

Tasca S3.01. Manipulació de taules

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

Crearemos una clave primaria **id** en la tabla **credit_card** que identificará con una clave unívoca cada registro, correspondiente a una tarjeta diferente.

El campo **credit_card_id** de la tabla **transaction** hace referencia al campo **id** de la tabla **credit_card**, así que, más adelante, definiremos una clave foránea que represente esta relación de 1 a N entre **credit_card.id** y **transaction.credit_card_id**. Para ello, definiremos el campo **id** de la tabla **credit_card** con el mismo tipo de datos de **credit_card_id** (varchar(15)).

La tabla **credit_card** contendrá datos relativos a la tarjeta: el número IBAN de la cuenta relacionada (alfanumérico), el número de la tarjeta (numérico que, en principio, es único; aquí una explicación de [cómo se calculan](#)), el número pin (numérico corto), el código de validación CVV de la tarjeta (numérico corto) y la fecha de caducidad, que, al no coincidir con el formato de fecha de MySQL, lo incorporaremos también como alfanumérico.

```

1  -- Creamos la tabla credit_card
2
3  CREATE TABLE IF NOT EXISTS credit_card (
4      id VARCHAR(15) PRIMARY KEY,
5      iban VARCHAR(255),
6      pan VARCHAR(255),
7      pin SMALLINT,
8      cvv SMALLINT,
9      expiring_date VARCHAR(15)
10 );

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:12:26	CREATE TABLE IF NOT EXISTS credit_card (0 row(s) affected	0.062 sec

Incorporamos los datos a la tabla:

```

1  -- Insertamos datos de credit_card
2
3  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
4  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
5  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
6  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
7  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
8  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
9  INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
10 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
11 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
12 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
13 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
14 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
15 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
16 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
17 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
18 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
19 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
20 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
21 INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
273	12:14:59	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (CcU-4835', 'PT... 1 row(s) affected	0.016 sec
274	12:14:59	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (CcU-4842', 'SA... 1 row(s) affected	0.000 sec
275	12:14:59	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (CcU-4849', 'SE... 1 row(s) affected	0.015 sec
276	12:15:00	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (CcU-4856', 'TR... 1 row(s) affected	0.000 sec

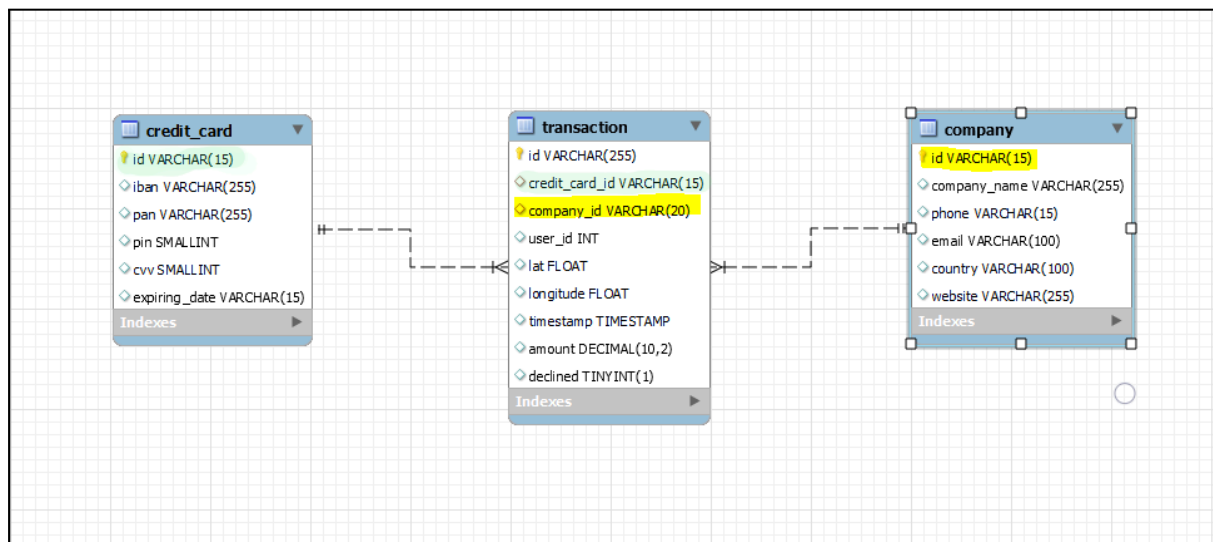
Tras incorporar los datos del fichero **datos_introducir_credit**, creamos la clave foránea **credit_card_fk** tal como explicamos más arriba.

```
1  -- Añadimos la clave foránea credit_card_id de la tabla transaction
2  -- tras rellenar la tabla credit_card
3
4  • ALTER TABLE transaction
5  ADD CONSTRAINT credit_card_fk
6  FOREIGN KEY (credit_card_id) REFERENCES credit_card(id);
```

Output

#	Time	Action	Message	Duration / Fetch
1	12:21:43	ALTER TABLE transaction ADD CONSTRAINT credit_card_fk FO...	586 row(s) affected Records: 586 Duplicates: 0 Warnings: 0	0.266 sec

Ejecutamos la ingeniería inversa para obtener el esquema:



Como vemos en el esquema, la nueva tabla **credit_card** está relacionada con la tabla **transaction** a través de la clave foránea **credit_card_fk**, que relaciona el **id** de la tarjeta de crédito de la tabla **credit_card** en una relación 1 a N con el campo **credit_card_id**, que identifica la tarjeta de crédito usada (en caso de que se use una tarjeta, pues el campo puede quedar nulo) en las transacciones almacenadas en la tabla **transaction**.

- 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.

Primero buscaremos el registro correspondiente al usuario CcU-2938 y comprobaremos que el IBAN no corresponde con el que nos notifica Recursos Humanos. A continuación, actualizaremos el dato correspondiente a este registro y volveremos a ejecutar la sentencia **SELECT** para confirmar que se ha realizado el cambio.

The screenshot shows a database management interface. At the top, a SQL query is entered in a text area:

```
1 -- Comprobamos que el IBAN del usuario CcU-2938 no corresponde
2 -- con el que nos notifica RRHH
3 • SELECT * FROM credit_card
4 WHERE id = 'CcU-2938';
```

Below the query, the 'Result Grid' displays the results of the query. The grid has columns: id, iban, pan, pin, cvv, and expiring_date. The first row shows the record for user CcU-2938 with the following values:

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

Below the result grid, the 'Action Output' section shows the execution details of the query:

	Time	Action	Response	Duration / Fetch Time
1	18:28:33	SELECT * FROM credit_card WHERE id = 'CcU-2938' LIMIT 0, 10000	1 row(s) returned	0.00094 sec / 0.000...

The screenshot shows the same database management interface. At the top, a new SQL query is entered in a text area:

```
38
39 -- Ejecutamos el cambio
40 • UPDATE credit_card
41 SET iban = 'R323456312213576817699999'
42 WHERE id = 'CcU-2938';
43
```

Below the query, the 'Action Output' section shows the execution details of the UPDATE query:

	Time	Action	Response	Duration / Fetch Time
1	18:30:53	UPDATE credit_card SET iban = 'R323456312213576817699999' WHERE id = 'CcU-2938'	1 row(s) affected Rows matched: 1 Changed: 1 Warni...	0.0020 sec

Don't Limit

```

1  -- Comprobamos que el IBAN del usuario CcU-2938 no corresponde
2  -- con el que nos notifica RRHM
3  • SELECT * FROM credit_card
4    WHERE id = 'CcU-2938';
5
6  -- Ejecutamos el cambio
7  • UPDATE credit_card
8    SET iban = 'R323456312213576817699999'
9    WHERE id = 'CcU-2938';
10
11 -- Comprobamos que el cambio se ha realizado correctamente
12 • SELECT * FROM credit_card
13   WHERE id = 'CcU-2938';

```

Result Grid

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

credit_card 12

Output

Action Output

#	Time	Action	Message
1	10:48:34	SELECT * FROM credit_card WHERE id = 'CcU-2938'	1 row(s) returned
2	10:48:58	UPDATE credit_card SET iban = 'R323456312213576817699999' WHERE id = 'CcU-2938'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
3	10:49:07	SELECT * FROM credit_card WHERE id = 'CcU-2938'	1 row(s) returned

Apply Revert

- Exercici 3

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

Creemos una sentencia **INSERT** para ingresar esta transacción, pero la inserción genera un error de integridad de las claves foráneas, ya que no existe información de la **credit_card_id** = 'CcU-9999' ni de la **company_id** = 'b-9999' en las tablas referenciadas correspondientes. Informamos a RRHH que nos facilite información de la empresa que ha realizado la transacción y de los datos de la tarjeta de crédito utilizada.

```

1 -- Transacción en transaction
2 • INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
3 VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', 9999, 829.999, -117.999, 111.11, 0);

```

The screenshot shows a database client interface with a query editor and an 'Action Output' pane. The query editor contains an SQL INSERT statement. The 'Action Output' pane shows a table with columns: Time, Action, Response, and Duration / Fetch Time. The first row shows an error: 'Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('transactions`.`transaction', CONSTRAINT 'transacti... 0.0015 sec'.

Para este ejercicio, crearé sendos registros en las tablas **company_id** y **credit_card** para mantener la integridad de la base de datos.

1

-- Tabla company

2

•

INSERT INTO company

3

VALUES ('b-9999', 'Eleternoaprendiz Data and DJ Ltd.', '0-1200-CALL-ME', 'eleternoaprendiz@proton.me', 'Spain', 'http://eleternoaprendiz.dj');

4

5

-- Tabla credit_card

6

•

INSERT INTO credit_card

7

VALUES ('CcU-9999', 'CE01234 5678 9012 3456', '0123 4567 8901 2345', 9999, 000, '12/28/28');

100%

93:7

Action Output

	Time	Action	Response	Duration / Fetch Time
1	18:38:47	INSERT INTO company VALUES ('b-9999', 'Eleternoaprendiz Data and DJ Ltd.', '0-1200-CALL-ME', 'eleternoaprendiz...	1 row(s) affected	0.0018 sec
2	18:39:00	INSERT INTO credit_card VALUES ('CcU-9999', 'CE01234 5678 9012 3456', '0123 4567 8901 2345', 9999, 000, '12/...	1 row(s) affected	0.0010 sec

Y, a continuació, insertamos el registro en la tabla **transaction**.

1

-- Tabla company

2

•

INSERT INTO company

3

VALUES ('b-9999', 'Eleternoaprendiz Data and DJ Ltd.', '0-1200-CALL-ME', 'eleternoaprendiz@proton.me', 'Spain', 'http://eleternoaprendiz.dj');

4

5

-- Tabla credit_card

6

•

INSERT INTO credit_card

7

VALUES ('CcU-9999', 'CE01234 5678 9012 3456', '0123 4567 8901 2345', 9999, 000, '12/28/28');

8

9

-- Transacción en transaction

10

•

INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)

11

VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', 9999, 829.999, -117.999, 111.11, 0);

100%

107:11

Action Output

	Time	Action	Response	Duration / Fetch Time
1	18:38:47	INSERT INTO company VALUES ('b-9999', 'Eleternoaprendiz Data and DJ Ltd.', '0-1200-CALL-ME', 'eleternoaprendiz...	1 row(s) affected	0.0018 sec
2	18:39:00	INSERT INTO credit_card VALUES ('CcU-9999', 'CE01234 5678 9012 3456', '0123 4567 8901 2345', 9999, 000, '12/...	1 row(s) affected	0.0010 sec
3	18:41:33	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) VALUES ('108B1...	1 row(s) affected	0.0014 sec

No se nos ha informado de la fecha de la transacción, así que el registro queda con el campo **timestamp** nulo.

1

-- Comprobamos que el registro se ha insertado correctamente

2

SELECT * FROM transaction

3

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

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

	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	NULL

transaction 4 ×

Apply

Revert

Output

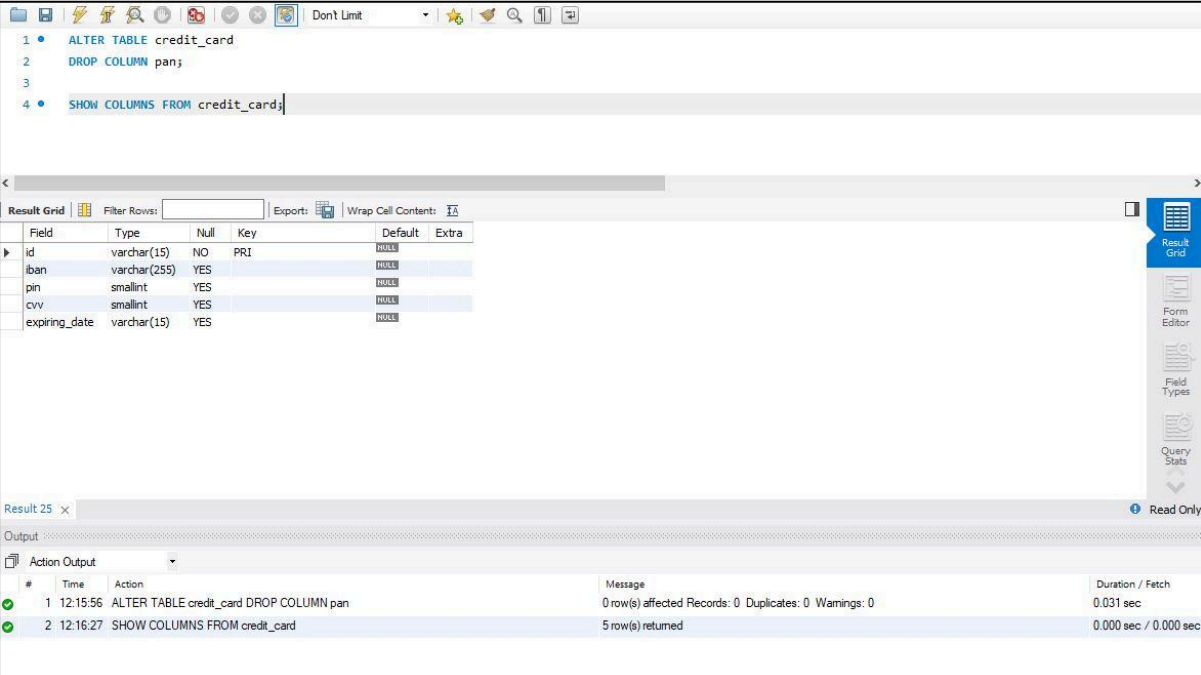
Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	12:26:50	SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-...	1 row(s) returned	0.000 sec / 0.000 sec

- Exercici 4

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

Usamos la sentencia **ALTER TABLE** para eliminar la columna **pan**. A continuación, ejecutamos la sentencia **SHOW COLUMNS** para la tabla **credit_card**, así mostramos los campos de la tabla y comprobamos que no aparece la columna **pan**.



The screenshot shows a database management interface with the following components:

- SQL Editor:** Contains the following queries:


```
1 ALTER TABLE credit_card
2 DROP COLUMN pan;
3
4 SHOW COLUMNS FROM credit_card;
```
- Result Grid:** Displays the structure of the **credit_card** table after the column **pan** has been dropped.

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(255)	YES		NULL	
pin	smallint	YES		NULL	
cvv	smallint	YES		NULL	
expiring_date	varchar(15)	YES		NULL	
- Action Output:** Shows the execution log of the queries.

#	Time	Action	Message	Duration / Fetch
1	12:15:56	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
2	12:16:27	SHOW COLUMNS FROM credit_card	5 row(s) returned	0.000 sec / 0.000 sec

Nivell 2

Exercici 1

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

Comprobamos que el registro existe. Ejecutamos la sentencia **DELETE**, teniendo cuidado de filtrar por el ID solicitado. A continuación, ejecutamos de nuevo la sentencia **SELECT** para comprobar que hemos borrado el registro.

The screenshot shows a database management tool interface. The top section displays a list of SQL queries:

```

1 • SELECT * FROM transaction
2   WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
3
4 • DELETE FROM transaction
5   WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
6
7 • SELECT * FROM transaction
8   WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';

```

Below the queries, there is a "Result Grid" section showing the results of the queries. The first query (SELECT) returned 1 row. The second query (DELETE) affected 1 row. The third query (SELECT) returned 0 rows.

The "Output" section shows the execution details for each query:

#	Time	Action	Message	Duration / Fetch
1	12:25:58	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	1 row(s) returned	0.000 sec / 0.000 sec
2	12:26:09	DELETE FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	1 row(s) affected	0.016 sec
3	12:26:21	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	0 row(s) returned	0.000 sec / 0.000 sec

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.

Creemos la vista mediante la sentencia **CREATE VIEW**, con el nombre indicado, **VistaMarketing**, y creamos la query mediante una left join entre las tablas **company** y **transaction**, usando la función de agregación **AVG**, agrupando por **id** de **company** para obtener el valor de venta medio de cada empresa y filtrando por las transacciones ejecutadas (campo **declined** = 0). Una vez creada la vista, ejecutamos una **SELECT** de todos los valores de la vista para visualizarla.

```

1  CREATE VIEW VistaMarketing AS
2  SELECT c.company_name Empresa, c.phone Teléfono, c.country País, round(AVG(t.amount),2) Media
3  FROM company c
4  JOIN transaction t
5  ON t.company_id = c.id
6  WHERE t.declined = 0
7  GROUP BY c.id;
8
9  -- Mostramos los datos de la vista
10 SELECT * FROM VistaMarketing
11 ORDER BY Media DESC;

```

Empresa	Teléfono	País	Media
Eget Ipsum Ltd	03 67 44 56 72	United States	481.86
Sed Id Limited	07 28 18 18 13	United States	477.51
Neque Tellus Incorporated	04 43 18 34 19	Ireland	477.10
Nunc Sit Incorporated	07 28 42 63 63	Norway	461.83
Non Magna LLC	06 71 73 13 17	United Kingdom	458.74
Maecenas Malesuada Fringilla Inc.	09 38 53 76 61	Netherlands	451.29
Erat LLP	03 18 88 77 79	Netherlands	448.44
Tortor Nunc Commodo Company	05 35 92 77 16	United States	447.11
Justo Eu Arcu Ltd	08 42 56 71 52	Italy	444.16
Pede Cum Ltd	07 62 26 48 38	Norway	442.32
Vestibulum Lorem PC	02 02 87 33 40	Belgium	428.40

#	Time	Action	Message	Duration / Fetch
1	12:42:42	CREATE VIEW VistaMarketing AS SELECT c.company_name Empresa, c.phone T...	0 row(s) affected	0.016 sec
2	12:44:32	SELECT * FROM VistaMarketing ORDER BY Media DESC	101 row(s) returned	0.000 sec / 0.000 sec

DESC fuera de vista

Exercici 3

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

Filtramos la vista a partir del campo definido **País** = 'Germany'.

1 • SELECT * FROM VistaMarketing

2 WHERE País = 'Germany';

100% 25:2

Result Grid

Filter Rows: Search

Export:

Empresa	Teléfono	País	Media
Ac Industries	09 34 65 40 60	Germany	396.15
Auctor Mauris Corp.	05 62 87 14 41	Germany	308.99
Ac Fermentum Incorporated	06 85 56 52 33	Germany	293.57
Aliquam PC	01 45 73 52 16	Germany	280.34
Rutrum Non Inc.	02 66 31 61 09	Germany	266.90
Nunc Interdum Incorporated	05 18 15 48 13	Germany	242.95
Convallis In Incorporated	06 66 57 29 50	Germany	60.99
Augue Foundation	06 88 43 15 63	Germany	15.05

VistaMarketing 6 Read Only

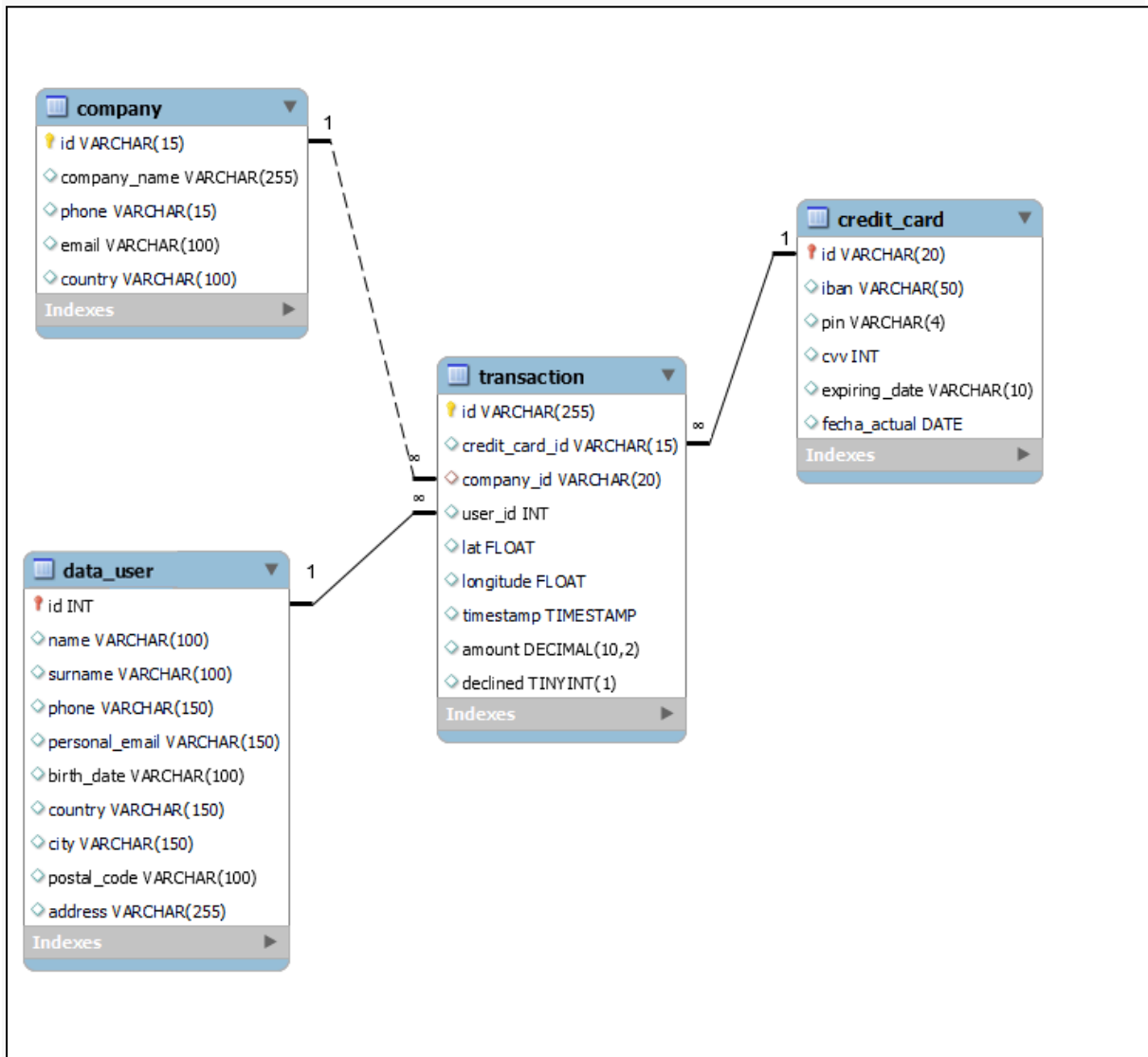
Action Output

	Time	Action	Response	Duration / Fetch Time
1	19:04:11	SELECT * FROM VistaMarketing WHERE País = 'Germany' LIMIT 0, 10000	8 row(s) returned	0.0022 sec / 0.00001...

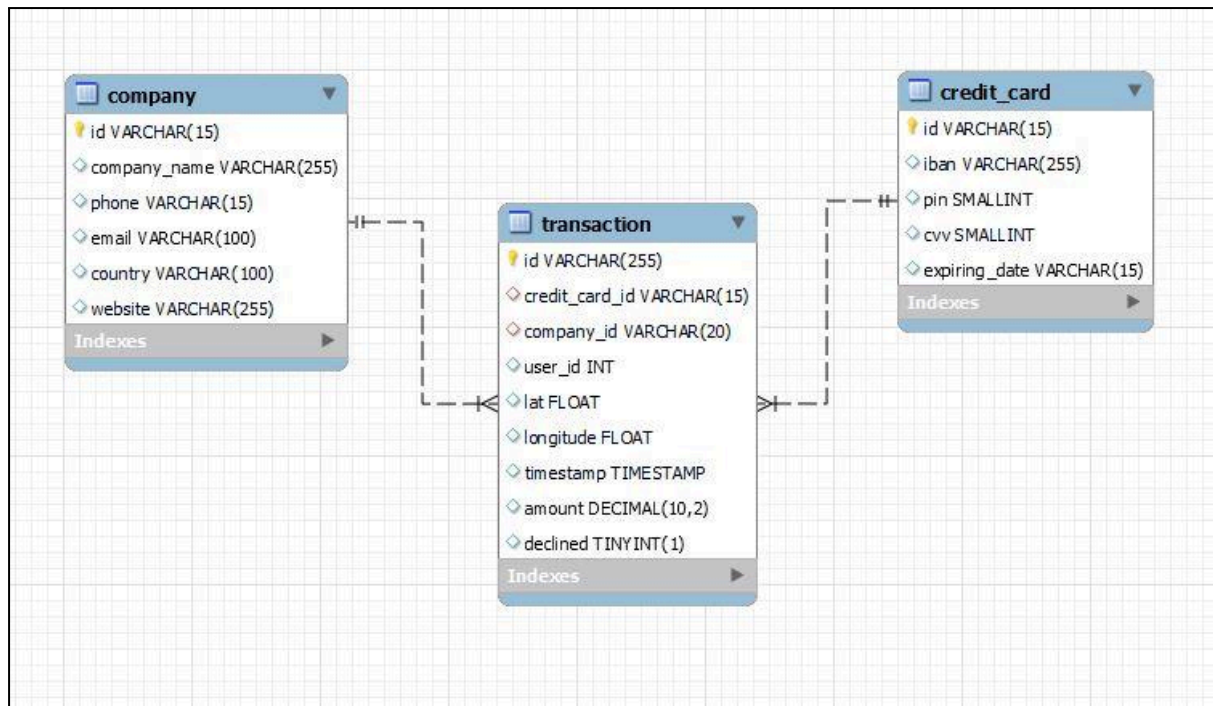
Nivell 3

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 comparamos la estructura de la base de datos que nos ha dejado nuestro compi de equipo con la que hemos estado trabajando durante estos ejercicios:



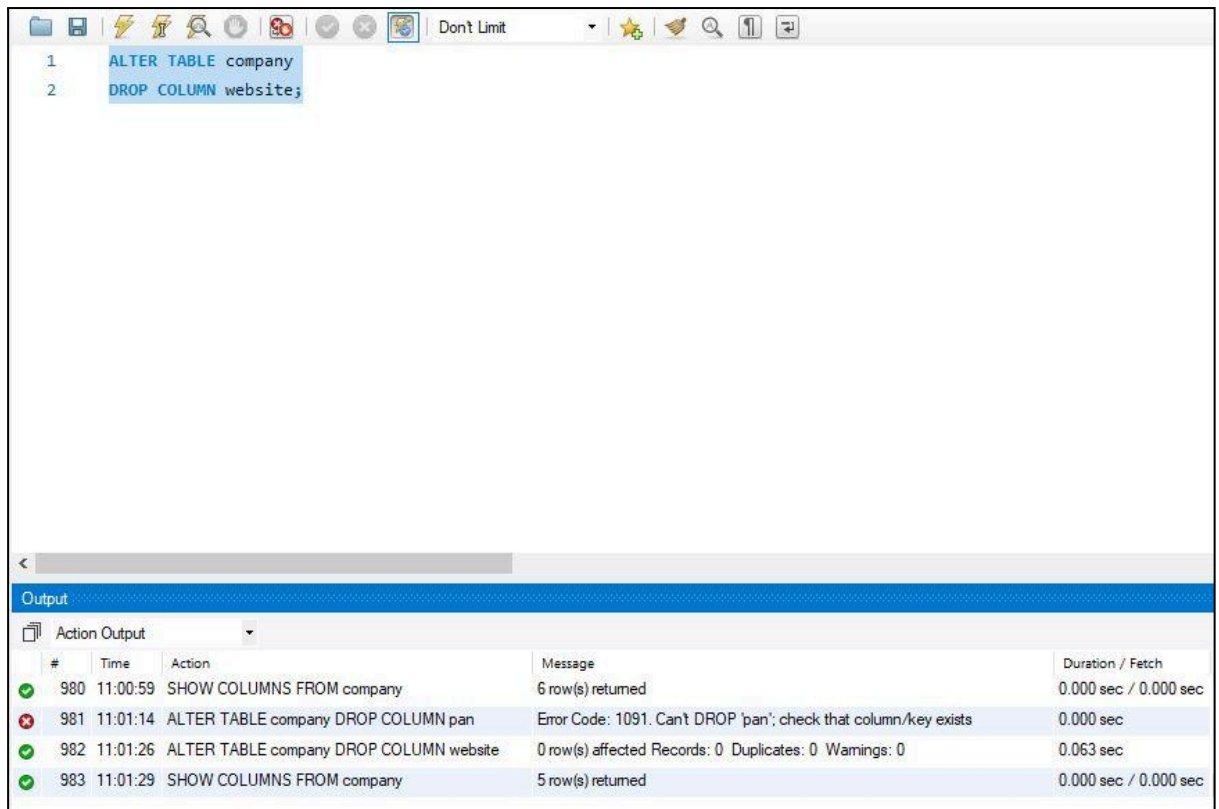
Observamos los siguientes cambios:

1. Se elimina el campo **website** de la tabla **company**;
2. Hay diferencias de tipo y extensión de datos entre las columnas de la tabla **credit_card** usada en los ejercicios y el modelo modificado;
3. Se ha añadido el campo **fecha_actual** con formato **DATE**;
4. Se crea la tabla **data_user** con la estructura del diagrama superior;
5. Se crea una clave foránea que referencia el campo **user_id** de la tabla **transaction**, que referencia al campo **id** de la tabla **data_user**.

NOTA IMPORTANTE: En el esquema de nuestro compi, se han creado claves foráneas desde los campos **id** de **data_user** y **credit_card** que apuntan a los campos **user_id** y **credit_card_id** respectivamente de la tabla **transaction**, cuando las claves foráneas se han de definir justo al revés: las tablas referenciadas han de ser **data_user** y **credit_card**, y no **transaction**. Así lo hicimos en el ejercicio 1 del nivel uno para la tabla **credit_card** y así lo haremos a continuación para **data_user**.

Por tanto, los cambios que tendremos que realizar serán:

1. Modificar la tabla **company** mediante **ALTER TABLE** y eliminar el campo con **DROP COLUMN**:



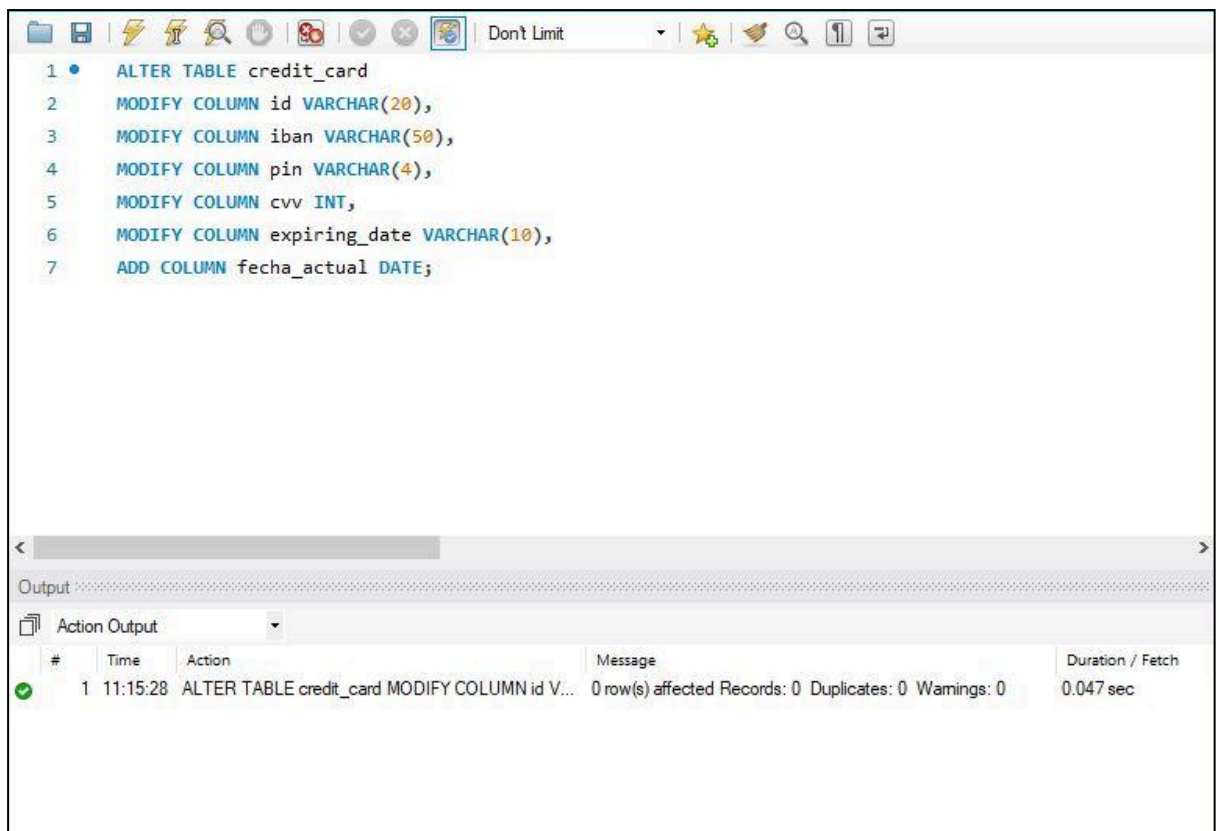
The screenshot shows a SQL IDE interface. The top pane contains a query with two lines:

```
1 ALTER TABLE company
2 DROP COLUMN website;
```

The bottom pane, titled "Output", shows the execution results of the queries. It contains a table with the following data:

#	Time	Action	Message	Duration / Fetch
980	11:00:59	SHOW COLUMNS FROM company	6 row(s) returned	0.000 sec / 0.000 sec
981	11:01:14	ALTER TABLE company DROP COLUMN pan	Error Code: 1091. Can't DROP 'pan'; check that column/key exists	0.000 sec
982	11:01:26	ALTER TABLE company DROP COLUMN website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.063 sec
983	11:01:29	SHOW COLUMNS FROM company	5 row(s) returned	0.000 sec / 0.000 sec

2. Cambiar los tipos de datos de la tabla **credit_card** y añadir el campo **fecha_actual** (que es un "falso amigo" del inglés *actual date*, "fecha real", tal como me señala Avelyn en la corrección) en la que transformaremos el campo alfanumérico **expiring_date** a formato **DATE**:



```

1 • ALTER TABLE credit_card
2   MODIFY COLUMN id VARCHAR(20),
3   MODIFY COLUMN iban VARCHAR(50),
4   MODIFY COLUMN pin VARCHAR(4),
5   MODIFY COLUMN cvv INT,
6   MODIFY COLUMN expiring_date VARCHAR(10),
7   ADD COLUMN fecha_actual DATE;

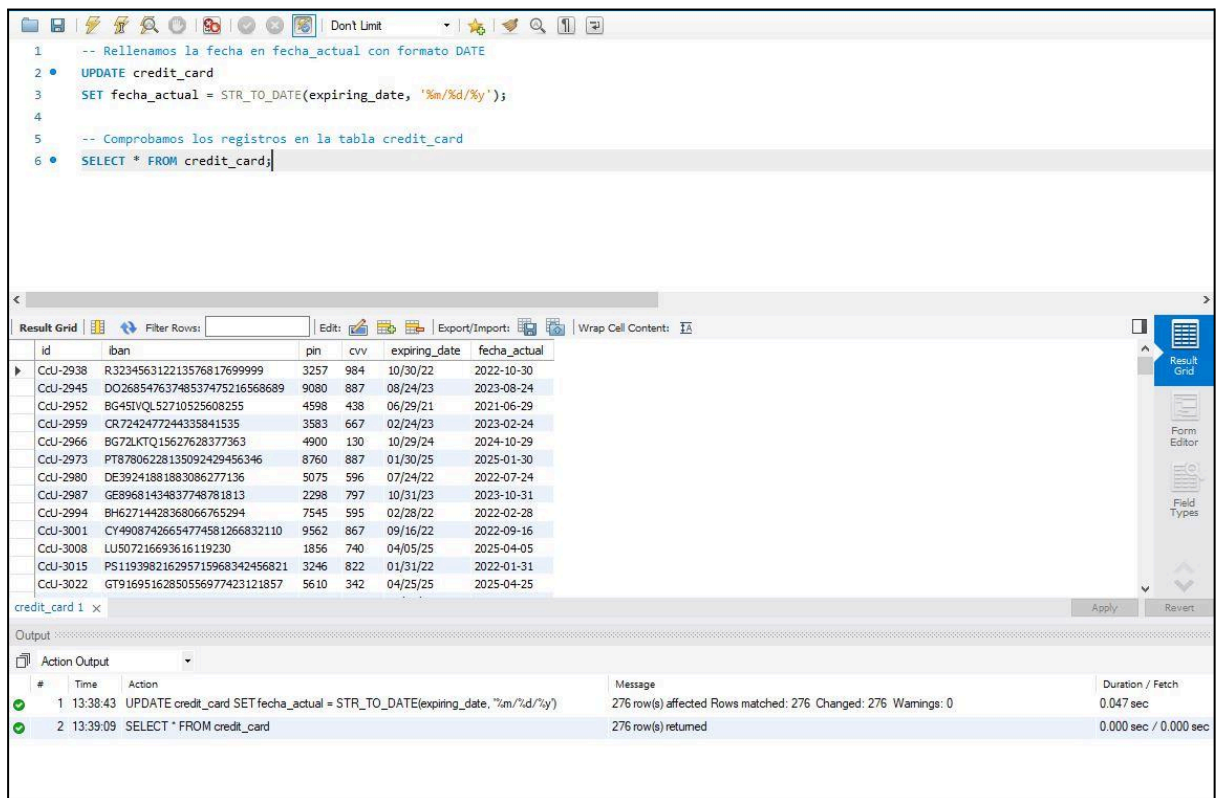
```

Output:

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:15:28	ALTER TABLE credit_card MODIFY COLUMN id V...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.047 sec

Transformamos los datos del campo **expiring_date** a **fecha_actual** con formato **DATE** y, a continuación, comprobamos los datos en la tabla:



```

1 -- Rellenamos la fecha en fecha_actual con formato DATE
2 • UPDATE credit_card
3   SET fecha_actual = STR_TO_DATE(expiring_date, '%m/%d/%y');
4
5 -- Comprobamos los registros en la tabla credit_card
6 • SELECT * FROM credit_card;

```

Result Grid

id	iban	pin	cvv	expiring_date	fecha_actual
CdU-2938	R323456312213576817699999	3257	984	10/30/22	2022-10-30
CdU-2945	DO26854763748537475216568689	9080	887	08/24/23	2023-08-24
CdU-2952	BG45IVQL52710525608255	4598	438	06/29/21	2021-06-29
CdU-2959	CR7242477244335841535	3583	667	02/24/23	2023-02-24
CdU-2966	BG72LKTQ15627628377363	4900	130	10/29/24	2024-10-29
CdU-2973	PT87806228135092429456346	8760	887	01/30/25	2025-01-30
CdU-2980	DE39241881883086277136	5075	596	07/24/22	2022-07-24
CdU-2987	GE89681434837748781813	2298	797	10/31/23	2023-10-31
CdU-2994	BH62714428368066765294	7545	595	02/28/22	2022-02-28
CdU-3001	CY49087426654774581266832110	9562	867	09/16/22	2022-09-16
CdU-3008	LU507216693616119230	1856	740	04/05/25	2025-04-05
CdU-3015	PS119398216295715968342456821	3246	822	01/31/22	2022-01-31
CdU-3022	GT91695162850556977423121857	5610	342	04/25/25	2025-04-25

credit_card 1 x

Output:

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:38:43	UPDATE credit_card SET fecha_actual = STR_TO_DATE(expiring_date, '%m/%d/%y')	276 row(s) affected Rows matched: 276 Changed: 276 Warnings: 0	0.047 sec
2	13:39:09	SELECT * FROM credit_card	276 row(s) returned	0.000 sec / 0.000 sec

- ```
1 CREATE TABLE IF NOT EXISTS user (
2 id INT PRIMARY KEY,
3 name VARCHAR(100),
4 surname VARCHAR(100),
5 phone VARCHAR(150),
6 email VARCHAR(150),
7 birth_date VARCHAR(100),
8 country VARCHAR(150),
9 city VARCHAR(150),
10 postal_code VARCHAR(100),
11 address VARCHAR(255)
12);
```
- Output
- Action Output
- | # | Time     | Action                                            | Message           | Duration / Fetch |
|---|----------|---------------------------------------------------|-------------------|------------------|
| 1 | 13:05:04 | CREATE TABLE IF NOT EXISTS user ( id INT PRIMA... | 0 row(s) affected | 0.047 sec        |

- ```

1  SET foreign_key_checks = 0;
2
3  -- Insertamos datos de user
4
5  INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
6  INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
7  INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
8  INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
9  INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
10 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
11 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
12 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
13 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
14 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
15 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
16 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
17 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
18 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
19 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
20 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
21 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
22 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
23 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
24 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (
25

```

Output

#	Time	Action	Message	Duration / Fetch
275	13:43:26	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, a...	1 row(s) affected	0.000 sec
276	13:43:26	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, a...	1 row(s) affected	0.015 sec
276	13:43:26	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, a...	1 row(s) affected	0.000 sec
277	13:43:26	SET foreign_key_checks = 1	0 row(s) affected	0.000 sec

5. En este punto, vamos a comprobar si no existe algún registro en la tabla **transaction** que no hayamos cargado en la tabla **user**:

The screenshot shows a database query tool interface. The SQL editor at the top contains the following query:

```
1 SELECT user_id FROM transaction
2 WHERE user_id NOT IN (SELECT id FROM user);
```

Below the editor, the 'Result Grid' tab is active, displaying a single row of results:

user_id
9999

The 'Output' section at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	13:09:40	SELECT user_id FROM transaction WHERE user_id NOT IN (SELECT id FROM u...	1 row(s) returned	0.000 sec / 0.000 sec

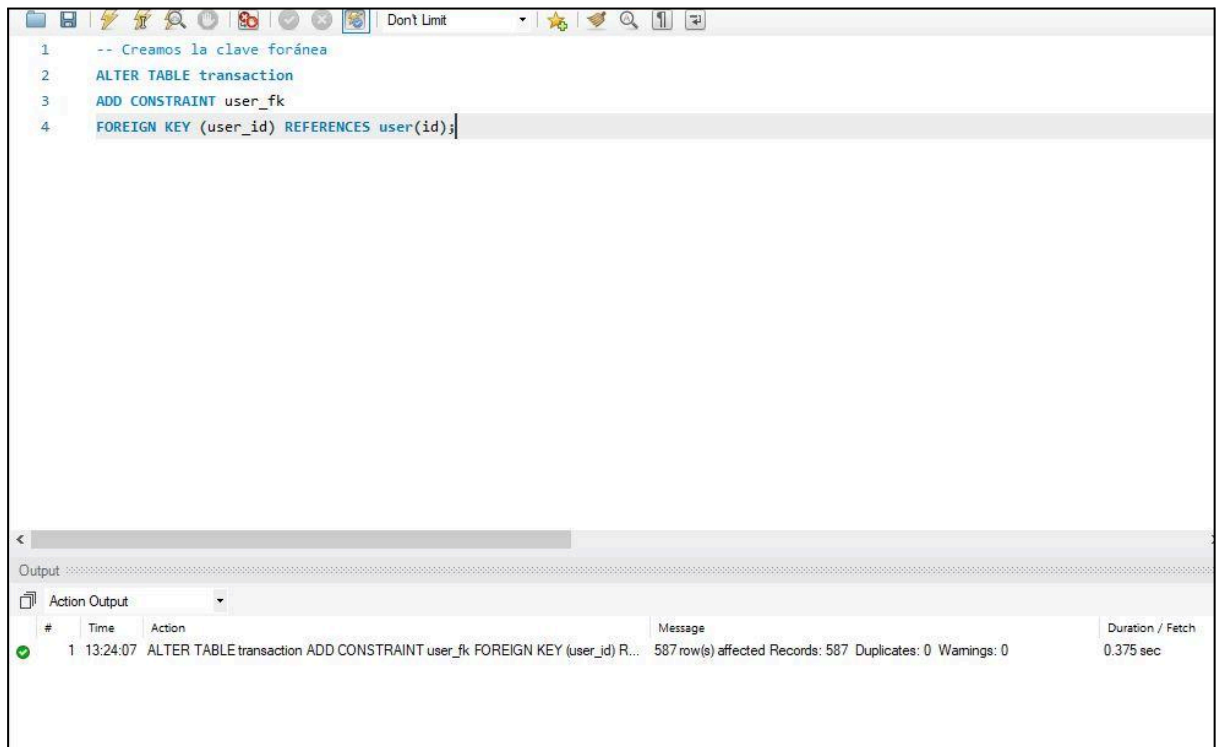
Y comprobamos que, efectivamente, existe un usuario que no hemos dado de alta, que corresponde con el que realizó la transacción del ejercicio 3 del nivel 1. Insertamos este usuario en la tabla **user**:

The screenshot shows the same database query tool interface. The SQL editor now contains an insert query:

```
1 -- Insertamos el usuario 9999
2 INSERT INTO user
3 VALUES (9999, 'Álex', 'Vidal', '666 666 666', 'eleternoaprendiz@proton.me',
4 'Mar 24, 1972', 'Spain', 'Barcelona', '08005', 'Patio, 117, Mi Casa, Particular');
```

The 'Output' section shows the execution log:

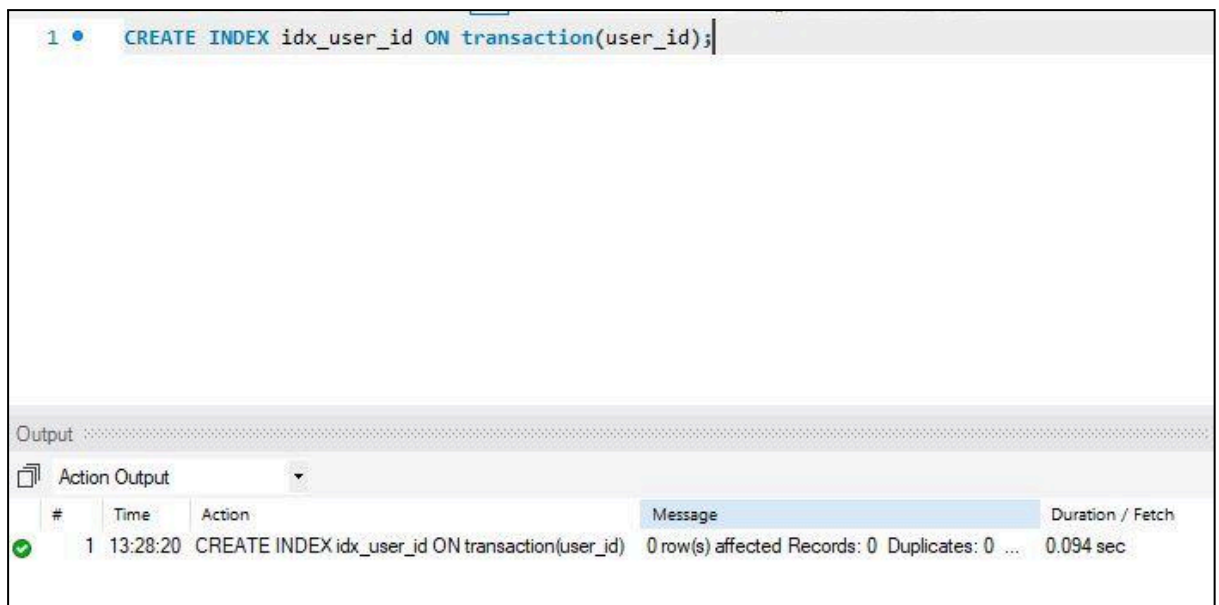
#	Time	Action	Message	Duration / Fetch
1	13:21:52	INSERT INTO user VALUES (9999, 'Álex', 'Vidal', '666 666 666', 'eleternoaprendiz@...	1 row(s) affected 0.016 sec	

6. Creamos la clave foránea **user_fk**:

```
1  -- Creamos la clave foránea
2  ALTER TABLE transaction
3  ADD CONSTRAINT user_fk
4  FOREIGN KEY (user_id) REFERENCES user(id);
```

Output

#	Time	Action	Message	Duration / Fetch
1	13:24:07	ALTER TABLE transaction ADD CONSTRAINT user_fk FOREIGN KEY (user_id) R...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	0.375 sec

7. Nos hemos saltado el paso de indexar el campo **user_id** para facilitar la búsqueda. Lo añadimos al procedimiento:

```
1 • CREATE INDEX idx_user_id ON transaction(user_id);
```

Output

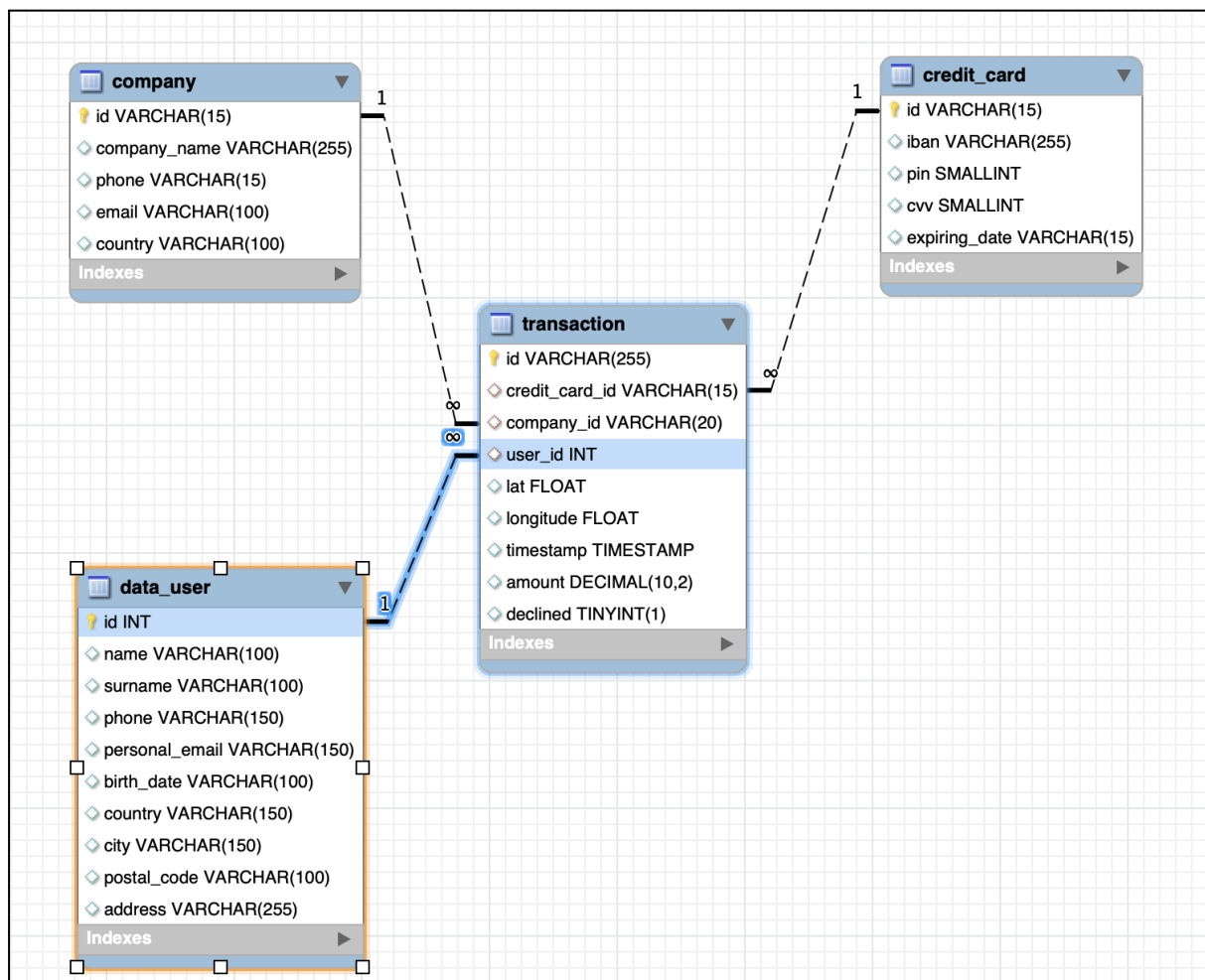
#	Time	Action	Message	Duration / Fetch
1	13:28:20	CREATE INDEX idx_user_id ON transaction(user_id)	0 row(s) affected Records: 0 Duplicates: 0 ...	0.094 sec

8. Nos quedan dos cambios más: cambiar el nombre de la columna **email** a **personal_email** en la tabla **user**, que tuvimos que mantener para introducir los datos del fichero **datos_introducir_user (1).sql** y, en caso que queramos, renombrar la tabla **user** a **data_user**:

1	•	ALTER TABLE user
2	•	RENAME COLUMN email TO personal_email;
3	•	
4	•	RENAME TABLE user TO data_user;

	Time	Action	Response	Duration / Fetch Time
✓ 1	12:52:32	ALTER TABLE user RENAME COLUMN email TO personal_email	0 row(s) affected Records: 0 Duplicates: 0 Warnings...	0.012 sec
✓ 2	12:52:35	RENAME TABLE user TO data_user	0 row(s) affected	0.210 sec

Al ejecutar todas las sentencias nos queda así el diagrama de la base de datos:



La diferencia entre las relaciones de este esquema y el que dejó el/la compi de trabajo es que aquí hay definidas claves foráneas en **transaction** que apuntan a los identificadores de **credit_card** y **data_user**.

Exercici 2

L'empresa també et sol·licita 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.
- Assegura't d'incloure informació rellevant de totes dues taules i utilitza àlies per a canviar de nom columnes segons sigui necessari.

Mostra els resultats de la vista, ordena els resultats de manera descendent en funció de la variable ID de transaction.

A las columnas del enunciado añadimos datos de contacto del usuario y un campo **Transacción** que indique a quien acceda a la vista si la transacción ha sido aceptada o rechazada; así puede ponerse en contacto con el usuario para solventar cualquier problema.

Creamos la vista y mostramos los datos de forma descendente mediante una sentencia **SELECT**, usando **order by desc** de la ID de la transacción:

1

CREATE VIEW InformeTecnico AS

2

SELECT

3

t.id NumTransaccion,

4

u.name Nombre,

5

u.surname Apellidos,

6

u.phone Teléfono,

7

u.personal_email 'Correo-e',

8

cc.iban IBAN,

9

c.company_name Media,

10

t.amount,

11

CASE

12

WHEN t.declined = 0 THEN 'Aceptada'

13

WHEN t.declined = 1 THEN 'Rechazada'

14

ELSE 'Error'

15

END AS 'Transacción'

16

FROM

17

transaction t

18

JOIN

19

data_user u ON t.user_id = u.id

20

JOIN

21

company c ON t.company_id = c.id

22

JOIN

23

credit_card cc ON t.credit_card_id = cc.id;

24

25

SELECT * FROM InformeTecnico

26

ORDER BY NumTransaccion DESC;

100%

30:26

Result Grid

Filter Rows: Search Export:

NumTransaccion	Nombre	Apellidos	Teléfono	Correo-e	IBAN	Media	amount	Transacción
FE96CE47-BD59-381C-E18-E3CA3D44E8FF	Kenyon	Hartman	082-871-7248	convallis.ante.lectus@yahoo.com	DO268547637485374752165868689	Magna A Neque Industries	480.13	Rechazada
FE809ED4-2DB6-55AC-C915-929516E4646B	Molly	Gilliam	0800 120 8023	donec@outlook.couk	SE2813123487163628531121	Nunc Interdum Incorporated	219.83	Aceptada
FD9CBCCD-8E1E-8DA1-4606-7E3ABF3A5A65	Linus	Willis	056-347-2535	ultrices.posuere@yahoo.couk	KW948532754781757886242955643	Nunc Interdum Incorporated	42.32	Aceptada
FD89D91B-AE8D-77DC-E450-88083FB03187	Hilda	Levy	088-867-5267	et.libero@yahoo.org	LT053237077744561475	Malesuada PC	200.72	Aceptada
FD2E9857-4148-BEEC-E9AD-59AA7ABA8290	Hedwig	Gilbert	084-204-8788	sem.eget@icloud.edu	GE84848451582810541526	Neque Tellus Imperdiet Corp.	78.29	Aceptada
FCE2AB9A-271D-2BDC-8E49-8D092A373391	Hakeem	Alford	(0111) 367 0184	adipiscing.ligula@google.edu	MD1234119525145401270486	Nunc Interdum Incorporated	335.56	Aceptada
FBD7E0D6-BA6B-F5BC-OCA9-EA4B8760100C	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	MU4132333444534342541344788855	Mauris Id Inc.	207.09	Rechazada
FAC76A80-8448-69AA-E892-426C2F12621C	Slade	Pooler	084-771-1363	amet@icloud.com	MT05JWCF58868200575771634583813	Arou LLP	304.95	Aceptada
FAAD3FFC-1A17-E141-43D3-359A5BA7CB3B	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	GE9015792884338134463	Lorem Eu Incorporated	149.84	Aceptada
FA053936-75D8-85FA-490D-9B624E1B920A	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	GT02497653655330848247645975	Non Justo Corp.	151.32	Aceptada
F85A7D75-2778-9D75-D776-3F41A828DE88	Sarah	Beck	(358) 691-4345	vitae.risus@aol.couk	VG1468087984174645729577	Ut Semper Foundation	135.93	Aceptada
F843DC08-CCB5-2444-1B4E-5966289FBA8B	Jasper	Landry	1-397-765-1118	consectetuer.euismod@aol.org	VG1468087984174645729577	Ut Semper Foundation	18.08	Aceptada
F5ACD74B-4275-5AA1-2414-46EF417636B98	Nora	Reeves	(01692) 29410	ac@aol.com	MD1234119525145401270486	Nunc Interdum Incorporated	148.97	Aceptada
F56FCA4A-0039-9F64-7376-85632B91121B	Lynn	Riddle	1-387-885-4057	vitae.alliquet@outlook.edu	CR7242477244335841535	Ut Semper Foundation	294.13	Aceptada
F55B3CE1-3379-E0BF-5AB9-6F4CC2C5479C	Sonya	Mckee	041-151-9737	magna.phasellus.dolor@google...	EE54153664818872885	Arou LLP	227.05	Aceptada
F4BCAE41-3B8E-EABD-9C24-466F7CEB9F9A	Chester	Haynes	059-783-3104	ut@protonmail.edu	CY94263537405015481188625576	Malesuada PC	182.01	Aceptada
F2B3E845-2E6D-E891-9D05-33D8ACE58DE4	Heather	Burks	053-586-8671	primis.in@protonmail.com	SM05022751049715477062682363	Malesuada PC	267.52	Aceptada
F28E108B-5418-4667-9514-2E2A823EC0C5	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	RO78DAFC05853348580208155	Pede Ultrices Ltd	114.89	Rechazada
F23CE886-5A74-63B3-8111-09FC98AA38011	Nero	Mills	056 4507 5712	adipiscing@protonmail.org	HU95215627749276573565556322	Arou LLP	146.34	Aceptada
F22BB361-E3CD-BC41-DD6A-A4694F175CD8	Aiko	Chaney	028-660-1876	ante.ipsum.primis@protonmail.ca	MD5723087436783068347555	Non Institute	385.28	Aceptada
F1A598A2-86C5-50A9-F1CE-FB1D9866C39	Craig	Shepherd	1-817-268-8038	feugiat@protonmail.net	R323456312213576817699999	Lorem Eu Incorporated	229.65	Aceptada

InformeTecnico 1

Read Only

Action Output

	Time	Action	Response	Duration / Fetch Time
1	12:56:14	CREATE VIEW InformeTecnico AS SELECT t.id NumTransaccion, u.name Nombre, u.surname Apellidos,...	0 row(s) affected	0.0032 sec
2	12:56:18	SELECT * FROM InformeTecnico ORDER BY NumTransaccion DESC LIMIT 0, 10000	587 row(s) returned	0.0063 sec / 0.0011 s...