

Informe Proyecto Final Base de Datos Avanzada

Base de Datos para un Ecommerce de Libros



Mishel Espinoza, Miguel Quenta, Valeria Zerain

Base de Datos Avanzadas

Facultad de Ingeniería y Arquitectura

Ingeniería de Sistemas Computacionales

26 de junio de 2025

Índice

Resumen (Abstract)	2
1. Introducción	2
2. Objetivos	3
a. Objetivo General	3
b. Objetivos Específicos	3
3. Requisitos	3
Funcionales	3
No funcionales	4
4. Diseño conceptual	4
Modelo Conceptual	4
Modelo Relacional	4
Diseño documental de la Base de Datos (Referencial y Embebido)	5
Diagrama de flujo del ETL	6
Diseño de la base de Base de Datos SnowFlake	6
5. Diseño de la Base de Datos	6
Modelo NoSQL	6
6. Arquitectura y justificación técnica	9
7. Implementación	10
Construcción de la solución	10
SP	10
View	10
Triggers	11
Functions	11
Índices	11
Transacciones ACID	11
Backups	12
Roles y Permisos	12
Base de datos Distribuidas	12
REDIS (CACHE)	13
Consultas MongoDB	13
Big Data, ETL, Diagrama Snowflake.	14
Herramientas y tecnologías utilizadas	15
8. Pruebas y evaluación	15
9. Conclusión y recomendaciones	22
10. Referencias	23

Resumen (Abstract)

Este proyecto aborda la problemática de la gestión eficiente de datos e información de un sistema de comercio electrónico que tiene como rubro principal la venta de libros en físico. Para ello se propone el diseño e implementación de base de datos SQL y NoSQL donde se tendrá una arquitectura híbrida que integra base de datos relacionales y no relacionales, así como funciones y arquitecturas robustas para su escalabilidad y mantenimiento que permitirá el seguimiento adecuado a la compra, venta y pedidos realizados por los usuarios. La solución brindada le da un enfoque completo a que los usuarios tengan un acceso a toda la información necesaria para tener un seguimiento de las compras y transacciones. Las herramientas y tecnologías utilizadas incluyeron MySQL, MongoDB y Docker, entre otras.

1. Introducción

En el contexto del mundo actual, los comercios digitales permiten tener mayor alcance y relevancia en el mercado, al ser una solución que se puede brindar para diferentes partes del mundo se debe garantizar una operatividad eficiente, tener un control de inventario, gestión de compra y venta de parte de los usuarios y la satisfacción que estos tienen al momento de realizar sus búsquedas y compras.

Este proyecto está enfocado a la venta de libros en formato físico, el cual necesita tener una estructura de datos organizada para poder tener acceso a la información de compra, un control de inventario preciso, aplicación de cupones y descuentos por antigüedad de los clientes, compras seguras y uso de filtros rápidos para poder buscar los libros, recomendaciones o productos deseados con la oportunidad de tener un feedback de parte de los clientes.

Los administradores del negocio necesitan esta claridad para evitar confusiones y tener el proceso de venta claro y organizado para que los usuarios se sientan satisfechos con su experiencia de compra. Los usuarios buscan poder hacer las compras con la mejor información y buscando las opciones más fiables para evitar estafas.

2. Objetivos

a. Objetivo General

Diseñar una gestión de datos para un sistema de comercio electrónico de libros para tener mediante la implementación de bases de datos que permitan una administración de los procesos de compra-venta, inventarios, seguimientos de pedidos y mejorar la experiencia de usuario.

b. Objetivos Específicos

- Analizar los requerimientos para el sistema.
- Diseñar la estructura y arquitectura de las bases de datos.
- Integrar mecanismos de seguridad, respaldo de datos y acceso eficiente.
- Evaluar diseño y escalabilidad del sistema.

3. **Requisitos**

Funcionales

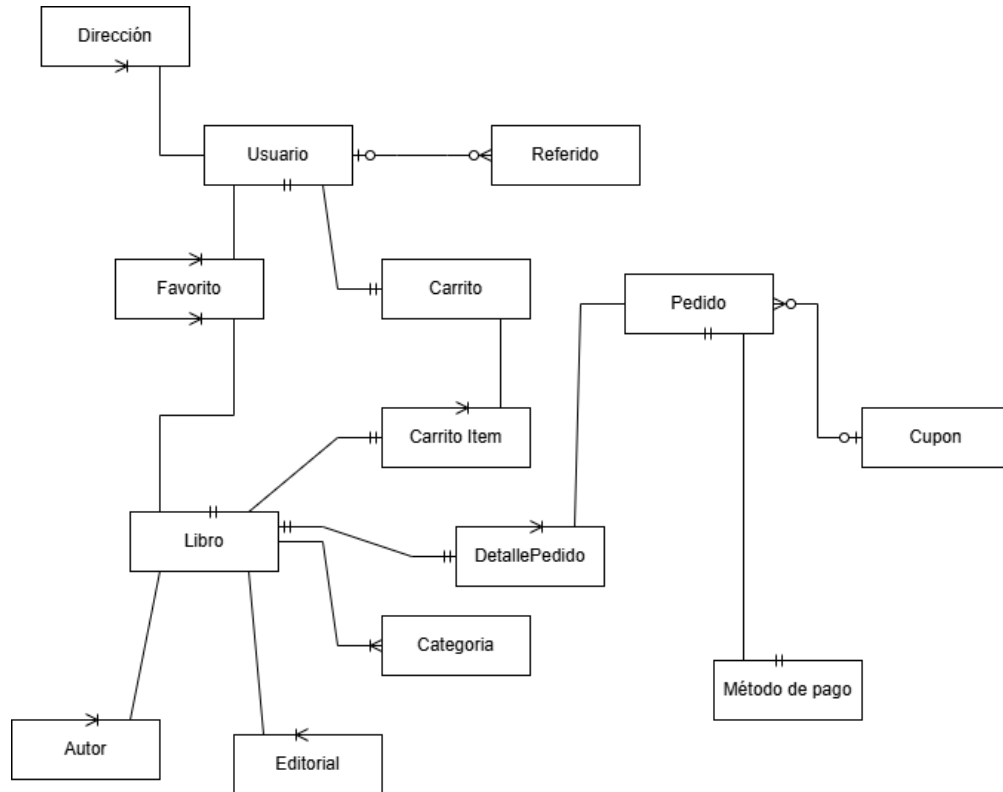
- Registrar clientes
- Tener un reporte de ventas por año para el administrador
- Aplicación y verificación de cupones
- Recomendaciones mediante códigos referidos a nuevos clientes
- Pedidos y verificación de stock para realizar un pedido
- Realizar pagos con verificación de finalización
- Registrar cupones por parte del administrador
- Generar reportes de venta anuales para el administrador
- Tener una wishlist por usuario
- Tener notificaciones a usuarios
- Los usuarios deben poder dar feedback de cada compra o mandar mensajes
- Calcular y mostrar la calificación promedio por libro y por usuario
- Detectar clientes frecuentes
- Poder aplicar promociones y precios nuevos por categoría
- Generar facturas de los pedidos
- Gestionar el envío para el seguimiento

No funcionales

- Tener backups automáticos cada 24 horas
- Datos de clientes sensibles ofuscados
- Hacer uso de la memoria caché para que sea más rápido
- Mantener una disponibilidad mínima del 99% para operaciones críticas como registro de compras o generación de facturas
- Registrar logs de acceso y actividad para auditoría
- Arquitectura distribuida con réplicas para la integridad ante fallos

4. Diseño conceptual

Modelo Conceptual



Modelo Relacional

- Usuario (id *pk*, nombre, apellido, email, password, fecha_registro)
- Autor (id *pk*, nombre, nacionalidad)
- Categoría (id *pk*, categoria)
- Editorial (id *pk*, editorial, país)
- Libro (isbn *pk*, título, descripción, precio, stock, autor_id *fk*, categoria_id *fk*, editorial_id *fk*, publicado)
- Favorito (id *pk*, usuario_id *fk*, libro_id *fk*, fecha_agregado)
- Referido (id *pk*, cliente_id *fk*, referido_id *fk*, fecha_referencia)
- Cupón (id *pk*, codigo, descuento_porcentaje, fecha_expiracion, max_uso, usos_actuales, activo)
- Carrito (id *pk*, usuario_id *fk*, fecha_creacion)
- CarritoItem (id *pk*, carrito_id *fk*, libro_isbn *fk*, cantidad)

- Pedido (id *pk*, usuario_id *fk*, fecha_pedido, estado, total, cupon_id *fk*, metodopago_id *fk*)
- DetallePedido (id *pk*, pedido_id *fk*, libro_isbn *fk*, cantidad, precio_unitario)
- MétodoPago (id *pk*, tipo, descripción)
- Dirección (id *pk*, usuario_id *fk*, dirección, ciudad, departamento, país, codigo_postal)
- LogPedido (id *pk*, pedido_id *fk*, accion, descripcion, fecha)

Diagramas de las tablas se encuentra en Anexo 1 y Anexo 2

Diseño documental de la Base de Datos (Referencial y Embebido)

En este caso se tiene una colección de comentarios que es referencial ya que hace referencia a los usuarios y libros.

También se tiene una colección de recomendaciones que es referencial ya que hace referencia al usuario y al libro.

Hay una colección de historial de precios que es referencial ya que se une con los libros y sus detalles.

Hay una colección de envíos que es referencial ya que se une con los pedidos y su estado.

Cada cliente tiene acceso a su Wishlist por lo que esta colección es referencial, ya que hace referencia a los clientes y a los libros.

Cada usuario tiene su sesión que hace referencia a la tabla de usuarios.

Las facturas son de modelo embebido ya que se guarda directamente todo el pedido y los detalles que tiene sin usar referencias a otras colecciones.

Diagrama de flujo del ETL

Para el ETL se van a extraer los datos de los usuarios para ver el comportamiento de compra que tienen en la plataforma para ver los procesos que se podrían mejorar o los que dan resultados, para esto se extraen los datos de las base de datos de MySQL, la cual almacena los datos de inventario y registros complementarios, PostgreSQL, la cual tiene la información de pedidos, usuarios y procesos de compras y MongoDB en la cual se encuentran colecciones como comentarios, historial de navegación, y recomendaciones.

Posteriormente se hará la transformación para tener todos los datos normalizados y estandarizados para poder facilitar su análisis.

Para finalizar se hará la carga de datos en un DataWarehouse para poder demostrar el análisis y el reporte final (Anexo 3).

Diseño de la base de Base de Datos Snowflake

Para la base de datos de Snowflake se quiere sacar un informe de las ventas que existen y el comportamiento de los usuario, las ventas realizadas por libro, categoría o método de pago, efectividad de cupones y la mayor tendencia a libros o las wishlist. La información será obtenida de las bases de datos de MySQL, PostgreSQL y MongoDB.

Las preguntas que se quieren responder son las siguientes:

- ¿Cuántos pedidos realiza en promedio un cliente?
- ¿Cuántas veces se utilizaron los cupones?
- Clientes más rentables
- Categorías más compradas
- Preferencia de uso de los métodos de pago dentro de la plataforma

5. Diseño de la Base de Datos

Modelo NoSQL

El módulo NoSQL se encarga de gestionar las interacciones que tienen los usuarios con los libros disponibles, así como la retroalimentación entre usuarios, relacionada con sus experiencias de compra y lectura. Este módulo gestiona las calificaciones que los clientes otorgan a los libros adquiridos, así como los comentarios que pueden dejar en cada producto.

- Usuarios: esta colección almacena información sobre los usuarios registrados en la plataforma.
 - `_id`: Identificador único generado por MongoDB. Para facilitar la demostración, se definió manualmente como ObjectId.
 - `teléfono`: Número de contacto del usuario.
 - `notificaciones`: Arreglo de objetos que almacena los mensajes enviados al usuario. Cada notificación incluye:
 - `mensaje`: Contenido de la notificación.
 - `leído`: Valor booleano que indica si fue leída.
 - `fecha`: Fecha y hora en la que fue generada.

- Libro: representa los libros disponibles en el catálogo de la tienda.
 - `_id`: Identificador único asignado automáticamente por MongoDB.
 - `seguimientos`: Arreglo de objetos que indica qué usuarios siguen el libro.
- Comentarios: contiene las reseñas y opiniones que los usuarios han dejado sobre los libros.
 - `_id`: Identificador único del comentario.
 - `libro_id`: Referencia al libro comentado.
 - `usuario_id`: Referencia al usuario que dejó el comentario.
 - `texto`: Contenido del comentario.
 - `calificacion`: Valor numérico de calificación del libro (1-5).
 - `fecha`: Fecha del comentario.
 - `reportes`: Arreglo de objetos que registran los reportes de otros usuarios sobre el comentario. Cada reporte incluye:
 - `fecha`: Fecha del reporte.
- Facturas: almacena las facturas generadas por cada compra.
 - `_id`: Identificador único de la factura.
 - `usuario_id`: Usuario que realizó la compra.
 - `fecha`: Fecha de emisión de la factura.
 - `total_pagado`: Monto total de la factura.
 - `items`: Arreglo de objetos que representan los productos adquiridos, con:
 - `titulo`: Título del libro.
 - `cantidad`: Unidades compradas.
 - `precio_unitario`: Precio por unidad.
- Envíos: contiene la información logística relacionada con los pedidos despachados.
 - `_id`: Identificador del envío.
 - `pedido_id`: Referencia al pedido o factura.
 - `empresa`: Empresa de transporte
 - `tracking`: Código de rastreo.
 - `estado`: Estado actual del envío
 - `fecha_envio`: Fecha de envío real.
 - `fecha_estimado`: Fecha estimada de entrega.
- Historial_precios: registra los cambios de precios que ha tenido un libro.
 - `_id`: Identificador del cambio de precio.

- libro_id: Referencia al libro cuyo precio cambió.
 - precio_anterior: Precio anterior.
 - precio_nuevo: Precio actualizado.
 - fecha_cambio: Fecha en que ocurrió el cambio.
- Wishlists: almacena los libros que un usuario ha marcado como deseados.
 - _id: Identificador único.
 - usuario_id: Referencia al usuario.
 - libros: Arreglo de identificadores de libros agregados a la lista de deseos.
 - fecha_creacion: Fecha de creación de la wishlist.
- Sesiones: registra el historial de sesiones de los usuarios en la plataforma.
 - _id: Identificador único de la sesión.
 - usuario_id: Referencia al usuario.
 - fecha_inicio: Fecha y hora de inicio de sesión.
 - fecha_fin: Fecha y hora de cierre de sesión.
 - ip: Dirección IP del dispositivo utilizado.
 - navegador: Navegador web utilizado.
- Recomendaciones: contiene libros recomendados a usuarios según su historial o intereses.
 - _id: Identificador del conjunto de recomendaciones.
 - usuario_id: Referencia al usuario.
 - libros_recomendados: Arreglo de objetos, cada uno con:
 - libro_id: Libro recomendado.
 - razon: Criterio o motivo de la recomendación.
 - fecha_generacion: Fecha en que se generó la recomendación

6. Arquitectura y justificación técnica

Para la arquitectura híbrida se utilizaron motores como MySQL, PostgreSQL, MongoDB y Redis.

En el caso de MySQL y PostgreSQL que son un sistema de gestión de bases de datos relacional las elegimos debido a su facilidad de uso, así como la robustez que ofrecen y la integración que tienen. Al combinar ambos motores se logra aprovechar las ventajas que tiene cada uno, PostgreSQL destaca por su cumplimiento de estándares SQL, su robustez y soporte para tipos de datos complejos, mientras que MySQL es reconocido por su velocidad y facilidad de uso en aplicaciones web y transaccionales. (Team, 2025)

Esta distribución se realizó con sharding, se tiene cuatro bases de datos master, dos en cada gestor y para la facilidad de lectura se tiene dos slave de cada máster, terminando con 8 slaves en total, esto debido a que así se distribuye mejor la carga y se balancea lo necesario. Basándonos en el siguiente diagrama para visualizar de mejor manera la arquitectura que maneja:

Para la parte de NoSQL se eligió utilizar MongoDB por la experiencia en el uso y también porque al ser una base de datos documental en la cual se puede tener esquemas flexibles nos da una ventaja al momento de hacer las colecciones de datos y tener la escalabilidad horizontal que MongoDB nos brinda (MongoDB, s. f.). Otra ventaja que da para este proyecto es que almacena datos en formato BSON (JSON binario), lo que permite un almacenamiento y una recuperación eficientes (Devoteam, 2024).

Para almacenar el caché se utilizó Redis, debido a su alto rendimiento y sus funciones de almacenamiento en memoria . Sirve para almacenar datos de acceso frecuente que se pueden recuperar rápidamente. Gracias a esta función, reduce la carga de la base de datos y mejora el rendimiento general de la aplicación (Wisniowski, 2025). Así que se lo consideró como la opción más segura y fácil de implementar para poder manejar los datos temporales en memoria.

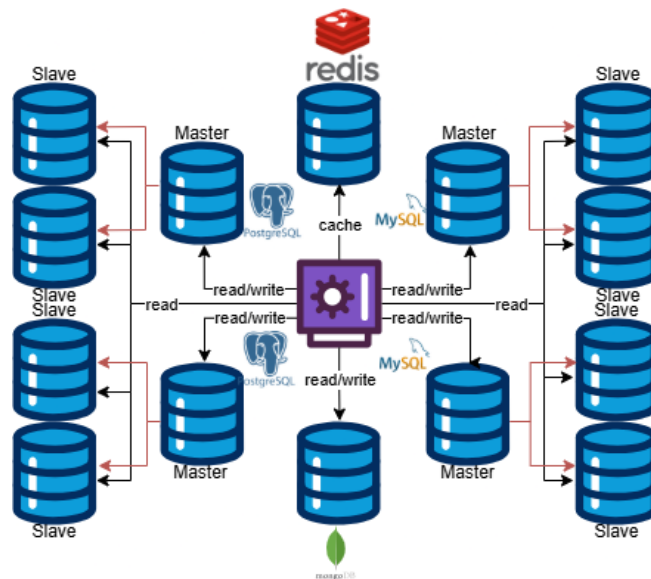


Figura 1: Diagrama de arquitectura de la base de datos

7. Implementación

Construcción de la solución

SP

1. Realizar el pago de un pedido, simulando la validación del pago.
2. Realizar un pedido completo, aplicando cupones, stock, cálculos e inserciones.
3. Validar y registrar el uso de un cupón.
4. Aplicar descuento masivo por categoría de libros.
5. Registrar cliente con validación de correos y opcionalmente tener un referido
6. Realizar un ranking con los clientes que tienen más referidos.

View

1. Para los reportes se tiene una vista con los pedidos, cupones y detalles de pedidos.
2. Para un manejo de catálogo e inventario se tiene una vista con libros, autores, editoriales y categorías.
3. Para que los usuarios puedan ingresar de manera rápida a su lista de favoritos se tiene una vista con los clientes y sus libros favoritos en un JSON.

Triggers

1. Aplicar un descuento automático si el cliente tiene más de 10 compras.
2. Insertar en un log cuando se descuenta el stock por una venta.
3. Si el stock de un libro llega a 0 lo marca como no disponible.
4. Insertar un log de compra luego de realizar y completar un pedido.
5. Registra un log si se elimina un pedido.
6. Muestra mensaje si el nuevo cliente fue referido.

Functions

1. Para reportes se tiene la cantidad de ventas por mes en un año específico.
2. Retorna el monto total monetario de ventas por mes en un año específico.
3. Contar cuántas compras realizó un cliente.
4. Para los filtrados se genera un json con categorías como clave y la lista de los libros como valores.
5. Ranking de clientes con más referencias.
6. Recuperar toda la información detallada de un pedido en específico.

Índices

Se aplican índices en la tabla de pedidos, en las columnas de pedidos y métodos de pago para realizar consultas de los tipos de métodos de pago que utilizan mayormente los usuarios para poder aplicar un mayor enfoque en esas áreas.

Transacciones ACID

Se tiene la implementación de transacciones ACID en los stored procedure realizar pedido y en realizar pago, ambos tienen un rollback y una excepción en caso de que falle el método de pago o que el pedido no se pueda realizar por la falla en el cupón que se está registrando al momento de la compra..

Backups

Se realizó la implementación de backups de los datos cada 24 horas a las 3 am con el uso de cron, así se tiene el respaldo cada día de los datos de las base de datos de PostgreSQL y MySQL.

Roles y Permisos

Los roles y permisos que se tendrán serán de administrador que tendrá acceso a todo, el contable que solo ve las partes transaccionales y el usuario que solo ve los libros y hace nuevas transacciones para las compras.

Base de datos Distribuidas

La arquitectura implementada utiliza un enfoque híbrido de sharding, con PostgreSQL y MySQL, su objetivo es mejorar el rendimiento, la disponibilidad y la escalabilidad (Figura 2).

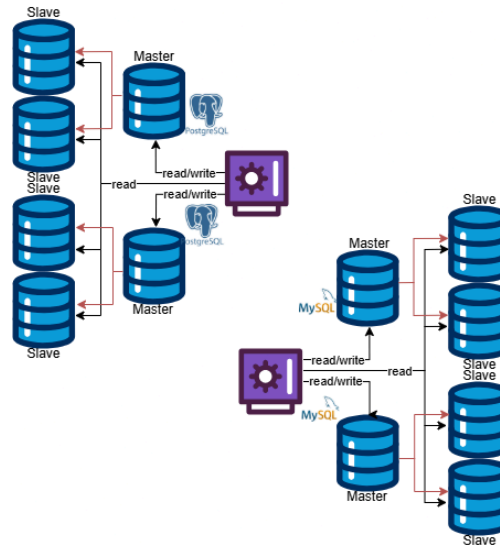


Figura 2: Arquitectura del Sharding

En este caso tenemos dos clusters separados de cada motor, ambos con el mismo patrón de sharding y replicación, cada motor tiene dos másters, las cuales se encargan de la lectura y escritura, los datos están shardeados entre los dos másters, lo que quiere decir que cada uno tiene una porción distinta del total de los datos. Cada máster tiene dos esclavos que reciben las copias de sus datos, estas solo se usan para la lectura, lo que permite aliviar la carga de los másters.

REDIS (CACHE)

Se decidió que el caché se aplicará al momento del iniciar una compra, se guardará en memoria caché los elementos del carrito de una persona y si no existe el dato en el cache lo crea, si existe lo lee directo. Este dato tiene una duración de 10 minutos porque es el tiempo promedio que se tendrá para completar la compra, así se tiene guardado todo el proceso hasta que se hace el pedido (Figura 3).

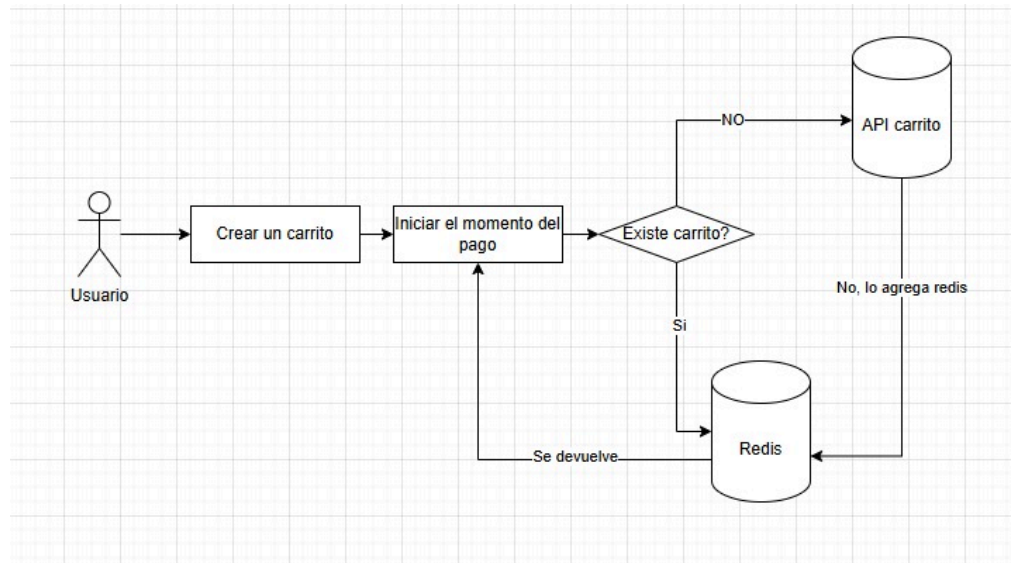


Figura 3: Diagrama de caché en carrito de compras

Consultas MongoDB

1. Cálculo promedio de la cantidad de comentarios que recibe un libro.
2. Mostrar los libros que se recomendaron.
3. Usuarios que realizan compras activamente en la página.
4. Libros sin comentarios.
5. Total de productos vendidos por categoría.
6. Total de ventas por libro
7. Clientes más rentables
8. Libros con stock o igual a n
9. Cupones más usados
10. Promedio de libros por carrito
11. Total gastado por país
12. Libros más deseados
13. Total por método de pago
14. Usuarios más activos por sesión
15. Libros con más cambios de precio
16. Usuarios con más libros en favoritos
17. Facturación por mes
18. Calificación promedio por usuario

19. Usuarios notificaciones no leídas

20. Libros más seguidos

Big Data, ETL, Diagrama Snowflake.

En el caso del ETL se seguirá la estructura mostrada en el Anexo 3 para la extracción, transformación y lectura de datos.

Para esta parte se realizará el análisis del comportamiento de compra de los usuarios para ver los libros más populares, para esto se tiene una tabla de hechos de venta que se relaciona con dimensiones de tiempo, usuario, métodos de pago, ubicación, inventario, libros y este último se relaciona con la categoría, editorial y autor (Figura 4).

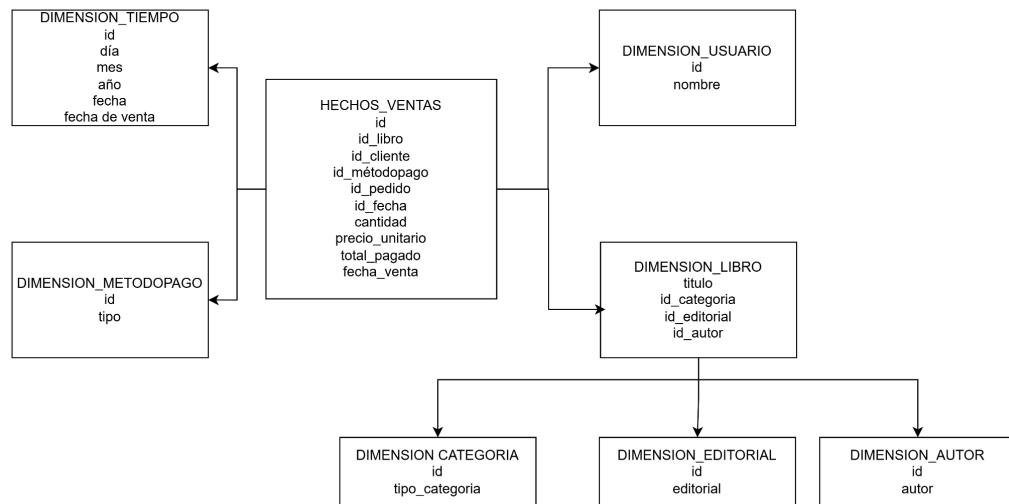


Figura 4: Diseño del Snowflake para el análisis de Big Data

Herramientas y tecnologías utilizadas

Se utiliza un Docker Compose File en el cual está la arquitectura del proyecto de forma modular, en esta se usa PostgreSQL con la imagen de Bitnami PostgreSQL 14. Para la parte de MySQL se incluye un servicio de base de datos MySQL 8. El sistema de caché está basado en Redis 6.2, el cual permite almacenar datos en memoria para mejorar el rendimiento general. Por último, se integra MongoDB 6.0, para gestionar datos no relacionales y documentos flexibles.

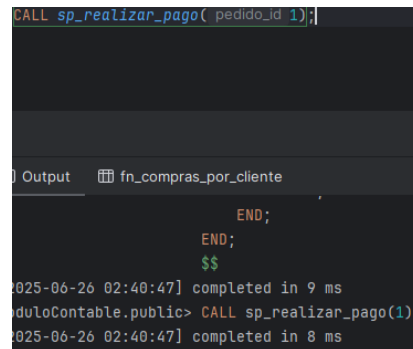
También se tiene el sharding de PostgreSQL y MySQL explicado anteriormente.

8. Pruebas y evaluación

- SP

- Realizar el pago de un pedido, simulando la validación del pago.

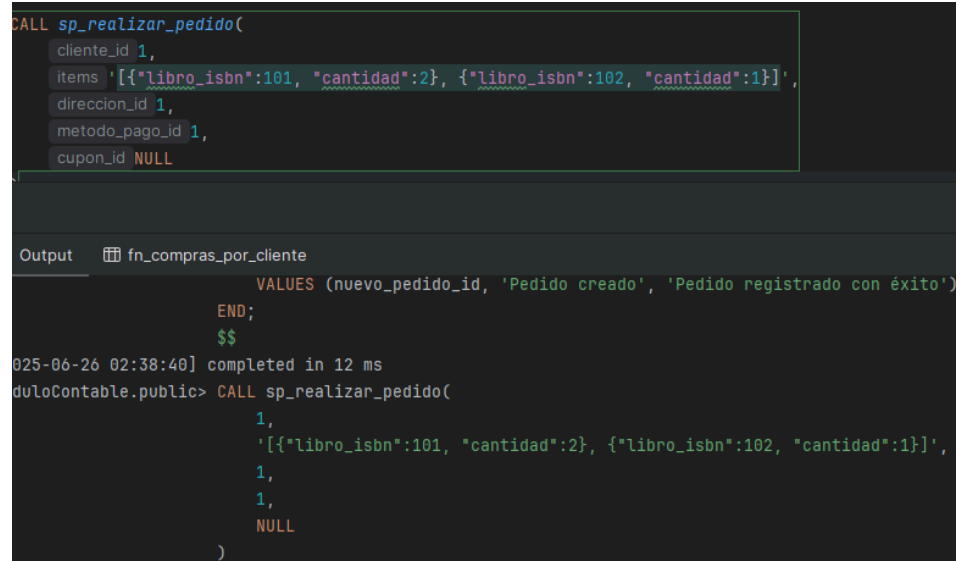
```
CALL sp_realizar_pago( pedido_id 1);
```



```
END;  
END;  
$$  
[2025-06-26 02:40:47] completed in 9 ms  
dueloContable.public> CALL sp_realizar_pago(1)  
[2025-06-26 02:40:47] completed in 8 ms
```

- Realizar un pedido completo, aplicando cupones, stock, cálculos e inserciones.

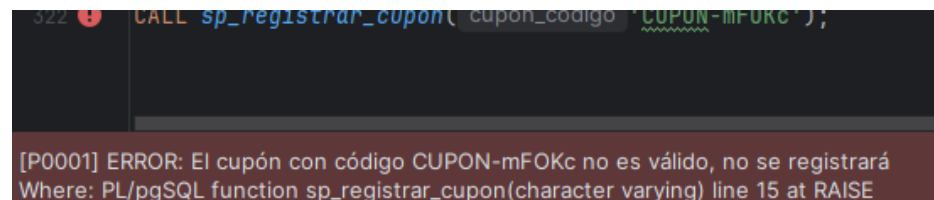
```
CALL sp_realizar_pedido(  
  cliente_id 1,  
  items '[{"libro_isbn":101, "cantidad":2}, {"libro_isbn":102, "cantidad":1}]',  
  direccion_id 1,  
  metodo_pago_id 1,  
  cupon_id NULL  
);
```



```
VALUES (nuevo_pedido_id, 'Pedido creado', 'Pedido registrado con éxito')  
END;  
$$  
[2025-06-26 02:38:40] completed in 12 ms  
dueloContable.public> CALL sp_realizar_pedido(  
  1,  
  '[{"libro_isbn":101, "cantidad":2}, {"libro_isbn":102, "cantidad":1}]',  
  1,  
  1,  
  NULL  
);
```

- Validar y registrar el uso de un cupón.

```
CALL sp_registrar_cupon( cupon_codigo 'CUPON-mFOKc');
```



```
[P0001] ERROR: El cupón con código CUPON-mFOKc no es válido, no se registrará  
Where: PL/pgSQL function sp_registrar_cupon(character varying) line 15 at RAISE
```

Registra si es válido:


```
CALL sp_registrar_cupon( cupon_codigo 'CUPON-BNJsv');

select * from cupones
```

Output modulosContable.public.cupones ventas_por_mes

```
2025-06-26 02:47:18] where: PL/pgSQL function sp_registrar_cup
modulosContable.public> select * from cupones
[2025-06-26 02:50:56] 100 rows retrieved starting from 1 in 106
modulosContable.public> CALL sp_registrar_cupon('CUPON-BNJsv')
[2025-06-26 02:51:07] completed in 19 ms
```

- Aplicar descuento masivo por categoría de libros:

Antes de aplicar el sp

	titulo	precio	categoria
1	Enim iusto delectus.	47.61	Ficción
2	Fugit ipsa distinctio consectetur.	88.50	Ficción
3	Sunt explicabo deserunt at asperiores.	74.95	Ficción
4	Recusandae maiores voluptatibus natus minus doloremque.	29.28	Ficción
5	Voluptates atque occaecati numquam consequatur error.	58.10	Ficción
6	Nihil porro tempore delectus perspiciatis.	98.36	Ficción
7	Mollitia suscipit quasi totam.	79.12	Ficción
8	Voluptatibus iusto temporibus animi veritatis.	52.60	Ficción
9	Delectus distinctio consectetur nisi id.	90.40	Ficción

Luego de aplicar el sp con un descuento del 10% a ficción:

```
call sp_aplicar_descuento_por_categoria( categoria 'Ficción', descuento 10);
```

Output Result 17

	titulo	precio	categoria
	Enim iusto delectus.	42.85	Ficción
	Fugit ipsa distinctio consectetur.	79.65	Ficción
	Sunt explicabo deserunt at asperiores.	67.46	Ficción
	Recusandae maiores voluptatibus natus minus doloremque.	26.35	Ficción
	Voluptates atque occaecati numquam consequatur error.	52.29	Ficción
	Nihil porro tempore delectus perspiciatis.	88.52	Ficción
	Mollitia suscipit quasi totam.	71.21	Ficción
	Voluptatibus iusto temporibus animi veritatis.	47.34	Ficción
	Delectus distinctio consectetur nisi id.	81.36	Ficción

- Registrar cliente con validación de correos y opcionalmente tener un referido

```
CALL sp_registrar_cliente(
  in_nombre 'Maria',
  in_apellido 'Gómez',
  in_email 'maria.gomez.bolannoz@mail.com',
  in_password '0traPass456',
  in_referido_id 3001 -- in_referido_id
);

select * from Usuarios u
inner join Referidos r on r.cliente_id = u.id
where email = 'maria.gomez.bolannoz@mail.com';
```

	fecha_registro	r.id	cliente_id	referido_id	fecha_referencia
1	2025-06-26 11:22:19	2	3003	3001	2025-06-26 11:22:19

- Ver el ranking de los 10 clientes con más referidos.

	full_name	numero_de_referencias
1	Clementina Cortes	3
2	Juana Álvaro	3
3	Juan Francisco Fuente	3
4	Amaro Córdoba	3
5	Lucio Machado	2
6	Sigfrido Chamorro	2
7	Rodrigo Casado	2
8	Jose Angel Cervantes	2
9	Aitana Cañete	2
10	Juanita Rios	2

- Funciones

- Cantidad de ventas por mes en un año específico.

```
SELECT * FROM fn_monto_ventas_por_mes( anio 2025);
```

mes	total_ventas
6	7783818.28

- Monto total monetario de ventas por mes en un año específico.

```
SELECT * FROM ventas_por_mes( año 2025);
```

mes	ventas_realizadas
6	30001

- Contar cuántas compras realizó un cliente.

```
select * from fn_compras_por_cliente( cliente_id 501);
```

fn_compras_por_cliente
58

- Recuperar toda la información detallada de un pedido en específico.

```
select * from fn_detalle_compra( in_pedido_id 120);
```

pedido_id	usuario_id	fecha_pedido	estado	total	cupon_codigo
120	566	2025-06-25 15:50:28.514398	entregado	180.61	<null>
120	566	2025-06-25 15:50:28.514398	entregado	180.61	<null>
120	566	2025-06-25 15:50:28.514398	entregado	180.61	<null>

- Views

- Vista con toda la información de los libros:

vista_libros_detallada	
isbn	int unsigned
titulo	varchar(200)
precio	decimal(10,2)
autor	varchar(100)
editorial	varchar(150)
categoría	varchar(100)

- Vista con toda la información de los pedidos:

vw_pedidos_completos	
pedido_id	integer
usuario_id	integer
fecha_pedido	timestamp
estado	varchar(20)
total	numeric(10,2)
cupon_codigo	varchar(50)
libro_jsbn	integer
cantidad	integer
precio_unitario	numeric(10,2)

- Vista con toda la información de los libros favoritos de los usuarios:

vw_libros_favoritos	
columns 4	
id	int unsigned
nombre	varchar(50)
apellido	varchar(50)
libros_favoritos	json

- Mongo

- Usuarios más activos

{ _id	{ sesiones
665caaa123abcde000000004	2
665caaa123abcde000000009	1
665caaa123abcde000000010	1
665caaa123abcde000000007	1

- Total de productos vendidos por categoría

	categoria ▼	total_vendidos ▼
1	Política	1
2	Realismo Mágico	8
3	Distopía	1
4	Clásico	1
5	Experimental	1
6	Filosofía	2
7	Infantil	1

- Mostrar los libros que se recomendaron.

	_id ▼	razon ▼	usuario_id ▼
1	685135832a69243d7120abd4	Te gustó Realismo Mágico	665caaa123abcde000000004
2	685135832a69243d7120abd4	Similares a tus intereses	665caaa123abcde000000004

- Promedio de compras libro

_id ▼	promedio_libros ▼
<null>	2

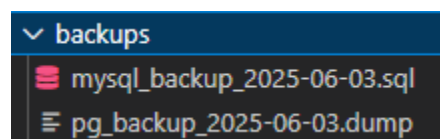
- Index: los índices aplicados mejoraron el tiempo y el costo de esta manera:

--- p time 8.048 ms e time 212.743 ms sin index

--- p time 1.081 ms e time 118.795 ms con index

Así vemos que mejoró el tiempo de consulta de manera considerable y que así se tiene una mayor rapidez al momento de juntar estas dos tablas que calculan la cantidad de dinero que se pagó por cada método y las veces que se usó cada método.

- Aplicación de backup: cada día a las 3:00 am se genera un backup con la fecha:



- Roles

Con el objetivo de establecer un control de acceso claro y seguro dentro del sistema, se han definido tres roles diferenciados, distribuidos entre los motores de bases de datos PostgreSQL y MySQL. Cada rol responde a

responsabilidades específicas, limitando los privilegios conforme a las funciones requeridas.

- 1. Rol contable (PostgreSQL).- El rol contable está destinado a perfiles orientados a la revisión financiera y auditoría. Se le han otorgado exclusivamente privilegios de lectura SELECT sobre las siguientes tablas: Pedidos, DetallesPedido, MetodosPago, Cupones y LogsPedidos.

2. Rol admin_operaciones (PostgreSQL).- El rol admin_operaciones está orientado a personal responsable de la gestión operativa del sistema. Se le ha concedido acceso completo SELECT, INSERT, UPDATE, DELETE sobre las mismas tablas que ocupa el rol contable.

3. Rol user (MySQL).- El rol user representa al usuario final del sistema y se encuentra definido en el entorno MySQL. Se limita a permisos de lectura SELECT sobre las tablas Libros, Autores, Categorías y Editoriales.

Este acceso restringido permite a los clientes consultar el catálogo de productos sin capacidad de alterar la información, resguardando así la integridad y consistencia del inventario publicado.

- Preguntas ETL

- ¿Cuántos pedidos realiza en promedio un cliente?

	promedio_pedidos_por_cliente	÷
		60

- ¿Cuántas veces se utilizaron los cupones? ¿Cuántas veces no se utilizaron?

	total_ventas_con_cupon	÷
1		17850

	total_ventas_sin_cupon	÷
1		41941

- Clientes más rentables

	id_cliente	nombre_completo	total_gastado
1	514	Jose Vilaplana	46123.83999999999
2	839	Toni Crespi	45913.529999999984
3	533	Custodia Revilla	44447.92000000001
4	505	Constanza Lledó	44123.90000000002
5	846	Ale Canales	43776.81999999997
6	615	Aurora Peláez	43567.58999999996
7	777	Reina Vizcaíno	43279.430000000015
8	706	Santiago Marin	42974.46999999998
9	702	Leandra Sacristán	42668.61999999999
10	747	Teresa Busquets	42410.92

- Categorías más compradas

	tipo_categoria	total_vendido
3	Tecnología	17926
4	Ficción	17768
5	Ciencia	17077

- Preferencia de uso de los métodos de pago dentro de la plataforma

	metodo_pago	cantidad_usos
1	Crypto	15135
2	Qr	15042
3	Tarjeta	14921
4	Paypal	14693

9. Conclusión y recomendaciones

El proyecto final para la plataforma integral de comercio electrónico de libros ha demostrado ser una solución robusta y escalable para la gestión de datos relacionados con inventario, compras, clientes, pedidos y comportamiento de usuarios. A través de la implementación de una arquitectura híbrida compuesta por bases de datos relacionales y no relacionales.

La elección de los diferentes motores permitió garantizar integridad, consistencia y disponibilidad de datos. La implementación de sharding y replicación en estos motores permite balancear la carga y aumentar la tolerancia a fallos. Por otro lado, MongoDB brindó flexibilidad para manejar documentos embebidos y referenciales como

comentarios, facturas, historial de navegación y recomendaciones, adaptándose al dinamismo del comportamiento del usuario.

Desde la perspectiva de Big Data, la integración de un flujo ETL hacia un modelo Snowflake permitió analizar patrones de comportamiento de compra, métodos de pago más usados, efectividad de los cupones y libros más deseados. De esta manera se puede ayudar a responder preguntas que pueden ayudar al funcionamiento del negocio.

El uso de Redis como sistema de caché optimizó operaciones en rutas críticas como el uso del carrito en un proceso de compra, reduciendo el tiempo de respuesta y la carga sobre las bases de datos principales. Además, la implementación de transacciones ACID, triggers, views, funciones, ofuscamiento de datos y stored procedures robustecieron la lógica de negocio, garantizando consistencia y trazabilidad en las operaciones.

La implementación de backup y replicación de datos brinda seguridad y una respuesta ante fallos que es importante al momento de tener procesos dentro de cualquier plataforma para no perder la información ni los datos y el hecho de aplicar roles y permisos nos da la seguridad de que la información está resguardada.

Después de este proyecto recomendamos realmente el tener siempre una arquitectura compatible, disponible y robusta desde un inicio, porque el realizar cambios desde la fase 1 hasta esta fase presentó algunos retos al momento de realizar las particiones y reestructurar las consultas, por lo que podemos decir que la migración no es tarea sencilla en estos casos. Pero dentro de todo se pudo aprender lo necesario para poder tener una arquitectura de datos robusta, segura y rápida en futuros proyectos.

10. Referencias

Team, A. A. (2025, 19 marzo). PostgreSQL vs. MySQL: A Comprehensive 8-Factor Comparison. Astera.

<https://www.astera.com/es/knowledge-center/postgresql-vs-mysql/#:~:text=Sobresale%20en%20el%20manejo%20de%20diversos%20formatos.que%20requiere%20ajustes%20para%20un%20rendimiento%20%C3%B3ptimo.>

MongoDB. (s. f.). Why use MongoDB and when to use it?

<https://www.mongodb-com.translate.google.com/resources/products/fundamentals/why-use-mongodb? x tr sl=en& x tr tl=es& x tr hl=es& x tr pto=wa>

Devoteam. (2024, 6 noviembre). MongoDB for Data Management: Handle Big Data Easily | Devoteam.

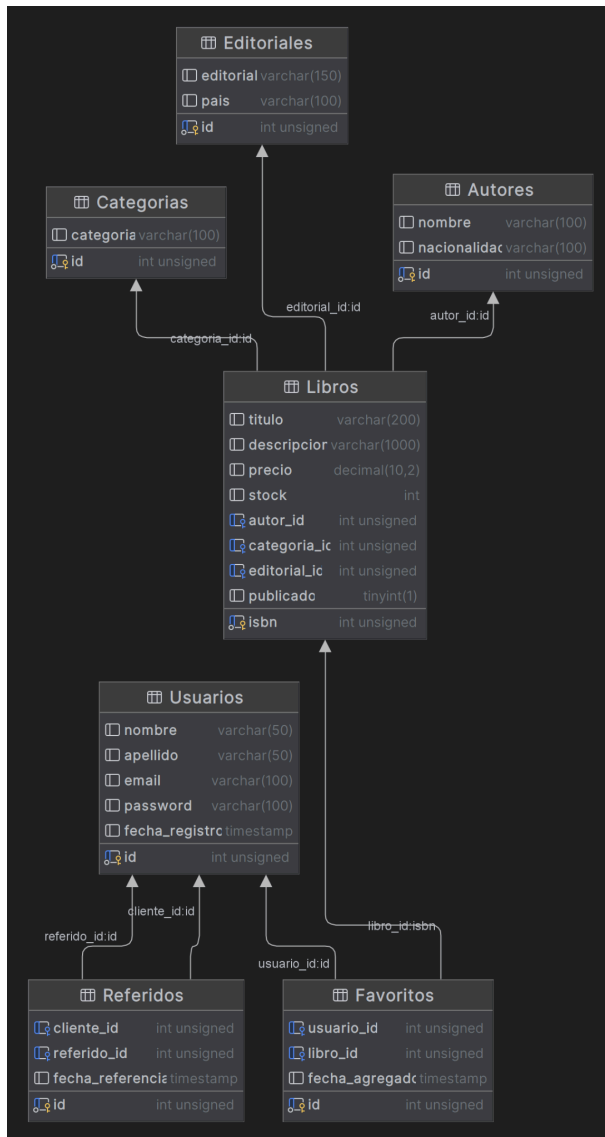
https://www-devoteam-com.translate.goog/expert-view/mongodb-for-data-management-handle-big-data-easily/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=rq#:~:text=MongoDB%20is%20a%20document%20database,enables%20efficient%20storage%20and%20retrieval.

Wisniowski, K. (2025, 4 junio). Redis® as Cache: How it Works and Why to Use it. Cloud Infrastructure Services.

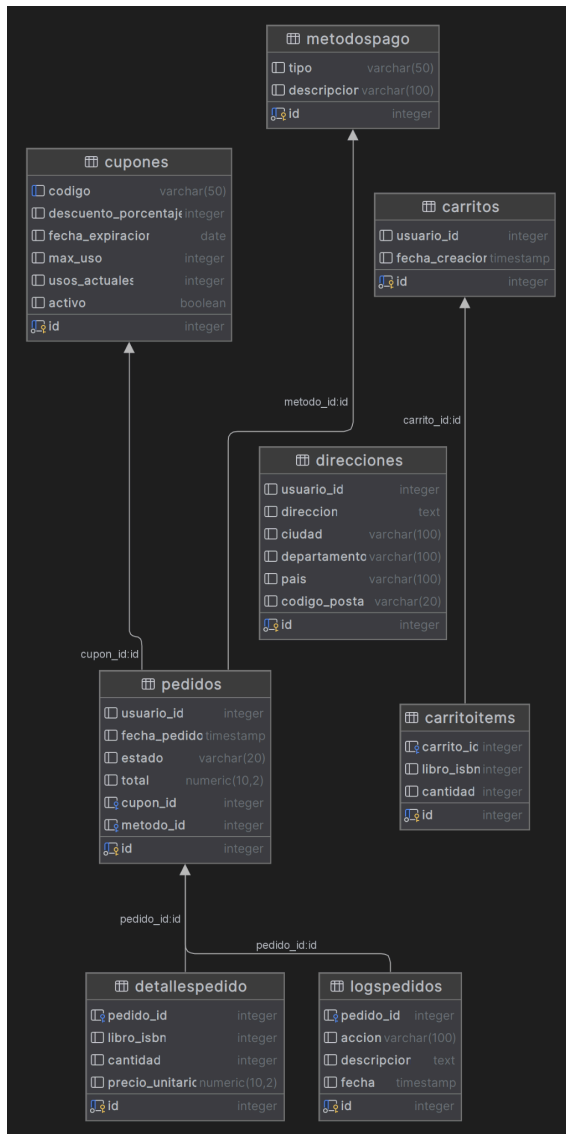
[https://cloudinfrastructureservices-co-uk.translate.goog/redis-as-cache-how-it-works-and-why-to-use-it/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=sge#:~:text=Pros%20y%20Contras\)-,Redis%C2%AE%20como%20cach%C3%A9:%20c%C3%B3mo%20funciona%20y%20por%20qu%C3%A9%20usarlo,est%C3%A1n%20reservados%20para%20Redis%20Ltd.](https://cloudinfrastructureservices-co-uk.translate.goog/redis-as-cache-how-it-works-and-why-to-use-it/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=sge#:~:text=Pros%20y%20Contras)-,Redis%C2%AE%20como%20cach%C3%A9:%20c%C3%B3mo%20funciona%20y%20por%20qu%C3%A9%20usarlo,est%C3%A1n%20reservados%20para%20Redis%20Ltd.)

Anexos

Anexo 1: Diagrama MySQL exportado de DataGrip



Anexo 2: Diagrama PostgreSQL exportado de DataGrip



Anexo 3: Diagrama de ETL

