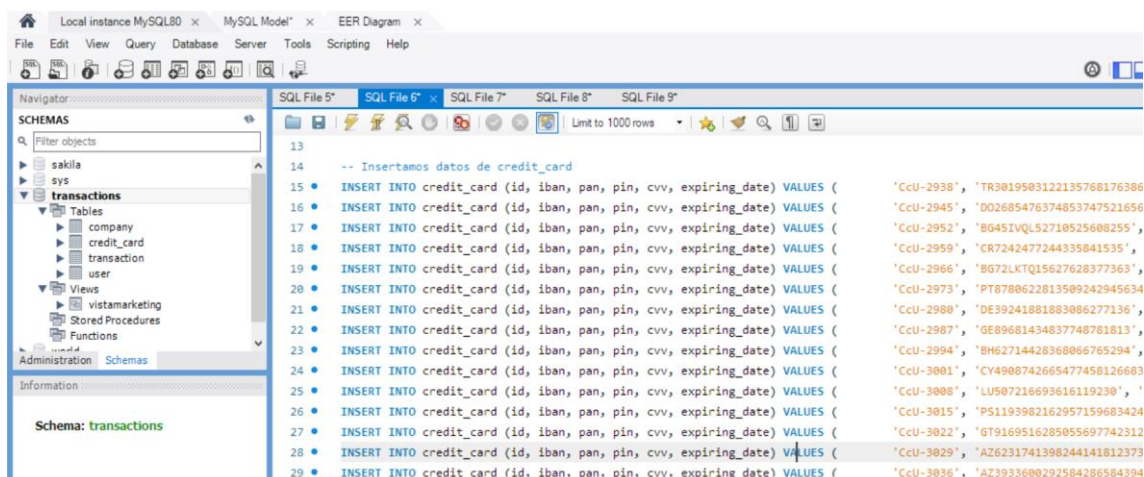


NIVEL 1

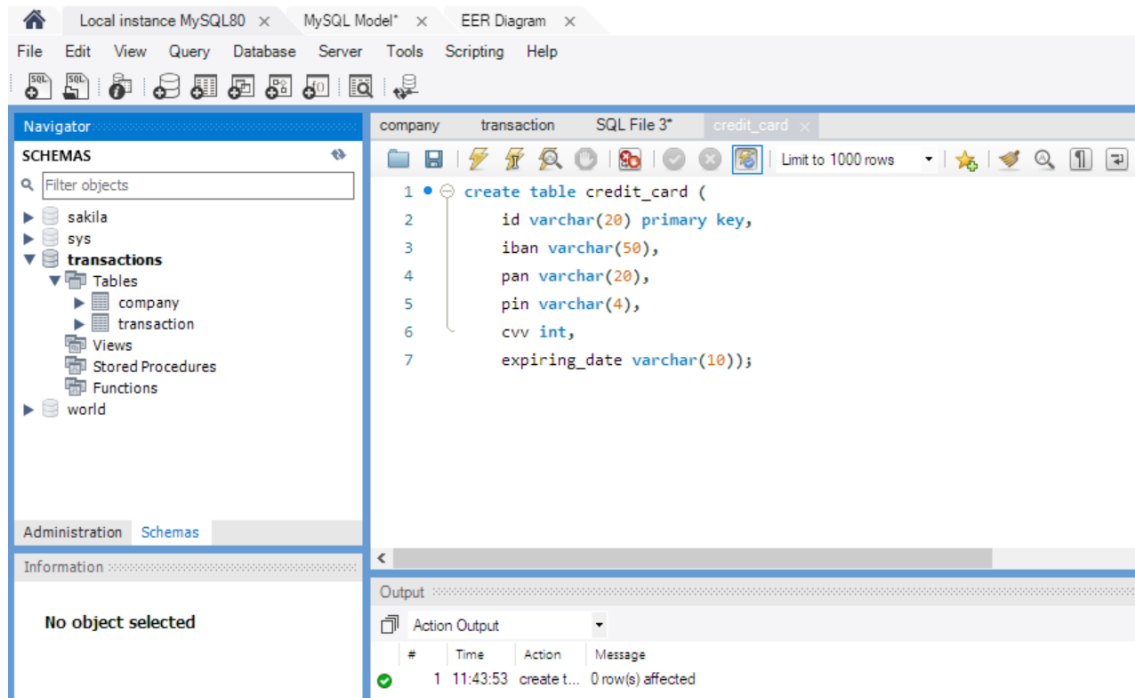
NIVEL 1-EXERCICI 1

La teva tasca és dissenyar i crear una taula anomenada "credit_card" que emmagatzemi detalls crucials sobre les targetes de crèdit. La nova taula ha de ser capaç d'identificar de manera única cada targeta i establir una relació adequada amb les altres dues taules ("transaction" i "company"). Després de crear la taula serà necessari que ingressis la informació del document denominat "dades_introduir_credit". Recorda mostrar el diagrama i realitzar una breu descripció d'aquest.

Para poder cargar los datos en una tabla 'credit_card' tendrá que tener la misma estructura, por tanto vemos como es la estructura de los datos que nos dan:

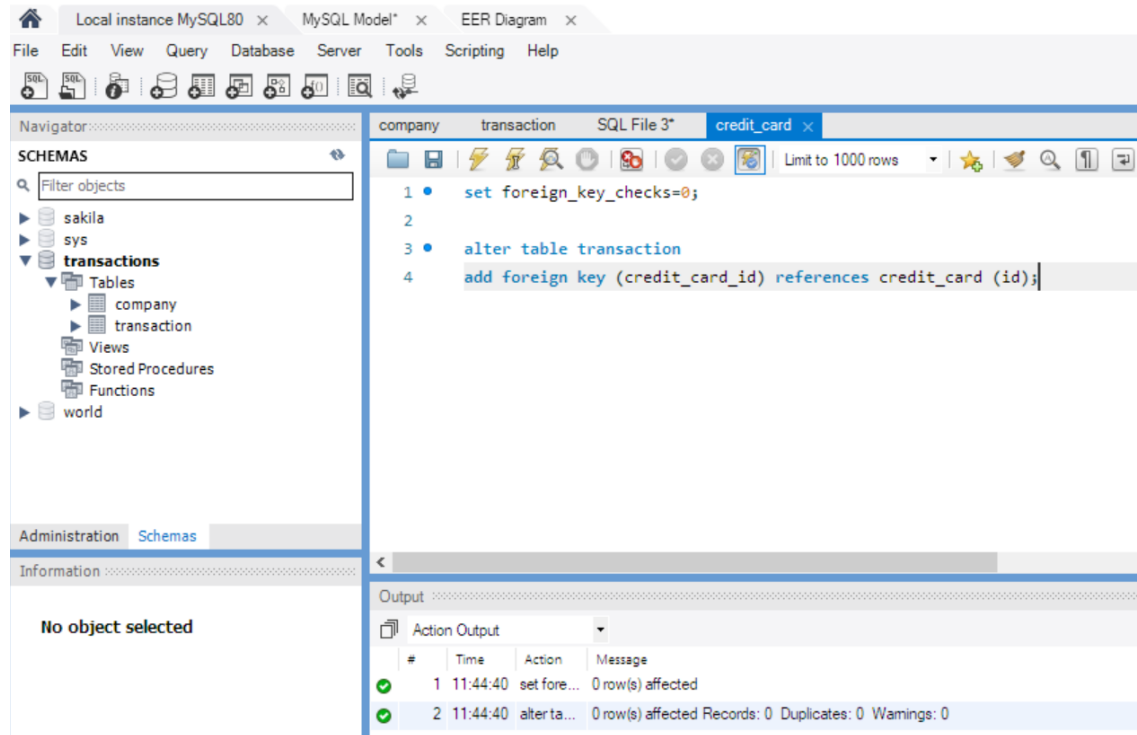


Creamos la tabla `credit_card` con la misma estructura de los datos que nos dan en el ejercicio en el archivo 'dades_introduir_credit'

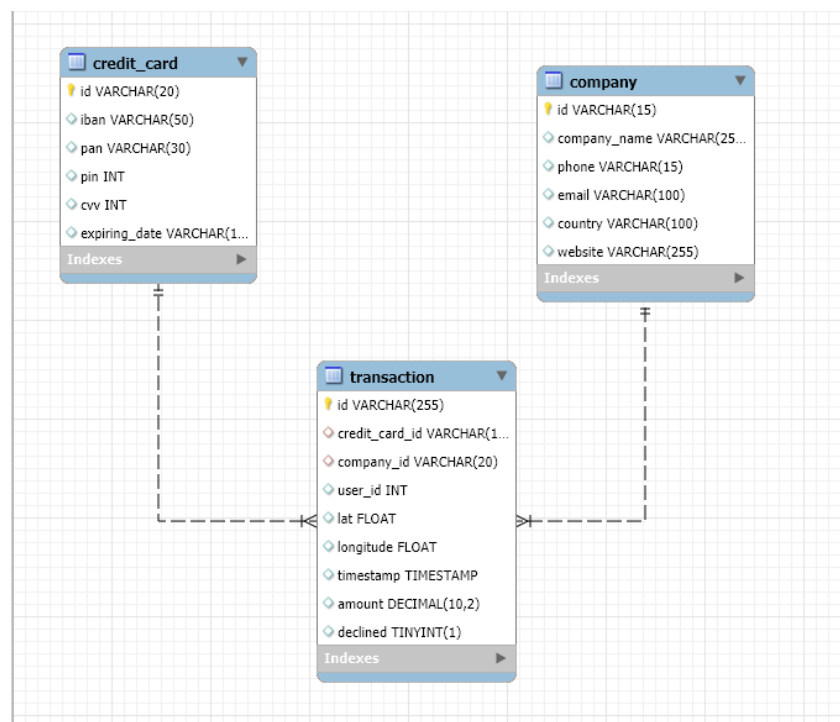


Referenciamos la primary key 'id' de la tabla recién creada con el campo 'credit_card_id'.

Si lo necesitamos deshabilitamos las restricciones de 'foreign key' con
`set foreign_key_checks=0;`



Quedando la Base de datos



NIVEL 1-EXERCICI 2

El departament de Recursos Humans ha identificat un error en el número de compte de l'usuari amb ID CcU-2938. La informació que ha de mostrar-se per a aquest registre és: R323456312213576817699999. Recorda mostrar que el canvi es va realitzar.

Después de cargar los datos suministrados en el ejercicio.

Vemos que está compuesta por 275 filas de datos.

The screenshot shows the MySQL Modeler interface. The Navigator pane on the left displays the database schema, including the 'transactions' database and the 'credit_card' table. The main query editor shows the following SQL query:

```
select * from credit_card
```

The Result Grid displays the first few rows of the query results:

	id	iban	pan	pin	cvv	expiring_date
▶	CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22
	CcU-2945	DO26854763748537475216568689	5142423821948828	9080	887	08/24/23
	CcU-2952	BG45IVQL52710525608255	4556 453 55 5287	4598	438	06/29/21
	CcU-2959	CR7242477244335841535	372461377349375	3583	667	02/24/23
	CcU-2966	BG72LKTQ15627628377363	448566 886747 7265	4900	130	10/29/24
	CcU-2973	PT878N6778135002420446346	544 58654 54343 384	8760	887	01/30/25

The Output pane at the bottom shows the execution results:

#	Time	Action	Message
1	11:47:36	select * ...	275 row(s) returned

Buscamos en la tabla 'credit_card' si existe ese usuario con id =CcU-2938

The screenshot shows the MySQL Modeler interface with the following SQL query:

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

The Result Grid displays the query results:

	id	iban	pan	pin	cvv	expiring_date
	CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22
▶	NULL		NULL	NULL	NULL	NULL

Nota: Si lo necesitamos deshabilitamos las restricciones de 'foreign key' con

```
set foreign_key_checks=0;
```

Con el comando update sustituimos el 'iban' y comprobamos el resultado

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'sakila' and 'sys' databases, and the 'transactions' database expanded to show the 'credit_card' table. The main window shows a query editor with the following SQL commands:

```
1 • update credit_card set iban = 'R323456312213576817699999' where id = 'CcU-2938';
2 • select * from credit_card
3 • where id = 'CcU-2938';
```

Below the query editor, the 'Result Grid' shows the data for the 'credit_card' table. The table has columns: id, iban, pan, pin, cvv, and expiring_date. The data row shows the updated record for 'CcU-2938'.

id	iban	pan	pin	cvv	expiring_date
CcU-2938	R323456312213576817699999	5424465566813633	3257	984	10/30/22

The bottom section of the interface shows the 'Table: credit_card' structure:

Columns:

- id: varchar(20) PK
- iban: varchar(50)
- pan: varchar(30)
- pin: int
- cvv: int
- expiring_date: varchar(10)

NIVEL 1-EXERCICI 3

En la taula "transaction" ingresa un nou usuari amb la següent informació:

Id 108B1D1D-5B23-A76C-55EF-C568E49A99DD

credit_card_id CcU-9999

company_id b-9999

user_id 9999

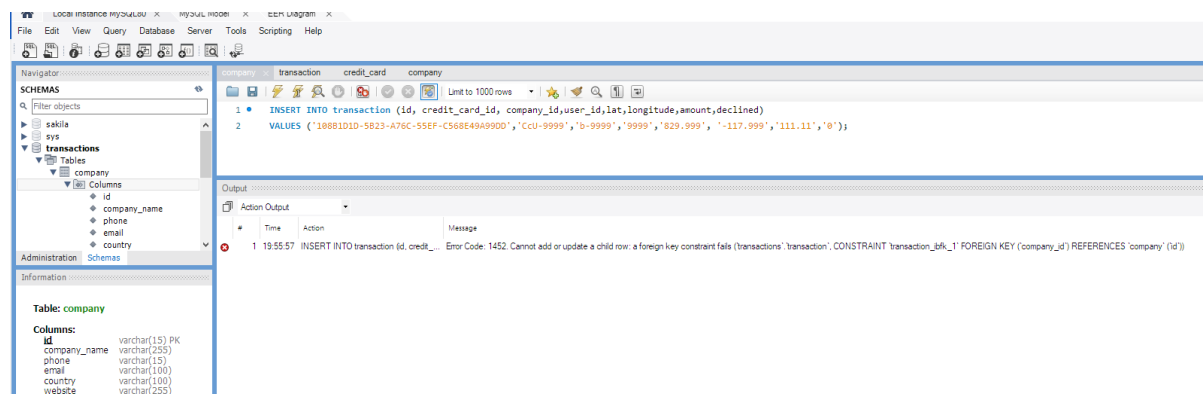
lat 829.999

longitude -117.999

amount 111.11

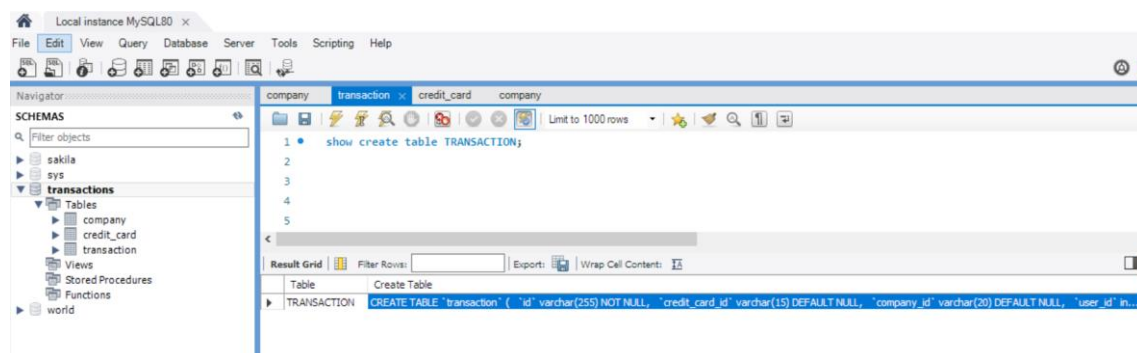
declined 0

al aplicar el comando 'insert' nos da un error relacionado con la foreign key.



Es un error que indica que hay una violación de la 'foreign key constraint' probablemente la tabla 'transaction' tiene restricciones que impiden añadir filas que cuya 'foreign key' no exista en la tabla padre

Mediante 'show create table' nos muestran las restricciones al crear la tabla 'transaction'

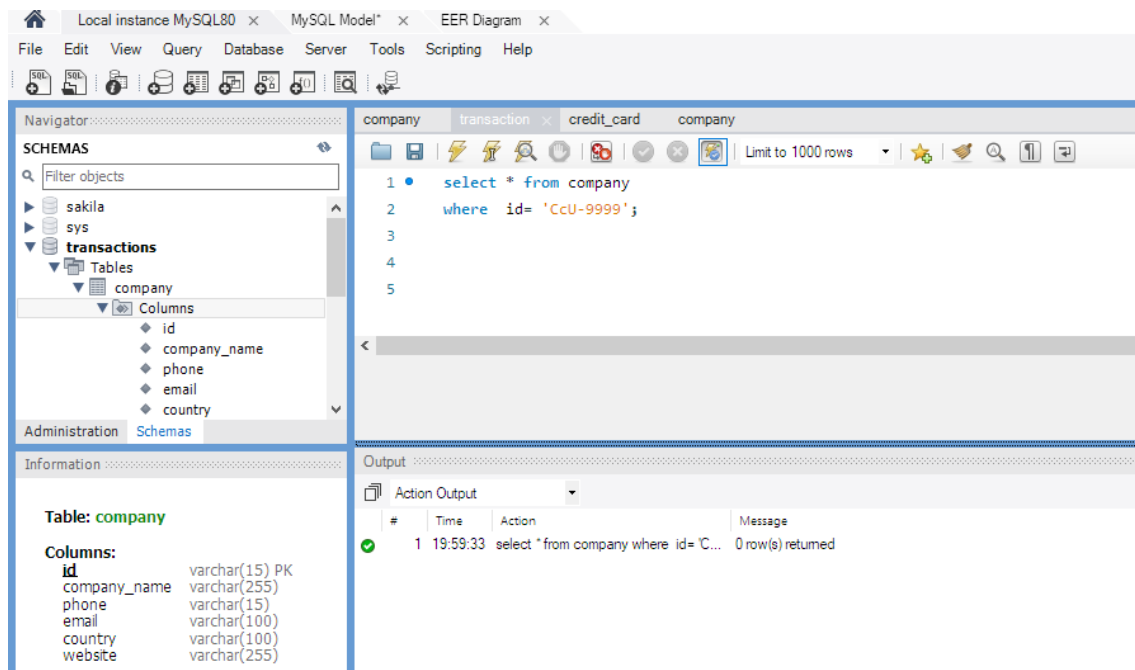


```

CREATE TABLE `transaction` (
  `id` varchar(255) NOT NULL,
  `credit_card_id` varchar(15) DEFAULT NULL,
  `company_id` varchar(20) DEFAULT NULL,
  `user_id` int DEFAULT NULL,
  `lat` float DEFAULT NULL,
  `longitude` float DEFAULT NULL,
  `timestamp` timestamp NULL DEFAULT NULL,
  `amount` decimal(10,2) DEFAULT NULL,
  `declined` tinyint(1) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `company_id` (`company_id`),
  KEY `credit_card_id` (`credit_card_id`),
  CONSTRAINT `transaction_ibfk_1` FOREIGN KEY (`company_id`)
REFERENCES `company` (`id`),
  CONSTRAINT `transaction_ibfk_2` FOREIGN KEY
(`credit_card_id`) REFERENCES `credit_card` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

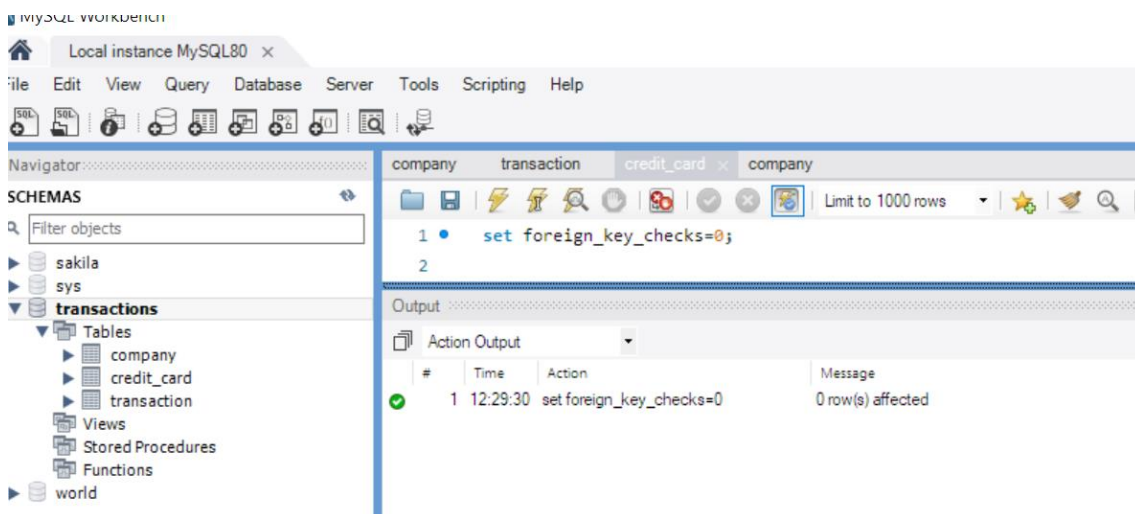
```

Si comprobamos que existe el valor de la foreign key 'credit_card_id' = CcU-999 en la tabla 'Company'



Efectivamente no existe ese valor para la 'primary key' por tanto no nos dejará introducir nuevos datos.

Utilizamos el comando 'set' para eliminar temporalmente estas restricciones



Ahora si introducimos los datos y vemos que ya es posible



Comprobamos que ha sido introducida la id con el valor correcto

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

sakila

sys

transactions

Tables

company

credit_card

transaction

Views

Stored Procedures

Functions

world

company transaction credit_card company

Limit to 1000 rows

1

2 • SELECT * FROM transaction

3 where id='108B1D1D-5B23-A76C-55EF-C568E49A99DD';

Result Grid

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

Administration Schemas

transaction 2 x

Information

Schema: transactions

Output

Action Output

#	Time	Action	Message
1	12:45:58	SELECT * FROM transaction where...	1 row(s) returned

NIVEL 1- EXERCICI 4

Des de recursos humans et sol·liciten eliminar la columna "pan" de la taula credit_*card. Recorda mostrar el canvi realitzat.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'transactions' expanded, showing tables 'company', 'credit_card', and 'transaction'. The main editor contains the following SQL script:

```
1 ALTER TABLE credit_card
2 DROP pan;
3
```

The 'Output' pane at the bottom shows the execution results:

#	Time	Action	Message
1	12:55:59	ALTER TABLE credit_card drop pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Comprobamos que el campo ya no existe

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'transactions' expanded. The main editor contains the following SQL script:

```
1 SELECT * FROM credit_card;
```

The 'Result Grid' shows the data returned by the query:

id	iban	pin	cvv	expiring_date
b-9999	NULL	NULL	NULL	NULL
CcU-2938	R323456312213576817699999	3257	984	10/30/22
CcU-2945	DO26854763748537475216568689	9080	887	08/24/23
CcU-2952	BG45IVQL52710525608255	4598	438	06/29/21
CcU-2959	CR7242477244335841535	3583	667	02/24/23
CcU-2966	BG72LKTQ15627628377363	4900	130	10/29/24
CcU-2973	PT87806228135092429456346	8760	887	01/30/25

The 'Output' pane at the bottom shows the execution results:

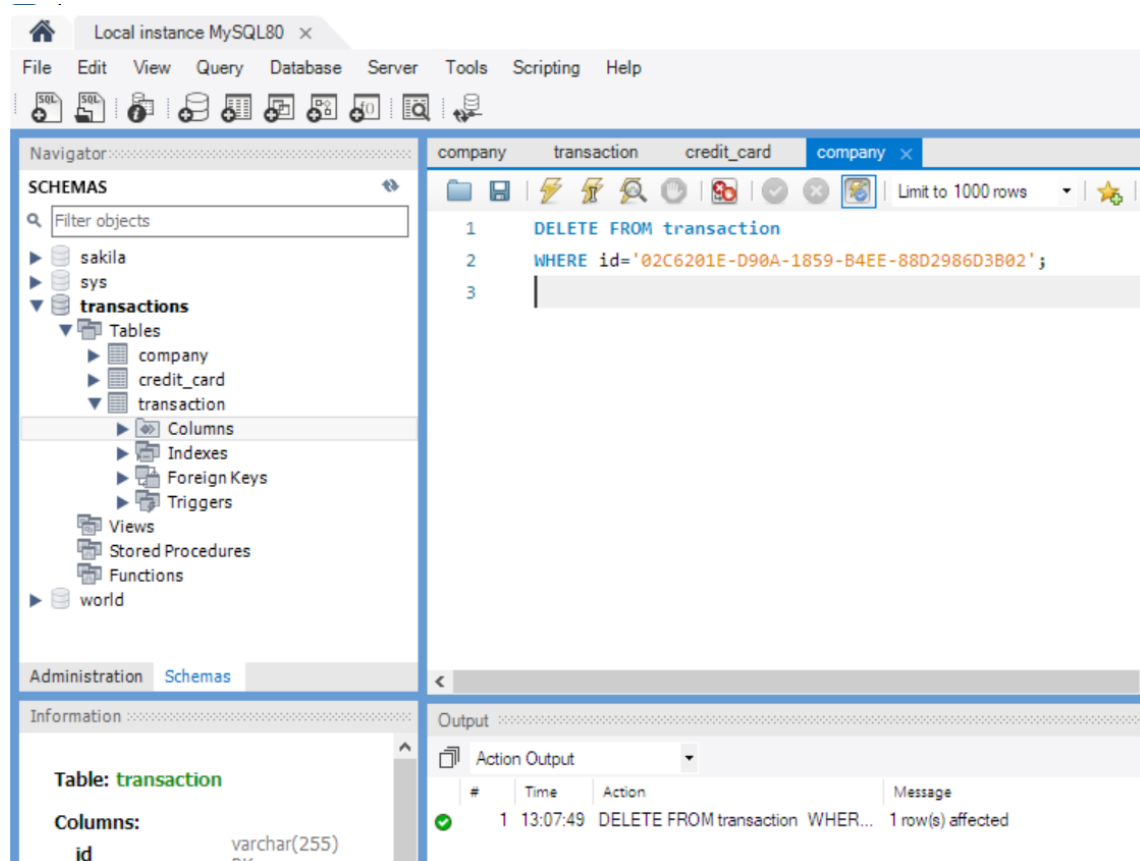
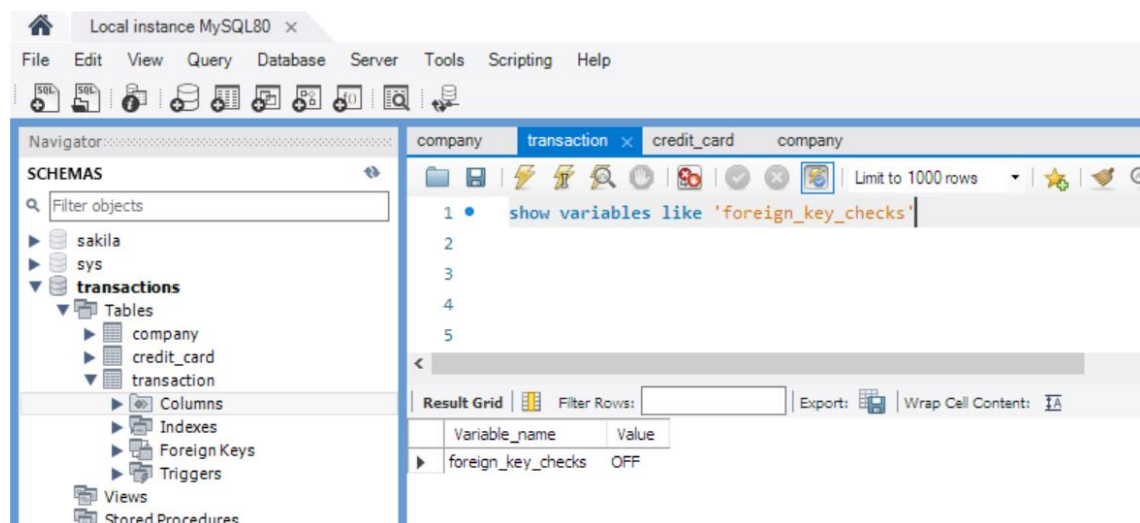
#	Time	Action	Message
1	12:58:53	SELECT * FROM credit_card LIMIT ...	276 row(s) returned

NIVEL 2- EXERCICI 1

Elimina de la taula transaction el registre amb ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de dades.

Mediante el comando 'DELETE FROM' borramos la transacción.

Nota: seguimos en la misma sesión del nivel 1 por tanto las restricciones de las foreign key siguen desactivadas. Sino habría que deshabilitarlas mediante el comando 'set'



Comprobamos que ya no existe la transacción

The screenshot shows the MySQL Workbench interface for a local instance of MySQL 8.0. The 'Schemas' pane on the left shows the 'sakila' database selected, with the 'transaction' table highlighted under the 'Tables' folder. The 'Query' tab is active, displaying a SQL query to delete a transaction record.

```
1 SELECT * FROM transaction
2 WHERE id='02C6201E-D90A-1859-B4EE-88D2986D3B02';
3
```

The 'Result Grid' shows the result of the query, which is an empty table with 9 columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. All values are NULL.

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

The 'Output' pane at the bottom shows the execution log with two actions:

#	Time	Action	Message
1	13:07:49	DELETE FROM transaction WHE...	1 row(s) affected
2	13:10:24	SELECT * FROM transaction WHE...	0 row(s) returned

The 'Information' pane on the left shows the structure of the 'transaction' table:

Table: transaction

Columns:

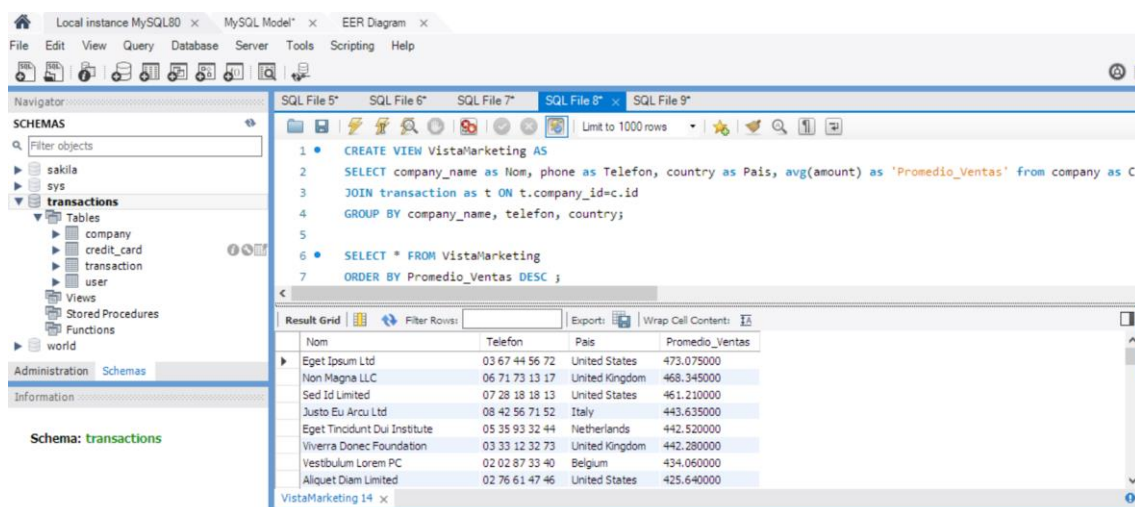
- id: varchar(255) PK
- credit_card_id: varchar(15)

NIVEL 2- Exercici 2

La secció de màrqueting desitja tenir accés a informació específica per a realitzar anàlisi i estratègies efectives. S'ha sol·licitat crear una vista que proporcioni detalls clau sobre les companyies i les seves transaccions. Serà necessària que creïs una vista anomenada VistaMarketing que contingui la següent informació: Nom de la companyia. Telèfon de contacte. País de residència. Mitjana de compra realitzat per cada companyia. Presenta la vista creada, ordenant les dades de major a menor mitjana de compra.

Creo una vista basada en una query en la cual mediante join entre 'company' y 'transaction' agrupada por nombre de compañía.

El output se realiza mediante un SELECT de la view creada como si fuera una tabla y ordenada por el 'Promedio_Ventas'



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'sakila' and 'sys' databases, and a 'transactions' schema containing tables 'company', 'credit_card', 'transaction', and 'user'. The main editor window shows the SQL script for creating the 'VistaMarketing' view. The script consists of two parts: a CREATE VIEW statement and a SELECT statement. The CREATE VIEW statement defines the view as a SELECT query joining 'company' and 'transaction' tables, grouping by company name, phone, and country, and calculating the average amount. The SELECT statement then queries the 'VistaMarketing' view, ordering the results by 'Promedio_Ventas' in descending order. The bottom pane shows the 'Result Grid' with 10 rows of data, including company names, phone numbers, countries, and average sales amounts.

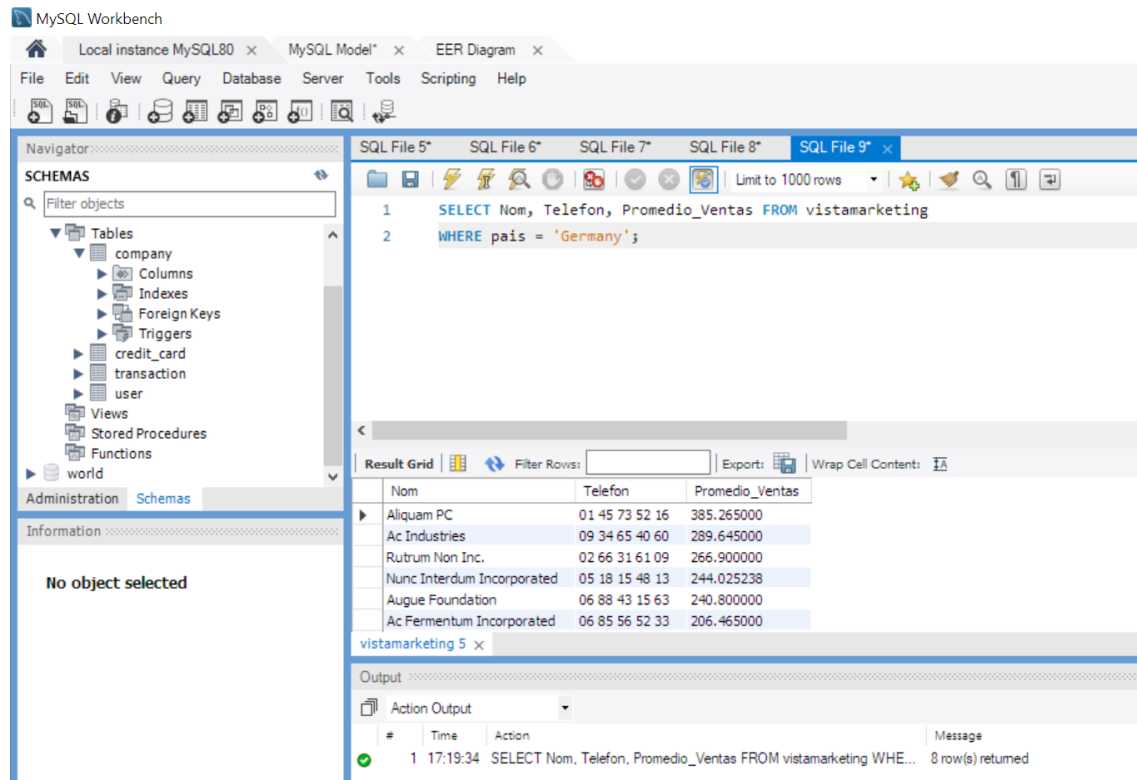
```
1 CREATE VIEW VistaMarketing AS
2 SELECT company_name as Nom, phone as Telefon, country as Pais, avg(amount) as 'Promedio_Ventas' from company as c
3 JOIN transaction as t ON t.company_id=c.id
4 GROUP BY company_name, telefon, country;
5
6 SELECT * FROM VistaMarketing
7 ORDER BY Promedio_Ventas DESC ;
```

Nom	Telefon	Pais	Promedio_Ventas
Eget Iosum Ltd	03 67 44 56 72	United States	473.075000
Non Magna LLC	06 71 73 13 17	United Kingdom	468.345000
Sed Id Limited	07 28 18 18 13	United States	461.210000
Justo Eu Arcu Ltd	08 42 56 71 52	Italy	443.635000
Eget Tincidunt Dui Institute	05 35 93 32 44	Netherlands	442.520000
Viverra Donec Foundation	03 33 12 32 73	United Kingdom	442.280000
Vestibulum Lorem PC	02 02 87 33 40	Belgium	434.060000
Aliquet Diam Limited	02 76 61 47 46	United States	425.640000

NIVEL 2-EXERCICI 3

Filtra la vista VistaMarketing per a mostrar només les companyies que tenen el seu país de residència en "Germany"

Se resuelve mediante una query como si se tratara de una tabla.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'world' selected. The main editor shows a SQL query in 'SQL File 9*':

```
1 SELECT Nom, Telefon, Promedio_Ventas FROM vistamarketing
2 WHERE pais = 'Germany';
```

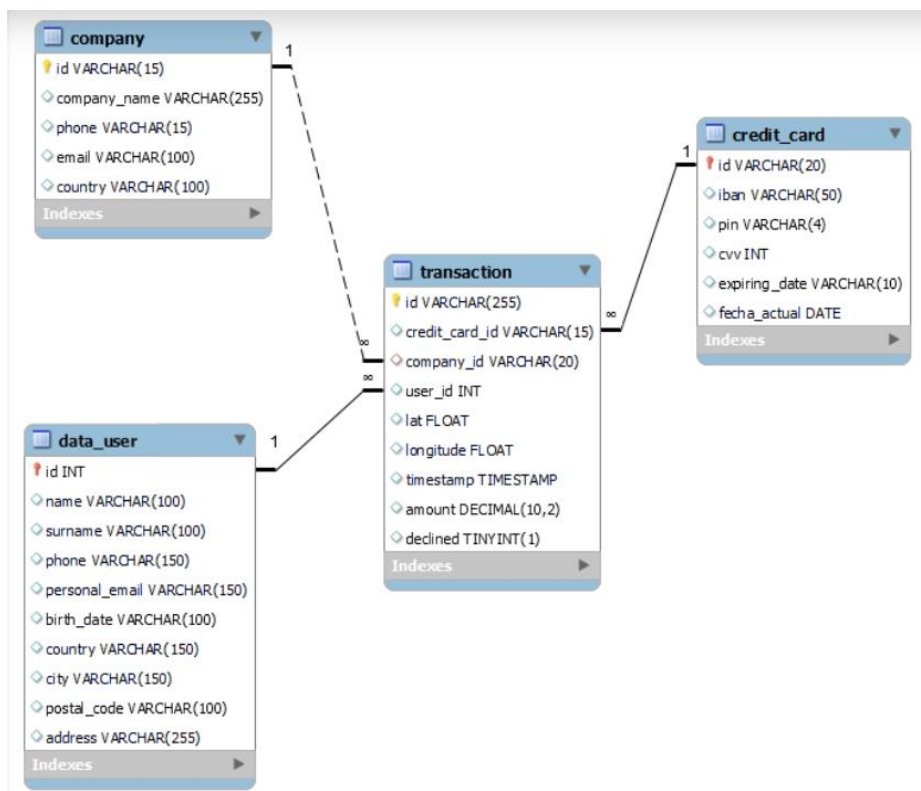
The 'Result Grid' shows the following data:

Nom	Telefon	Promedio_Ventas
Aliquam PC	01 45 73 52 16	385.265000
Ac Industries	09 34 65 40 60	289.645000
Rutrum Non Inc.	02 66 31 61 09	266.900000
Nunc Interdum Incorporated	05 18 15 48 13	244.025238
Augue Foundation	06 88 43 15 63	240.800000
Ac Fermentum Incorporated	06 85 56 52 33	206.465000

The 'Output' pane at the bottom shows the execution message: '1 17:19:34 SELECT Nom, Telefon, Promedio_Ventas FROM vistamarketing WHE... 8 row(s) returned'.

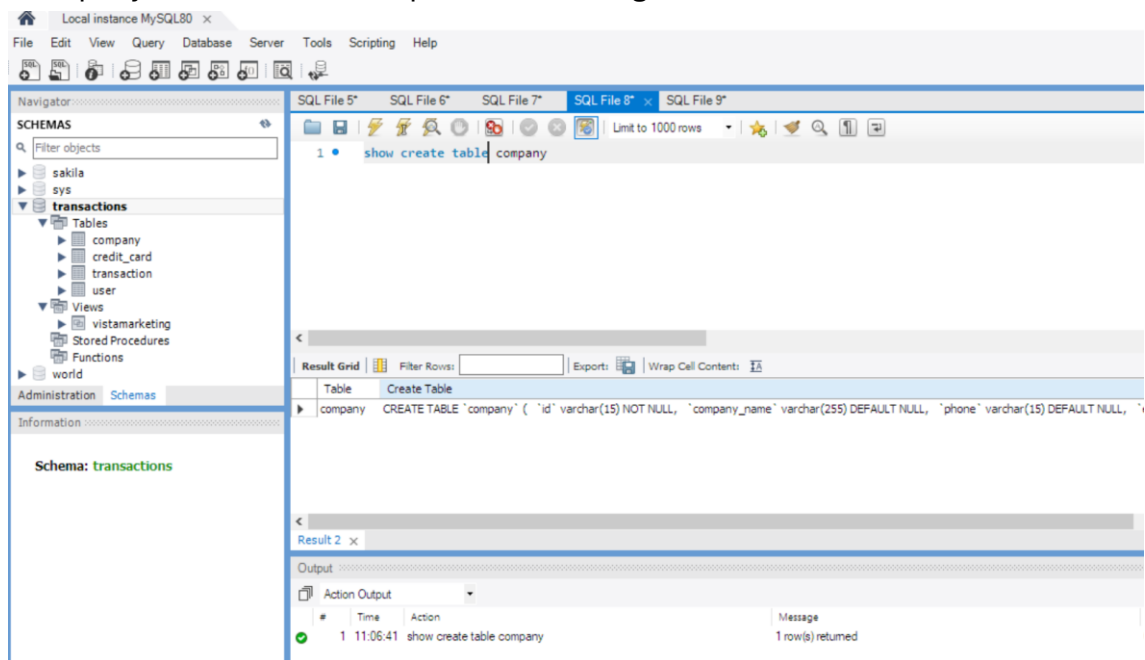
NIVEL 3- EXERCICI 1

La setmana vinent tindràs una nova reunió amb els gerents de màrqueting. Un company del teu equip va realitzar modificacions en la base de dades, però no recorda com les va realitzar. Et demana que l'ajudis a deixar els comandos executats per a obtenir el següent diagrama:

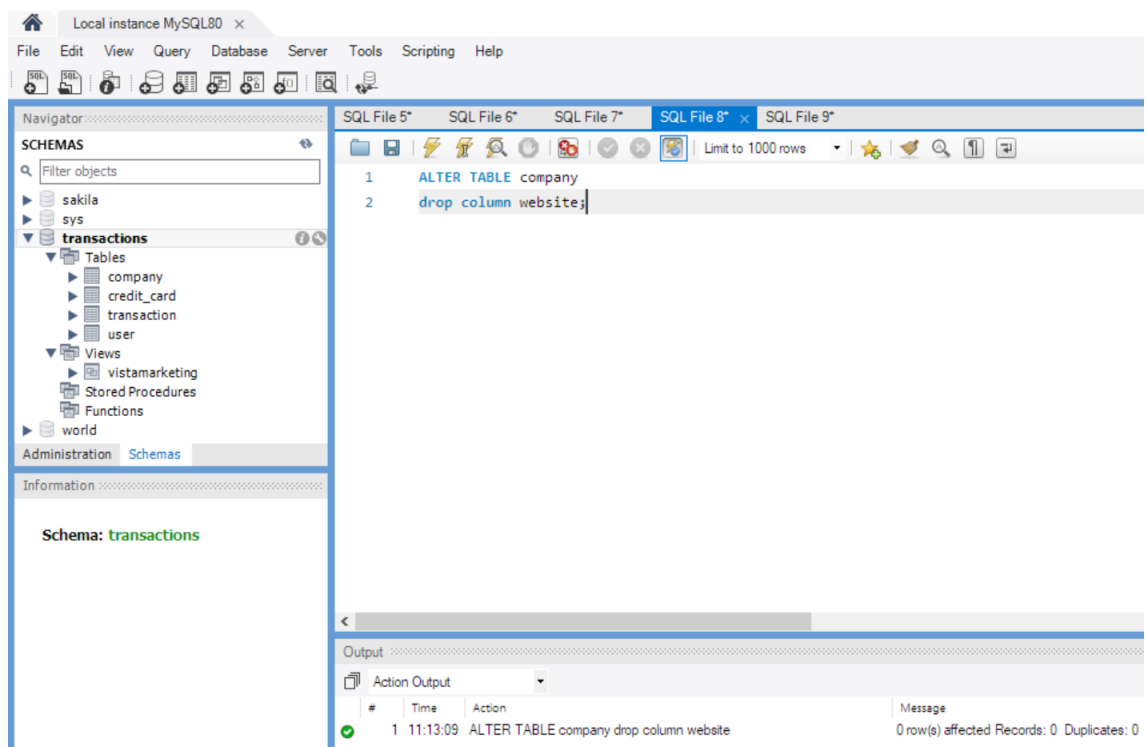


Vamos comparando la estructura de la BBDD del gráfico con lo que tenemos cargado en mysql.

Por ejemplo, uso SHOW CREATE TABLE company para ver la estructura de tabla 'company' con la estructura que me indica el gráfico.



Compruebo que no tiene el campo 'website' por tanto elimino el campo



Compruebo el resultado con el comando Describe

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS

Filter objects

sakila
sys
▼ transactions

Tables

- company
- credit_card
- transaction
- user

Views

- vistamarketing

Stored Procedures

Functions

world

Administration Schemas

Information: Schema: transactions

SQL File 5* SQL File 6* SQL File 7* SQL File 8* x SQL File 9*

1 • 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	

Result 3 x

Output

Action Output

#	Time	Action	Message
1	11:13:09	ALTER TABLE company drop column website	0 row(s) affected Record
2	11:14:17	describe company	5 row(s) returned

Repetimos el mismo proceso con 'credit_card'

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS

Filter objects

sakila
sys
▼ transactions

Tables

- company
- credit_card
- transaction
- user

Views

- vistamarketing

Stored Procedures

Functions

world

Administration Schemas

Information: Schema: transactions

SQL File 5* SQL File 6* SQL File 7* SQL File 8* x SQL File 9*

1 • describe credit_card

2 • ;

Result Grid

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(10)	YES		NULL	

Result 6 x

Output

Action Output

#	Time	Action	Message
1	11:19:05	describe credit_card	5 row(s) returned

Añadimos a la tabla 'credit_card' la columna 'fecha_actual' y comprobamos que la estructura es igual a la estructura del ejercicio.

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following SQL statement is entered:

```
1 • ALTER TABLE credit_card
2 • add fecha_actual DATE;
3 • DESCRIBE Credit_card;
```

The Navigator pane on the left shows the 'transactions' database selected, with the 'credit_card' table highlighted. The 'Result Grid' pane displays the structure of the 'credit_card' table after the modification:

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

The 'Output' pane shows the execution results:

#	Time	Action	Message
1	12:09:50	ALTER TABLE credit_card add fecha_actual DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2	12:09:51	DESCRIBE Credit_card	6 row(s) returned

Repetimos el proceso con la tabla 'User' y comprobamos que tenemos que cambiar el nombre a la tabla por 'Data_user' y cambiar el nombre a la columna 'email' por 'personal_email'. Comprobamos con DESCRIBE.

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following SQL statement is entered:

```
1 • DESCRIBE Data_user;
2
```

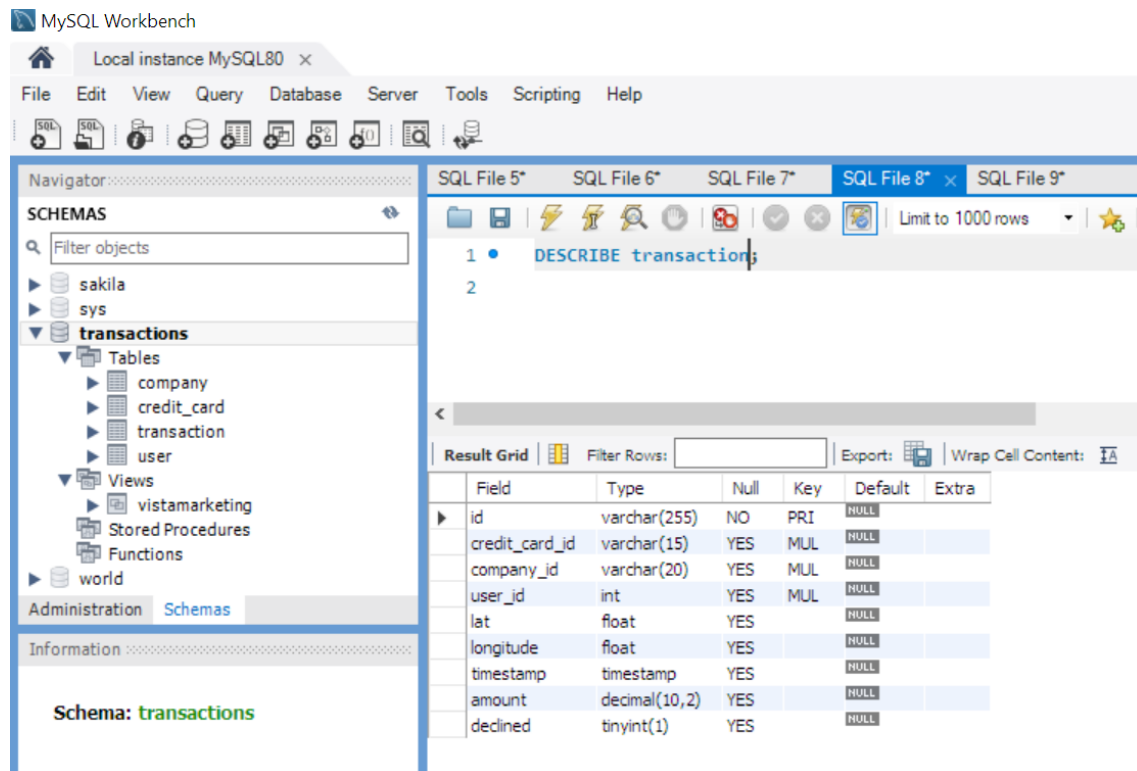
The Navigator pane on the left shows the 'transactions' database selected, with the 'user' table highlighted. The 'Result Grid' pane displays the structure of the 'Data_user' table:

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

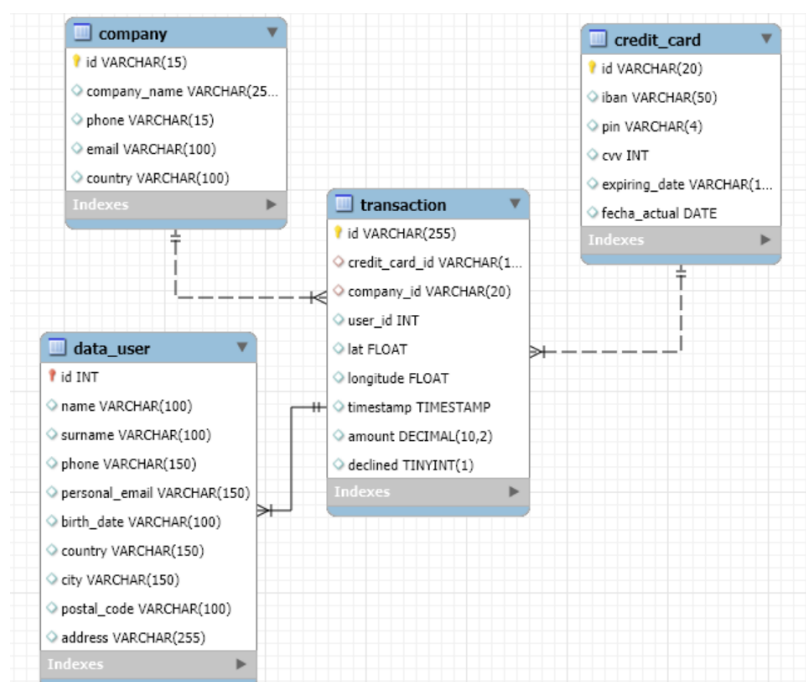
The 'Output' pane shows the execution results:

#	Time	Action	Message
1	12:09:50	ALTER TABLE credit_card add fecha_actual DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2	12:09:51	DESCRIBE Credit_card	6 row(s) returned

Realizamos las mismas comprobaciones para 'transaction' y es igual al ejemplo del ejercicio.



Analizamos ahora las relaciones entre tablas. Realizamos la visualización de las relaciones de la base de datos mediante MYSQL



Vemos que las relaciones entre tablas son las mismas en ambas BBDD por tanto no hay nada que modificar.