

# SPRINT 3

## 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.

Respuesta:

--- Se crea la tabla 'credit\_card'

```
11 • CREATE TABLE IF NOT EXISTS credit_card (  
12     id VARCHAR(15) NOT NULL,  
13     iban VARCHAR(50),  
14     pan VARCHAR(50),  
15     pin VARCHAR(4),  
16     cvv VARCHAR(3),  
17     expiring_date VARCHAR(15),  
18     PRIMARY KEY (id)  
19 );
```

Output

#	Time	Action	Message	Duration / Fetch
1	20:24:01	CREATE TABLE IF NOT EXISTS credit_card (id VARCHAR...	0 row(s) affected	0.047 sec

--- Se cargan los valores de la tabla 'credit\_card'.

```
1 -- Insertamos datos de credit_card  
2 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2938', 'TR301950312213576'  
3 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2945', 'D0268547637485374'  
4 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2952', 'BG45IVQL527105256'  
5 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2959', 'CR724247724433584'  
6 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2966', 'BG72LKTQ156276283'  
7 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2973', 'PT878062281350924'  
8 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2980', 'DE392418818830862'  
9 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2987', 'GE896814348377487'  
10 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2994', 'BH627144283680667'  
11 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-3001', 'CY490874266547745'  
12 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-3008', 'LU507216693616119'
```

Output

#	Time	Action	Message	Duration / Fetch
4999	20:27:59	INSERT INTO credit_card (id, iban, pan, pin, cvv, expirin...	1 row(s) affected	0.016 sec
5000	20:27:59	INSERT INTO credit_card (id, iban, pan, pin, cvv, expirin...	1 row(s) affected	0.015 sec

--- También se verifica que la Tabla 'transaction' se encuentre conectada con la tabla 'credit\_card' por medio de la FK (credit\_card\_id). Para ellos, se visualiza el diagrama o 'schema'.

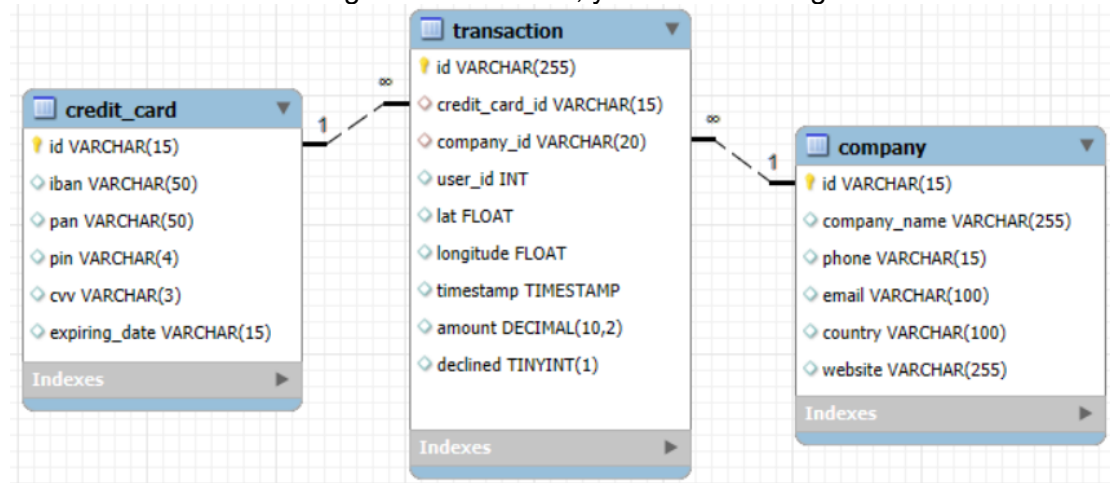
--- Como no se están vinculadas ambas tablas, se procede a asignar la FK en la tabla 'transaction'.

```
23 • ALTER TABLE transaction  
24     ADD FOREIGN KEY (credit_card_id) REFERENCES credit_card (id);  
25
```

Output

#	Time	Action	Message	Duration / Fetch
1	20:30:36	ALTER TABLE transaction ADD FOREIGN KEY (credit_car...	100000 row(s) affected Records: 100000 Duplicates: 0 W...	2.672 sec

--- Se visualiza nuevamente el diagrama o 'schema', y se obtiene el siguiente:



--- Se describe el diagrama o 'schema':

Se observa, que en el modelo se puede identificar que la Tabla 'transactions' puede denominarse la tabla de 'Hechos' y las tablas company y credit\_card son tablas de 'dimensiones'.

La nueva tabla 'credit\_card' se vincula directamente con la tabla 'transactions' en una relación de '1 a muchos', siendo que una tarjeta de crédito puede tener muchas transacciones.

La columna que conecta ambas tablas tiene diferente nombre:

Tabla	Nombre de columna
transaction	credit_card_id
credit_card	id

En la tabla 'credit\_card':

La PK es la columna 'id' que identifica a cada tarjeta de crédito con un registro único.

Las demás columnas de la nueva tabla son datos que describen a cada tarjeta de crédito: iban (cuenta bancaria vinculada a la tarjeta), pan (num. de tarjeta), pin (num. 4 dígitos para seguridad), cvv (num 3 dígitos para seguridad), expiring\_date (fecha de caducidad de tarjeta).

La nueva tabla 'credit\_card' no tiene una relación directa con la tabla 'company'.

## Ejercicio 2

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

**Respuesta:**

Paso 1: se verifica que exista la tarjeta de crédito y se verifica el iban, en la tabla 'credit\_card':

```

30 • SELECT id, iban
31 FROM credit_card
32 WHERE id = 'CcU-2938';
33
  
```

id	iban
CcU-2938	TR301950312213576817638661
NULL	NULL

credit\_card 14 x Apply Revert

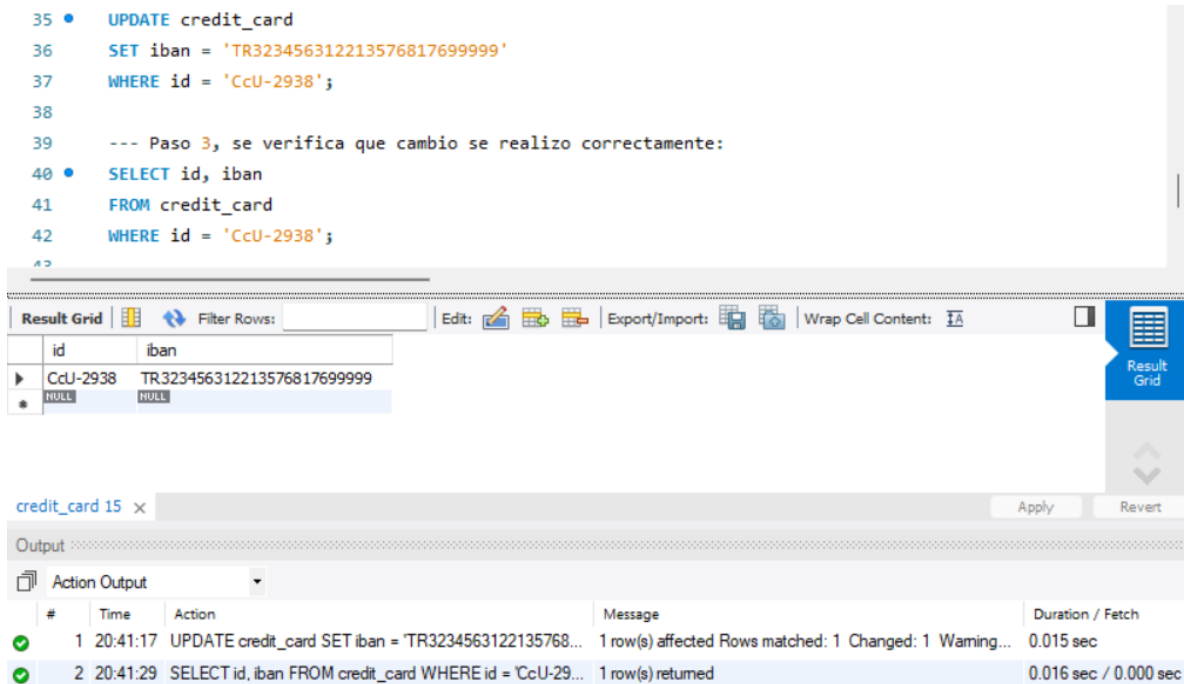
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	20:39:05	SELECT id, iban FROM credit_card WHERE id = 'CcU-29...	1 row(s) returned	0.000 sec / 0.000 sec

Paso 2: se realiza el cambio solicitado de iban y Paso 3, se verifica que el cambio se realizó correctamente:

```
35 • UPDATE credit_card
36 SET iban = 'TR323456312213576817699999'
37 WHERE id = 'CcU-2938';
38
39 --- Paso 3, se verifica que cambio se realizo correctamente:
40 • SELECT id, iban
41 FROM credit_card
42 WHERE id = 'CcU-2938';
```



id	iban
CcU-2938	TR323456312213576817699999

credit\_card 15 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	20:41:17	UPDATE credit_card SET iban = 'TR3234563122135768...	1 row(s) affected Rows matched: 1 Changed: 1 Warning...	0.015 sec
✓ 2	20:41:29	SELECT id, iban FROM credit_card WHERE id = 'CcU-29...	1 row(s) returned	0.016 sec / 0.000 sec

### 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
- lat 829.999
- longitude -117.999
- amount 111.11
- declined 0

#### Respuesta:

Para introducir el registro solicitado en la tabla 'transaction', es necesario verificar que la información correspondiente a las otras tablas ('credit\_card' y 'company') existan, de lo contrario habrá que crear primero el id correspondiente en cada tabla y luego podrá realizarse el ingreso en la tabla 'transaction'. Esto se realiza para mantener la integridad referencial de la base de datos.

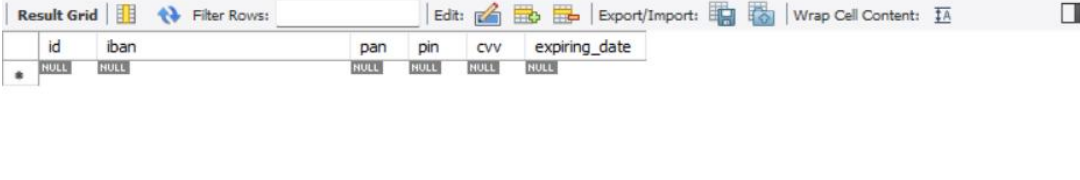
No obstante, para efectos de esta operación, inicialmente solo se crean los 'id' correspondientes a las tablas 'credit\_card' y 'company', por ser parte de la información brindada y por ser necesario para ingresar la nueva transacción en la tabla 'transaction'. Quedando pendiente completar la información de estos 'id' en cada tabla ('credit\_card' y 'company'), información que se debe solicitar al depto correspondiente de forma inmediata posterior para mantener correctamente la base de datos.

A continuación se describen los pasos realizados para efectuar esta operación solicitada:

## Tabla 'credit\_card'

Paso 1: se verifica que exista en tabla 'credit\_card' el id 'CcU-9999'.

```
57 • SELECT *
58 FROM credit_card
59 WHERE id = 'CcU-9999';
```



	id	iban	pan	pin	cvv	expiring_date
*	NULL	NULL	NULL	NULL	NULL	NULL

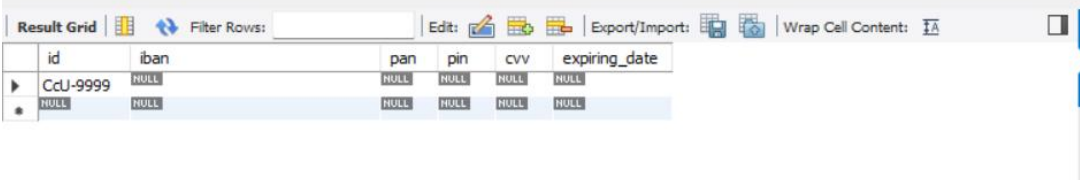
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	21:08:25	SELECT * FROM credit_card WHERE id = 'CcU-9999'	0 row(s) returned	0.015 sec / 0.000 sec

Paso 2: Se crea el id 'CcU-9999' en la tabla 'credit\_card' (porque no existe) y Se verifica que el nuevo id esté en la tabla 'credit\_card'.

```
62 • INSERT INTO credit_card (id)
63 VALUES ('CcU-9999');
64
65 • SELECT *
66 FROM credit_card
67 WHERE id = 'CcU-9999';
```



	id	iban	pan	pin	cvv	expiring_date
▶	CcU-9999	NULL	NULL	NULL	NULL	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL

credit\_card 17 x

Output

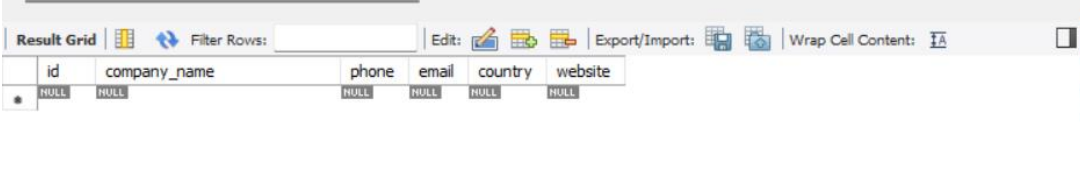
Action Output

#	Time	Action	Message	Duration / Fetch
1	21:10:47	INSERT INTO credit_card (id) VALUES ('CcU-9999')	1 row(s) affected	0.016 sec
2	21:10:52	SELECT * FROM credit_card WHERE id = 'CcU-9999'	1 row(s) returned	0.000 sec / 0.000 sec

## Tabla 'company'

Paso 1: se verifica que exista en tabla 'company' el id 'b-9999'.

```
70 • SELECT *
71 FROM company
72 WHERE id = 'b-9999';
73
```



	id	company_name	phone	email	country	website
*	NULL	NULL	NULL	NULL	NULL	NULL

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	21:15:35	SELECT * FROM company WHERE id = 'b-9999'	0 row(s) returned	0.000 sec / 0.000 sec

Paso 2: Se crea el id 'b-9999' en la tabla 'company' (porque no existe) y Se verifica que el nuevo id esté en la tabla 'company'.

```

75 • INSERT INTO company (id)
76   VALUES ('b-9999');
77
78 • SELECT *
79   FROM company
80  WHERE id = 'b-9999';

```

id	company_name	phone	email	country	website
b-9999	NULL	NULL	NULL	NULL	NULL
*	NULL	NULL	NULL	NULL	NULL

company 19 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	21:17:49	INSERT INTO company (id) VALUES (b-9999)	1 row(s) affected	0.016 sec
✓ 2	21:17:56	SELECT * FROM company WHERE id = b-9999'	1 row(s) returned	0.000 sec / 0.000 sec

## Tabla 'transaction'

Paso 1: Se verifica si existe o no en la tabla 'transaction' el id '108B1D1D-5B23-A76C-55EF-C568E49A99DD'

```

83 • SELECT *
84   FROM transaction
85  WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';

```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	dec
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
*								

transaction 20 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	21:43:14	SELECT * FROM transaction WHERE id = '108B1D1D-5B...	0 row(s) returned	0.000 sec / 0.000 sec

Paso 2: Se ingresa nuevo id y su información en tabla 'transaction' y Se verifica que el nuevo registro esté en la tabla 'transaction'.

```

88 • INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
89   VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', '829.999', '-117.999', '111.11', '0');
90
91 • SELECT *
92   FROM transaction
93  WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';

```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B1D1D-5B23-A76C-55EF-C568E49A99DD	CcU-9999	b-9999	9999	829.999	-117.999	NULL	111.11	0
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

transaction 21 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	21:47:22	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)	1 row(s) affected	0.015 sec
✓ 2	21:47:30	SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55E...	1 row(s) returned	0.000 sec / 0.000 sec

## Ejercicio 4

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

### Respuesta:

Paso 1: se verifica que en la tabla 'credit\_card' exista la columna 'pan'.

The screenshot shows a database query editor with the command `DESCRIBE credit_card;` entered. Below the editor, the 'Result Grid' displays the table structure:

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pan	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	varchar(3)	YES		NULL	
expiring_date	varchar(15)	YES		NULL	

Below the result grid, the 'Output' section shows the 'Action Output' table:

#	Time	Action	Message	Duration / Fetch
1	21:51:27	DESCRIBE credit_card	6 row(s) returned	0.016 sec / 0.000 sec

Paso 2: se elimina la columna 'pan' y se verifica que ya no aparezca en la tabla 'credit\_card'.

The screenshot shows a database query editor with the following commands entered:

```
101 • ALTER TABLE credit_card
102 DROP pan;
103
104 • DESCRIBE credit_card;
```

Below the editor, the 'Result Grid' displays the table structure after the drop operation:

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	varchar(3)	YES		NULL	
expiring_date	varchar(15)	YES		NULL	

Below the result grid, the 'Output' section shows the 'Action Output' table:

#	Time	Action	Message	Duration / Fetch
1	21:54:37	ALTER TABLE credit_card DROP pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.047 sec
2	21:54:42	DESCRIBE credit_card	5 row(s) returned	0.000 sec / 0.000 sec



## NIVEL 2

### Ejercicio 1

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

Respuesta:

Paso 1: se verifica que en la tabla 'transaction' exista el registro con ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD.

```
113 • SELECT *
114 FROM transaction
115 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
000447FE-B650-4DCF-85DE-C7ED0EE1CAAD	CcS-5019	b-2370	438	41.5972	12.2218	2016-12-21 20:07:18	155.63	0

transaction 24 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:00:07	SELECT * FROM transaction WHERE id = '000447FE-B650-4DCF-85D...	1 row(s) returned	0.015 sec / 0.000 sec

Paso 2: se elimina el registro con ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD y se verifica que ya no aparezca en la tabla 'transaction'.

```
118 • DELETE
119 FROM transaction
120 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
121
122 • SELECT *
123 FROM transaction
124 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
----	----------------	------------	---------	-----	-----------	-----------	--------	----------

transaction 25 x Apply Revert

Output

Action Output

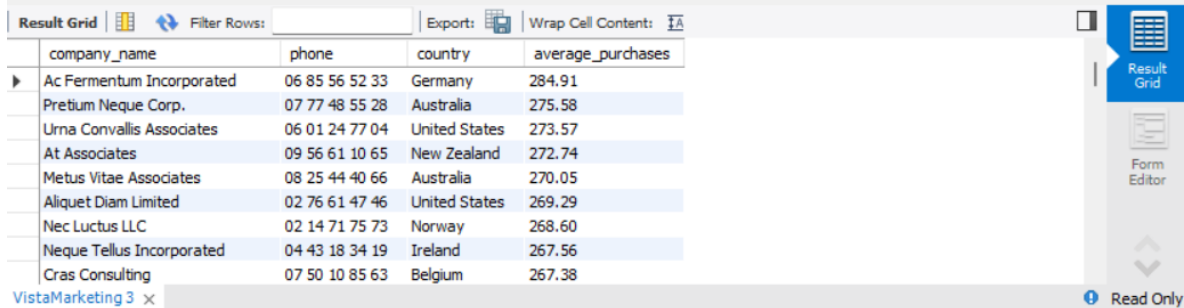
#	Time	Action	Message	Duration / Fetch
1	22:02:40	DELETE FROM transaction WHERE id = '000447FE-B650-4DCF-85DE...	1 row(s) affected	0.015 sec
2	22:02:49	SELECT * FROM transaction WHERE id = '000447FE-B650-4DCF-85D...	0 row(s) returned	0.000 sec / 0.000 sec

### Ejercicio 2

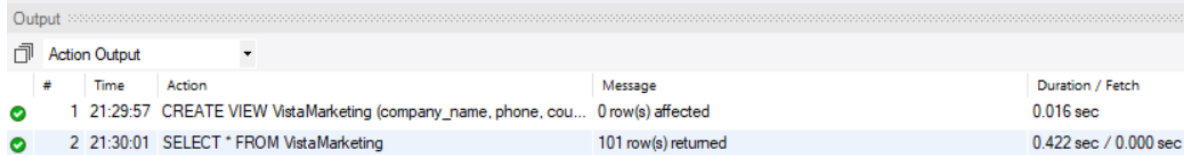
La sección de marketing desea tener acceso a información específica para realizar análisis y estrategias efectivas. Se ha solicitado crear una vista que proporcione detalles clave sobre las compañías y sus transacciones. Será necesaria que crees una vista llamada VistaMarketing que contenga la siguiente 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.

### Respuesta:

```
131 • CREATE VIEW VistaMarketing (company_name, phone, country, average_purchases) AS
132 SELECT c.company_name, c.phone, c.country, ROUND(AVG(t.amount),2) AS average_purchases
133 FROM company c
134 JOIN transaction t ON c.id = t.company_id
135 WHERE t.declined = 0
136 GROUP BY c.company_name, c.phone, c.country
137 ORDER BY average_purchases DESC;
138
139 • SELECT * FROM VistaMarketing;
```



company_name	phone	country	average_purchases
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.91
Pretium Neque Corp.	07 77 48 55 28	Australia	275.58
Urna Convallis Associates	06 01 24 77 04	United States	273.57
At Associates	09 56 61 10 65	New Zealand	272.74
Metus Vitae Associates	08 25 44 40 66	Australia	270.05
Aliquet Diam Limited	02 76 61 47 46	United States	269.29
Nec Luctus LLC	02 14 71 75 73	Norway	268.60
Neque Tellus Incorporated	04 43 18 34 19	Ireland	267.56
Cras Consulting	07 50 10 85 63	Belgium	267.38



#	Time	Action	Message	Duration / Fetch
1	21:29:57	CREATE VIEW VistaMarketing (company_name, phone, cou...	0 row(s) affected	0.016 sec
2	21:30:01	SELECT * FROM VistaMarketing	101 row(s) returned	0.422 sec / 0.000 sec

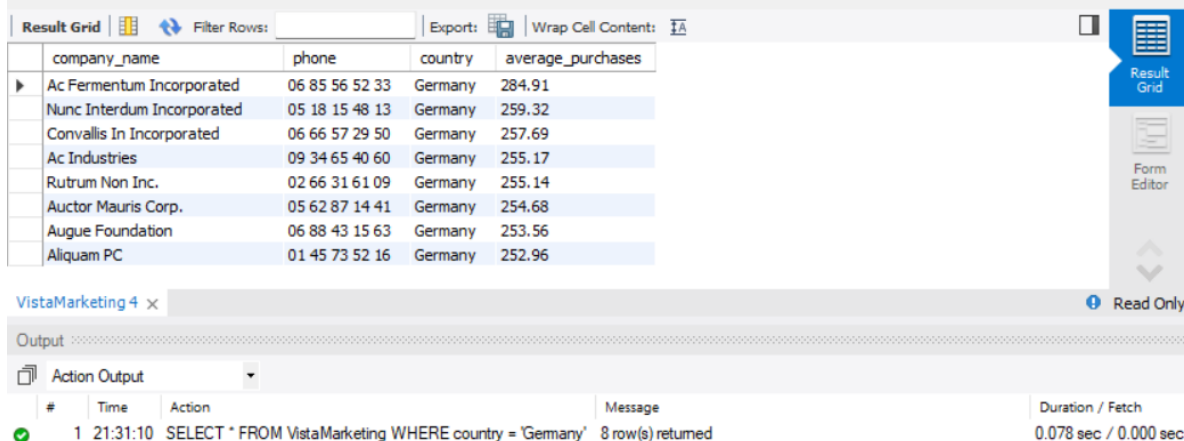
Se ha creado una Vista con la información solicitada, no obstante, se precisa que la columna 'amount' de la tabla de transacciones al indicar el importe de cada transacción, se han entendido para este caso como las 'compras' solicitadas por el dpto. de Marketing y en ella se ha basado el cálculo del promedio solicitado. Asimismo, al ser una Vista 'final' para el dpto. de Marketing se ha filtrado las transacciones, tomándose solo aquellas que no han sido declinadas ( $t.declined = 0$ ), esto para brindar una información clara de las 'compras' efectivas, y que las estrategias y/o campañas que se realicen a futuro estén adecuadamente orientadas a las empresas y/o países que realmente las producen.

### Ejercicio 3

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

### Respuestas:

```
143 • SELECT * FROM VistaMarketing
144 WHERE country = 'Germany';
```



company_name	phone	country	average_purchases
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.91
Nunc Interdum Incorporated	05 18 15 48 13	Germany	259.32
Convallis In Incorporated	06 66 57 29 50	Germany	257.69
Ac Industries	09 34 65 40 60	Germany	255.17
Rutrum Non Inc.	02 66 31 61 09	Germany	255.14
Auctor Mauris Corp.	05 62 87 14 41	Germany	254.68
Augue Foundation	06 88 43 15 63	Germany	253.56
Aliquam PC	01 45 73 52 16	Germany	252.96

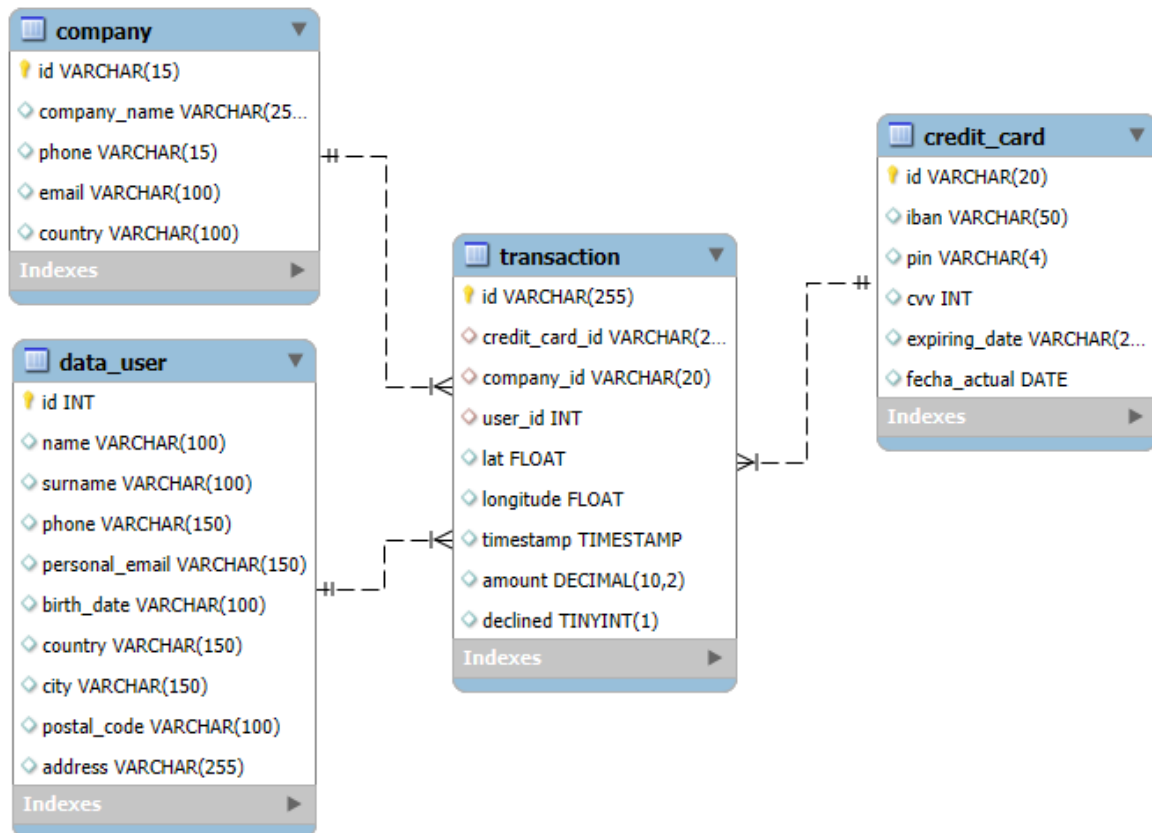
En base a la vista anterior, se ha filtrado como se pide solo las empresas que corresponden con Alemania ('Germany').



### 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:



Recordatorio:

En esta actividad, es necesario que describas el "paso a paso" de las tareas realizadas. Es importante realizar descripciones sencillas, simples y fáciles de comprender. Para realizar esta actividad deberás trabajar con los archivos denominados "estructura\_datos\_user" y "datos\_introducir\_user". Recuerda seguir trabajando sobre el modelo y las tablas con las que ya has trabajado hasta ahora.

#### Respuesta:

Paso 1: Se crea la tabla 'user'.

```
156 --- Paso 1: Se crea la tabla 'user'
157 CREATE TABLE IF NOT EXISTS user (
158     id CHAR(10) PRIMARY KEY,
159     name VARCHAR(100),
160     surname VARCHAR(100),
161     phone VARCHAR(150),
162     email VARCHAR(150),
163     birth_date VARCHAR(100),
164     country VARCHAR(150),
165     city VARCHAR(150),
166     postal_code VARCHAR(100),
167     address VARCHAR(255)
168 );
169
```

Output			
Action Output			
#	Time	Action	Message
1	23:39:53	CREATE TABLE IF NOT EXISTS user (id CHAR(10) PRIMARY KEY, na...	0 row(s) affected

Duration / Fetch  
0.078 sec

Paso 2: Se cargan los datos de la tabla 'user'.

```

1 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "151"
2 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "152"
3 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "153"
4 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "154"
5 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "155"
6 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "156"
7 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "157"
8 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "158"
9 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "159"
10 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "160"
11 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "161"
12 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "162"
13 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "163"
14 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "164"
15 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "165"
16 • INSERT INTO user (id, name, surname, phone, email, birth date, country, citv, postal code, address) VALUES ( "166"

```

Output

#	Time	Action	Message	Duration / Fetch
4999	23:42:07	INSERT INTO user (id, name, surname, phone, email, birth_date, coun...	1 row(s) affected	0.000 sec
5000	23:42:07	INSERT INTO user (id, name, surname, phone, email, birth_date, coun...	1 row(s) affected	0.000 sec

Paso 3: En la tabla 'user', cambiar el tipo de dato de 'id' a INT y nombre de columna 'email' por 'personal\_email'.

--- Se visualiza información actual de la tabla 'user', para verificar columnas e información.

```

172 • DESCRIBE user;

```

Result Grid

Field	Type	Null	Key	Default	Extra
id	char(10)	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
surname	varchar(100)	YES		NULL	
phone	varchar(150)	YES		NULL	
email	varchar(150)	YES		NULL	
birth_date	varchar(100)	YES		NULL	
country	varchar(150)	YES		NULL	
city	varchar(150)	YES		NULL	
postal_code	varchar(100)	YES		NULL	
address	varchar(255)	YES		NULL	

Result 4 x

Output

#	Time	Action	Message	Duration / Fetch
1	23:46:27	DESCRIBE user	10 row(s) returned	0.015 sec / 0.000 sec

--- Se realiza el cambio del tipo de dato de 'id' de CHAR a INT y nombre de columna 'email' por 'personal\_email'. Luego se visualiza nuevamente la tabla, para verificar cambios realizados.

```

175 • ALTER TABLE user
176     MODIFY id INT;
177
178 • ALTER TABLE user
179     RENAME COLUMN email TO personal_email;
180
181 • DESCRIBE user;

```

Result Grid

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
surname	varchar(100)	YES		NULL	
phone	varchar(150)	YES		NULL	
personal_email	varchar(150)	YES		NULL	
birth_date	varchar(100)	YES		NULL	
country	varchar(150)	YES		NULL	
city	varchar(150)	YES		NULL	
postal_code	varchar(100)	YES		NULL	
address	varchar(255)	YES		NULL	

Result 5 x

Output

#	Time	Action	Message	Duration / Fetch
1	23:50:03	ALTER TABLE user MODIFY id INT	5000 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.546 sec
2	23:50:07	ALTER TABLE user RENAME COLUMN email TO personal_email	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
3	23:50:10	DESCRIBE user	10 row(s) returned	0.000 sec / 0.000 sec

--- Paso 4: En la tabla 'user', agregar el user\_id = '9999', que se había ingresado en la tabla 'transaction' como parte de la información de una nueva transacción (ejercicio 3 - Nivel 1).

En dicho momento, la tabla user no existía, pero ahora es necesario incorporar ese dato para evitar problemas futuros entre las tablas 'user' y 'transaction', ya que los 'user\_id' de la tabla 'transaction' deben existir previamente en la tabla 'user'.

Igual que en el ejercicio 3 – Nivel 1, solo se crea el 'id' correspondiente a la tabla 'user' por ser información brindada e ingresada en la tabla 'transaction', quedando pendiente completar la información de este 'id' en la tabla 'user', información que se debe solicitar al depto correspondiente de forma inmediata posterior para mantener correctamente la base de datos.

--- Se verifica si existe o no en tabla 'user' el id '9999'.

```
189 • SELECT *
190 FROM user
191 WHERE id = '9999';
```

id	name	surname	phone	personal_email	birth_date	country	city	postal_code	address

user 6 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:01:06	SELECT * FROM user WHERE id = '9999'	0 row(s) returned	0.000 sec / 0.000 sec

--- Se crea el id '9999' en la tabla 'user' (porque no existe) y se verifica que el nuevo id esté en la tabla 'user'.

```
194 • INSERT INTO user (id)
195 VALUES ('9999');
196
197 • SELECT *
198 FROM user
199 WHERE id = '9999';
```

id	name	surname	phone	personal_email	birth_date	country	city	postal_code	address
9999									

user 7 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:02:38	INSERT INTO user (id) VALUES ('9999')	1 row(s) affected	0.000 sec
2	00:02:42	SELECT * FROM user WHERE id = '9999'	1 row(s) returned	0.000 sec / 0.000 sec

Paso 5: actualizar tabla 'transaction' con FK 'user\_id' para establecer la relación de 1 a N, entre la tabla 'data\_user' y la tabla 'transaction'.

```
202 • ALTER TABLE transaction
203 ADD FOREIGN KEY (user_id) REFERENCES user(id);
204
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:06:34	ALTER TABLE transaction ADD FOREIGN KEY (user_id) REFERENCE...	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0	3.546 sec

Paso 6: Se renombra la tabla 'user' como 'data\_user'

```
206 • RENAME TABLE user TO data_user;
207
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:10:55	RENAME TABLE user TO data_user	0 row(s) affected	0.047 sec

Paso 7: en la tabla 'company' eliminar la columna 'website'.

--- Se verifican las columnas de la tabla 'company'

210 • DESCRIBE company;

Result Grid

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

Result 8 x Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:14:54	DESCRIBE company	6 row(s) returned	0.000 sec / 0.000 sec

212 • ALTER TABLE company

213 DROP website;

214

215 • DESCRIBE company;

Result 9 x Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:17:38	ALTER TABLE company DROP website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
2	00:17:41	DESCRIBE company	5 row(s) returned	0.016 sec / 0.000 sec

--- Paso 8: en la tabla 'credit\_card' agregar nueva columna 'fecha\_actual' DATE

--- se verifica si existe la columna 'fecha\_Actual' DATE en la tabla 'credit\_card'

219 • DESCRIBE credit\_card;

Result Grid

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	varchar(3)	YES		NULL	
expiring_date	varchar(15)	YES		NULL	

Result 10 x Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:21:36	DESCRIBE credit_card	5 row(s) returned	0.000 sec / 0.000 sec

--- se agrega la nueva columna 'fecha\_actual' DATE y se verifica nuevamente la tabla 'credit\_card'

```
221 • ALTER TABLE credit_card
222   ADD fecha_actual DATE;
223
224 • DESCRIBE credit_card;
225
```

The screenshot shows the SQL IDE interface. The 'Result Grid' displays the structure of the 'credit\_card' table after adding the 'fecha\_actual' column. The 'Action Output' pane shows the execution of two commands: an ALTER TABLE statement and a DESCRIBE statement.

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	varchar(3)	YES		NULL	
expiring_date	varchar(15)	YES		NULL	
fecha_actual	date	YES		NULL	

#	Time	Action	Message	Duration / Fetch
1	00:25:39	ALTER TABLE credit_card ADD fecha_actual DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.047 sec
2	00:25:43	DESCRIBE credit_card	6 row(s) returned	0.015 sec / 0.000 sec

--- Paso 9: en la tabla 'credit\_card' cambiar los tipos de datos de 'id' a VARCHAR(20), 'cvv' a INT y 'expiring\_date' a VARCHAR(20).

--- se verifica las columnas e información de la tabla 'credit\_card'.

```
228 • DESCRIBE credit_card;
229
```

The screenshot shows the SQL IDE interface. The 'Result Grid' displays the structure of the 'credit\_card' table. The 'Action Output' pane shows the execution of a DESCRIBE command.

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	varchar(3)	YES		NULL	
expiring_date	varchar(15)	YES		NULL	
fecha_actual	date	YES		NULL	

#	Time	Action	Message	Duration / Fetch
1	00:30:16	DESCRIBE credit_card	6 row(s) returned	0.000 sec / 0.000 sec

--- se realiza el cambio de tipo de dato en las variables solicitadas (id, cvv y expiring\_date) y luego se verifica los cambios en dicha tabla.

```
230 • ALTER TABLE credit_card
231   MODIFY id VARCHAR(20),
232   MODIFY cvv INT,
233   MODIFY expiring_date VARCHAR(20);
234
235 • DESCRIBE credit_card;
236
```

The screenshot shows the SQL IDE interface. The 'Result Grid' displays the structure of the 'credit\_card' table after modifying the data types. The 'Action Output' pane shows the execution of an ALTER TABLE statement and a DESCRIBE statement.

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

#	Time	Action	Message	Duration / Fetch
1	00:32:08	ALTER TABLE credit_card MODIFY id VARCHAR(20), ...	5001 row(s) affected Records: 5001 Duplicates: 0 War...	0.296 sec
2	00:32:12	DESCRIBE credit_card	6 row(s) returned	0.000 sec / 0.000 sec



Paso 10: en la tabla 'transaction' cambiar el tipo de dato de 'credit\_card\_id' VARCHAR(15) por VARCHAR(20).

-- se verifica las columnas e información de la tabla 'transaction'.

```
239 • DESCRIBE transaction;
```

Result Grid

Field	Type	Null	Key	Default	Extra
id	varchar(255)	NO	PRI		
credit_card_id	varchar(15)	YES	MUL		
company_id	varchar(20)	YES	MUL		
user_id	int	YES	MUL		
lat	float	YES			
longitude	float	YES			
timestamp	timestamp	YES			

Result 14 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:34:03	DESCRIBE transaction	9 row(s) returned	0.015 sec / 0.000 sec

-- se realiza el cambio solicitado del tipo de dato de 'credit\_card\_id' y luego verificar tabla 'transaction'.

```
241 • ALTER TABLE transaction
```

```
242   MODIFY credit_card_id VARCHAR(20);
```

```
243  
244 • DESCRIBE transaction;
```

Result Grid

Field	Type	Null	Key	Default	Extra
id	varchar(255)	NO	PRI		
credit_card_id	varchar(20)	YES	MUL		
company_id	varchar(20)	YES	MUL		
user_id	int	YES	MUL		
lat	float	YES			
longitude	float	YES			
timestamp	timestamp	YES			

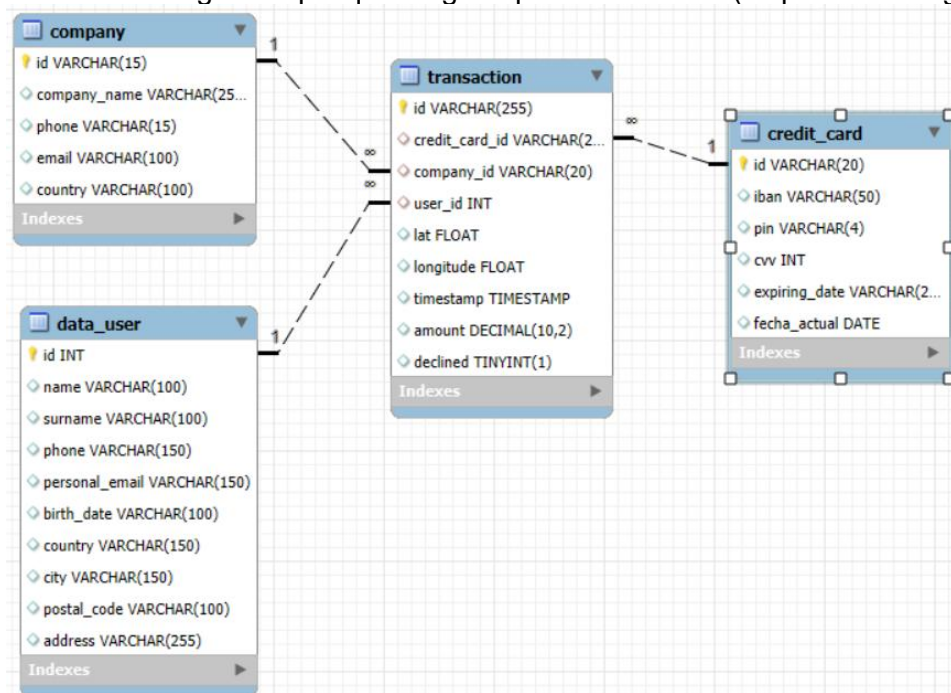
Result 15 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:37:18	ALTER TABLE transaction MODIFY credit_card_id VAR...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec
2	00:37:21	DESCRIBE transaction	9 row(s) returned	0.000 sec / 0.000 sec

Finalmente, se obtiene el diagrama que queda igual que el solicitado: (se presenta 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 columna ID de transacción.

**Respuesta:**

```
251 CREATE VIEW InformeTecnico (id_transaccion, nombre_usuario, apellido_usuario, iban_tarj_cred, nombre_empresa, importe_transaccion, fecha_transaccion, hora_transaccion,
252 lat_transaccion, longitud_transaccion, transaccion_declinada, fecha_exp_tarj_cred, pais_usuario, pais_empresa) AS
253 SELECT t.id, d.name, d.surname, e.iban, c.company_name, t.amount, DATE(t.timestamp), TIME(t.timestamp), t.lat, t.longitude, t.declined, e.expiring_date, d.country, c.country
254 FROM transaction t
255 JOIN company c ON c.id = t.company_id
256 JOIN data_user d ON d.id = t.user_id
257 JOIN credit_card e ON e.id = t.credit_card_id
258 ORDER BY t.id DESC;
259
260 SELECT * FROM InformeTecnico;
```

id_transaccion	nombre_usuario	apellido_usuario	iban_tarj_cred	nombre_empresa	importe_transaccion	fecha_transaccion	hora_transaccion	lat_transaccion	longitud_transaccion	tr
FFFD31D6-9495-47CE-B54A-70B8E1CC274B	Bmrglj	Tprvmrc	XX794814451211289182490922	Turpis Company	74.54	2021-08-05	12:26:24	55.6384	-3.6166	0
FFFC76D0-ECF0-4985-A2D0-B2A7675998FC	Dfiedl	Vlqcdl	XX636251701647892036676034	Amet Nulla Donec Corporation	148.91	2023-06-17	19:10:30	52.0635	4.30862	0
FFFC9E80-27C7-4ADE-98F2-7533EF40F126	Securp	Faofvqfy	XX162677143304223631437567	Nunc Interdum Incorporated	234.22	2019-10-30	02:31:37	59.3327	18.0753	0
FFFB27D0-F53A-4D5D-9666-E5307C53CC84	Gzjzpa	Urzjzlh	XX395114267082019952567052	Viverra Donec Foundation	349.13	2024-05-13	03:42:03	39.0271	-8.10523	0
FFFB2CE-234E-408C-A8EF-F9CAD577224A	Yshmq	Zpsjleed	XX8845462156537570367941	Convallis In Incorporated	247.39	2022-12-17	20:57:55	52.5223	13.4049	0
FFFB178-6CD2-4D9F-99B0-49AE068809B1	Jevepx	Xwcvzwrm	XX321405515711654384711481	Mus Aenean Eget Foundation	438.13	2023-02-08	10:01:48	52.5489	5.48015	0
FFFB67C9-17B5-4B1F-AFD9-F8023AAA449E	Fqjngd	Lvhfyxi	XX278446342932680979729426	Cras Vehicula Aliquet Industries	448.63	2015-04-12	10:13:11	55.0648	-2.97111	0
FFF7042D-18C6-4D0D-823C-4D90A4AC8F26	Njoraa	Egqsquii	XX405009272572550082027209	Placerat LLP	163.35	2017-07-20	22:47:05	39.0305	-8.48542	0
FFF66D04-4244-47F6-9210-E5D1DCB99D80	Lopzaj	Itgrfyay	XX63376659736627454015125	Pede Cum Ltd	288.76	2017-05-18	06:22:25	32.7781	-96.801	0
FFF5C660-4441-436D-8D27-E6C5B618622	Gmnbnu	Oxdvhkl	XX237820256172646394016483	At Associates	53.31	2018-11-30	01:39:51	39.9472	-75.1665	0
FFF54F54-8439-41F0-8DD2-F7332DC1ACAD	Gqcfyy	Mpifnltn	XX802723943240147612158718	Enim Conditum Ltd	128.48	2021-03-18	16:30:41	52.5717	5.23023	0

InformeTecnico 17 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:49:31	CREATE VIEW InformeTecnico (id_transaccion, nombre_usuario, apellido_usuario, iban_tarj_cred, nombre_empresa, imp...	0 row(s) affected	0.031 sec
2	00:49:37	SELECT * FROM InformeTecnico	100000 row(s) returned	0.687 sec / 0.125 sec

Al ser un Informe técnico, se entiende que otro profesional lo utilizará como insumo para realizar reportes finales. Por ello se ha considerado útil adicionar las siguientes columnas:

De la tabla 'transaction': importe, fecha y hora, latitud y longitud (coordenadas de ubicación), declinada (si fueron aceptadas o no).

De la tabla 'credit\_card': fecha de expiración de la tarjeta de crédito.

De la tabla 'data\_user': país del usuario.

De la tabla 'company': país de empresa.

Asimismo, se han considerado todas las transacciones realizadas, declinadas o no.

Se ha realizado esta selección de información, debido a que el analista técnico puede utilizarla para realizar diversos reportes, como por ejemplo:

Realizar reportes de análisis de las transacciones efectivas (t.declined=0) identificando los países y ciudades dentro de ellos donde se ubique el mayor o menor importe de transacciones efectivas acumulado, o identificar los usuarios / empresas con un mayor importe acumulado en total o por países y ciudades, incluso podría analizarse el período de tiempo en que esto ocurre (días de la semana con mayor cantidad e importe de transacciones e incluso las horas en que se realizan).

Reportes similares se pueden realizar, pero orientados a las transacciones declinadas (t.declined=1), que luego podría complementarse con un análisis de causas posibles y consecuentes acciones o campañas para mitigarlas.

Revisión con ayuda de Gabriel.