

Sprint 3: Gestió de taules, índex i vistes

Tarea S3.01. Manipulación de tablas

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 manera 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 "dades_introduir_credit". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

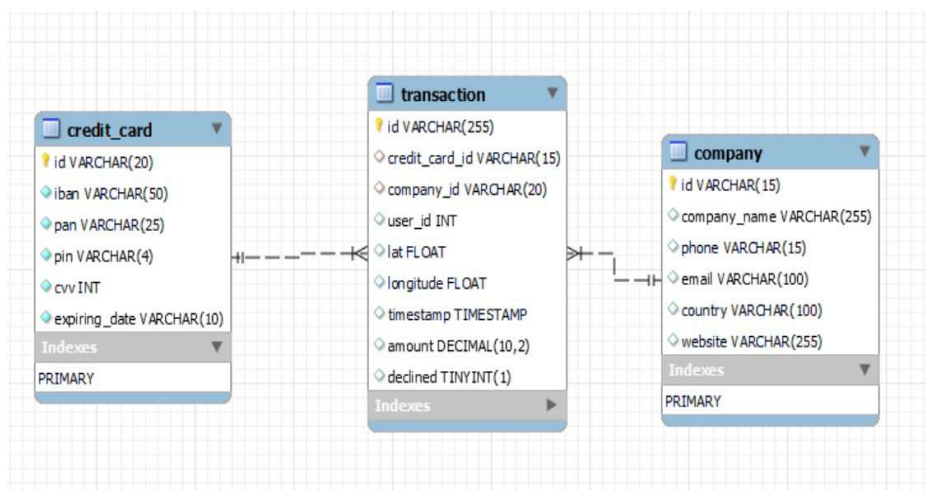


Imagen1. Diagrama entidad relación de la base de datos transactions.

En este ejercicio, se crea la tabla credit_card mediante el siguiente código.

```
CREATE TABLE IF NOT EXISTS credit_card (
    id VARCHAR(20) NOT NULL PRIMARY KEY,
    iban VARCHAR(50) NOT NULL,
    pan VARCHAR(25) NOT NULL,
    pin VARCHAR(4) NOT NULL,
    cvv INT NOT NULL,
    expiring_date VARCHAR(10) NOT NULL
);
```

También se crea la relación entre las tablas transaction y credit_card. En la tabla transaction, se añade una **FOREIGN KEY**, la clave Credit_Card_id de la tabla transaction. Ésta queda vinculada con la **clave primaria** id de la tabla Credit_card. Una tarjeta de crédito puede tener diferentes transacciones. Y diferentes transacciones se pueden haber realizado con una misma tarjeta de crédito. La relación entre tablas es de **1:N**.

```
ALTER TABLE transaction ADD CONSTRAINT credit_card_id
FOREIGN KEY (credit_card_id) REFERENCES credit_card(id)
ON DELETE CASCADE
ON UPDATE CASCADE;
```

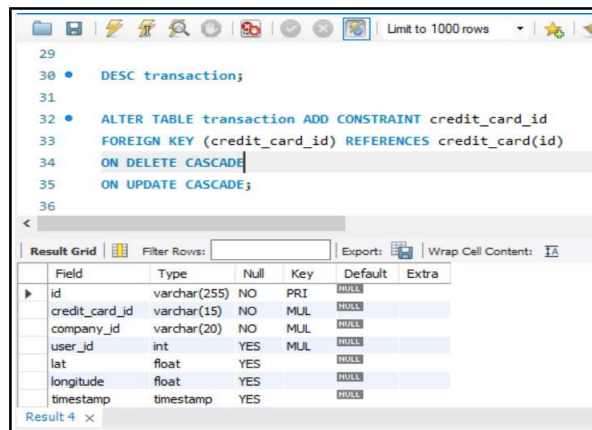


Imagen2. Imagen de la creación de la clave foránea credit card_id que se relaciona con el campo id de la tabla credit_card.

Por último, se cargan todos los datos de la tabla credit_card proporcionados en el ejercicio.

Ejercicio 2

El departamento de Recursos Humanos ha identificado un error en el número de cuenta del usuario con ID CcU-2938. La información que debe mostrarse para este registro es: R323456312213576817699999. Recuerda mostrar que el cambio se realizó.

Primero, buscamos el registro con ID, CcU-2938.

```
SELECT * FROM credit_card WHERE id = 'CcU-2938';
```

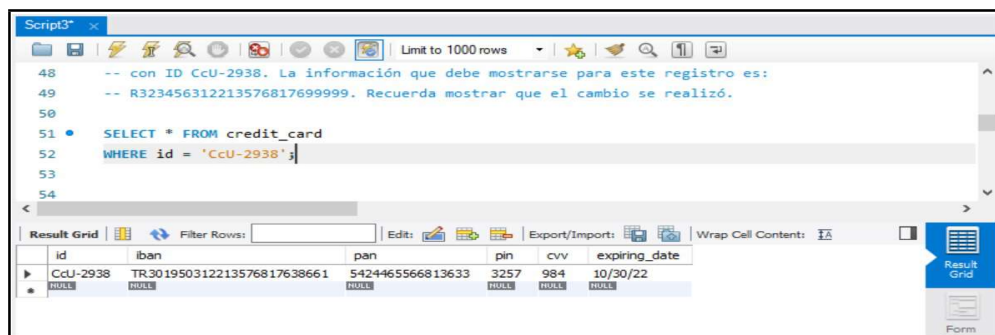


Imagen3. Captura de pantalla de la búsqueda del registro a modificar en la tabla Credit_Card.

Después, se modifica el registro

```
UPDATE credit_card
SET iban = 'R323456312213576817699999'
WHERE id = 'CcU-2938';
```

Por último, comprobamos que el registro se ha modificado con éxito.

```
SELECT *  
FROM credit_card  
WHERE iban= 'R323456312213576817699999';
```

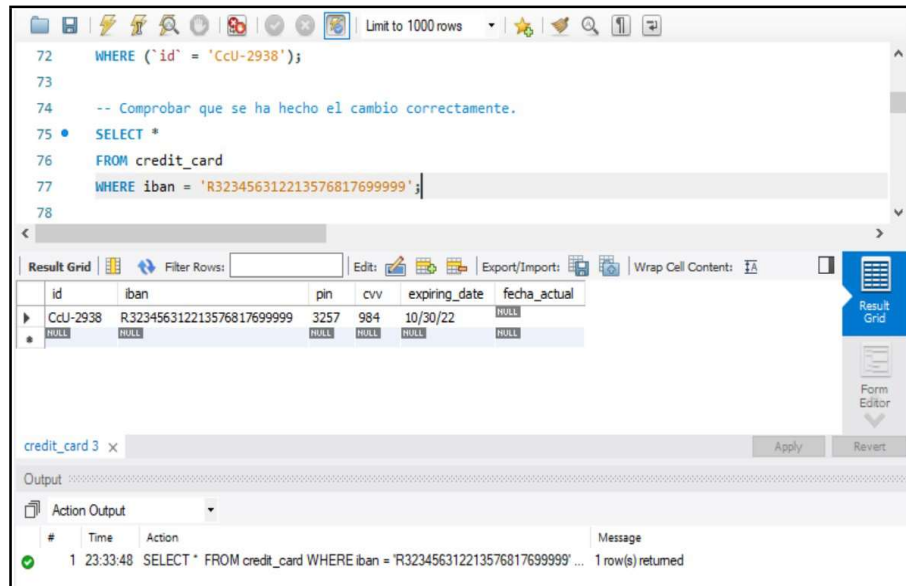


Imagen4. Captura de pantalla del registro con los datos modificados.

Ejercicio 3

En la tabla "transaction" ingresa un nuevo usuario con la siguiente información:

Id 108B1D1D-5B23-A76C-55EF-C568E49A99DD (y demás datos)

- Comprobamos si esos datos existen en la tabla

```
SELECT * FROM credit_card  
WHERE id = 'CcU-9999'
```

Para mi, para poder añadir una transacción, primero se debe dar de alta la company en la BBDD, al igual que los datos de la credit_card. Por un tema de consistencia de la BBDD, se aplican CONSTRAINT. (Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails)

Todo y que no se debería, insertar para mantener la consistencia de nuestra base de datos. Se va a introducir el registro en cuestión. Para ello debemos **desactivar** todas las FOREIGN KEY de la traba transaction con la instrucción **CHECK**.

```
SET FOREIGN_KEY_CHECKS = 0;
```

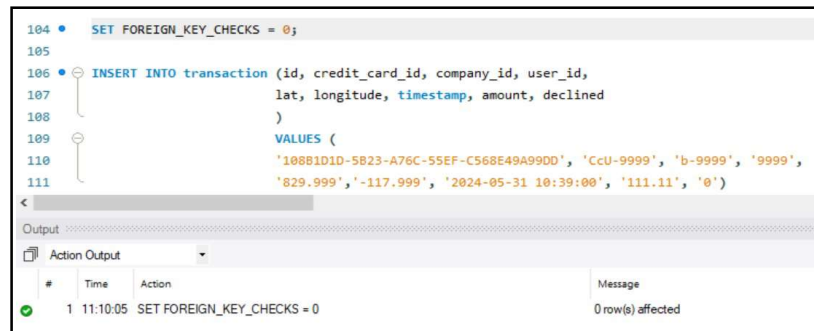


Imagen5. Captura de pantalla de la ejecución de la instrucción `SET FOREIGN_KEY_CHECKS = 0` exitosamente.

Seguidamente, se ejecuta la consulta siguiente para introducir el registro en la tabla transaction.

```

INSERT INTO transaction (id, credit_card_id, company_id, user_id,
                        lat, longitude, timestamp, amount, declined
                        )
VALUES (
    '108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999',
    '829.999', '-117.999', '2024-05-31 10:39:00', '111.11', '0');

```

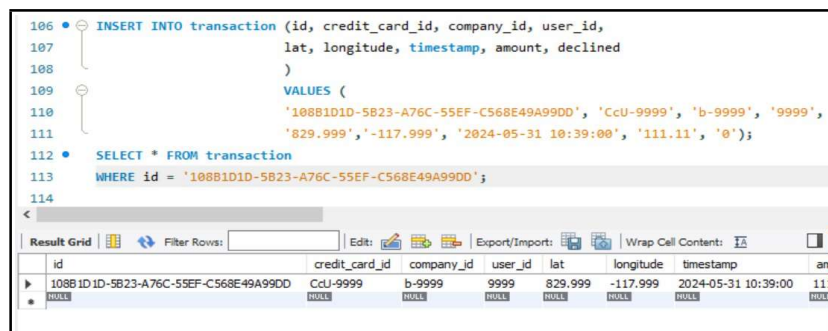


Imagen6. Captura de pantalla de la comprobación de la existencia del registro añadido a la tabla transaction.

Por último, después de añadir el registro, activamos las FOREIGN KEY.

```
SET FOREIGN_KEY_CHECKS = 1;
```

Ejercicio 4

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

```
DESC credit_card;
```

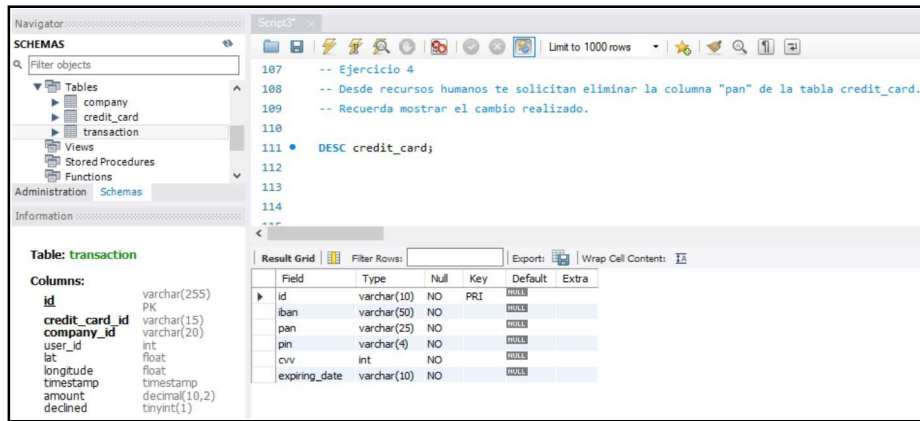


Imagen7. Captura de pantalla de los diferentes campos que forman la tabla credit_card, donde aparece el campo pan.

A continuación la Query para eliminar el campo pan de la tabla Credit_card.

ALTER TABLE credit_card DROP COLUMN pan;

DESC credit_card;

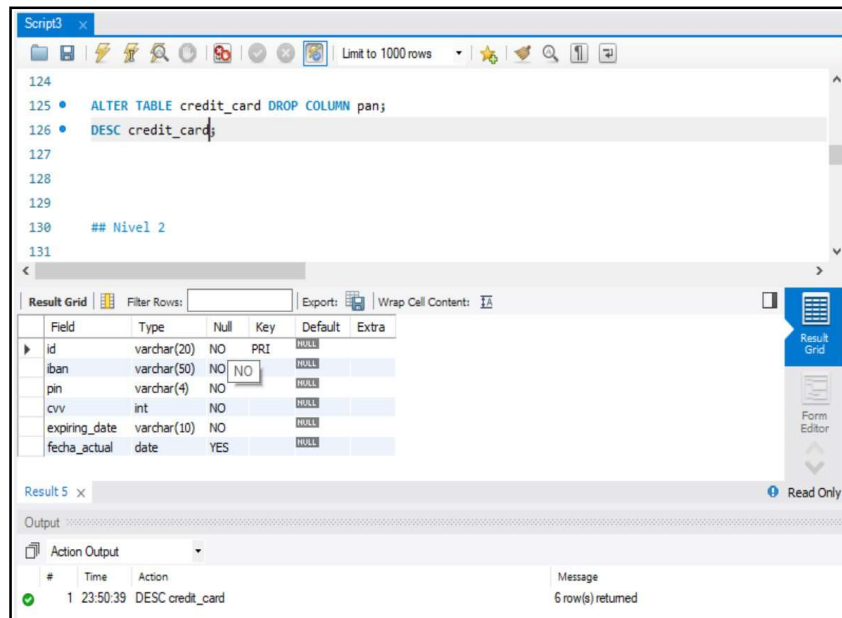


Imagen8. Captura de pantalla de los campos de la tabla credit_card, donde ya no aparece el campo pan.

NIVEL 2

Ejercicio 1

Elimina de la tabla transaction el registro con ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de datos.

Primero, comprobamos que está el registro

```
SELECT * FROM transaction
WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
```

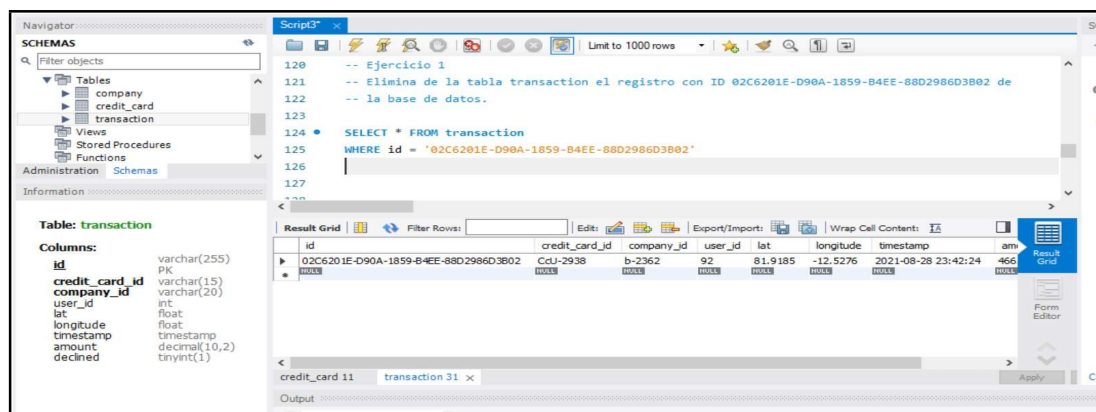


Imagen9. Captura de pantalla del registro correspondiente en la tabla transaction.

Segundo lo eliminamos

```
DELETE FROM transaction
WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
```

#	Time	Action	Message
6	10:58:19	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2...	1 row(s) returned
7	10:59:38	DELETE FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2...	1 row(s) affected

Imagen10. Captura de pantalla del mensaje de consola que demuestra la eliminación del registro.

Tercero volvemos a comprobar que ya no existe.

```
SELECT * FROM transaction
WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
```

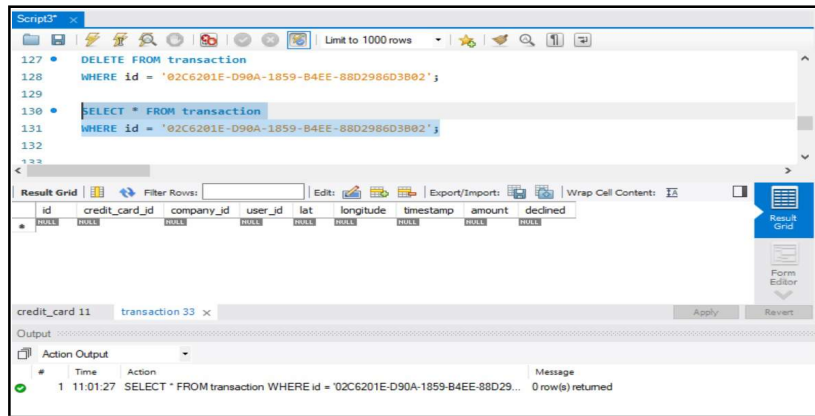


Imagen11. Captura de pantalla de la búsqueda del registro. Comprobación de la correcta eliminación.

No se obtiene ningún resultado con ese valor de filtro. La eliminación del registro se ha realizado con éxito.

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. Promedio de compra realizado por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor.

CREATE VIEW vistaMarketing **AS**

SELECT company.company_name, company.phone, company.country, avg(transaction.amount) **AS** media

FROM company

INNER JOIN transaction

ON transaction.company_id = company.id

GROUP BY company.id

ORDER BY media **DESC**;

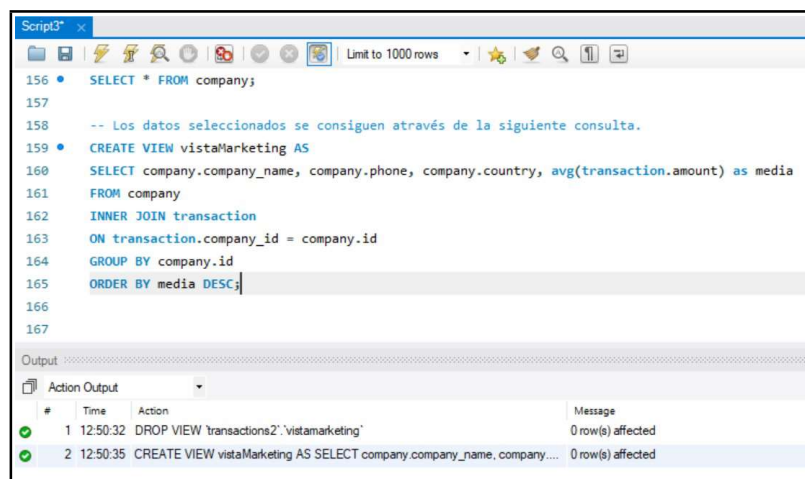
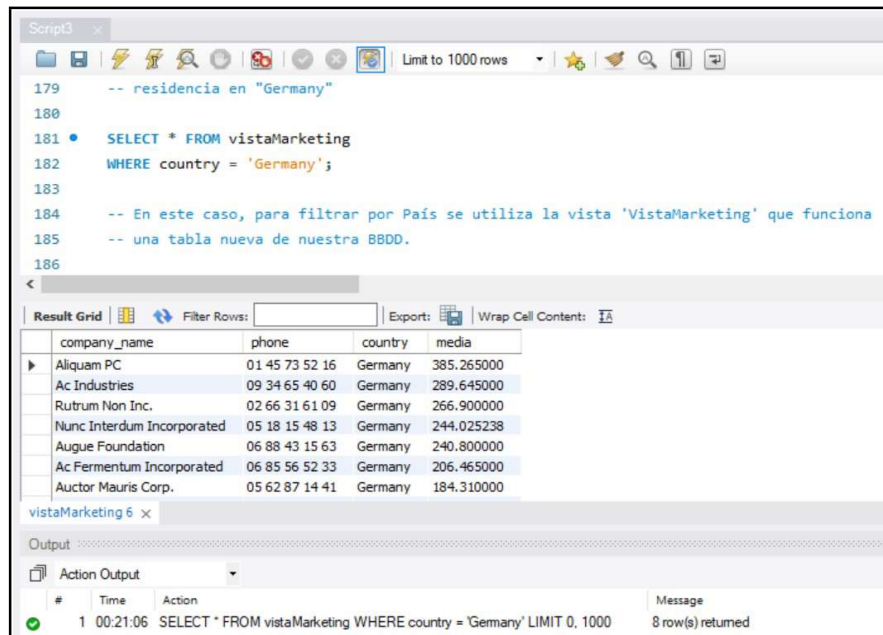


Imagen12. En la izquierda de la imagen, se puede la vista llamada VistaMarketing.

Ejercicio 3

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



The screenshot shows a database management tool interface. At the top, there's a script editor with a SQL query: `SELECT * FROM vistaMarketing WHERE country = 'Germany';`. Below the script editor, there's a "Result Grid" showing the results of the query. The grid has four columns: `company_name`, `phone`, `country`, and `media`. There are 8 rows of data, all with `Germany` as the country. Below the result grid, there's an "Output" section showing the execution details: "SELECT * FROM vistaMarketing WHERE country = 'Germany' LIMIT 0, 1000" and "8 row(s) returned".

```
179 -- residencia en "Germany"
180
181 • SELECT * FROM vistaMarketing
182 WHERE country = 'Germany';
183
184 -- En este caso, para filtrar por País se utiliza la vista 'VistaMarketing' que funciona
185 -- una tabla nueva de nuestra BBDD.
186
```

company_name	phone	country	media
Aliquam PC	01 45 73 52 16	Germany	385.265000
Ac Industries	09 34 65 40 60	Germany	289.645000
Rutrum Non Inc.	02 66 31 61 09	Germany	266.900000
Nunc Interdum Incorporated	05 18 15 48 13	Germany	244.025238
Augue Foundation	06 88 43 15 63	Germany	240.800000
Ac Fermentum Incorporated	06 85 56 52 33	Germany	206.465000
Auctor Mauris Corp.	05 62 87 14 41	Germany	184.310000

Output: SELECT * FROM vistaMarketing WHERE country = 'Germany' LIMIT 0, 1000 8 row(s) returned

Imagen13. Consulta en la vista VistaMarketing de los registros que pertenecen al país Germany.

NIVEL 3

Ejercicio 1

La semana próxima 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:

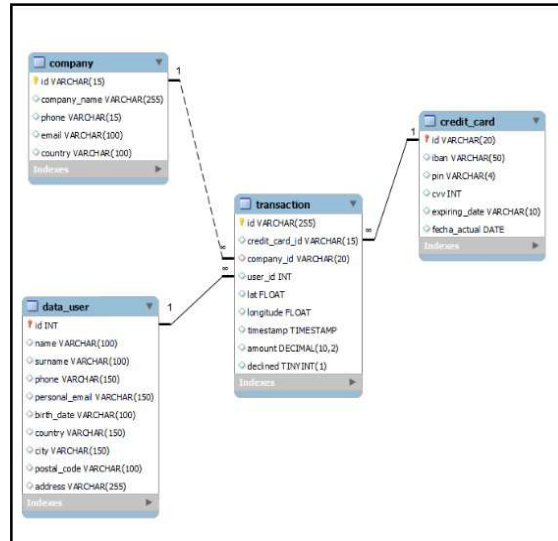


Imagen 14. Diagrama entidad relación al cual debemos llegar:

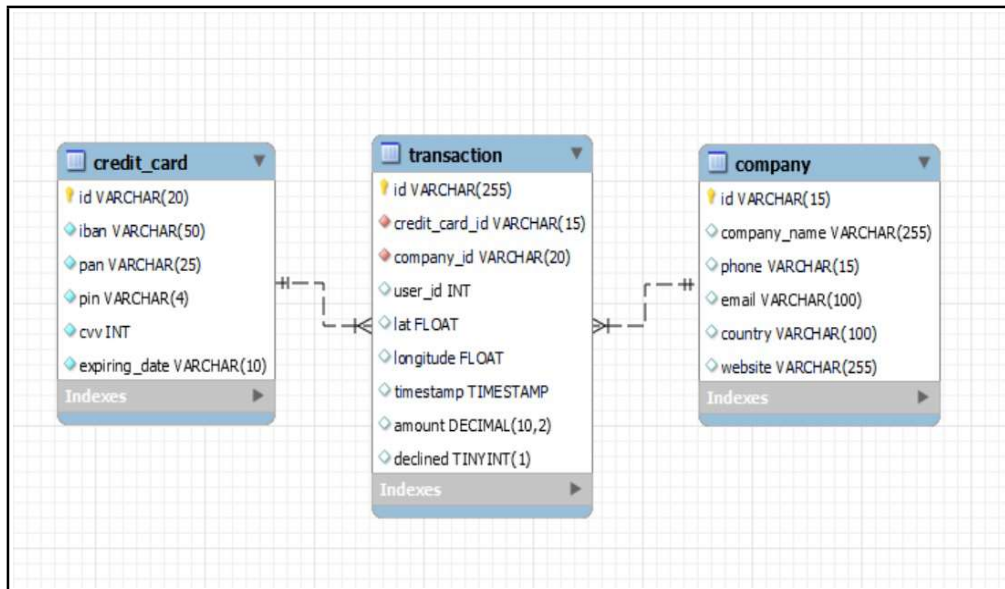
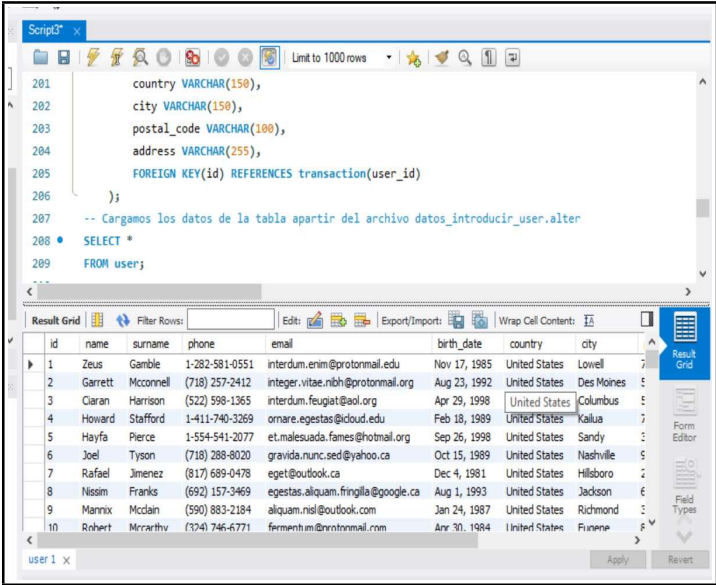


Imagen 15. Diagrama inicial del cuál parto para llegar al esquema de la imagen 12.

1) Primero, creamos la tabla **data_user** y añadimos todos los datos, ejecutando los archivos facilitados.
Creamos la tabla user

```
CREATE INDEX idx_user_id ON transaction(user_id);
CREATE TABLE IF NOT EXISTS user (
    id INT PRIMARY KEY,
    name VARCHAR(100),
    surname VARCHAR(100),
    phone VARCHAR(150),
    email VARCHAR(150),
    birth_date VARCHAR(100),
    country VARCHAR(150),
    city VARCHAR(150),
    postal_code VARCHAR(100),
    address VARCHAR(255),
    FOREIGN KEY(id) REFERENCES transaction(user_id)
);
```



The screenshot shows a database management interface. The top pane displays SQL code for creating the 'user' table and inserting data. The bottom pane shows a 'Result Grid' with 10 rows of data. The columns are: id, name, surname, phone, email, birth_date, country, and city. The data includes names like Zeus, Garrett, Claran, Howard, Hayfa, Joel, Rafael, Nissim, Mannix, and Robert, with their respective surnames, phone numbers, email addresses, birth dates, and locations.

id	name	surname	phone	email	birth_date	country	city
1	Zeus	Gamble	1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	Lowell
2	Garrett	Mcconnell	(718) 257-2412	integer.vitae.nibh@protonmail.org	Aug 23, 1992	United States	Des Moines
3	Claran	Harrison	(522) 598-1365	interdum.feugiat@aol.org	Apr 29, 1998	United States	Columbus
4	Howard	Stafford	1-411-740-3269	ornare.egestas@icloud.edu	Feb 18, 1989	United States	Kailua
5	Hayfa	Pierce	1-554-541-2077	et.malesuada.fames@hotmail.org	Sep 26, 1998	United States	Sandy
6	Joel	Tyson	(718) 288-8020	gravidamunc.sed@yahoo.ca	Oct 15, 1989	United States	Nashville
7	Rafael	Jimenez	(817) 689-0478	egest@outlook.ca	Dec 4, 1981	United States	Hillsboro
8	Nissim	Franks	(692) 157-3469	egestas.aliquam.fringilla@google.ca	Aug 1, 1993	United States	Jackson
9	Mannix	Mcdain	(590) 883-2184	aliquam.nisl@outlook.com	Jan 24, 1987	United States	Richmond
10	Robert	Mccarthy	(724) 746-6771	fementum@protonmail.com	Apr 30, 1984	United States	Fusene

Imagen 16. Representación de los datos introducidos en la tabla user, después de ser creada.

Cargamos los datos de la tabla apartir del archivo **datos_introducir_user**.

Primero elimino la **FOREIGN KEY** que hay en la tabla user, ya que no tiene sentido.

```
ALTER TABLE user DROP FOREIGN KEY user_ibfk_1;
```

Se define la relación entre la tabla transaction y la tabla tabla user através de una **FOREIGN KEY** en la tabla transaction.

```
ALTER TABLE transaction ADD CONSTRAINT user_id
```

```
FOREIGN KEY (user_id) REFERENCES user(id)
```

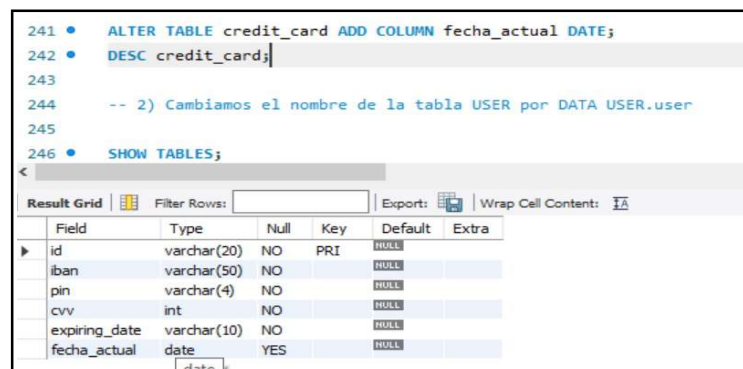
```
ON DELETE CASCADE
```

```
ON UPDATE CASCADE;
```

2) Añadimos a la tabla *Credit-Card*, la columna **fecha_actual** DATE

```
ALTER TABLE credit_card ADD COLUMN fecha_actual DATE;
```

```
DESC credit_card;
```



```
241 • ALTER TABLE credit_card ADD COLUMN fecha_actual DATE;
242 • DESC credit_card;
243
244 -- 2) Cambiamos el nombre de la tabla USER por DATA USER.user
245
246 • SHOW TABLES;
```

Field	Type	Null	Key	Default	Extra
id	varchar(20)	NO	PRI	HULL	
iban	varchar(50)	NO		HULL	
pin	varchar(4)	NO		HULL	
cvv	int	NO		HULL	
expiring_date	varchar(10)	NO		HULL	
fecha_actual	date	YES		HULL	

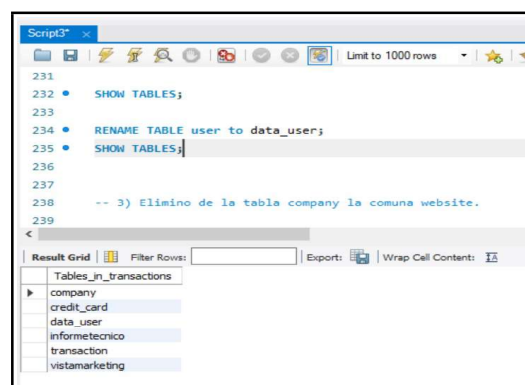
Imagen 17. Listado de los campos de la tabla *credit_card* donde aparece el campo nuevo creado, *fecha_actual*.

5) Cambiamos el nombre de la tabla USER por DATA USER.user

```
SHOW TABLES;
```

```
RENAME TABLE user to data_user;
```

```
SHOW TABLES;
```



```
231
232 • SHOW TABLES;
233
234 • RENAME TABLE user to data_user;
235 • SHOW TABLES;
236
237 -- 3) Elimino de la tabla company la comuna website.
238
239
```

Tables_in_transactions
company
credit_card
data_user
informetecnico
transaction
vistamarketing

Imagen 18. Captura de pantalla donde se listan las diferentes tablas de la BBDD transactions. Se puede ver que el cambio de nombre de la tabla user se ha hecho efectivo a *data_user*.

5) Elimino de la tabla company la comuna website.

ALTER TABLE company **DROP COLUMN** website;

DESC company;

```
240 • ALTER TABLE company DROP COLUMN website;
241 • DESC company;
242
243
244
```

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI		
company_name	varchar(255)	YES			
phone	varchar(15)	YES			
email	varchar(100)	YES			
country	varchar(100)	YES			

Imagen 19. Imagen donde aparecen los campos de company. Y no aparece el campo Website que se ha eliminado.

La imagen final del diagrama entidad relación de nuestra base de datos transactions, después de las modificaciones, queda de la siguiente manera.

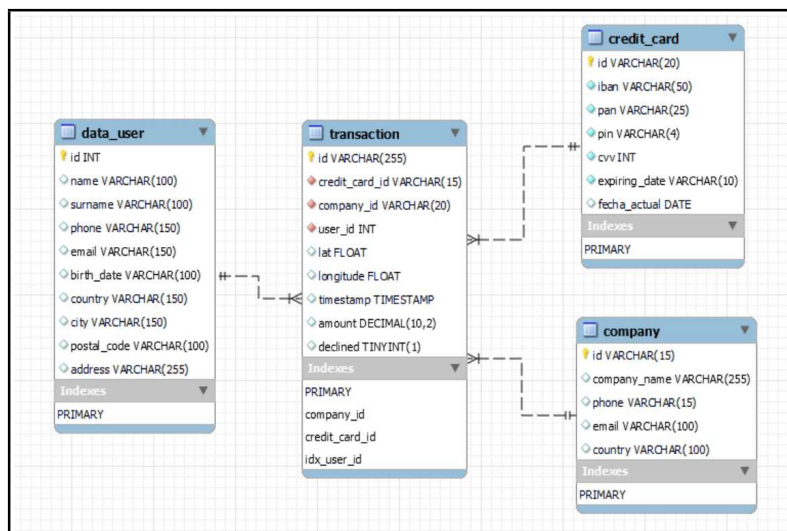


Imagen20. Diagrama de entidad relación final, después de los cambios solicitados.

Ejercicio 2

La empresa también te solicita 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.

Asegúrate de incluir información relevante de ambas tablas y utiliza alias para cambiar de nombre columnas según sea necesario.

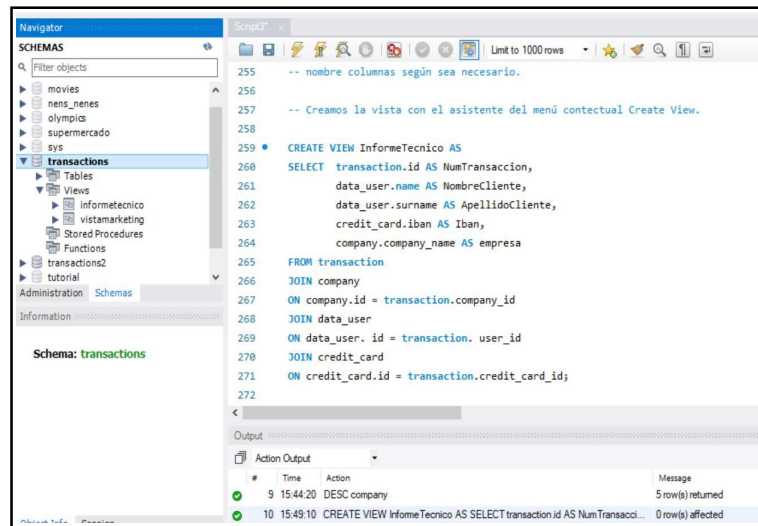


Imagen21. Creación de la vista InformeTecnico.