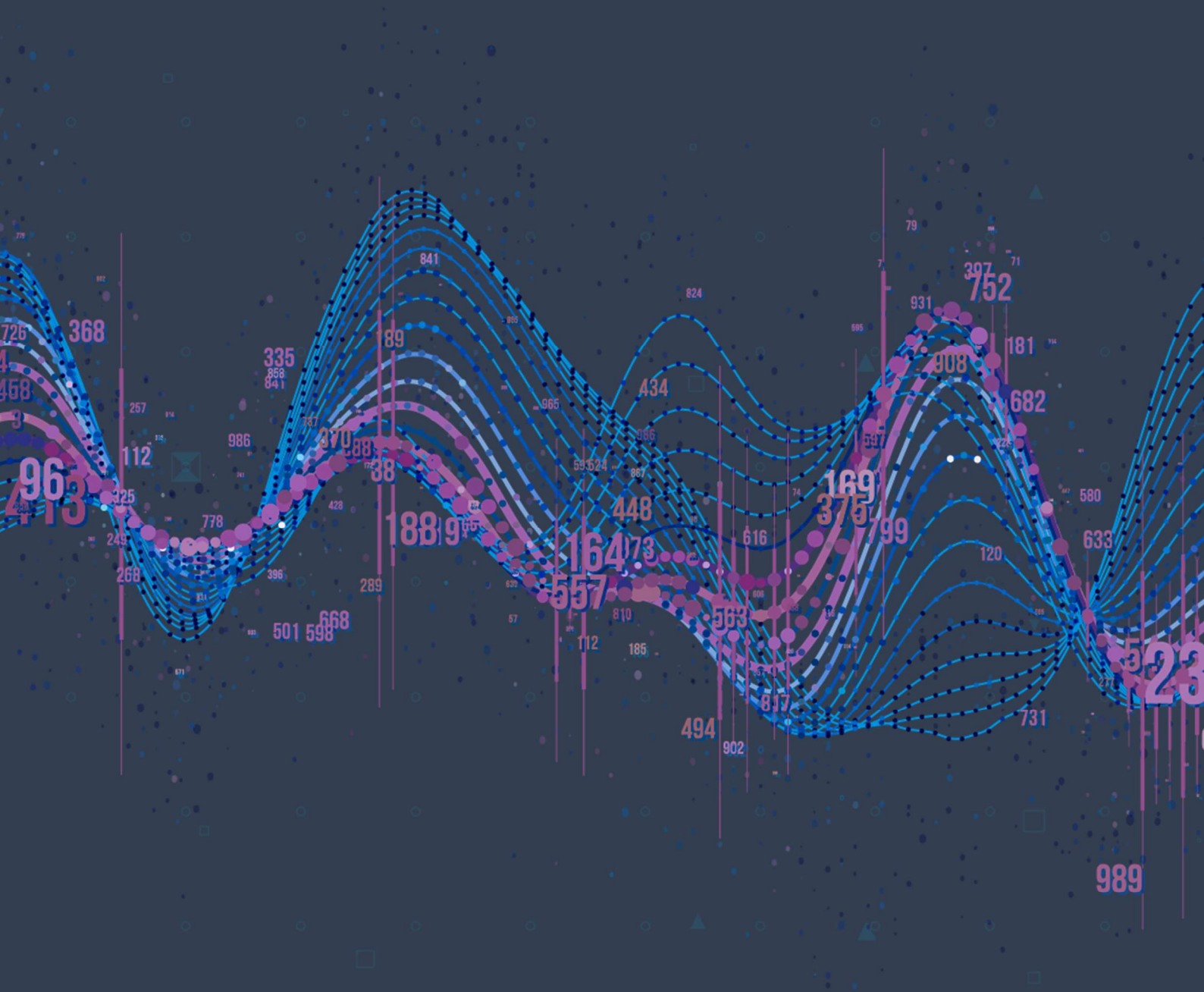
PROYECTO SQL – CODERHOUSE
MUSIC STORE DATABASE



- 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

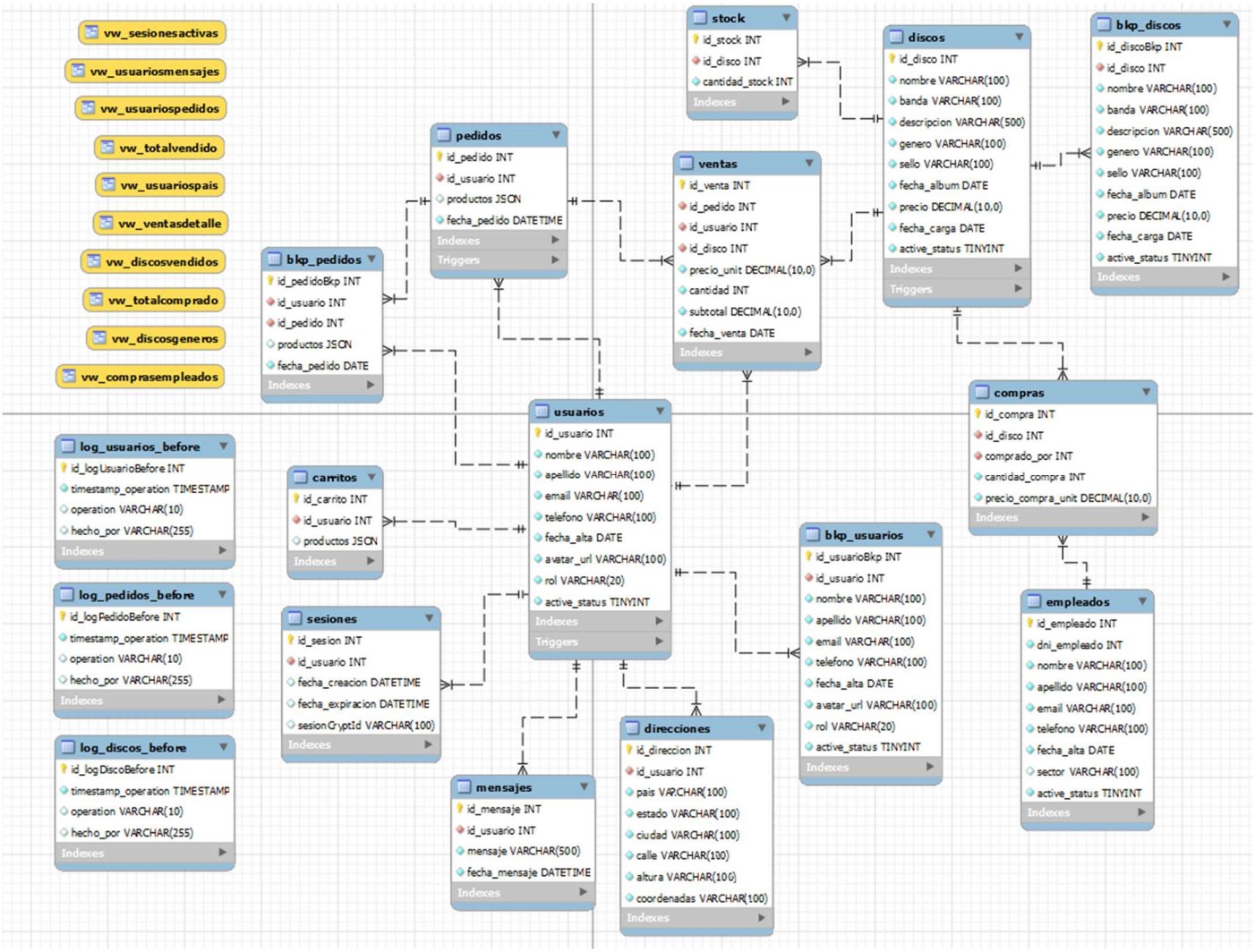
- USUARIOS

- 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 informacion del carrito pasa al pedido,el carrito se vacia 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 (precio_unit * 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_usuariosPais: 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_pedidosPais: 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 (STORED 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 adjunta dumps.

HERRAMIENTAS UTILIZADAS

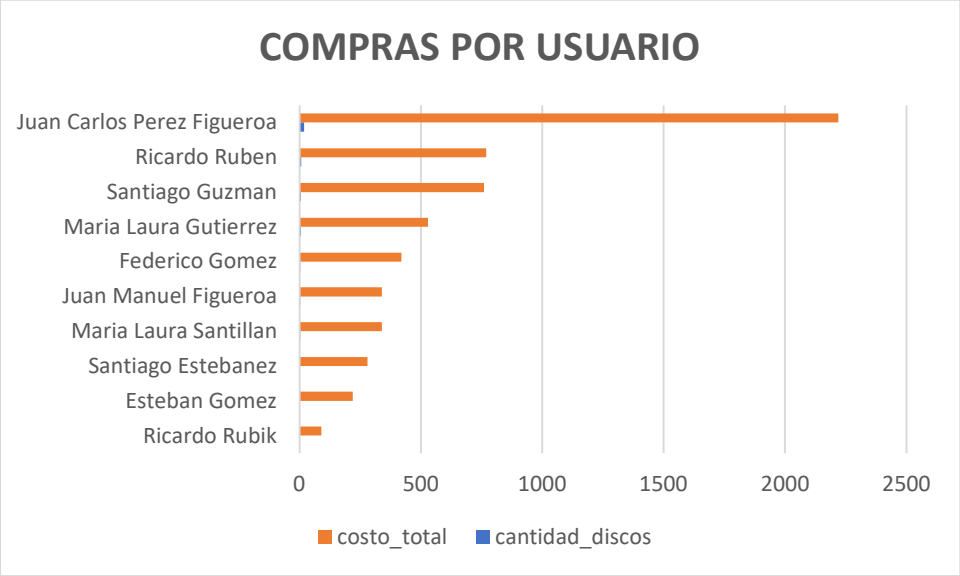
- Para la creación y gestión de base de datos se usó **MySql** con **MySql Workbench**.

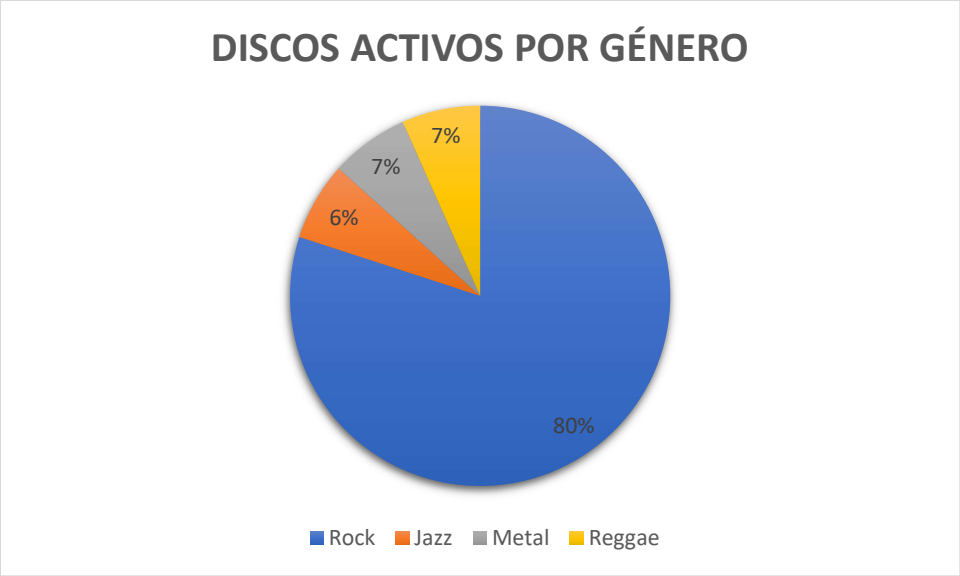
Nota:

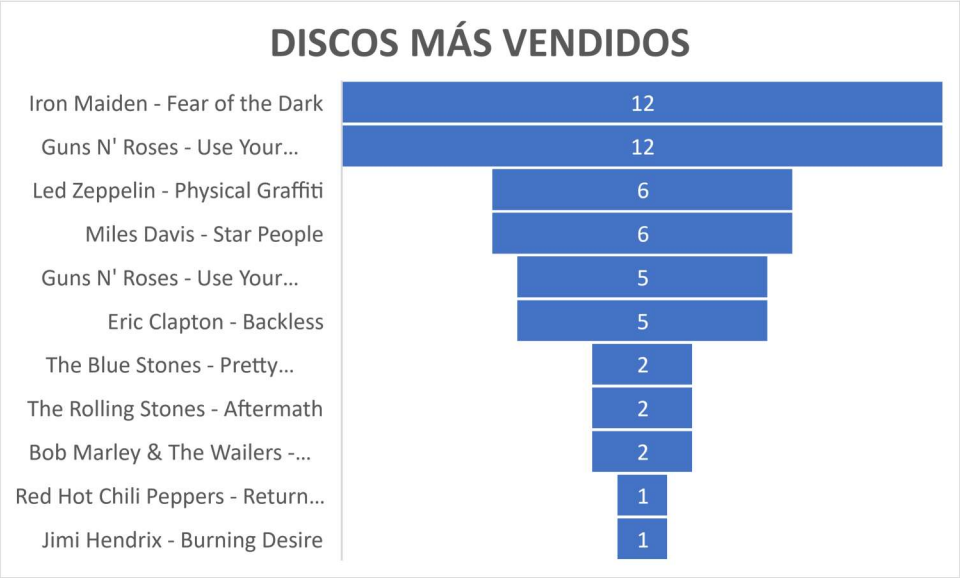
Para creación y configuración de la BD, así como la inserción de datos, se adjunta una carpeta 'scripts' y un documento de texto detallando el orden de ejecución.

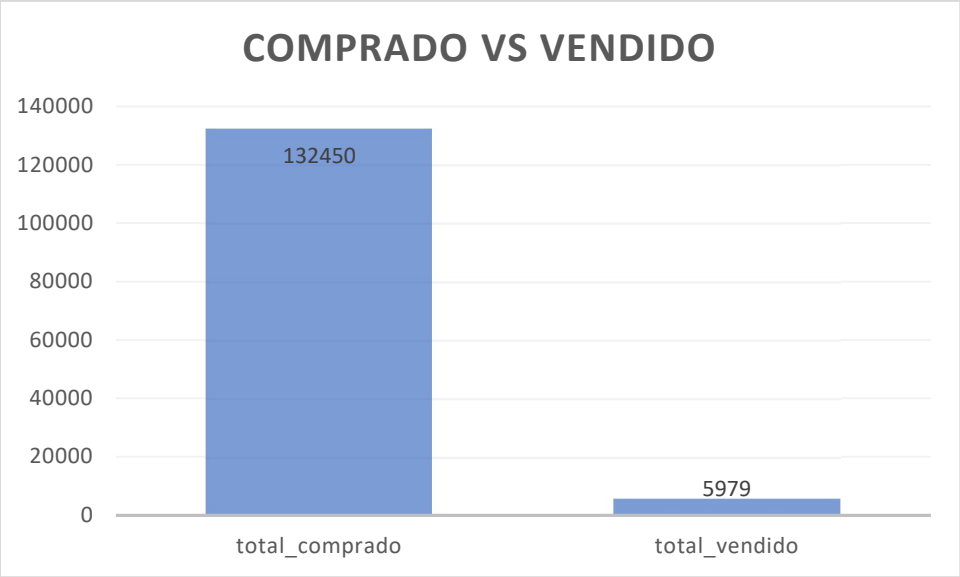
INFORME

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
7%
7%
6%
80%
Rock
Jazz
Metal
Reggae


DISCOS MÁS VENDIDOS
Iron Maiden - Fear of the Dark
Guns N' Roses - Use Your...
Led Zeppelin - Physical Graffiti
Miles Davis - Star People
Guns N' Roses - Use Your...
Eric Clapton - Backless
The Blue Stones - Pretty...
The Rolling Stones - Aftermath
Bob Marley & The Wailers -...
Red Hot Chili Peppers - Return...
Jimi Hendrix - Burning Desire
12
12
6
6
5
5
2
2
2
1
1


COMPRADO VS VENDIDO
140000
120000
100000
80000
60000
40000
20000
0
132450
5979
total_comprado
total_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.