

Tasca S3.01. Manipulació de taules

Descripció

- En aquest sprint se simula una situació empresarial en la qual hauràs de realitzar diverses manipulacions a les taules d'una base de dades. A més, treballaràs amb índexs i vistes per optimitzar consultes i organitzar la informació.
- Continuaràs treballant amb la base de dades que conté informació d'un marketplace, un entorn similar a Amazon on diverses empreses venen els seus productes a través d'un canal en línia. En aquesta activitat, començaràs a treballar amb dades relacionades amb targetes de crèdit.
- Afegeix les taules al model segons correspongui:
 - Nivell 1: Taula "credit_card"
 - Nivell 3 : Taula "user"

Nivell 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_introducir_credit". Recorda mostrar el diagrama i realitzar una breu descripció d'aquest.

- Primero mediante código realizamos la creación de la tabla credit_card agregando todas las columnas necesarias, ejecutamos y refrescamos la base de datos para ver que se haya creado.

The screenshot shows a MySQL Workbench interface with a query editor window titled "Tasca S3.01*". The code in the editor is as follows:

```
1 USE transactions;
2
3 -- Creamos la tabla credit_card
4 CREATE TABLE IF NOT EXISTS credit_card (
5     id VARCHAR(15) PRIMARY KEY,
6     iban VARCHAR(50),
7     pan VARCHAR(35),
8     pin VARCHAR(4),
9     cvv INT,
10    expiring_date VARCHAR(15)
11 );
12
13
14
```

The "Output" tab at the bottom shows the results of the execution:

#	Time	Action	Message
1	11:02:07	CREATE TABLE IF NOT EXISTS credit_card (id VARCHAR(15) PRIMARY KEY, iban VARCHAR...)	0 row(s) affected

- Una vez creada la tabla en la base de datos Transactions cargamos los datos de la tabla credit_card mediante el archivo [datos_introducir_sprint3_credit.sql](#) y lo ejecutamos.

Tasca S3.01* | datos_introducir_sprint3_credit (...) | SQlAdditions ...

Dont Limit

1 • Insertamos datos de credit_card

2 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2983', 'TR301950312213576817638861', '54246566813633', '0')

3 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2945', 'D026854763748537475216568689', '5142423821948828', '0')

4 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2952', 'B6451VQL52710852506255', '4556 453 55 5287', '45')

5 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2959', 'CR724247724435841535', '372461377349375', '3583', '0')

6 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2966', 'B672LKTQ1562768377363', '448566 886747 7265', '1')

7 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2973', 'PT8780622813592429456346', '544 58654 54343 384', '0')

8 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2980', 'DE39241881883086277136', '492400 7145845969', '51')

9 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2987', 'GE9861434357748781813', '3763 747687 76666', '2')

10 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-2994', 'BH62721442836806675294', '344283273252593', '754')

11 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3001', 'CY49887426654774581266832110', '511722 924833 22')

12 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3008', 'U50000000000000000000000000000000', '448574446433884', '1856')

13 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3015', 'PS11398216295715968342456821', '3784 662233 1731')

14 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3022', 'GT9169516250556977423121857', '5164 1379 4842 35')

15 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3029', 'A262371413982441418123739746', '3429 279566 7763')

16 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3036', 'U50000000000000000000000000000000', '3768 451556 48761')

17 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3043', 'IN648814331014852179535', '455676 6437463635', '0')

18 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3050', 'U50000000000000000000000000000000', '4024007123722', '1')

19 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3057', 'LU91822574697545215', '3484 621767 21237', '680')

20 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3064', 'PS1469655454925377627273133', '3467 732741 268')

21 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3071', 'W08923814763512', '3464 789562 23352', '6625', '1')

22 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Ccu-3078', 'IS025127145884623279548733', '4539 322 74 2377', '0')

- Una vez creada la tabla credit_card con todos los datos cargados realizaremos la conexión con la tabla transaction a través de la FK.
- Código

The screenshot shows the MySQL Workbench interface. In the top panel, a SQL editor window titled "Tasca S3.01*" contains the following SQL script:

```

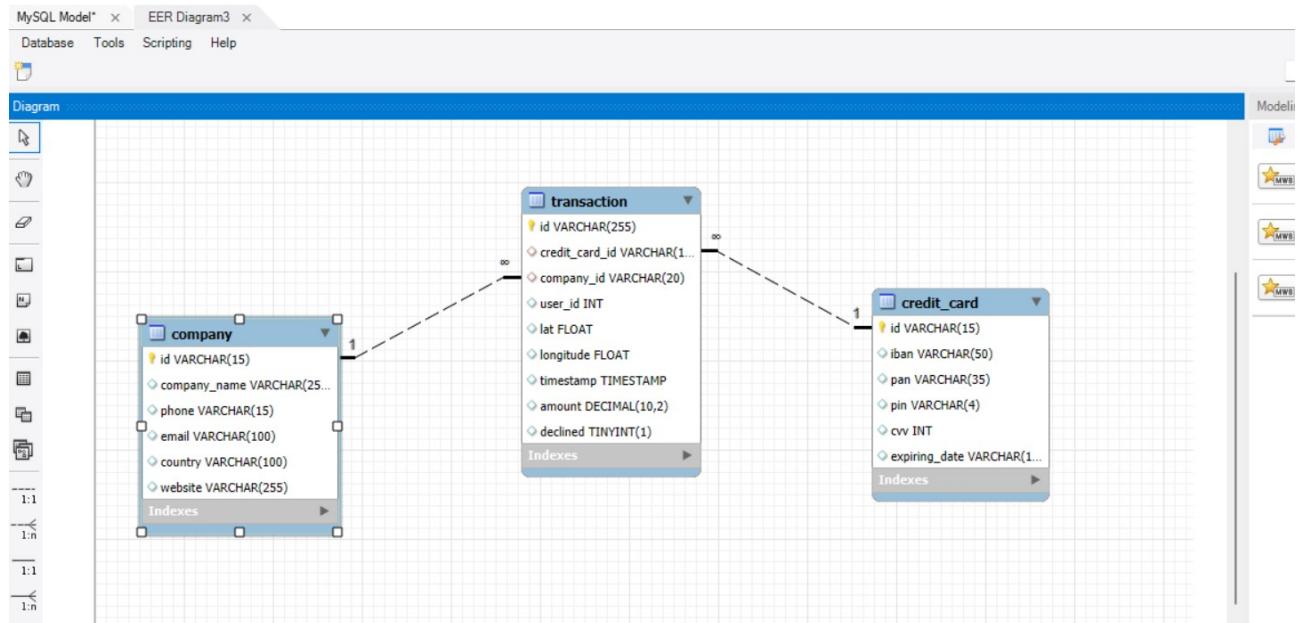
1 • USE transactions;
2
3 -- Creamos la tabla credit_card
4 • CREATE TABLE IF NOT EXISTS credit_card (
5     id VARCHAR(15) PRIMARY KEY,
6     iban VARCHAR(50),
7     pan VARCHAR(35),
8     pin VARCHAR(4),
9     cvv INT,
10    expiring_date VARCHAR(15)
11 );
12
13
14 • ALTER TABLE transaction
15 ADD CONSTRAINT fk_transaction_credit_card
16 FOREIGN KEY (credit_card_id)
17 REFERENCES credit_card(id);

```

The right side of the interface has a status bar with the message: "Automatic context I disabled. Use the toc manually get help f current caret positio l toggle automatic l". Below the editor is an "Output" pane showing the results of the executed queries:

#	Time	Action	Message	Duration / Fetc
5002	11:14:12	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcS-9579', 'XX29639309158...', ...)	1 row(s) affected	0.000 sec
5003	11:14:12	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcS-9580', 'XX78125888985...', ...)	1 row(s) affected	0.000 sec
5004	11:14:12	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcS-9581', 'XX91567051640...', ...)	1 row(s) affected	0.000 sec
5005	11:14:23	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card FOREIGN KEY (credit_card_id) REFERENCES credit_card(id);	100000 rows(s) affected Records: 100000 Duplicates: 0 Warnings: 0	1.390 sec

- Visualización del esquema EER



- En esta ocasión observamos que se ha creado la conexión entre la nueva tabla creada credit_card y la tabla transaction a través de la columna credit_card_id como clave foranea con la Primary Key de la tabla credit_card (id).

Exercici 2

- El departament de Recursos Humans ha identificat un error en el número de compte associat a la targeta de crèdit amb ID CcU-2938. La informació que ha de mostrar-se per a aquest registre és: TR323456312213576817699999. Recorda mostrar que el canvi es va realitzar.

```
29 •  SELECT iban
30   FROM credit_card
31   WHERE id = 'CcU-2938';
32
33   -- Para modificarlo realizaremos el siguiente código
34
35 •  UPDATE credit_card SET iban = 'TR323456312213576817699999'
36   WHERE id = 'CcU-2938';
```

The screenshot shows the 'Result Grid' tab of the SQL Assistant interface. It displays a single row with the column 'iban' containing the value 'TR301950312213576817638661'. The interface includes standard navigation buttons like back, forward, and search, along with export and wrap cell content options.

- Primero hemos realizado una consulta para ver el dato que teníamos que modificar, seguidamente hemos realizado la consulta Update que hay abajo de la imagen la hemos ejecutado, y finalmente mediante la primera consulta comprobamos que se haya cambiado el dato que necesitamos cambiar.

This screenshot shows a more comprehensive view of the SQL Assistant session. It includes the original code, the update statement, and the results of both queries. The 'credit card 2' tab at the bottom shows the execution log with two entries: an update at 11:44:29 and a select at 11:44:51. The message pane indicates 1 row affected, 1 row matched, 0 changes, and 0 warnings.

```
-- Exercici 2
-- El departament de Recursos Humans ha identificat un error en el número de compte associat a la targeta de crèdit amb ID CcU-2938.
-- La informació que ha de mostrar-se per a aquest registre és: TR323456312213576817699999.
-- Recorda mostrar que el canvi es va realitzar.

SELECT iban
FROM credit_card
WHERE id = 'CcU-2938';

-- Para modificarlo realizaremos el siguiente código

UPDATE credit_card SET iban = 'TR323456312213576817699999'
WHERE id = 'CcU-2938';

+-----+-----+-----+
| iban |      |
+-----+-----+
| TR301950312213576817638661 |      |
+-----+-----+
```

Exercici 3 Modificar la forma

- En la taula "transaction" ingressa 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

- Primero introducimos en la tabla company el nuevo id 'b-9999'.

The screenshot shows the SQL Server Management Studio interface with a script window titled 'Tasca S3.01*' containing the following code:

```
-- EXERCICI 3  
-- En la taula "transaction" ingressa un nou usuari amb la següent informació:  
40 -- Id 108B1D1D-5B23-A76C-55EF-C568E49A99DD  
41 -- credit_card_id CcU-9999  
42 -- company_id b-9999  
43 -- user_id 9999  
44 -- lat 829.999  
45 -- longitude -117.999  
46 -- amount 111.11  
47 -- declined 0  
48  
49 -- Primero introducimos un nuevo valor en la tabla company  
50 • INSERT INTO company (id)  
51 VALUES ('b-9999');
```

The output pane shows the execution results:

#	Time	Action	Message
1	11:44:08	INSERT INTO company (id) VALUES ('b-9999')	1 row(s) affected

- Introducimos en la tabla credit_card el nuevo id 'CcU-9999'

The screenshot shows the SQL Server Management Studio interface with a script window titled 'Tasca S3.01*' containing the following code:

```
-- user_id 9999  
44 -- lat 829.999  
45 -- longitude -117.999  
46 -- amount 111.11  
47 -- declined 0  
48  
49 -- Primero introducimos un nuevo valor en la tabla company  
50 • INSERT INTO company (id)  
51 VALUES ('b-9999');  
52  
53 -- Seguidamente realizamos lo mismo en la tabla credit_card  
54 • INSERT INTO credit_card (id)  
55 VALUES ('CcU-9999');
```

The output pane shows the execution results:

#	Time	Action	Message
1	11:44:08	INSERT INTO company (id) VALUES ('b-9999')	1 row(s) affected
2	11:56:21	INSERT INTO credit_card (id) VALUES ('CcU-9999')	1 row(s) affected

- Seguidamente introducimos la siguiente información en la tabla transaction.

```

Server Tools Scripting Help
Tasca 33.01* | datos_introducir_sprint3_user
File Edit View Insert Object Tools Help
49 -- Primero introducimos un nuevo valor en la tabla company
50 • INSERT INTO company (id)
51   VALUES ('b-9999');
52
53 -- Seguidamente realizamos lo mismo en la tabla credit_card
54 • INSERT INTO credit_card (id)
55   VALUES ('ccU-9999');
56
57 -- Insertamos la siguiente informacióen en la tabla transaction
58
59 • INSERT INTO transaction (Id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
60   VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', '829.999', '-117.999', '111.11', 0);
61
62
63

```

Output

#	Time	Action	Message
1	12:06:49	INSERT INTO transaction (Id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) VALUE...	1 row(s) affected

- Finalmente comprobamos que los datos de hayan cargado correctamente realizando la siguiente consulta.

```

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

```

Exercici 4

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

```

Server Tools Scripting Help
Tasca 33.01* | datos_introducir_sprint3_credit...
File Edit View Insert Object Tools Help
57 WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';
58
59 -- Exercici 4
60 -- Des de recursos humans et sol·liciten eliminar la columna "pan" de la taula credit_card.
61 -- Recorda mostrar el canvi realitzat.
62
63 • ALTER TABLE credit_card
64   DROP COLUMN pan;
65
66 • SELECT *
67   FROM credit_card;
68

```

Result Grid

id	iban	pin	cvv	expiring_date
Cd5-4857	XX4857591835292505850771	1819	467	09/27/25
Cd5-4858	XX5581768137002436094025	3964	817	12/28/28
Cd5-4859	XX7826930491423553609370	4983	277	11/26/26
Cd5-4860	XX5559590368835304645299	6876	661	07/27/27
Cd5-4861	XX2035182877195191627307	5710	398	04/25/26

credit_card 4 <

Output

#	Time	Action	Message
1	12:44:20	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2	12:44:51	SELECT * FROM credit_card	5000 row(s) returned

- Para eliminar la columna ‘pan’ de la tabla credit_card realizamos el siguiente código.

```
ALTER TABLE credit_card
DROP COLUMN pan;
```

- Utilizamos la sentencia ALTER TABLE de la tabla en este caso que queremos realizar el cambio credit_card, mediante el comando DROP COLUMN seleccionamos el nombre de la columna que queremos eliminar, una vez ejecutado ese código realizamos la siguiente consulta para ver si se han realizado los cambios.

```
SELECT *
FROM credit_card;
```

También podemos realizar un DESCRIBE credit_card; para comprobar que la columna se ha eliminado.

- Finalmente hemos comprobado que la columna ‘pan’ ha sido eliminada de la tabla credit_card.

Nivell 2

Exercici 1

· Elimina de la taula transaction el registre amb ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD de la base de dades.

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following code:

```
66 •  SELECT *
67   FROM credit_card;
68
69 -- Nivell 2
70 -- Exercici 1
71 -- Elimina de la taula transaction el registre amb ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD de la base de dades.
72
73 •  DELETE FROM transaction
74 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
75
76 •  SELECT * FROM transaction
77 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
```

The results grid below shows the structure of the transaction table:

#	id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
*	HULL	HULL	HULL	HULL	HULL	HULL	HULL	HULL	HULL

The output pane at the bottom shows the execution log:

#	Time	Action	Message
1	12:44:20	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2	12:44:51	SELECT * FROM credit_card	5000 row(s) returned
3	13:03:05	DELETE FROM transaction WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD'	1 row(s) affected
4	13:04:03	SELECT * FROM transaction WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD'	0 row(s) returned

-Para realizar la eliminación de un registro de la tabla realizaremos el siguiente código.

```
DELETE FROM transaction  
WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
```

- Utilizamos la sentencia DELETE para indicarle que queremos eliminar de la tabla transaction y en el WHERE le ponemos la condición , en este caso el registro que queremos eliminar, para que solamente nos elimine ese registro.

- Una vez ejecutado el código anterior comprobamos que realmente el registro haya sido eliminado tal y como se nos había solicitado.

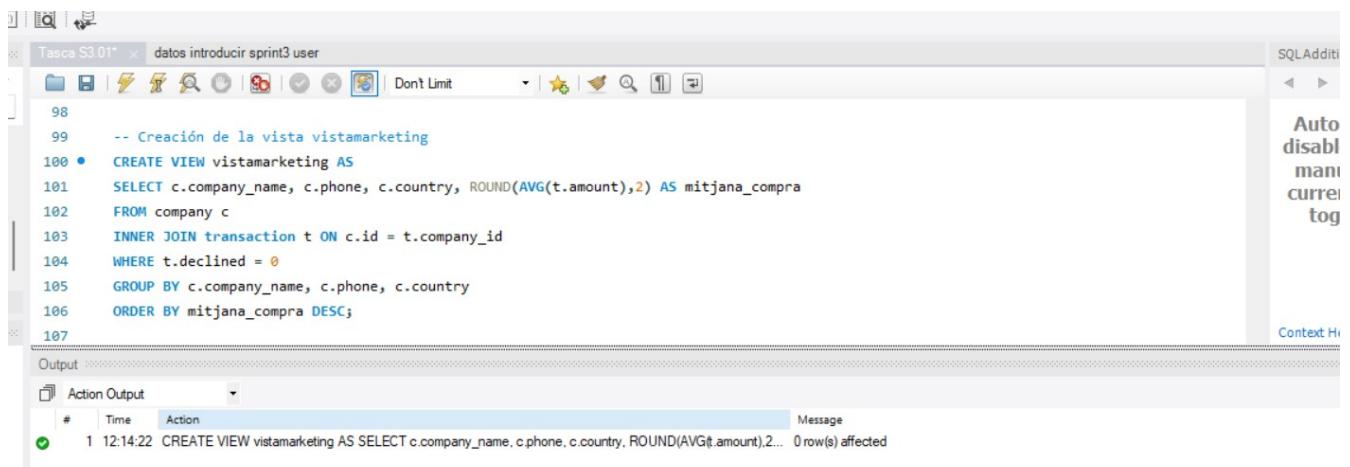
```
SELECT *  
FROM transaction  
WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
```

- Como podemos observar en la imagen de arriba no nos devuelve ningún resultado, con lo que la eliminación del registro se ha realizado.

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.

- Primero Creamos la vista (vistamarketing) mediante el siguiente código, una vez realizado el código lo ejecutamos y refrescamos Schemas para que nos aparezca en la pestaña de Views.



```
98  
99 -- Creación de la vista vistamarketing  
100 • CREATE VIEW vistamarketing AS  
101 SELECT c.company_name, c.phone, c.country, ROUND(AVG(t.amount),2) AS mitjana_compra  
102 FROM company c  
103 INNER JOIN transaction t ON c.id = t.company_id  
104 WHERE t.declined = 0  
105 GROUP BY c.company_name, c.phone, c.country  
106 ORDER BY mitjana_compra DESC;  
107
```

- Seguidamente realizamos la consulta para obtener la media de compras ordenada de mayo a menor con la siguiente consulta.

Tasca S3.01* x datos introducir sprint3 user

```
103     INNER JOIN transaction t ON c.id = t.company_id
104 WHERE t.declined = 0
105 GROUP BY c.company_name, c.phone, c.country
106 ORDER BY mitjana_compra DESC;
107
108 • SELECT *
109 FROM vistamarkeeting;
110
111 -- Exercici 3
112 -- Filtra la vista VistaMarketing per a mostrar només les companyies que tenen el seu país de residència en "Germany"
113
```

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

company_name	phone	country	mitjana_compra
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 95 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 48 38 34 19	Ireland	267.56
Cras Consulting	07 50 10 85 63	Belgium	267.38
Sed LLC	01 63 36 26 52	Belgium	266.61
Erat Tempus Ltd	02 47 31 55 72	United States	266.77

vistamarkeeting 3 x

Output

Action Output

#	Time	Action	Message
1	16:12:21	SELECT * FROM vistamarkeeting	101 row(s) returned

Exercici 3

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

Tasca S3.01* | datos introducir sprint3 user

Dont Limit

```
112 -- Filtra la vista VistaMarketing per a mostrar només les companyies que tenen el seu país de residència en "Germany"
113
114 • SELECT * FROM vistamarketing
115 WHERE country = 'Germany';
116
```

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

company_name	phone	country	mitjana_compra
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
Rubrum 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

vistamarketing 6 < Read Only Context |

Action Output

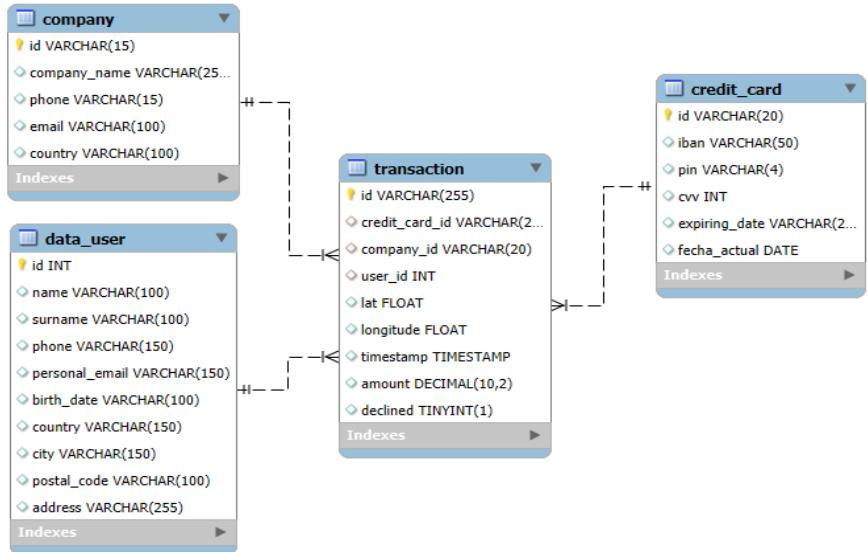
#	Time	Action	Message
1	12:14:22	CREATE VIEW vistamarketing AS SELECT c.company_name, c.phone, c.country, ROUND(AVG(c.amount),...	0 row(s) affected
2	12:15:59	SELECT * FROM vistamarketing	101 row(s) returned
3	12:18:01	SELECT * FROM vistamarketing WHERE country = 'Germany'	8 row(s) returned

- En esta ocasión realizamos un SELECT * de la vista VISTAMARKETING y en el WHERE le ponemos la condición que el COUNTRY = alemania para obtener el resultado de la consulta.

Nivell 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:



- Primero de todo creamos la tabla data_user , ejecutamos y refrescamos el schema.

Screenshot of MySQL Workbench showing the creation of the data_user table:

```

-- Nivell 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:

-- creamos la tabla data_user:
CREATE TABLE IF NOT EXISTS user (
    id CHAR(10) 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)
);
  
```

The screenshot shows the SQL Editor tab with the code above. Below it, the Output tab shows the execution results:

#	Time	Action	Message
1	12:24:41	CREATE TABLE IF NOT EXISTS user (id CHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARC...)	0 row(s) affected

- El siguiente paso es cargar todos los datos correspondientes a la tabla data_user.

```
1 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("151", "Meghan", "Hay"
2 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("152", "Hakeem", "Al"
3 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("153", "Keegan", "Pug"
4 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("154", "Cooper", "Bul"
5 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("155", "Joshua", "Rus"
6 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("156", "Remedios", "C
7 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("157", "Philip", "Car
8 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("158", "Fatima", "Dye
9 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("159", "Kyllynn", "Ae
10 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("160", "Lael", "Moody
11 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("161", "Nora", "Reeve
12 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("162", "Francesca", "
13 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("163", "Denton", "Bla
14 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("164", "Preston", "Mc
15 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("165", "Nora", "Cantr
16 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("166", "Matthew", "Wc
17 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("167", "Sheila", "Dic
18 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("168", "Donna", "Rive
19 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("169", "Linda", "Gair
20 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("170", "William", "Be
21 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("171", "Gary", "Robbi
22 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("172", "Yoko", "Calhc
```

Output:

#	Time	Action	Message	Dur
4998	12:30:11	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	VALUES (0.00
4999	12:30:11	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	VALUES (0.00
5000	12:30:11	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	VALUES (0.00
5001	12:30:11	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	VALUES (0.00

- Cambiamos el nombre de la tabla de user a data_user.

```
127     surname VARCHAR(100),
128     phone VARCHAR(150),
129     email VARCHAR(150),
130     birth_date VARCHAR(100),
131     country VARCHAR(150),
132     city VARCHAR(150),
133     postal_code VARCHAR(100),
134     address VARCHAR(255)
135   );
136
137   -- Cambio el nombre de la tabla de user a data_user con el rename.
138 •  ALTER TABLE user RENAME TO data_user;
139
```

Output:

#	Time	Action	Message
---	------	--------	---------

- Realizamos el cambio de nombre de la columna email por personal_email.

```

Tasca S3.01* | datos introducir sprint3 user
132     city VARCHAR(150),
133     postal_code VARCHAR(100),
134     address VARCHAR(255)
135 );
136
137 -- Cambio el nombre de la tabla de user a data_user con el rename.
138 • ALTER TABLE user RENAME TO data_user;
139
140 -- Realizamos el cambio de la columna email por personal_email en la tabla data_user
141
142 • ALTER TABLE data_user
143     CHANGE COLUMN email personal_email VARCHAR (150) NULL;
144
Output
Action Output
# Time Action
1 12:34:59 ALTER TABLE data_user CHANGE COLUMN email personal_email VARCHAR (150) NULL
Message
0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

```

- Tenemos que crear en la tabla data_user el nuevo usuario del ejercicio 3 del primer nivel.

```

Server Tools Scripting Help
Tasca S3.01* | datos introducir sprint3 user
137 -- Cambio el nombre de la tabla de user a data_user con el rename.
138 • ALTER TABLE user RENAME TO data_user;
139
140 -- Realizamos el cambio de la columna email por personal_email en la tabla data_user
141
142 • ALTER TABLE data_user
143     CHANGE COLUMN email personal_email VARCHAR (150) NULL;
144 -- Crear en la tabla data_user el nuevo usuario del ejercicio 3 del primer nivel
145
146 • INSERT INTO data_user (id)
147     Values ('9999');
148 --
149
Output
Action Output
# Time Action
1 12:34:59 ALTER TABLE data_user CHANGE COLUMN email personal_email VARCHAR (150) NULL
0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2 12:39:50 INSERT INTO data_user (id) Values ('9999')
1 row(s) affected
Message
1 row(s) affected

```

- Previamente a realizar la conexión entre las dos tablas tenemos que cambiar el tipo de dato de la columna id de la tabla data_user de (CHAR) a INT para que coincidan el tipo de datos con la FK de la tabla transaction.

```

Server Tools Scripting Help
Tasca S3.01* | datos introducir sprint3 user
148 --
149
150 -- Una vez cargados todos los datos de la tabla realizamos un cambio del tipo de datos para poder generar la relación
151 -- ya que si no coincide el tipo de dato no podrá realizar la conexión en el EER.
152
153 • ALTER TABLE data_user MODIFY COLUMN id INT;
154
Output
Action Output
# Time Action
1 16:51:12 ALTER TABLE data_user MODIFY COLUMN id INT
Message
5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

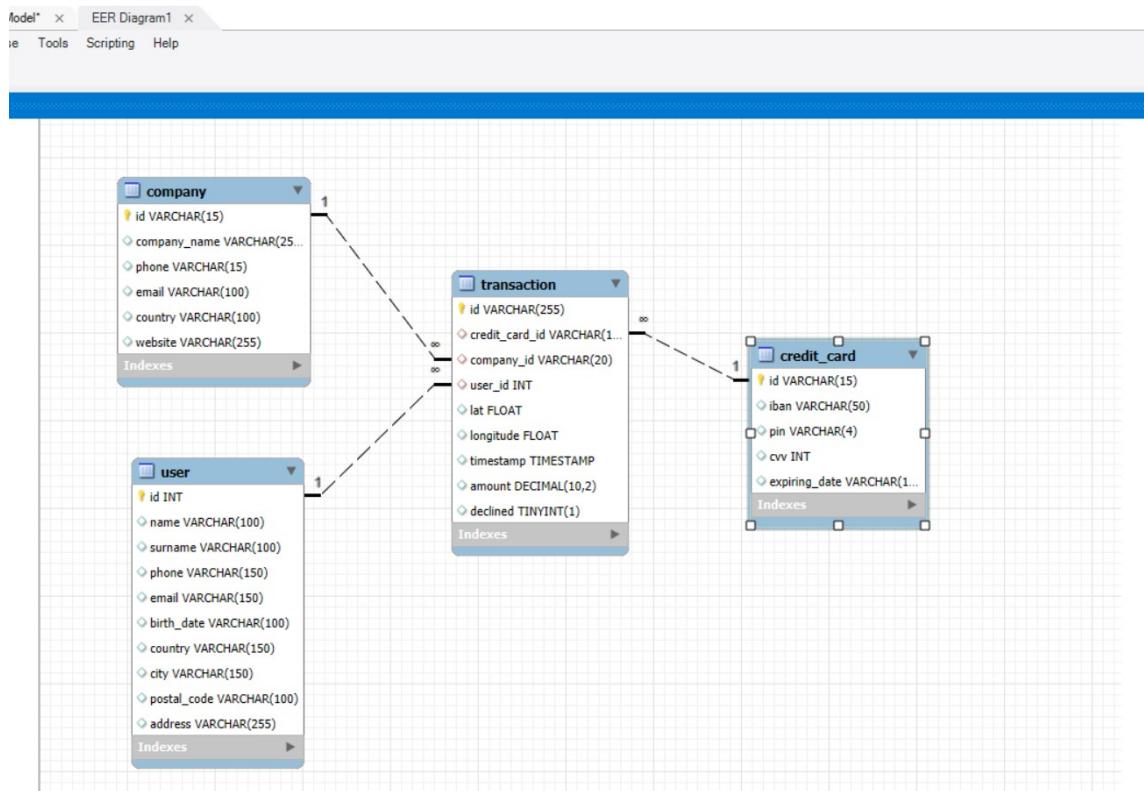
```

- Generamos la relación entre la tabla transaction y data_user con el siguiente código.

```

Server Tools Scripting Help
Tasca S3.01* | datos introducir sprint3 user
154
155 -- Crear relación entre la tabla transaction y data_user
156 • ALTER TABLE transaction
157 ADD CONSTRAINT fk_transaction_data_user
158 FOREIGN KEY (user_id)
159 REFERENCES data_user(id);
160
Output
Action Output
# Time Action
1 16:51:12 ALTER TABLE data_user MODIFY COLUMN id INT
2 17:04:36 SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD'
3 17:10:14 ALTER TABLE transaction ADD CONSTRAINT fk_transaction_data_user FOREIGN KEY (user_id) REFER...
Message
5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0
1 row(s) returned
100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0
  
```

- Realizamos el EER y obtenemos la siguiente imagen.



- Eliminamos la columna website de la tabla company mediante el siguiente código.

```

129
137      -- Eliminamos la columna website de la tabla company
138
139 •  ALTER TABLE company
140     DROP COLUMN website;
***
```

Output

#	Time	Action
1	13:07:20	ALTER TABLE company DROP COLUMN website

Message

0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

- Ahora rompemos la relación entre la tabla transaction y la tabla credit_card para realizar unas modificaciones.

```

169      -- Romper la relación entre transaction y credit_card
170
171 •  ALTER TABLE transaction DROP FOREIGN KEY FK_transaction_credit_card;
***
```

Output

#	Time	Action
1	17:37:55	ALTER TABLE transaction DROP FOREIGN KEY FK_transaction_credit_card

Message

0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

- Realizamos el cambio de longitud de la variable id en la tabla credit_card y realizamos seguidamente un describe para ver que se haya realizado correctamente.

Tasca S3.01* | datos introducir sprint3 user

```

171 •  ALTER TABLE transaction DROP FOREIGN KEY FK_transaction_credit_card;
172
173      -- Cambiamos la longitud de la variable id en la tabla credit_card
174
175 •  ALTER TABLE credit_card MODIFY COLUMN id varchar (20);
176 •  DESCRIBE credit_card;
```

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(15)	YES		NULL	

Result 7

Output

#	Time	Action
1	17:37:55	ALTER TABLE transaction DROP FOREIGN KEY FK_transaction_credit_card
2	17:51:09	ALTER TABLE credit_card MODIFY COLUMN id varchar (20)
3	17:51:41	DESCRIBE credit_card

Message

0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

5 row(s) returned

- Realizamos el cambio de longitud de la variable credit_card_id de la tabla transaction para que las 2 sean del mismo tipo y longitud.

The screenshot shows the MySQL Workbench interface with the following details:

- Server:** MySQL Model*
- Tools:** EER Diagram4
- Scripting:** SQLA
- Help:** A dis m cu
- Current Connection:** Tasca S3.01* - datos introducir sprint3 user
- SQL Editor:**

```

180 • ALTER TABLE transaction MODIFY COLUMN credit_card_id varchar (20);
181 • DESCRIBE transaction;
    
```
- Result Grid:** Shows the structure of the transaction table after modification. The columns are:

Field	Type	Null	Key	Default	Extra
id	varchar(255)	NO	PRI	NULL	
credit_card_id	varchar(20)	YES	MUL	NULL	
company_id	varchar(20)	YES	MUL	NULL	
user_id	int	YES	MUL	NULL	
lat	float	YES		NULL	
- Action Output:**

#	Time	Action
1	17:58:25	ALTER TABLE transaction MODIFY COLUMN credit_card_id varchar (20)
2	17:59:14	DESCRIBE transaction

Message:
0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
9 row(s) returned

- Ahora creamos la columna nueva fecha_actual en la tabla credit_card.

The screenshot shows the MySQL Workbench interface with the following details:

- Server:** MySQL Model*
- Tools:** EER Diagram4
- Scripting:** SQLA
- Help:** Aul disal ma curr tc
- Current Connection:** Tasca S3.01* - datos introducir sprint3 user
- SQL Editor:**

```

-- Creamos la nueva columna fecha_actual en la tabla credit_card
184
185 • ALTER TABLE credit_card
186 ADD COLUMN fecha_actual DATE NULL;
187
188 • DESCRIBE credit_card;
    
```
- Result Grid:** Shows the structure of the credit_card table after adding the new column. The columns are:

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(15)	YES		NULL	
fecha_actual	date	YES		NULL	
- Action Output:**

#	Time	Action
1	17:58:25	ALTER TABLE transaction MODIFY COLUMN credit_card_id varchar (20)
2	17:59:14	DESCRIBE transaction
3	18:11:33	DESCRIBE credit_card
4	18:13:53	ALTER TABLE credit_card ADD COLUMN fecha_actual DATE NULL
5	18:14:25	DESCRIBE credit_card

Message:
0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
9 row(s) returned
5 row(s) returned
0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
6 row(s) returned

- Realizamos el cambio de la longitud de la variable expiring_date de la tabla credit_card

The screenshot shows the SQL Assistant interface with the following details:

- SQL Editor:** Shows the following SQL code:


```

186 ADD COLUMN fecha_actual DATE NULL;
187
188 -- cambiar la longitud de la variable expiring_date de la tabla credit_card
189
190 • ALTER TABLE credit_card MODIFY COLUMN expiring_date varchar (20);
191 • DESCRIBE credit_card;
```
- Result Grid:** Displays the structure of the credit_card table with the following columns:

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	
- Action Output:** Shows the log of actions with their times and messages:

#	Time	Action	Message
1	17:58:25	ALTER TABLE transaction MODIFY COLUMN credit_card_id varchar (20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2	17:59:14	DESCRIBE transaction	9 row(s) returned
3	18:11:33	DESCRIBE credit_card	5 row(s) returned
4	18:13:53	ALTER TABLE credit_card ADD COLUMNfecha_actual DATE NULL	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
5	18:14:25	DESCRIBE credit_card	6 row(s) returned
6	18:43:32	ALTER TABLE credit_card MODIFY COLUMN expiring_date varchar (20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
7	18:43:46	DESCRIBE credit_card	6 row(s) returned

- Creamos la relación entre las tablas transaction y credit_card después de haber realizado todas las modificaciones

The screenshot shows the SQL Assistant interface with the following details:

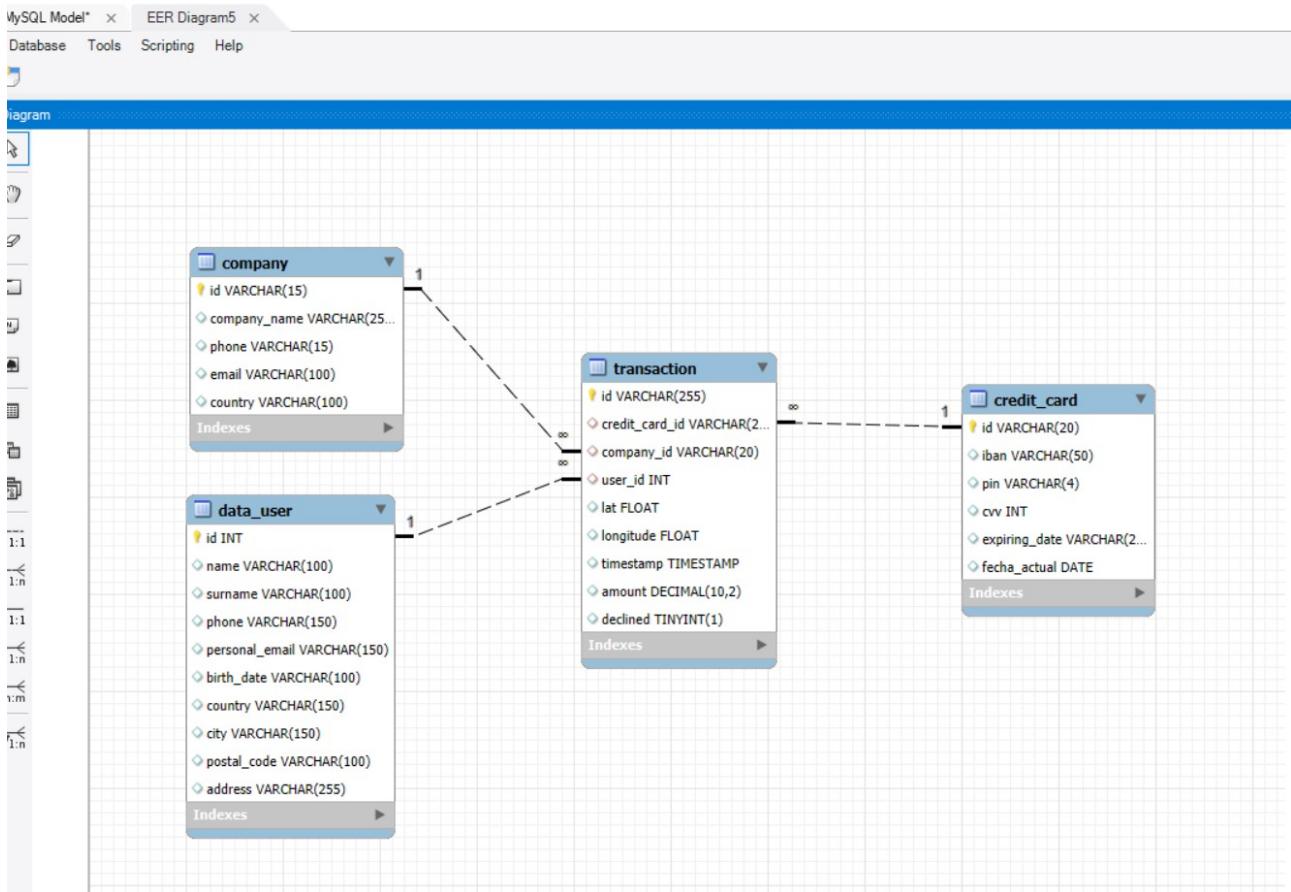
- SQL Editor:** Shows the following SQL code:


```

191 • DESCRIBE credit_card;
192
193 -- Despues de realizar todos los cambios volveremos a conectar la tabla transaction con la tabla credit_card
194
195 • ALTER TABLE transaction
196 ADD CONSTRAINT fk_transaction_credit_card
197 FOREIGN KEY (credit_card_id)
198 REFERENCES credit_card(id);
```
- Action Output:** Shows the log of actions with their times and messages:

#	Time	Action	Message
1	18:51:15	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card FOREIGN KEY (credit_card_id) ...	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0

- Realizamos el diagrama EER y obtenemos el siguiente resultado



Exercici 2

L'empresa també us demana crear una vista anomenada "InformeTecnico" que contingui la següent informació:

- ID de la transacció
- Nom de l'usuari/ària
- Cognom de l'usuari/ària
- IBAN de la targeta de crèdit usada.
- Nom de la companyia de la transacció realitzada.
- Assegureu-vos d'incloure informació rellevant de les taules que coneixereu i utilitzeu àlies per canviar de nom columnes segons calgui.

Mostra els resultats de la vista, ordena els resultats de forma descendente en funció de la variable ID de transacción

- Primero creamos la nueva vista ‘ InformeTecnico’ con las columnas solicitadas y realizando las diferentes uniones entre tablas mediante los JOINS y finalmente ponemos un order by de la columna id_transaccio en orden descendente para que la vista se ordene de esa manera. Una vez generada la vista visualizamos el resultado con un SELECT * FROM informetecnico;

The screenshot shows a SQL query editor window with the following details:

- Title Bar:** 'Tasca S3.01*' and 'datos introducir sprint3 user'.
- Toolbar:** Includes icons for file operations, search, and database navigation.
- Query Area:** Displays the SQL code for creating the view and executing it.


```

212 • CREATE VIEW InformeTecnico AS
213     SELECT
214         t.id AS id_transaccio,
215         u.name AS nom_usuari,
216         u.surname AS cognom_usuari,
217         cc.iban AS iban_targeta,
218         c.company_name AS nom_companyia
219     FROM transaction t
220     JOIN data_user u ON t.user_id = u.id
221     JOIN credit_card cc ON t.credit_card_id = cc.id
222     JOIN company c ON t.company_id = c.id
223     ORDER BY id_transaccio DESC;
224
225 • SELECT * FROM informetecnico;
      
```
- Result Grid:** Shows the results of the SELECT query. The columns are id_transaccio, nom_usuari, cognom_usuari, iban_targeta, and nom_companyia. The data includes various user names, company names, and IBAN numbers.
- Action Output:** Shows two log entries:

#	Time	Action	Message
1	19:00:04	CREATE VIEW InformeTecnico AS SELECT t.id AS id_transaccio, u.name AS nom_usuari, u.suma...	0 row(s) affected
2	19:01:07	SELECT * FROM informetecnico	100000 row(s) returned