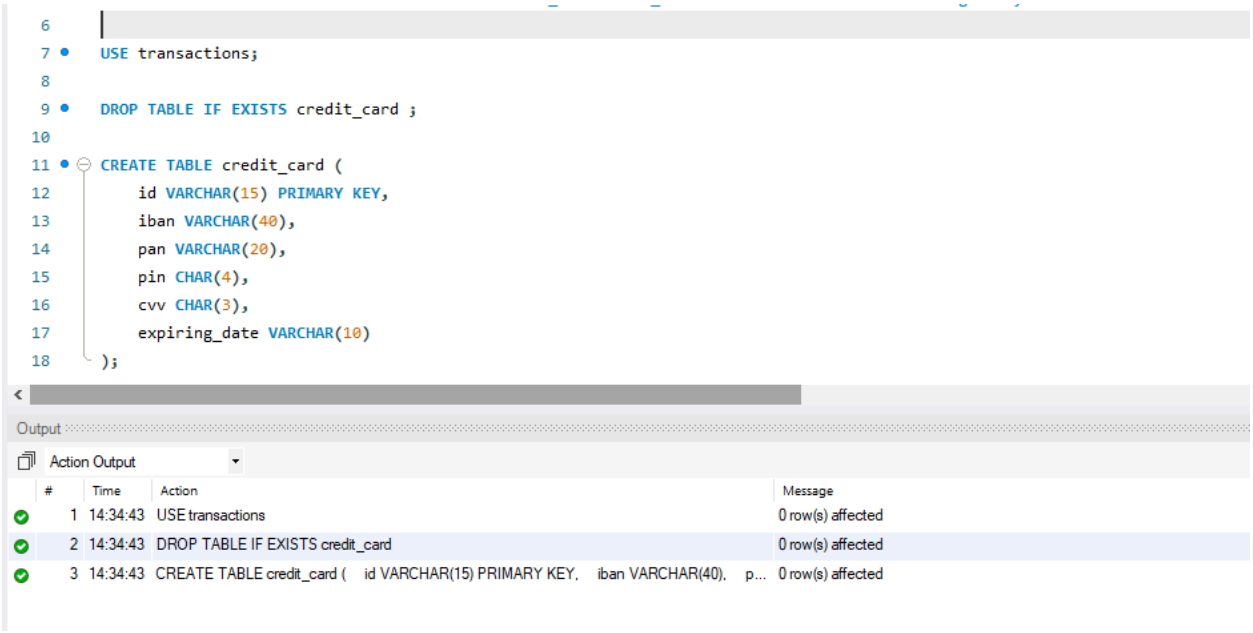
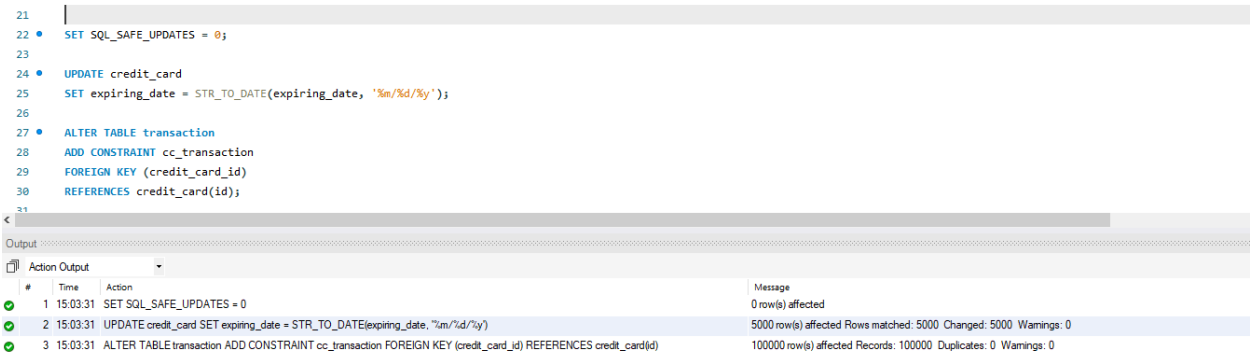
★☆☆ Nivel 1

Ejercicio 1

Tu tarea es diseñar y crear una tabla llamada "credit_card" que almacene detalles cruciales sobre las tarjetas de crédito. La nueva tabla debe ser capaz de identificar de forma única cada tarjeta y establecer una relación adecuada con las otras dos tablas ("transaction" y "company"). Después de crear la tabla será necesario que ingreses la información del documento denominado "datos_introducir_credit". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

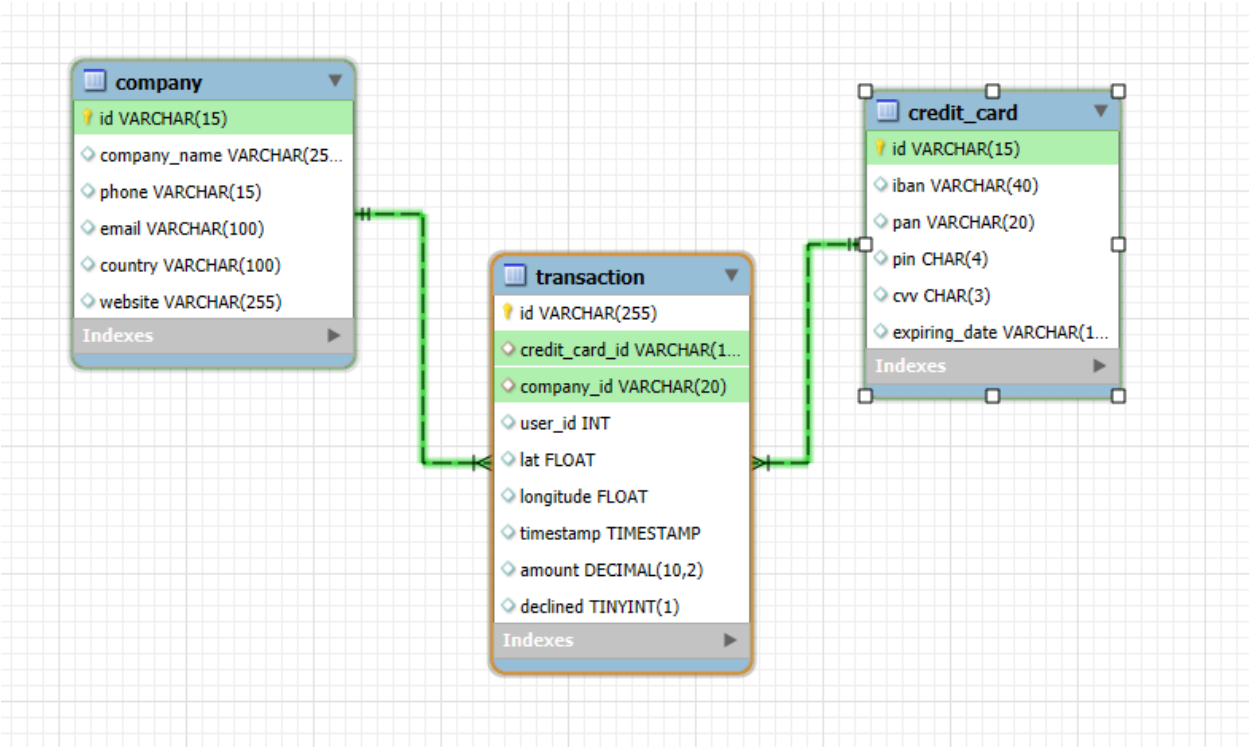


Explicación código: usando la BBDD transactions cree la tabla credit_card con sus columnas, formato y longitudes de las mismas estableciendo como PRIMARY KEY id. Como habia que hacer varias comprobaciones agregue el código DROP TABLE IF EXISTS. Luego ingrese la información del documento "datos_introducir_credit". En pin y cvv utilice CHAR ya que este se usa para registros de longitud fija y es menos pesado que VARCHAR para el motor y por ende más eficiente.



Explicación código: para poder actualizar la columna expiring_date tuve que desactivar las restricciones a las actualizaciones con SET SQL_SAFE_UPDATES = 0. Luego con un UPDATE actualice el formato de la columna al

formato estándar de date. Por último creamos la relación entre transaction y credit_card con fk credit_card_id y nombrandola como cc_transaction.



company(1) ————————< (N) transaction (N) >———————— (1) credit_card

El modelo de base de datos representa un sistema de transacciones financieras en el que se registran operaciones realizadas mediante tarjetas de crédito y asociadas a empresas. El diseño se compone de tres tablas principales: credit_card, company y transaction.

La tabla central del modelo es transaction, que funciona como una entidad dependiente que relaciona las tarjetas de crédito utilizadas y las empresas involucradas en cada operación.

Relaciones entre tablas

credit_card → transaction

- Relación: Uno a muchos.
- Descripción: una tarjeta de crédito puede estar asociada a múltiples transacciones.
- Clave foránea: transaction.credit_card_id hace referencia a credit_card.id.
- Rol:
    - credit_card: Tabla padre
    - transaction: Tabla hija

company → transaction

- Relación: Uno a muchos
- Descripción: Una empresa puede estar asociada a múltiples transacciones.
- Clave foránea: transaction.company_id hace referencia a company.id.
- Rol:
    - company: Tabla padre
    - transaction: Tabla hija

Ejercicio 2

El departamento de Recursos Humanos ha identificado un error en el número de cuenta asociado a su tarjeta de crédito con ID CcU-2938. La información que debe mostrarse para este registro es: TR323456312213576817699999. Recuerda mostrar que el cambio se realizó.

```
36 •  UPDATE credit_card
37    SET iban = 'TR323456312213576817699999'
38    WHERE id = 'CcU-2938';
39
40 •  SELECT iban
41    FROM credit_card
42    WHERE id = 'CcU-2938';
43
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

iban
TR323456312213576817699999

credit_card 1

Output

Action Output

#	Time	Action	Message
1	15:48:15	UPDATE credit_card SET iban = 'TR323456312213576817699999' WHERE id = 'CcU-2938'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0
2	15:48:15	SELECT iban FROM credit_card WHERE id = 'CcU-2938'	1 row(s) returned

Ejercicio 3

En la tabla "transaction" ingresa una nueva transacción con la siguiente información:

Id	108B1D1D-5B23-A76C-55EF-C568E49A99DD
credit_card_id	CcU-9999
company_id	b-9999
user_id	9999
lato	829.999
longitud	-117.999
amunt	111.11
declined	0

```
55 •  INSERT INTO credit_card (id)
56    VALUES ('CcU-9999');
57
58 •  INSERT INTO company (id)
59    VALUES ('b-9999');
60
61 •  INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
62    VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', 9999, 829.999, -117.999, 111.11, 0);
63
```

Output

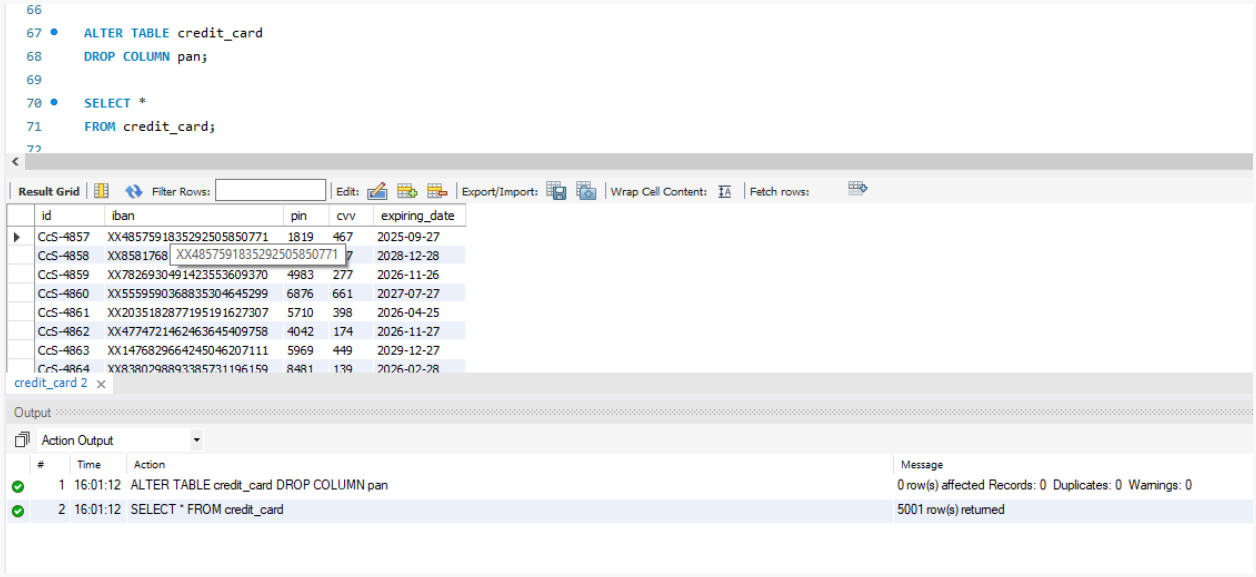
Action Output

#	Time	Action	Message
1	15:50:56	INSERT INTO credit_card (id) VALUES ('CcU-9999')	1 row(s) affected
2	15:50:56	INSERT INTO company (id) VALUES ('b-9999')	1 row(s) affected
3	15:50:56	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) VALUES ('108B1D1D-5B23-A76C-55EF-C5...	1 row(s) affected

Explicación código: aquí se pedía agregar una fila entera en transaction, y me pedían ingresar un nuevo registro tanto de company_id como de credit_card_id, que no existían en las tablas padres (company y credit_card) por lo que tuve que agregar esos valores antes de insertarlos en la tabla hija (transaction) para que no quedaran valores huérfanos.

Ejercicio 4

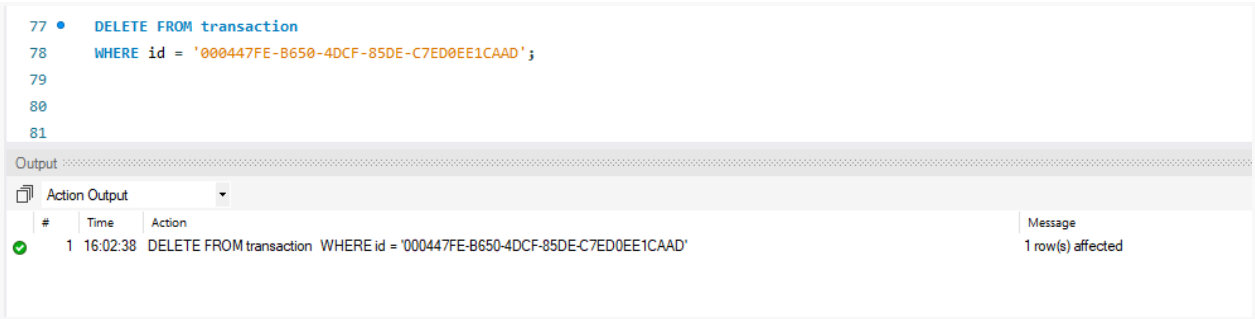
Desde recursos humanos te solicitan eliminar la columna "pan" de la tabla credit_card. Recuerda mostrar el cambio realizado.



★★☆ Nivel 2

Ejercicio 1

Elimina de la tabla transacción el registro con ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD de la base de datos.



Ejercicio 2

La sección de marketing desea tener acceso a información específica para realizar análisis y estrategias efectivas. Se ha solicitado crear una vista que proporcione detalles clave sobre las compañías y sus transacciones. Será necesaria que crees una vista llamada VistaMarketing que contenga la siguiente información: Nombre de la compañía. Teléfono de contacto. País de residencia. Media de compra realizado por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor promedio de compra.

```sql
CREATE VIEW VistaMarketing AS
SELECT c.company_name AS nombre_compania,
       c.phone AS telefono_contacto,
       c.country AS pais_residencia,
       AVG(t.amount) AS media_compra
FROM company c
JOIN transaction t ON c.id = t.company_id
GROUP BY c.id, c.company_name
ORDER BY media_compra DESC;

SELECT *
FROM VistaMarketing;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nombre_compania	telefono_contacto	pais_residencia	media_compra
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.867160
Pretium Neque Corp.	07 77 48 55 28	Australia	276.158330
Urna Convallis Associates	06 01 24 77 04	United States	274.235011
At Associates	09 56 61 10 65	New Zealand	272.214870
Metus Vitae Associates	08 25 44 40 66	Australia	270.080965
Aliquet Diam Limited	02 76 61 47 46	United States	269.599181

VistaMarketing 9

Output

Action Output

#	Time	Action	Message
1	16:16:42	CREATE VIEW VistaMarketing AS SELECT c.company_name AS nombre_compania, c.phone AS telefono_contacto, c.country AS ...	0 row(s) affected
2	16:16:42	SELECT * FROM VistaMarketing	101 row(s) returned

Explicación código: cree la vista con CREATE VIEW y le puse alias a las columnas para su mejor comprensión.

Ejercicio 3

Filtra la vista VistaMarketing para mostrar sólo las compañías que tienen su país de residencia en "Germany".

```sql
SELECT *
FROM VistaMarketing
WHERE pais_residencia = 'Germany';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nombre_compania	telefono_contacto	pais_residencia	media_compra
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.867160
Nunc Interdum Incorporated	05 18 15 48 13	Germany	259.319156
Convallis In Incorporated	06 66 57 29 50	Germany	257.745376
Ac Industries	09 34 65 40 60	Germany	255.147288
Rutrum Non Inc.	02 66 31 61 09	Germany	255.136927
Auctor Mauris Corp.	05 62 87 14 41	Germany	254.765518
Augue Foundation	06 88 43 15 63	Germany	253.505000
Aliquam PC	01 45 73 52 16	Germany	253.136923

VistaMarketing 10

Output

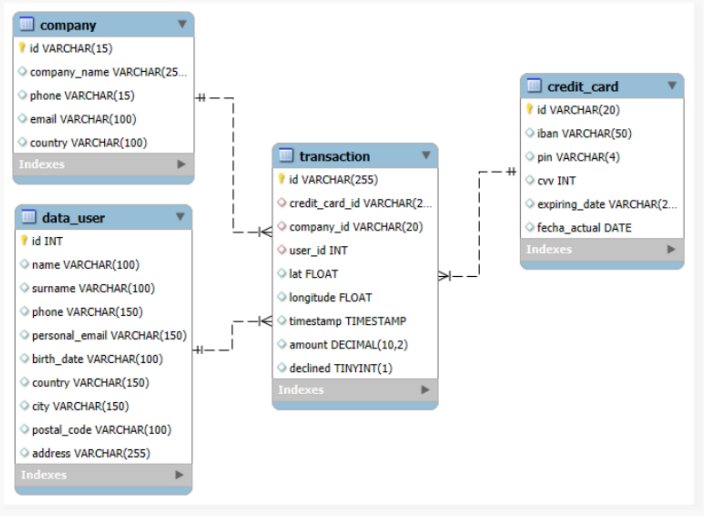
Action Output

#	Time	Action	Message
1	16:17:57	SELECT * FROM VistaMarketing WHERE pais_residencia = 'Germany'	8 row(s) returned

★★★ Nivel 3

Ejercicio 1

La próxima semana tendrás una nueva reunión con los gerentes de marketing. Un compañero de tu equipo realizó modificaciones en la base de datos, pero no recuerda cómo las realizó. Te pide que le ayudes a dejar los comandos ejecutados para obtener el siguiente diagrama:



Comandos ejecutados:

1) creamos tabla user y luego cargamos base de datos

```
116 •   DROP TABLE IF EXISTS user;
117 •   CREATE TABLE IF NOT EXISTS user (
118         id CHAR(10) PRIMARY KEY,
119         name VARCHAR(100),
120         surname VARCHAR(100),
121         phone VARCHAR(150),
122         email VARCHAR(150),
123         birth_date VARCHAR(100),
124         country VARCHAR(150),
125         city VARCHAR(150),
126         postal_code VARCHAR(100),
127         address VARCHAR(255)
128     );
129
```

Output

Action Output

#	Time	Action	Message
1	16:17:57	SELECT * FROM VistaMarketing WHERE pais_residencia = 'Germany'	8 row(s) returned
2	16:20:59	DROP TABLE IF EXISTS user	0 row(s) affected, 1 warning(s): 1051 Unknown table 'transactions.user'
3	16:20:59	CREATE TABLE IF NOT EXISTS user (id CHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARCHAR(15...	0 row(s) affected

2) corregimos datos distintos en este caso user.id CHAR lo cambiamos a INT como figura en la imagen así queda igual a la tabla transaction.

```
133 •   ALTER TABLE user
134     MODIFY COLUMN id INT;
135
```

Output

Action Output

#	Time	Action	Message
1	16:22:48	ALTER TABLE user MODIFY COLUMN id INT	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

3) renombramos la columna email por personal_email.

```
137 •  ALTER TABLE user
138    RENAME COLUMN email TO personal_email;
139
```

Output

Action Output

#	Time	Action	Message
1	16:25:03	ALTER TABLE user RENAME COLUMN email TO personal_email	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

4) cambiamos el nombre de la tabla a data_user.

```
141 •  RENAME TABLE user TO data_user;
142
```

Output

Action Output

#	Time	Action	Message
1	16:26:12	RENAME TABLE user TO data_user	0 row(s) affected

5) eliminamos la columna website.

```
144 •  ALTER TABLE company
145    DROP COLUMN website;
146
```

Output

Action Output

#	Time	Action	Message
1	16:27:18	ALTER TABLE company DROP COLUMN website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

6) en transaction cambiamos la longitud de los datos credit_card_id de 15 a 20.

```
148 •  ALTER TABLE transaction
149    MODIFY COLUMN credit_card_id VARCHAR(20);
```

Output

Action Output

#	Time	Action	Message
1	16:28:07	ALTER TABLE transaction MODIFY COLUMN credit_card_id VARCHAR(20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

7) en credit_card cambiamos la longitud de los datos id a VARCHAR(20), iban a VARCHAR(50), cambiamos el cvv a INT y expiring_date VARCHAR(20).

```
153 •  ALTER TABLE credit_card
154    MODIFY COLUMN id VARCHAR(20),
155    MODIFY COLUMN iban VARCHAR(50),
156    MODIFY COLUMN cvv INT,
157    MODIFY COLUMN expiring_date VARCHAR(20);
158
```

Output

Action Output

#	Time	Action	Message
1	16:29:01	ALTER TABLE credit_card MODIFY COLUMN id VARCHAR(20), MODIFY COLUMN iban VARCHAR(50), MODIFY COLUMN cvv INT, MODIF...	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

8) añadimos columna fecha_actual DATE.

```
160 •  ALTER TABLE credit_card
161    ADD COLUMN fecha_actual DATE;
162
```

Output

Action Output

#	Time	Action	Message
1	16:31:15	ALTER TABLE credit_card ADD COLUMN fecha_actual DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

9) agregamos dato huérfano faltante en tabla data_user.

```
173 •  INSERT INTO data_user (id)
174    VALUES (9999);
175
```

Output

Action Output

#	Time	Action	Message
1	16:45:11	ALTER TABLE transaction ADD CONSTRAINT user_transaction FOREIGN KEY (user_id) REFERENCES data_user(id)	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`transactions`.`#sql-154c_9`, CONSTRAINT `user_transaction`...
2	16:45:24	INSERT INTO data_user (id) VALUES (9999)	1 row(s) affected

10) relacionamos tablas data_user y transaction.

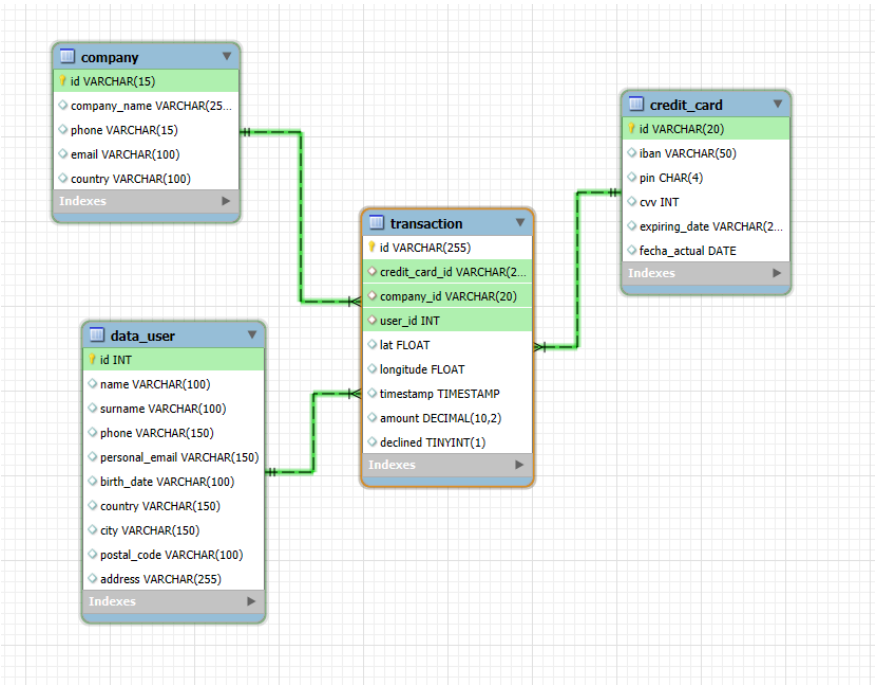
```
169 •  ALTER TABLE transaction
170    ADD CONSTRAINT user_transaction
171    FOREIGN KEY (user_id)
172    REFERENCES data_user(id);
173
```

Output

Action Output

#	Time	Action	Message
1	16:46:07	ALTER TABLE transaction ADD CONSTRAINT user_transaction FOREIGN KEY (user_id) REFERENCES data_user(id)	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0

11) diagrama final:



Ejercicio 2

La empresa también le pide crear una vista llamada "InformeTecnico" que contenga la siguiente información:

ID de la transacción
Nombre del usuario/a
Apellido del usuario/a
IBAN de la tarjeta de crédito usada.
Nombre de la compañía de la transacción realizada.

Asegúrese de incluir información relevante de las tablas que conocerá y utilice alias para cambiar de nombre columnas según sea necesario.
Muestra los resultados de la vista, ordena los resultados de forma descendente en función de la variable ID de transacción.

```sql
DROP VIEW IF EXISTS InformeTecnico;
CREATE VIEW InformeTecnico AS
SELECT t.id AS id_transaccion,
       u.name AS nombre_usuario,
       u.surname AS apellido_usuario,
       c.company_name AS nombre_compania,
       cc.iban AS iban
FROM company c
JOIN transaction t ON c.id = t.company_id
JOIN data_user u ON t.user_id = u.id
JOIN credit_card cc ON cc.id = t.credit_card_id
ORDER BY t.id DESC;

SELECT *
FROM InformeTecnico;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

id_transaccion	nombre_usuario	apellido_usuario	nombre_compania	iban
FFFD31D6-9495-47CE-B54A-7DB8E1CC274B	Bmrgli	Tprvvmrc	Turpis Company	XX7948144512112891824909 22
FFFCF76D-ECF0-4985-A2D0-B2A7B75998FC	Dfrled	Vilqcjdl	Amet Nulla Donec Corporation	XX6362517016478920366760 34
FFFC9E8D-27C7-4ADE-98F2-7533EF4DF126	Securp	Faofvqfy	Nunc Interdum Incorporated	XX1626771433042236314375 67
FFFB270D-F53A-4D5D-9666-E5307C53CC84	Ggzjpa	Uirzjulh	Viverra Donec Foundation	XX3951142670820199525670 52
FFF9E3CE-234E-408C-A8EF-F9CAD577224A	Yshimq	Zpsjsleed	Convallis In Incorporated	XX8845462156537570367941
FFF9E178-6CD2-4DF9-99B0-49AE068809B1	Jevepx	Xwcwzwnm	Mus Aenean Eget Foundation	XX3214055157116543847114 81
FFF867C9-17B5-4B1F-AFD9-F8023AAA449E	Fqlngd	Lvhfqyxi	Cras Vehicula Aliquet Industries	XX2784463429326809797294 26
FFF7042D-18C6-4DDD-823C-4D90A4AC8F26	Njoraa	Egsqcuii	Placerat LLP	XX4050092725725500820272 09

InformeTecnico 13

Output

Action Output

#	Time	Action	Message
1	17:01:43	DROP VIEW IF EXISTS InformeTecnico	0 row(s) affected
2	17:01:43	CREATE VIEW InformeTecnico AS SELECT t.id AS id_transaccion, u.name AS nombre_usuario, u.surname AS apellido_usuario, ...	0 row(s) affected
3	17:01:43	SELECT * FROM InformeTecnico	100000 row(s) returned