

[Ver script de SQL](#)
[Ver Sprint 4 en PDF](#)



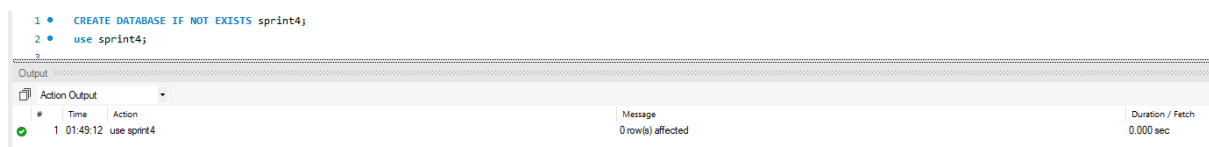
Nivel 1

Diseña una base de datos con un esquema de estrella que contenga, al menos 4 tablas de las que puedas realizar las siguientes consultas:

Sql de Base de Datos sprint4

-- Creación base de datos sprint4

```
CREATE DATABASE IF NOT EXISTS sprint4;  
use sprint4;
```



Creacion de Tablas: Transacciones, usuarios, producto y tarjeta de credito.

Tabla Fact (hechos): transacciones (la tabla principal que contiene las ventas/operaciones). Tablas Dimensión: usuarios, productos, y tarjetas_credito.

Sql de Tabla Transactions

-- Creación de la tabla Transactions

```
CREATE TABLE transactions (  
  id VARCHAR(50) NOT NULL PRIMARY KEY, -- Identificador único  
  card_id VARCHAR(50) NULL,           -- Puede ser NULL si no hay tarjeta asociada  
  business_id VARCHAR(50) NULL,       -- Puede ser NULL si no hay negocio asociado  
  timestamp TIMESTAMP NOT NULL,       -- Marca de tiempo obligatoria  
  amount DECIMAL(10, 2) NOT NULL,     -- El monto de la transacción no debe ser NULL  
  declined BOOLEAN NOT NULL,          -- Indica si la transacción fue rechazada  
  product_ids TEXT NULL,              -- Puede ser NULL si no hay productos asociados  
  user_id INT NULL,                  -- Puede ser NULL si el usuario no está identificado  
  lat DECIMAL(15, 10) NULL,          -- Puede ser NULL si la ubicación no está disponible  
  longitude DECIMAL(15, 10) NULL     -- Puede ser NULL si la ubicación no está disponible  
);
```

```

3 • CREATE TABLE transactions (
4     id VARCHAR(50) NOT NULL PRIMARY KEY, -- Identificador único
5     card_id VARCHAR(50) NULL,           -- Puede ser NULL si no hay tarjeta asociada
6     business_id VARCHAR(50) NULL,       -- Puede ser NULL si no hay negocio asociado
7     timestamp TIMESTAMP NOT NULL,       -- Marca de tiempo obligatoria
8     amount DECIMAL(10, 2) NOT NULL,     -- El monto de la transacción no debe ser NULL
9     declined BOOLEAN NOT NULL,          -- Indica si la transacción fue rechazada
10    product_ids TEXT NULL,               -- Puede ser NULL si no hay productos asociados
11    user_id INT NULL,                    -- Puede ser NULL si el usuario no está identificado
12    lat DECIMAL(15, 10) NULL,            -- Puede ser NULL si la ubicación no está disponible
13    longitude DECIMAL(15, 10) NULL      -- Puede ser NULL si la ubicación no está disponible
14 );
15

```

Output			
Action Output			
#	Time	Action	Message
✓ 1	09:48:04	CREATE TABLE transactions (id VARCHAR(50) NOT NULL PRIMARY KEY, -- Identificador único card_id VARC...	0 row(s) affected

Sql de Tabla companies

-- Creación de la tabla companies

```

CREATE TABLE companies (
    company_id VARCHAR(150) NOT NULL PRIMARY KEY, -- Identificador único de la compañía
    company_name VARCHAR(100) NOT NULL,          -- Nombre de la compañía es obligatorio
    phone VARCHAR(20) NULL,                       -- Puede ser NULL si no hay teléfono disponible
    email VARCHAR(100) NULL,                       -- Puede ser NULL si no hay email disponible
    country VARCHAR(50) NULL,                      -- Puede ser NULL si no hay país disponible
    website VARCHAR(255) NULL                      -- Puede ser NULL si no hay sitio web disponible
);

```

```

16 • CREATE TABLE companies (
17     company_id VARCHAR(6) NOT NULL PRIMARY KEY, -- Identificador único de la compañía
18     company_name VARCHAR(100) NOT NULL,         -- Nombre de la compañía es obligatorio
19     phone VARCHAR(20) NULL,                     -- Puede ser NULL si no hay teléfono disponible
20     email VARCHAR(100) NULL,                    -- Puede ser NULL si no hay email disponible
21     country VARCHAR(50) NULL,                   -- Puede ser NULL si no hay país disponible
22     website VARCHAR(255) NULL                   -- Puede ser NULL si no hay sitio web disponible
23 );
24

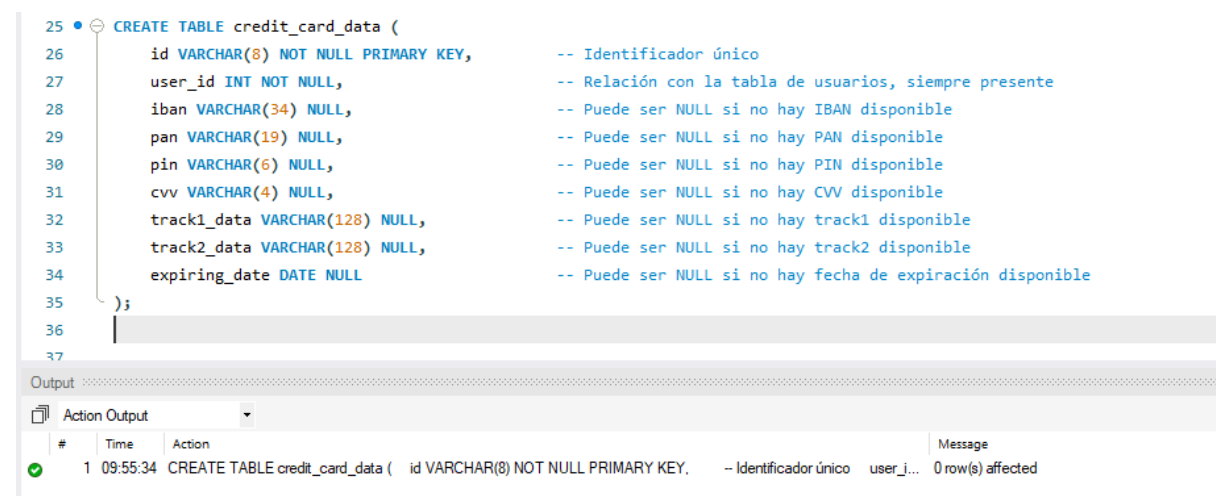
```

Output			
Action Output			
#	Time	Action	Message
✓ 1	09:54:12	CREATE TABLE companies (company_id VARCHAR(6) NOT NULL PRIMARY KEY, -- Identificador único de la co...	0 row(s) affected

Sql de Tabla credit_card_data

-- Creación de la tabla credit_card_data

```
CREATE TABLE credit_card_data (  
    id VARCHAR(8) NOT NULL PRIMARY KEY,      -- Identificador único  
    user_id INT NOT NULL,                    -- Relación con la tabla de usuarios, siempre presente  
    iban VARCHAR(34) NULL,                  -- Puede ser NULL si no hay IBAN disponible  
    pan VARCHAR(19) NULL,                   -- Puede ser NULL si no hay PAN disponible  
    pin VARCHAR(6) NULL,                    -- Puede ser NULL si no hay PIN disponible  
    cvv VARCHAR(4) NULL,                    -- Puede ser NULL si no hay CVV disponible  
    track1_data VARCHAR(128) NULL,          -- Puede ser NULL si no hay track1 disponible  
    track2_data VARCHAR(128) NULL,          -- Puede ser NULL si no hay track2 disponible  
    expiring_date DATE NULL                 -- Puede ser NULL si no hay fecha de expiración disponible  
);
```



The screenshot shows a SQL IDE with a query editor and an output window. The query editor contains the SQL code for creating the credit_card_data table, with line numbers 25 to 37. The output window shows the execution result of the CREATE TABLE statement, indicating that 0 rows were affected.

```
25 CREATE TABLE credit_card_data (  
26     id VARCHAR(8) NOT NULL PRIMARY KEY,      -- Identificador único  
27     user_id INT NOT NULL,                    -- Relación con la tabla de usuarios, siempre presente  
28     iban VARCHAR(34) NULL,                  -- Puede ser NULL si no hay IBAN disponible  
29     pan VARCHAR(19) NULL,                   -- Puede ser NULL si no hay PAN disponible  
30     pin VARCHAR(6) NULL,                    -- Puede ser NULL si no hay PIN disponible  
31     cvv VARCHAR(4) NULL,                    -- Puede ser NULL si no hay CVV disponible  
32     track1_data VARCHAR(128) NULL,          -- Puede ser NULL si no hay track1 disponible  
33     track2_data VARCHAR(128) NULL,          -- Puede ser NULL si no hay track2 disponible  
34     expiring_date DATE NULL                 -- Puede ser NULL si no hay fecha de expiración disponible  
35 );  
36  
37
```

Output

#	Time	Action	Message
1	09:55:34	CREATE TABLE credit_card_data (id VARCHAR(8) NOT NULL PRIMARY KEY, -- Identificador único user_id...	0 row(s) affected

Sql de Tabla users

-- Creacion de la tabla users

```
CREATE TABLE users (  
    id INT NOT NULL PRIMARY KEY,            -- Identificador único  
    name VARCHAR(100) NOT NULL,             -- El nombre del usuario no puede ser NULL  
    surname VARCHAR(100) NOT NULL,          -- El apellido del usuario no puede ser NULL  
    phone VARCHAR(50) NULL,                 -- Puede ser NULL si no hay teléfono disponible  
    email VARCHAR(150) NULL,                -- Puede ser NULL si no hay email disponible  
    birth_date DATE NULL,                   -- Puede ser NULL si no hay fecha de nacimiento disponible  
    country VARCHAR(50) NULL,               -- Puede ser NULL si no hay país disponible  
    city VARCHAR(100) NULL,                 -- Puede ser NULL si no hay ciudad disponible  
    postal_code VARCHAR(20) NULL,           -- Puede ser NULL si no hay código postal disponible  
    address VARCHAR(255) NULL               -- Puede ser NULL si no hay dirección disponible  
);
```

```

37 • CREATE TABLE users (
38     id INT NOT NULL PRIMARY KEY,           -- Identificador único
39     name VARCHAR(100) NOT NULL,             -- El nombre del usuario no puede ser NULL
40     surname VARCHAR(100) NOT NULL,          -- El apellido del usuario no puede ser NULL
41     phone VARCHAR(50) NULL,                 -- Puede ser NULL si no hay teléfono disponible
42     email VARCHAR(150) NULL,                -- Puede ser NULL si no hay email disponible
43     birth_date DATE NULL,                   -- Puede ser NULL si no hay fecha de nacimiento disponible
44     country VARCHAR(50) NULL,               -- Puede ser NULL si no hay país disponible
45     city VARCHAR(100) NULL,                 -- Puede ser NULL si no hay ciudad disponible
46     postal_code VARCHAR(20) NULL,           -- Puede ser NULL si no hay código postal disponible
47     address VARCHAR(255) NULL               -- Puede ser NULL si no hay dirección disponible
48 );

```

Output

#	Time	Action	Message
1	09:57:09	CREATE TABLE users (id INT NOT NULL PRIMARY KEY, -- Identificador único name VARCHAR(100) N...	0 row(s) affected

Sql de Tabla products

-- Creacion de la tabla products

```

CREATE TABLE products (
    id INT NOT NULL PRIMARY KEY,           -- Identificador único
    product_name VARCHAR(255) NOT NULL,    -- El nombre del producto no puede ser NULL
    price DECIMAL(10, 2) NOT NULL,         -- El precio no puede ser NULL
    color VARCHAR(7) NULL,                 -- Puede ser NULL si no hay color disponible
    weight DECIMAL(5, 2) NULL,             -- Puede ser NULL si no hay peso disponible
    warehouse_id VARCHAR(10) NULL          -- Puede ser NULL si no hay almacén asociado
);

```

```

50 • CREATE TABLE products (
51     id INT NOT NULL PRIMARY KEY,           -- Identificador único
52     product_name VARCHAR(255) NOT NULL,    -- El nombre del producto no puede ser NULL
53     price DECIMAL(10, 2) NOT NULL,         -- El precio no puede ser NULL
54     color VARCHAR(7) NULL,                 -- Puede ser NULL si no hay color disponible
55     weight DECIMAL(5, 2) NULL,             -- Puede ser NULL si no hay peso disponible
56     warehouse_id VARCHAR(10) NULL          -- Puede ser NULL si no hay almacén asociado
57 );

```

Output

#	Time	Action	Message
1	10:00:51	CREATE TABLE products (id INT NOT NULL PRIMARY KEY, -- Identificador único product_name VARC...	0 row(s) affected

Importacion de datos Transactions companies companies credit_card_data users products

Configurar la carpeta C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/ en MySQL y copiar los archivos necesarios para la carga de datos.

Paso 1: Ubicar la carpeta de donde se importa por inercia

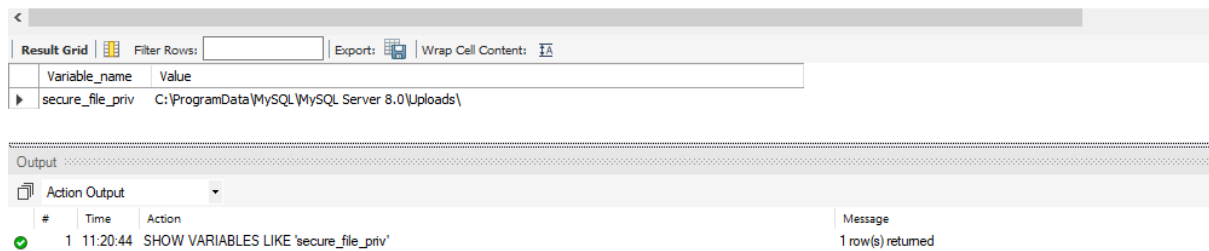
- MySQL 8.0: Asegúrate de que MySQL 8.0 está correctamente instalado en tu sistema.
- Archivos CSV: Los archivos de datos a cargar deben estar disponibles en formato CSV.

SHOW VARIABLES LIKE 'secure_file_priv';

Para saber la ubicación de la carpeta C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\ donde ir los archivos csv para ser importados

La carpeta Uploads de MySQL suele encontrarse en la siguiente ubicación en sistemas Windows:

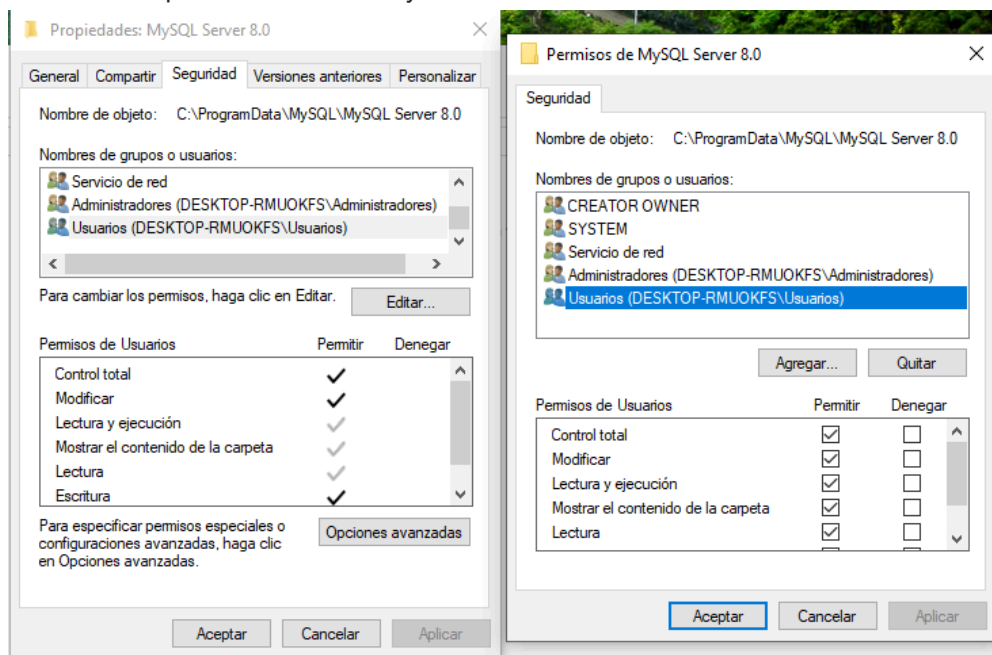
```
1 • SHOW VARIABLES LIKE 'secure_file_priv';
```



Paso 2: Configuración de permisos la carpeta

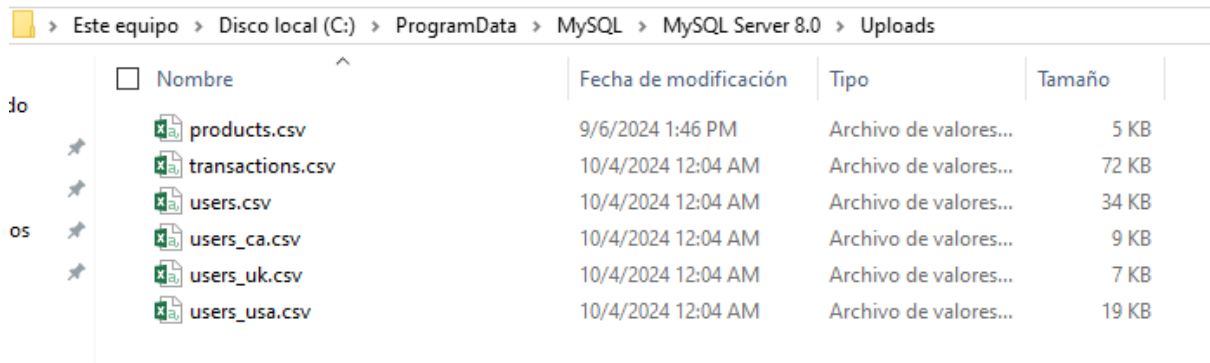
Asegúrate de que el servicio MySQL tiene permisos adecuados para acceder y leer archivos en la carpeta Uploads.

1. Haz clic derecho en la carpeta Uploads.
2. Selecciona Propiedades y ve a la pestaña Seguridad.
3. Asegúrate de que el usuario que ejecuta MySQL (normalmente NT AUTHORITY\SYSTEM o mysql) tiene permisos de lectura y escritura.



Paso 3: Copiar y pegar los archivos csv en la carpeta Upload

1. Abre el explorador de archivos y navega a C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/.
2. Copia los archivos CSV que deseas cargar desde tu ubicación original a la carpeta Uploads.



Este equipo > Disco local (C:) > ProgramData > MySQL > MySQL Server 8.0 > Uploads					
	Nombre	Fecha de modificación	Tipo	Tamaño	
do	products.csv	9/6/2024 1:46 PM	Archivo de valores...	5 KB	
	transactions.csv	10/4/2024 12:04 AM	Archivo de valores...	72 KB	
	users.csv	10/4/2024 12:04 AM	Archivo de valores...	34 KB	
	users_ca.csv	10/4/2024 12:04 AM	Archivo de valores...	9 KB	
	users_uk.csv	10/4/2024 12:04 AM	Archivo de valores...	7 KB	
	users_usa.csv	10/4/2024 12:04 AM	Archivo de valores...	19 KB	

Paso 3: Reiniciar MySQL desde Windows Power Shell como administrador

1. Abre Windows PowerShell como administrador.
2. Detén el servicio MySQL con el comando:

```
net stop Mysql80
```

3. Inicia el servicio MySQL nuevamente con el comando:

```
net start Mysql80
```

Esto reinicia MySQL para aplicar cualquier cambio en la configuración.

```
Administrador: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\WINDOWS\system32> net stop MySQL80
El servicio de MySQL80 está deteniéndose.
El servicio de MySQL80 se detuvo correctamente.

PS C:\WINDOWS\system32> net start MySQL80
El servicio de MySQL80 está iniciándose.....
El servicio de MySQL80 se ha iniciado correctamente.

PS C:\WINDOWS\system32>
```

Importacion de datos a la tabla Transactions

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv' ---- Especifica que se está cargando datos desde un archivo

INTO TABLE transactions ----- Define que los datos se insertarán en la tabla transactions.

FIELDS TERMINATED BY ',' ----- Especifica que los campos en el archivo CSV están separados por punto y coma (;).

ENCLOSED BY '''' ----- Indica que los valores de los campos
están encerrados entre comillas dobles (").
LINES TERMINATED BY '\n' -----Indica que cada línea en el archivo está
separada por un salto de línea (\n)
IGNORE 1 LINES ----- ignorar la primera línea del archivo, que
contiene los encabezados de las columnas.
(id, card_id, business_id, timestamp, amount, declined, product_ids, user_id, lat, longitude); -----Mapea las
columnas del archivo CSV a las columnas correspondientes de la tabla

```

3 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv'
4 INTO TABLE transactions
5 FIELDS TERMINATED BY ','
6 ENCLOSED BY ''''
7 LINES TERMINATED BY '\n'
8 IGNORE 1 LINES
9 (id, card_id, business_id, timestamp, amount, declined, product_ids, user_id, lat, longitude);

```

Output

#	Time	Action	Message
1	10:57:14	use sprint4	0 row(s) affected
2	10:58:32	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv' INTO TABLE transaction...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0

Importacion de datos a la tabla Companies

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
ENCLOSED BY ''''
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(company_id, company_name, phone, email, country, website);

```

3 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv'
4 INTO TABLE companies
5 FIELDS TERMINATED BY ','
6 ENCLOSED BY ''''
7 LINES TERMINATED BY '\n'
8 IGNORE 1 LINES
9 (company_id, company_name, phone, email, country, website);

```

Output

#	Time	Action	Message	Duration / Fetch
1	17:23:47	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv' INTO TAB...	Error Code: 1261. Row 1 doesn't contain data for all columns	0.000 sec
2	17:25:39	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv' INTO TAB...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0	0.157 sec

Importación de datos a la tabla credit_card_data

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_card_data.csv'
INTO TABLE credit_card_data
FIELDS TERMINATED BY ','
ENCLOSED BY ''''
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(id, user_id, iban, pan, pin, cvv, track1_data, track2_data, @expiring_date)
SET expiring_date = STR_TO_DATE(@expiring_date, '%m/%d/%y');


```

3 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_card_data.csv'
4 INTO TABLE credit_card_data
5 FIELDS TERMINATED BY ','
6 ENCLOSED BY '"'
7 LINES TERMINATED BY '\n'
8 IGNORE 1 LINES
9 (id, user_id, iban, pan, pin, cvv, track1_data, track2_data, @expiring_date)
10 SET expiring_date = STR_TO_DATE(@expiring_date, '%m/%d/%y');
11

```

#	Time	Action	Message	Duration / Fetch
1	17:43:22	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_card_data.csv' INTO TABLE credit_card_data	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0	0.453 sec

Importacion de datos a la tabla Products

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv'
INTO TABLE products
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(id, name, surname, phone, email, birth_date, country, city, postal_code, address);

```

```

3 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv'
4 INTO TABLE products
5 FIELDS TERMINATED BY ',' ENCLOSED BY '"'
6 LINES TERMINATED BY '\n'
7 IGNORE 1 LINES
8 (id, product_name, @price, color, weight, warehouse_id)
9 SET price = REPLACE(@price, '$', '');
10
11

```

#	Time	Action	Message	Duration / Fetch
1	18:09:10	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv' INTO TABLE products	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0	0.171 sec

Importación de datos a la tabla Users

- Se utilizo para python para retirar las "" y las comillas en algunos registros de la columna address
- Se utilizo para python para darle formato %d-%b-%y a los registros de la columna birth_date
- Luego se procedio a importar los datos a la tabla Users.
-

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa_cleaned.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(id, name, surname, phone, email, birth_date, country, city, postal_code, address);

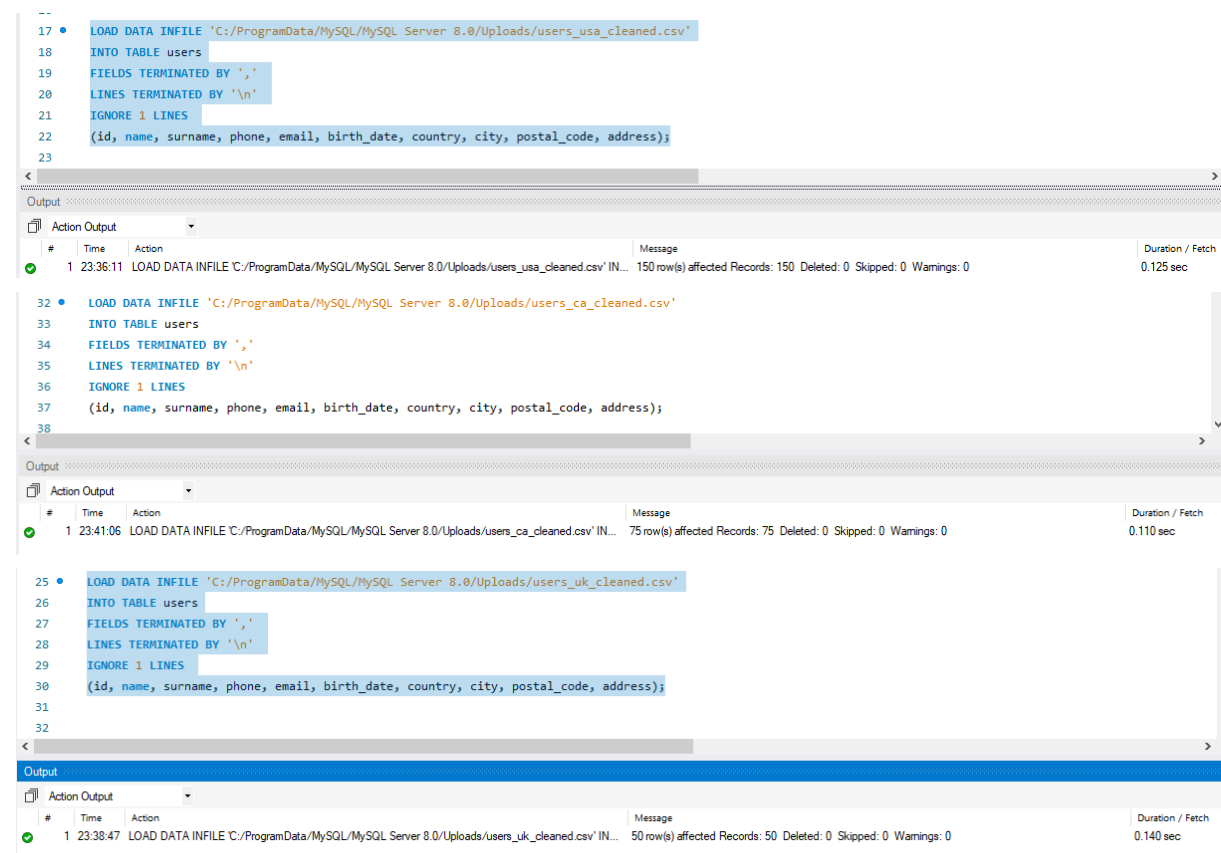
```

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk_cleaned.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(id, name, surname, phone, email, birth_date, country, city, postal_code, address);

```

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca_cleaned.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(id, name, surname, phone, email, birth_date, country, city, postal_code, address);
```



The screenshot shows a MySQL command window with three SQL commands and their corresponding output messages.

Command 1:

```
17 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa_cleaned.csv'
18 INTO TABLE users
19 FIELDS TERMINATED BY ','
20 LINES TERMINATED BY '\n'
21 IGNORE 1 LINES
22 (id, name, surname, phone, email, birth_date, country, city, postal_code, address);
23
```

Output 1:

#	Time	Action	Message	Duration / Fetch
1	23:36:11	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa_cleaned.csv' IN...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0	0.125 sec

Command 2:

```
32 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca_cleaned.csv'
33 INTO TABLE users
34 FIELDS TERMINATED BY ','
35 LINES TERMINATED BY '\n'
36 IGNORE 1 LINES
37 (id, name, surname, phone, email, birth_date, country, city, postal_code, address);
38
```

Output 2:

#	Time	Action	Message	Duration / Fetch
1	23:41:06	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca_cleaned.csv' IN...	75 row(s) affected Records: 75 Deleted: 0 Skipped: 0 Warnings: 0	0.110 sec

Command 3:

```
25 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk_cleaned.csv'
26 INTO TABLE users
27 FIELDS TERMINATED BY ','
28 LINES TERMINATED BY '\n'
29 IGNORE 1 LINES
30 (id, name, surname, phone, email, birth_date, country, city, postal_code, address);
31
32
```

Output 3:

#	Time	Action	Message	Duration / Fetch
1	23:38:47	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk_cleaned.csv' IN...	50 row(s) affected Records: 50 Deleted: 0 Skipped: 0 Warnings: 0	0.140 sec

Esquema estrella de la base de datos

-- Alterar a las tablas agregando las relaciones

darle esquema de estrella

-- Agregar clave foránea para `user_id` que referencia a la tabla `users`

ALTER TABLE transactions

ADD CONSTRAINT fk_user

FOREIGN KEY (user_id) REFERENCES users(id);

-- Agregar clave foránea para `card_id` que referencia a la tabla `credit_card_data`

ALTER TABLE transactions

ADD CONSTRAINT fk_card

FOREIGN KEY (card_id) REFERENCES credit_card_data(id);

-- Agregar clave foránea para `business_id` que referencia a la tabla `companies`

ALTER TABLE transactions

ADD CONSTRAINT fk_business

FOREIGN KEY (business_id) REFERENCES companies(company_id);

```

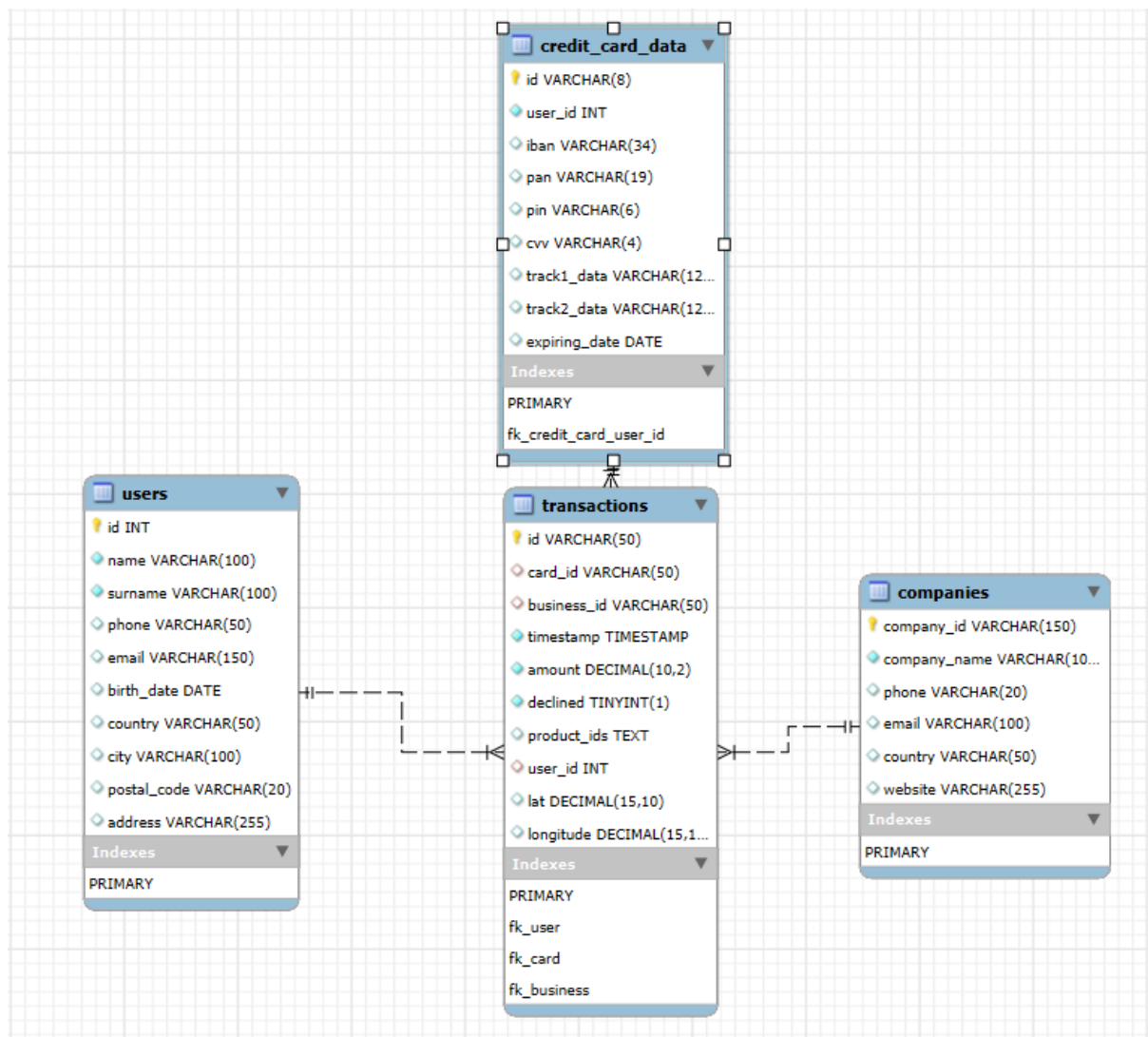
146 -- Agregar clave foránea para 'user_id' que referencia a la tabla 'users'
147 • ALTER TABLE transactions
148   ADD CONSTRAINT fk_user
149   FOREIGN KEY (user_id) REFERENCES users(id);
150
151 -- Agregar clave foránea para 'card_id' que referencia a la tabla 'credit_card_data'
152 • ALTER TABLE transactions
153   ADD CONSTRAINT fk_card
154   FOREIGN KEY (card_id) REFERENCES credit_card_data(id);
155
156 -- Agregar clave foránea para 'business_id' que referencia a la tabla 'companies'
157 • ALTER TABLE transactions
158   ADD CONSTRAINT fk_business
159   FOREIGN KEY (business_id) REFERENCES companies(company_id);
160

```

Output

#	Time	Action	Message	Duration / Fetch
1	00:22:34	ALTER TABLE transactions ADD CONSTRAINT fk_user FOREIGN KEY (user_id) REFERENCES users(id);	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	3.297 sec
2	00:22:38	ALTER TABLE transactions ADD CONSTRAINT fk_card FOREIGN KEY (card_id) REFERENCES credit_card_data(id);	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	3.485 sec
3	00:22:41	ALTER TABLE transactions ADD CONSTRAINT fk_business FOREIGN KEY (business_id) REFERENCES companies(company_id);	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	2.859 sec

ER de la base de datos



Ejercicio 1:

Realiza una subconsulta que muestre a todos los usuarios con más de 30 transacciones utilizando al menos 2 tablas.

Sql de tabla Usuarios con más de 30 transacciones

```
-- Usuarios con más de 30 transacciones
SELECT u.id, u.name, u.surname, COUNT(t.id) AS transaction_count
FROM users u
JOIN transactions t ON u.id = t.user_id
GROUP BY u.id, u.name, u.surname
HAVING COUNT(t.id) > 30
LIMIT 0, 5000;
```

Explicación del Sql

El objetivo de la consulta es obtener una lista de usuarios que han realizado más de 30 transacciones. Para ello, se cuenta la cantidad de transacciones de cada usuario, y solo aquellos usuarios que cumplen con este criterio son incluidos en los resultados. Además, se limita el resultado a un máximo de 100 registros.

Paso a paso:

1. Inicio de la consulta
 - Definir las columnas a seleccionar:
 - u.id: el identificador único del usuario.
 - u.name: el nombre del usuario.
 - u.surname: el apellido del usuario.
 - COUNT(t.id): el conteo de transacciones realizadas por ese usuario, usando el campo id de la tabla transactions.
2. SELECT u.id, u.name, u.surname, COUNT(t.id) AS transaction_count
3. Especificar la tabla base (tabla users)
 - Utilizar la tabla users como la tabla principal, a la que se le hará uniones para obtener más datos relacionados.
4. FROM users u
5. Realizar la unión (JOIN) con la tabla transactions
 - Unir la tabla transactions con la tabla users mediante una relación en la cual el user_id en la tabla transactions es igual al id del usuario en la tabla users.
6. JOIN transactions t ON u.id = t.user_id
7. Agrupar los datos por usuario (GROUP BY)
 - Agrupar los resultados para que cada fila en el resultado corresponda a un usuario único. Esto nos permitirá contar las transacciones de cada usuario.
8. GROUP BY u.id, u.name, u.surname
9. Filtrar usuarios con más de 30 transacciones (HAVING)
 - Usar la cláusula HAVING para filtrar a aquellos usuarios cuyo número de transacciones (contadas en el paso anterior) sea mayor a 30.
10. HAVING COUNT(t.id) > 30
11. Limitar el número de resultados (LIMIT)

- Limitar el número de resultados a 5000, comenzando desde el primer resultado. Esto mejora el rendimiento y evita sobrecargar el sistema si hay muchos usuarios.

12. LIMIT 0, 100;

The screenshot shows a SQL query execution interface. The query is as follows:

```

136 • SELECT u.id, u.name, u.surname, COUNT(t.id) AS total_transactions
137 FROM users u
138 JOIN transactions t ON u.id = t.user_id
139 GROUP BY u.id, u.name, u.surname
140 HAVING COUNT(t.id) > 30
141 LIMIT 0, 100;

```

The results are displayed in a table with the following columns: id, name, surname, total_transactions. The data shown is:

id	name	surname	total_transactions
92	Lynn	Riddle	39
267	Ocean	Nelson	52
272	Hedvig	Gilbert	76
275	Kenyon	Hartman	48

Below the table, the output section shows the execution details:

```

# Time Action Message Duration / Fetch
1 01:11:44 SELECT u.id, u.name, u.surname, COUNT(t.id) AS total_transactions FROM users u JOIN transactions... 4 row(s) returned 0.000 sec / 0.000 sec

```

Ejercicio 2:

Muestra la media de amount por IBAN de las tarjetas de crédito en la compañía Donec Ltd., utiliza por lo menos 2 tablas.

Sql de tabla Usuarios con más de 30 transacciones

-- media de amount por IBAN de las tarjetas de crédito en la compañía Donec Ltd

```

SELECT co.company_name,
       cc.iban,
       ROUND(AVG(t.amount), 2) AS promedio_suma_transacciones
FROM companies co
INNER JOIN transactions t ON co.company_id = t.business_id
INNER JOIN credit_card_data cc ON t.card_id = cc.id
WHERE co.company_name = 'Donec Ltd'
GROUP BY cc.iban;

```

Explicación del SQL

Paso a paso:

1. SELECT co.company_name, cc.iban, ROUND(AVG(t.amount), 2) AS promedio_suma_transacciones:
 - Objetivo: Seleccionar los siguientes campos:
 - co.company_name: El nombre de la compañía.
 - cc.iban: El IBAN de las tarjetas de crédito.
 - ROUND(AVG(t.amount), 2): Calcular el promedio de los montos (amount) de las transacciones, redondeado a 2 decimales.
2. FROM companies co:
 - Objetivo: Se define la tabla principal companies y se le asigna el alias co.
3. INNER JOIN transactions t ON co.company_id = t.business_id:

- Objetivo: Realizar un INNER JOIN entre las tablas companies y transactions, donde company_id en companies coincida con business_id en transactions. Esto permite relacionar las transacciones con las compañías.
- 4. INNER JOIN credit_card_data cc ON t.card_id = cc.id:
 - Objetivo: Realizar un INNER JOIN entre las tablas transactions y credit_card_data, usando t.card_id = cc.id. Esto permite acceder a los datos de las tarjetas de crédito, en particular al IBAN.
- 5. WHERE co.company_name = 'Donec Ltd':
 - Objetivo: Filtrar las filas para que solo se incluyan las transacciones que están asociadas con la compañía cuyo nombre es 'Donec Ltd'.
- 6. GROUP BY cc.iban:
 - Objetivo: Agrupar los resultados por el IBAN de las tarjetas de crédito (cc.iban). Cada grupo representa un conjunto de transacciones realizadas con el mismo IBAN.

```
178 • SELECT co.company_name,
179         cc.iban,
180         ROUND(AVG(t.amount), 2) AS promedio_suma_transacciones
181 FROM companies co
182 INNER JOIN transactions t ON co.company_id = t.business_id
183 INNER JOIN credit_card_data cc ON t.card_id = cc.id
184 WHERE co.company_name = 'Donec Ltd'
185 GROUP BY cc.iban;
```

company_name	iban	promedio_suma_transacciones
Donec Ltd	PT87806228135092429456346	203.72

Result 17 x

Output

#	Time	Action	Message	Duration / Fetch
1	01:53:44	SELECT co.company_name, cc.iban, ROUND(AVG(t.amount), 2) AS promedio_suma_transa...	1 row(s) returned	0.031 sec / 0.000 sec

Author

Nombre:

Yimmy Beltran

Información de Contacto

- LinkedIn: <https://www.linkedin.com/in/gianmarco-beltran-13959b232/>
- GitHub: <https://github.com/ciberzerone>
- Correo Electrónico: gianmarcobeltran@gmail.com