizan plataforma online, contiene información básica.

### USUARIOS

**id\_usuario** INT NOT NULL AUTO\_INCREMENT: id único a cada usuario (PK)

**nombre** VARCHAR (100) NOT NULL: nombre del usuario

**apellido** VARCHAR (100) NOT NULL: apellido del usuario

**email** VARCHAR (100) NOT NULL: e-mail del usuario

**telefono** VARCHAR (100) NOT NULL: teléfono del usuario

**fecha\_alta** DATE NOT NULL: fecha de alta el usuario

**avatar\_url** VARCHAR (100) NOT NULL: avatar (enlace a foto de perfil) del usuario

**rol** VARCHAR (20) NOT NULL: rol del usuario (user/admin)

**active\_status** TINYINT NOT NULL: estado del usuario (1- activo, 0- no activo)

**PRIMARY KEY (id\_usuario)**: la llave primaria es id\_usuario



- Tabla **direcciones**: informacion detallada de las direcciones de usuarios de la plataforma online.

#### DIRECCIONES

**id\_direccion** INT NOT NULL AUTO\_INCREMENT: id único a cada dirección (PK)

**id\_usuario** INT NOT NULL: id del usuario a quien pertenece la dirección (FK)

**pais** VARCHAR (100) NOT NULL: país de la dirección usuario

**estado** VARCHAR (100) NOT NULL: estado de la dirección usuario

**ciudad** VARCHAR (100) NOT NULL: ciudad de la dirección usuario

**calle** VARCHAR (100) NOT NULL: calle de la dirección del usuario

**altura** VARCHAR (100) NOT NULL: altura de la dirección del usuario

**coordenadas** VARCHAR (100) NOT NULL: coordenadas de la dirección del usuario

**PRIMARY KEY (id\_direccion)**: la llave primaria es id\_direccion

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario) ON DELETE CASCADE**: llave foránea id\_usuario que apunta a la llave primaria id\_usuario de la tabla usuarios

- Tabla **discos**: información detallada de los discos (albums de música) que están a la venta.

#### DISCOS

**id\_disco** INT NOT NULL AUTO\_INCREMENT: id único a cada disco (PK)

**nombre** VARCHAR (100) NOT NULL: nombre del disco

**banda** VARCHAR (100) NOT NULL: banda del disco

**descripcion** VARCHAR (500) NOT NULL: descripción del disco

**genero** VARCHAR (100) NOT NULL: género musical del disco

**sello** VARCHAR (100) NOT NULL: sello discográfico del disco

**fecha\_album** DATE NOT NULL: fecha de lanzamiento del disco

**precio** DECIMAL NOT NULL: precio de venta del disco

**fecha\_carga** DATE NOT NULL: fecha de carga del disco al sistema

**active\_status** TINYINT NOT NULL: estado del disco (1- activo, 0- no activo)

**PRIMARY KEY (id\_disco)**: la llave primaria es id\_disco



- Tabla **compras**: detalle de las compras de discos que realiza la empresa para stockear.

#### COMPRAS

**id\_compra** INT NOT NULL AUTO\_INCREMENT: id único a cada compra (PK)

**id\_disco** INT NOT NULL: id del disco (FK)

**comprado\_por** INT NOT NULL: empleado que realiza la compra (FK)

**cantidad\_compra** INT NOT NULL: cantidad de discos compradas

**precio\_compra\_unit** DECIMAL NOT NULL: precio unitario del disco

**PRIMARY KEY (id\_compra)**: la llave primaria es id\_compra

**FOREIGN KEY (id\_disco) REFERENCES discos (id\_disco)**: llave foránea id\_disco apunta a la llave primaria id\_disco de la tabla discos

**FOREIGN KEY (comprado\_por) REFERENCES empleados (id\_empleado)**: llave foránea comprado\_por apunta a la llave primaria id\_empleado de la tabla empleados

- Tabla **stock**: cantidad de productos (stock es lo disponible para venta, a diferencia de compras que es una gestion más bien comercial).

#### STOCK

**id\_stock** INT NOT NULL AUTO\_INCREMENT: id único a cada producto (disco) del stock (PK)

**id\_disco** INT NOT NULL: id del disco (FK)

**cantidad\_stock** INT NOT NULL: cantidad de stock del producto

**PRIMARY KEY (id\_stock)**: la llave primaria es id\_stock

**FOREIGN KEY (id\_disco) REFERENCES discos (id\_disco) ON DELETE CASCADE**: llave foránea id\_disco que apunta a la llave primaria id\_disco de la tabla discos

- Tabla **carritos**: selección de productos que el usuario va eligiendo dentro de la plataforma online y antes de finalizar la compra.

#### CARRITOS

**id\_carrito** INT NOT NULL AUTO\_INCREMENT: id único a cada carrito (PK)

**id\_usuario** INT NOT NULL: id del usuario del carrito (FK)

**productos** JSON: productos del carrito en formato JSON

**PRIMARY KEY (id\_carrito)**: la llave primaria es id\_carrito

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario) ON DELETE CASCADE**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios





- Tabla **pedidos**: órdenes de compra que recibe el sistema desde la plataforma online. Al confirmar una orden, la información del carrito pasa al pedido, el carrito se vacía y el pedido pasa al sector de ventas quienes gestionan la misma.

#### PEDIDOS

**id\_pedido** INT NOT NULL AUTO\_INCREMENT: id único a cada pedido (PK)

**id\_usuario** INT NOT NULL: id del usuario del pedido (FK)

**productos** JSON: productos del pedido en formato JSON

**fecha\_pedido** DATETIME NOT NULL: fecha y hora en que realizó el pedido

**PRIMARY KEY (id\_pedido)**: la llave primaria es id\_pedido

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario)**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios

- Tabla **mensajes**: mensajes enviados a través de la plataforma online, desde los usuarios hacia el administrador.

#### MENSAJES

**id\_mensaje** INT NOT NULL AUTO\_INCREMENT: id único a cada mensaje (FK)

**id\_usuario** INT NOT NULL: id del usuario del mensaje (FK)

**mensaje** VARCHAR (500) NOT NULL: mensaje enviado

**fecha\_mensaje** DATETIME NOT NULL: fecha y hora del mensaje

**PRIMARY KEY (id\_mensaje)**: la llave primaria es id\_mensaje

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario) ON DELETE CASCADE**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios

- Tabla **sesiones**: sesiones de usuario en la plataforma online (las sesiones se validan por el backend en cada login y pueden durar una cantidad de tiempo).

#### SESIONES

**id\_sesion** INT NOT NULL AUTO\_INCREMENT: id único a cada sesión de usuario (PK)

**id\_usuario** INT NOT NULL: id del usuario de la sesión (FK)

**fecha\_creacion** DATETIME: fecha y hora de creación de la sesión

**fecha\_expiracion** DATETIME: fecha y hora de expiración de la sesión

**sesionCryptId** VARCHAR(100): id de sesión encriptado para validar la misma con el backend

**PRIMARY KEY (id\_sesion)**: la llave primaria es id\_sesion

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario) ON DELETE CASCADE**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios

- Tabla **ventas**: información de los productos vendidos. Esta tabla se carga mediante disparador al realizar pedidos.

## VENTAS

**id\_venta** INT NOT NULL AUTO\_INCREMENT: id de la venta (PK)

**id\_pedido** INT NOT NULL: id del pedido generador de la venta (FK)

**id\_usuario** INT NOT NULL: id del usuario que hizo el pedido (a quién se vendió) (FK)

**id\_disco** INT NOT NULL: id del producto (disco) comprado (FK)

**precio\_unit** DECIMAL NOT NULL: precio unitario del disco comprado

**cantidad** INT NOT NULL: cantidad comprada del disco

**subtotal** DECIMAL NOT NULL: subtotal de la venta ( $\text{precio\_unit} * \text{cantidad}$ )

**fecha\_venta** DATE NOT NULL: fecha en que se realizó la venta

**PRIMARY KEY (id\_venta)**: la llave primaria es id\_venta

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario)**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios

**FOREIGN KEY (id\_disco) REFERENCES discos (id\_disco)**: llave foránea id\_disco apunta a la llave primaria id\_disco de la tabla discos

**FOREIGN KEY (id\_pedido) REFERENCES pedidos (id\_pedido)**: llave foránea id\_pedido apunta a la llave primaria id\_pedido de la tabla pedidos





- Tabla **log\_pedidos\_before**: log de pedidos (para auditoría). Esta tabla se carga mediante disparador al realizar pedidos.

#### LOG\_PEDIDOS\_BEFORE

**id\_logPedidoBefore** INT NOT NULL AUTO\_INCREMENT: id único al log (PK)

**timestamp\_operation** TIMESTAMP NOT NULL: fecha y hora del log

**operation** VARCHAR (10): tipo de operación realizada

**hecho\_por** VARCHAR(255): usuario que realizó la operación

**PRIMARY KEY (id\_logPedidoBefore)**: la llave primaria es id\_logPedidoBefore

- Tabla **bkp\_pedidos**: backup de pedidos. Esta tabla se carga mediante disparador al realizar pedidos.

#### BKP\_PEDIDOS

**id\_pedidoBkp** INT NOT NULL AUTO\_INCREMENT: id único al backup de pedido (PK)

**id\_usuario** INT NOT NULL: id del usuario que realizó el pedido (FK)

**id\_pedido** INT NOT NULL: id del pedido (FK)

**productos** JSON: productos del pedido en formato JSON

**fecha\_pedido** DATE NOT NULL: fecha y hora en que realizó el pedido

**PRIMARY KEY (id\_pedidoBkp)**: la llave primaria es id\_pedidoBkp

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario)**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios

**FOREIGN KEY (id\_pedido) REFERENCES pedidos (id\_pedido)**: llave foránea id\_pedido apunta a la llave primaria id\_pedido de la tabla pedidos



- Tabla **log\_usuarios\_before**: log de usuarios (para auditoría). Esta tabla se carga mediante disparador al crear usuarios de la plataforma.

#### LOG\_USUARIOS\_BEFORE

**id\_logUsuarioBefore** INT NOT NULL AUTO\_INCREMENT: id único al log (PK)

**timestamp\_operation** TIMESTAMP NOT NULL: fecha y hora del log

**operation** VARCHAR (10): tipo de operación realizada (ej: insert)

**hecho\_por** VARCHAR(255): usuario que realizó la operación

**PRIMARY KEY (id\_logUsuarioBefore)**: la llave primaria es id\_logUsuarioBefore

- Tabla **bkp\_usuarios**: backup de usuarios. Esta tabla se carga mediante disparador al cargar usuarios de la plataforma.

#### BKP\_USUARIOS

**id\_usuarioBkp** INT NOT NULL AUTO\_INCREMENT: id único al backup de usuario (PK)

**id\_usuario** INT NOT NULL: id del usuario (FK)

**nombre** VARCHAR (100) NOT NULL: nombre del usuario

**apellido** VARCHAR (100) NOT NULL: apellido del usuario

**email** VARCHAR (100) NOT NULL: e-mail del usuario

**telefono** VARCHAR (100) NOT NULL: teléfono del usuario

**fecha\_alta** DATE NOT NULL: fecha de alta del usuario

**avatar\_url** VARCHAR (100) NOT NULL: avatar (enlace a foto de perfil) del usuario

**rol** VARCHAR (20) NOT NULL: rol del usuario (user/admin)

**active\_status** TINYINT NOT NULL: estado del usuario (1- activo, 0- no activo)

**PRIMARY KEY (id\_usuarioBkp)**: la llave primaria es id\_usuarioBkp

**FOREIGN KEY (id\_usuario) REFERENCES usuarios (id\_usuario)**: llave foránea id\_usuario apunta a la llave primaria id\_usuario de la tabla usuarios



- Tabla **log\_discos\_before**: log de usuarios (para auditoría). Esta tabla se carga mediante disparador al cargar los productos (discos).

#### LOG\_DISCOS\_BEFORE

**id\_logDiscoBefore** INT NOT NULL AUTO\_INCREMENT: id único al log (PK)

**timestamp\_operation** TIMESTAMP NOT NULL: fecha y hora del log

**operation** VARCHAR (10): tipo de operación realizada (ej: insert)

**hecho\_por** VARCHAR (255): usuario que realizó la operación

**PRIMARY KEY (id\_logDiscoBefore)**: la llave primaria es id\_logDiscoBefore

- Tabla **bkp\_discos**: backup de discos Esta tabla se carga mediante disparador al cargar los productos (discos).

#### BKP\_DISCOS

**id\_discoBkp** INT NOT NULL AUTO\_INCREMENT: id único al backup de producto (disco) (PK)

**id\_disco** INT NOT NULL: id del disco (FK)

**nombre** VARCHAR (100) NOT NULL: nombre del disco

**banda** VARCHAR (100) NOT NULL: banda del disco

**descripcion** VARCHAR (500) NOT NULL: descripción del disco

**genero** VARCHAR (100) NOT NULL: género musical del disco

**sello** VARCHAR (100) NOT NULL: sello discográfico del disco

**fecha\_album** DATE NOT NULL: fecha de lanzamiento del disco

**precio** DECIMAL NOT NULL: precio de venta del disco

**fecha\_carga** DATE NOT NULL: fecha de carga de disco al sistema

**active\_status** TINYINT NOT NULL: estado del disco (1- activo, 0- no activo)

**PRIMARY KEY (id\_discoBkp)**: la llave primaria es id\_discoBkp

**FOREIGN KEY (id\_disco) REFERENCES discos (id\_disco)**: llave foránea id\_disco apunta a la llave primaria id\_disco de la tabla discos

## DESCRIPCIÓN DE VISTAS (VIEWS)

- **vw\_usuariosPaís:** usuarios activos por país.

Tablas utilizadas: usuarios, direcciones.

pais	cantidad_usuarios_activos
------	---------------------------

- **vw\_usuariosMensajes:** mensajes por usuario.

Tablas utilizadas: usuarios, mensajes.

id_usuario	nombre_completo	cantidad_mensajes
------------	-----------------	-------------------

- **vw\_usuariosPedidos:** pedidos y discos comprados por usuario.

Tablas utilizadas: usuarios, pedidos, ventas.

id_usuario	nombre_completo	cantidad_pedidos	cantidad_discos	costo_total
------------	-----------------	------------------	-----------------	-------------

- **vw\_sesionesActivas:** sesiones de usuario activas en la plataforma online.

Tablas utilizadas: usuarios, sesiones.

id_usuario	nombre_completo	sesion_creacion	sesion_expiracion	sesion_cryptId
------------	-----------------	-----------------	-------------------	----------------

- **vw\_ventasDetalle:** detalle de ventas.

Tablas utilizadas: ventas, discos.

id_venta	id_pedido	id_usuario	id_disco	precio	cantidad	subtotal_venta
----------	-----------	------------	----------	--------	----------	----------------

- **vw\_totalVendido:** cantidad de pedidos, ítems pedidos y total vendido.

Tablas utilizadas: ventas, discos.

cantidad_pedidos	items_vendidos	total_vendido
------------------	----------------	---------------

- **vw\_totalComprado:** cantidad de compras, ítems comprados y total comprado.

Tablas utilizadas: compras, discos.

cantidad_compras	items_comprados	total_comprado
------------------	-----------------	----------------

- **vw\_comprasEmpleados:** compras (para stock) por empleado.

Tablas utilizadas: compras, empleados.

id_empleado	nombre_completo	cantidad_compras	cantidad_discos	costo_total
-------------	-----------------	------------------	-----------------	-------------

- **vw\_discosGeneros:** discos activos por género musical.

Tablas utilizadas: discos.

genero	cantidad_discos_activos
--------	-------------------------

- **vw\_discosVendidos:** discos más vendidos.

Tablas utilizadas: ventas, discos.

id_disco	disco	discos_vendidos
----------	-------	-----------------

## DESCRIPCIÓN DE FUNCIONES (FUNCTIONS)

- **fn\_precioPesos:** convertir precio en pesos (ingresando de parámetro: precio en dólares).
- **fn\_pedidosPaís:** cantidad de pedidos por país (ingresando de parámetro: nombre de país).
- **fn\_pedidosUsuario:** cantidad de pedidos por usuario (ingresando de parámetro: e-mail de usuario).
- **fn\_itemsUsuario:** cantidad de productos (discos) comprados por usuario (ingresando de parámetro: e-mail de usuario).
- **fn\_comprasEmpleadoStock:** cantidad de compras por empleado (ingresando de parámetro: id de empleado).
- **fn\_itemsEmpleadoStock:** cantidad de productos (discos) comprados para stock por empleado (ingresando de parámetro: id de empleado).
- **fn\_totalBalance:** diferencia entre total vendido y total comprado (no recibe parámetros).

## DESCRIPCIÓN DE PROCEDIMIENTOS (STORE PROCEDURES)

- **sp\_ventasUsuario:** listar ventas/pedidos por usuario, ventas en detalle (ingresando e-mail de usuario).
- **sp\_comprasEmpleado:** listar compras realizadas por empleados para stock, compras en detalle (ingresando id de empleado).
- **sp\_ordenTabla:** ordenar tabla (ingresando nombre de tabla, campo a ordenar, orden (asc o desc)).
- **sp\_insertarDisco:** insertar disco en tabla discos (ingresando los datos respectivos al disco).
- **sp\_insertarUsuario:** insertar usuario en tabla usuarios (ingresando los datos respectivos al usuario).
- **sp\_insertEmpleado:** insert empleado en tabla empleados (ingresando los datos respectivos al empleado).
- **sp\_desactivarDisco:** 'desactivar' disco, poniendo el active\_status en 0 (ingresando id de disco).
- **sp\_desactivarUsuario:** 'desactivar' usuario, poniendo el active\_status en 0 (ingresando id de usuario).
- **sp\_desactivarEmpleado:** 'desactivar' empleado, poniendo el active\_status en 0 (ingresando id de empleado).
- **sp\_activarDisco:** 'activar' disco, poniendo el active\_status en 1 (ingresando id de disco).
- **sp\_activarUsuario:** 'activar' usuario, poniendo el active\_status en 1 (ingresando id de usuario).
- **sp\_activarEmpleado:** 'activar' empleado, poniendo el active\_status en 1 (ingresando id de empleado).

- **sp\_eliminarDisco**: eliminar disco de tabla discos (ingresando id de disco)
- **sp\_eliminarUsuario**: eliminar usuario de tabla usuarios (ingresando id de usuario)
- **sp\_eliminarEmpleado**: eliminar empleado de tabla empleados (ingresando id de empleado)

## DESCRIPCIÓN DE DISPARADORES (TRIGGERS)

- **tr\_after\_insertPedido\_venta**: se dispara después de cargar datos en tabla pedidos, cargando tabla ventas.
- **tr\_before\_insertPedido\_log**: se dispara antes de cargar datos en tabla pedidos, cargando logs de la operación (auditoría).
- **tr\_after\_insertPedido\_bkp**: se dispara después de cargar datos en tabla pedidos, cargando tabla bkp\_pedidos (backup).
- **tr\_before\_insertUsuario\_log**: se dispara antes de cargar datos en tabla usuarios, cargando logs de la operación (auditoría).
- **tr\_after\_insertUsuario\_bkp**: se dispara después de cargar datos en tabla usuarios, cargando tabla bkp\_usuarios (backup).
- **tr\_before\_insertDisco\_log**: se dispara antes de cargar datos en tabla discos, cargando logs de la operación (auditoría).
- **tr\_after\_insertDisco\_bkp**: se dispara después de cargar datos en tabla discos, cargando tabla bkp\_discos (backup).

## DESCRIPCIÓN DE USUARIOS (DCL - DATA CONTROL LANGUAGE)

- **Usuario "root@localhost"**: permisos root sobre la base de datos.
- **Usuario "juanp@localhost"**: permisos de lectura sobre la base de datos.
- **Usuario "marian@localhost"**: permisos de lectura, escritura y actualización sobre la base de datos.

## BACKUP

- Se realizan 3 backups: de datos, de estructura, y de datos y estructura. Se encuentran en la carpeta dumps.

## HERRAMIENTAS UTILIZADAS

- Para la creación y gestión de base de datos se usó **MySQL** con **MySQL Workbench**.

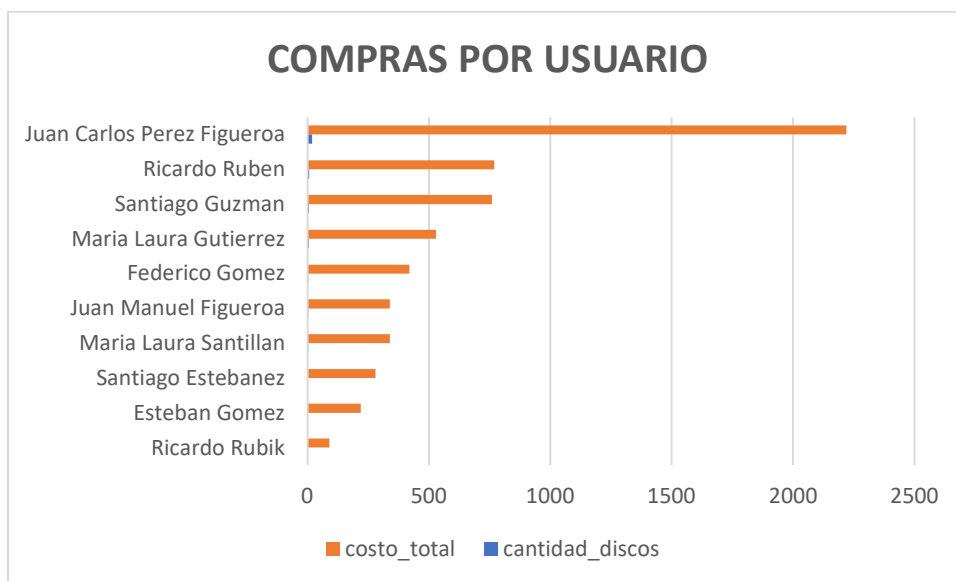
### Nota:

Para creación y configuración de la BD, así como la inserción de datos, se adjunta una carpeta 'scripts' y un documento de texto detallando el orden de ejecución.

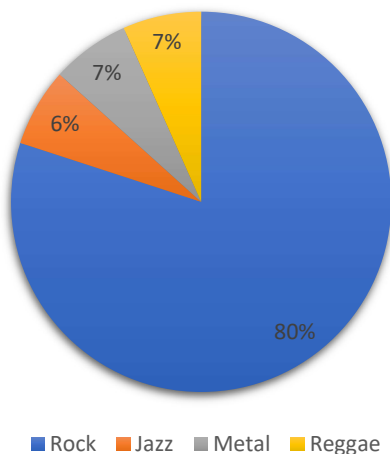


## INFORME

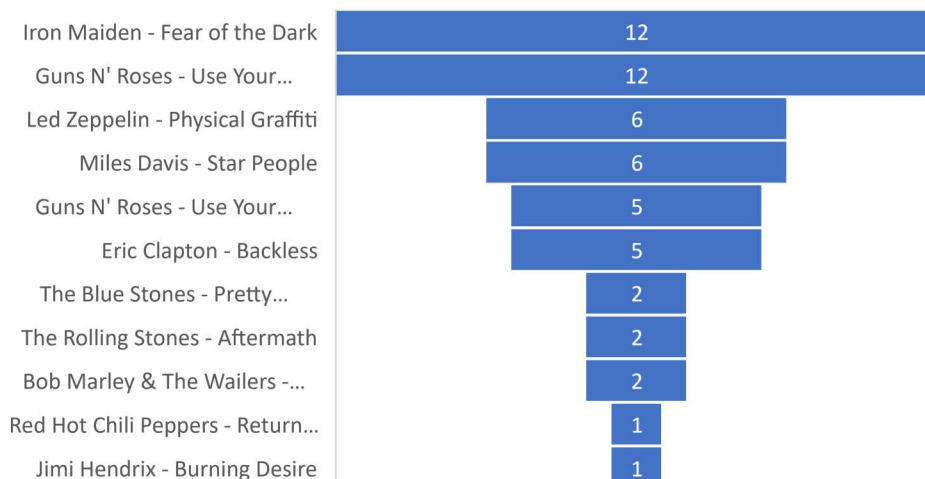
Usando la base de datos configurada, se obtiene información de relevancia al estado de negocio de la empresa. Se muestran algunos gráficos obtenidos a partir de la misma.



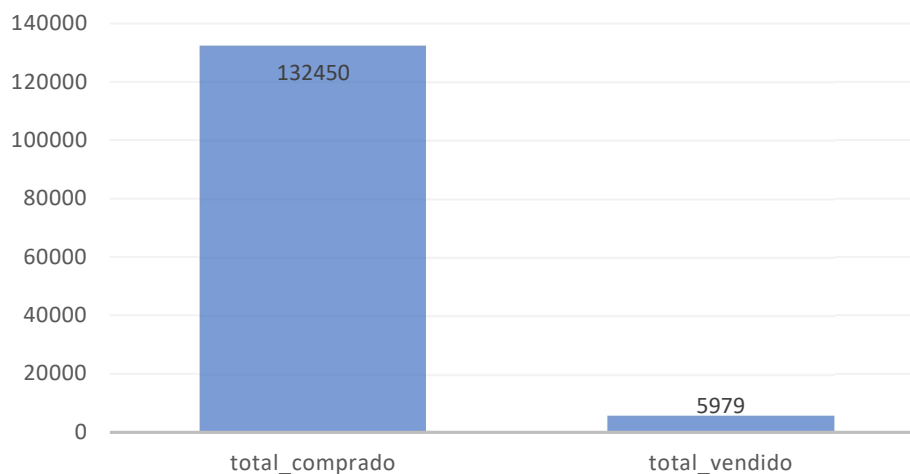
## DISCOS ACTIVOS POR GÉNERO



## DISCOS MÁS VENDIDOS



## COMPRADO VS VENDIDO



## CONCLUSIÓN

Con las tablas generadas, así como las vistas, funciones, procedimientos y disparadores implementados en la base de datos, se puede obtener información de relevancia y así tener un panorama general del estado de negocio de la empresa, permitiendo mejorar la toma de decisiones.