

Tarea S3.01. Manipulación de tablas

En este sprint, se simula una situación empresarial en la que debes realizar diversas manipulaciones en las tablas de la base de datos. A su vez, deberás trabajar con índices y vistas. En esta actividad, continuarás trabajando con la base de datos que contiene información de una empresa dedicada a la venta de productos en línea. En esta tarea, comenzarás a trabajar con información relacionada con tarjetas de crédito.

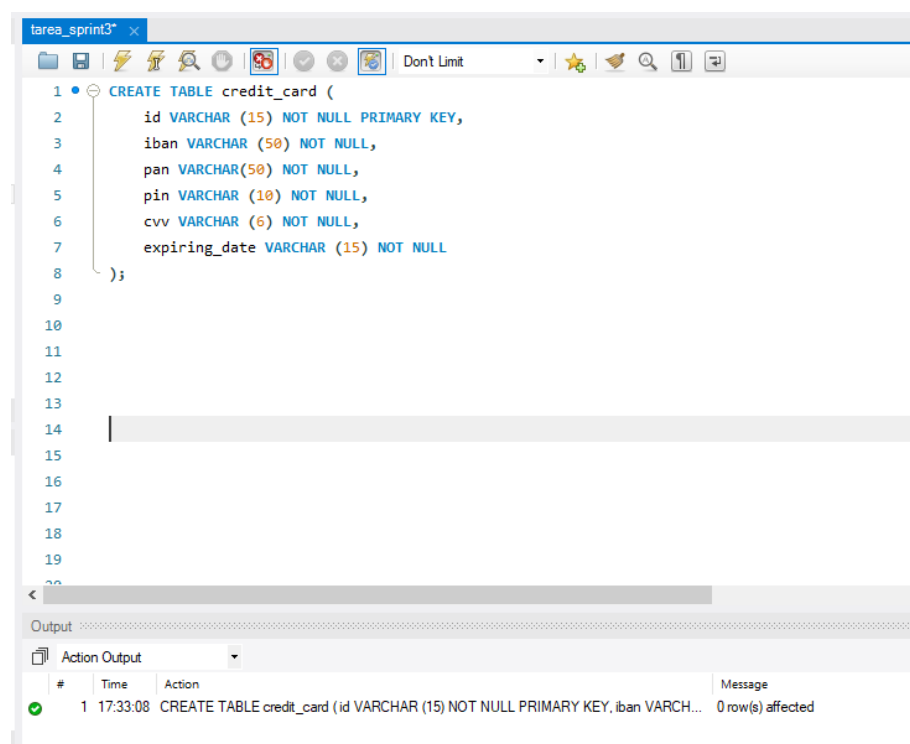
NIVEL 1

- Ejercicio 1

Tu tarea es diseñar y crear una tabla llamada "credit_card" que almacene detalles cruciales sobre las tarjetas de crédito. La nueva tabla debe ser capaz de identificar de manera única cada tarjeta y establecer una relación adecuada con las otras dos tablas ("transaction" y "company"). Después de crear la tabla, será necesario que ingreses la información del documento denominado "datos_introducir_credit". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

Con el comando CREATE TABLE creo la tabla Credit_Card con todos sus campos, los campos son los que aparecen en el archivo 'datos_introducir_credit', al crear la tabla he indicado que tipo de datos son y sus peculiaridades cómo longitud, si no debe ser nulo...Indico también cuál será la Primary Key.

Los datos son obligatorios (no nulos) porque son datos necesarios para hacer transacciones con las tarjetas.



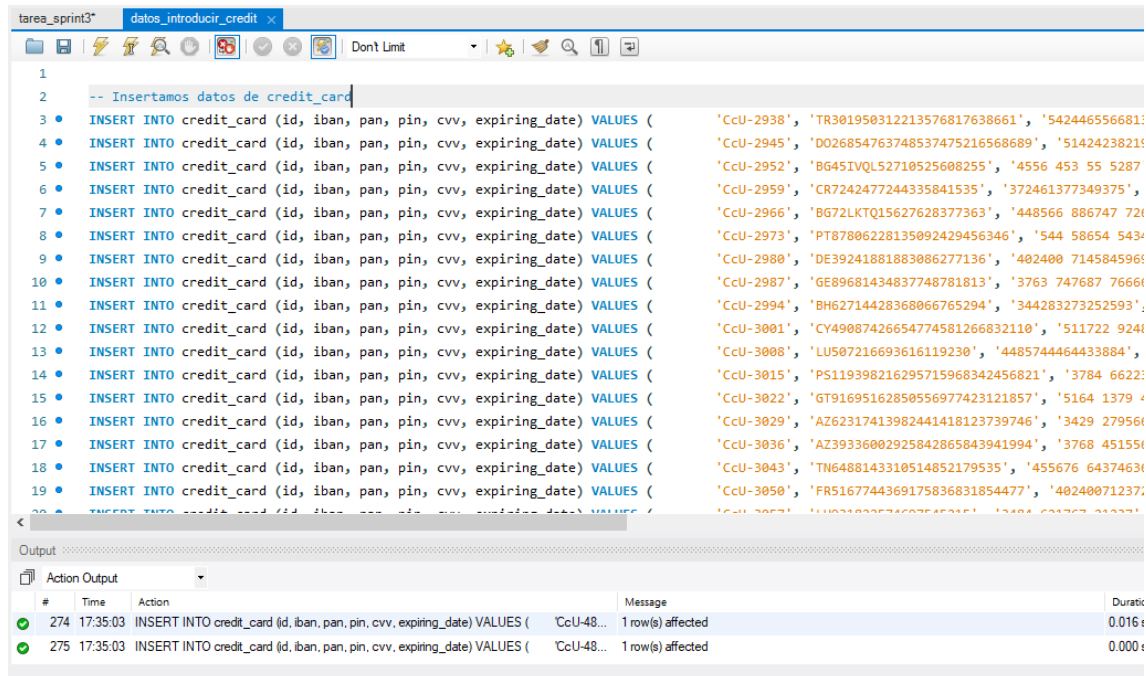
```
1 CREATE TABLE credit_card (  
2     id VARCHAR (15) NOT NULL PRIMARY KEY,  
3     iban VARCHAR (50) NOT NULL,  
4     pan VARCHAR(50) NOT NULL,  
5     pin VARCHAR (10) NOT NULL,  
6     cvv VARCHAR (6) NOT NULL,  
7     expiring_date VARCHAR (15) NOT NULL  
8 );  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20
```

Output

#	Time	Action	Message
✓ 1	17:33:08	CREATE TABLE credit_card (id VARCHAR (15) NOT NULL PRIMARY KEY, iban VARCH...	0 row(s) affected

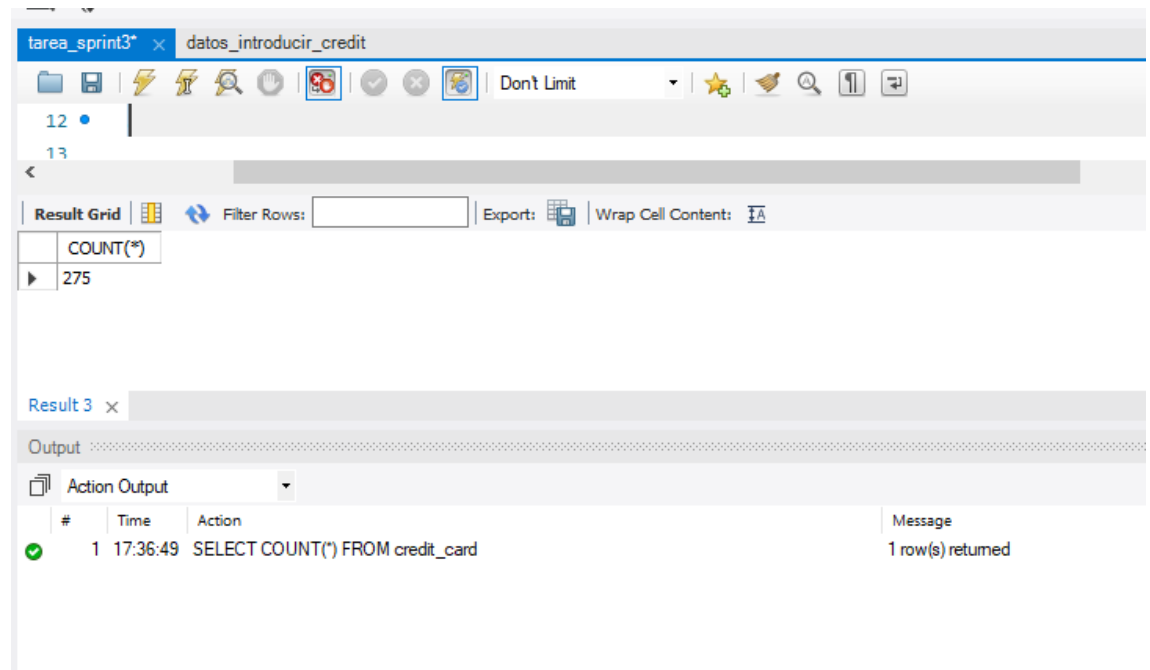
Tarea S3.01. Manipulación de tablas

A continuación de crear la tabla he introducido los datos del archivo abriendo el script y ejecutando el insert, y comprobando cuantos se han cargado en credit_card



```
1
2 -- Insertamos datos de credit_card
3 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2938', 'TR301950312213576817638661', '542446556681
4 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2945', 'D026854763748537475216568689', '5142423821
5 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2952', 'BG45IVQL52710525608255', '4556 453 55 5287
6 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2959', 'CR7242477244335841535', '372461377349375',
7 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2966', 'BG72LKTQ15627628377363', '448566 886747 72
8 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2973', 'PT87806228135092429456346', '544 58654 543
9 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2980', 'DE39241881883086277136', '402400 714584596
10 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2987', 'GE89681434837748781813', '3763 747687 766
11 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2994', 'BH62714428368066765294', '344283273252593',
12 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3001', 'CY49087426654774581266832110', '511722 924
13 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3008', 'LU507216693616119230', '4485744464433884',
14 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3015', 'PS119398216295715968342456821', '3784 6622
15 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3022', 'GT91695162850556977423121857', '5164 1379 4
16 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3029', 'AZ62317413982441418123739746', '3429 27956
17 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3036', 'AZ39336002925842865843941994', '3768 45155
18 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3043', 'TN6488143310514852179535', '455676 6437463
19 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3050', 'FR5167744369175836831854477', '40240071237
20 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3057', 'LU31833573463745315', '3484 631767 31333
```

#	Time	Action	Message	Duration
274	17:35:03	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-48...	1 row(s) affected	0.016 s
275	17:35:03	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-48...	1 row(s) affected	0.000 s



```
12 •
13
```

Result Grid
COUNT(*)
275

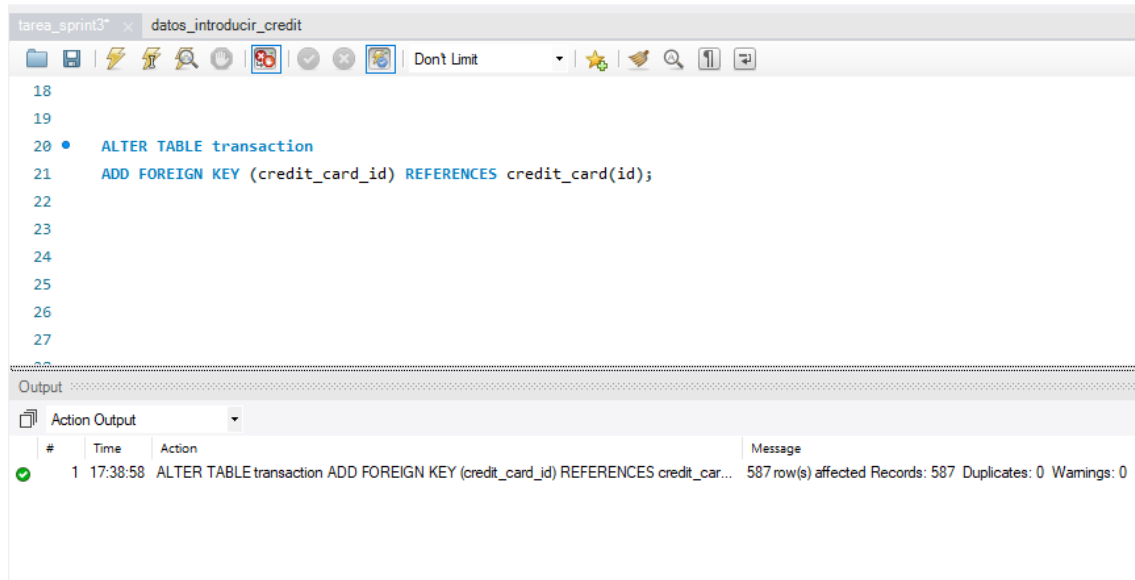
Result 3 x

#	Time	Action	Message
1	17:36:49	SELECT COUNT(*) FROM credit_card	1 row(s) returned

Tarea S3.01. Manipulación de tablas

Indico que para la tabla transaction el credit_card_id será clave foránea apuntando a el campo id de la tabla credit_card.

En la tabla transaction el company_id será clave foránea apuntando a el campo id de la tabla company. (Esto ya viene heredado del sprint2)



The screenshot shows a SQL IDE window titled 'tarea_sprint3*' with a sub-tab 'datos_introducir_credit'. The SQL editor contains the following code:

```
18
19
20 • ALTER TABLE transaction
21   ADD FOREIGN KEY (credit_card_id) REFERENCES credit_card(id);
22
23
24
25
26
27
28
```

Below the editor is the 'Output' pane, which shows the execution results:

#	Time	Action	Message
✓ 1	17:38:58	ALTER TABLE transaction ADD FOREIGN KEY (credit_card_id) REFERENCES credit_car...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0

A continuación voy a especificar el tipo de dato que he escogido para cada campo y porqué:

Id: Identificador de la tarjeta de crédito. Es la PRIMARY KEY de la tabla ya que es el diferenciador único de cada tarjeta, no se puede repetir. No puede aparecer en blanco, por ello NOT NULL. Es de tipo VARCHAR para que pueda contener datos de tipo alfanuméricos y que pueda ser variable su longitud. He indicado una longitud máxima de 15 (me he fijado en el campo credit card id de transaction). No puede ser un tipo de dato numérico ya que en el caso de contener ceros iniciales serían anulados y tampoco realizaremos operaciones matemáticas con él. Tiene que ser del mismo tipo que el credit card id de transaction

Iban: Identificador único para las cuentas bancarias, en este caso es el que está vinculado a la tarjeta. No puede aparecer en blanco, por ello NOT NULL. Es de tipo VARCHAR para que pueda contener datos de tipo alfanuméricos y pueda ser variable su longitud ya que dependiendo del país de origen de la cuenta puede variar. He indicado una longitud máxima de 50.

Pan: Numero de la tarjeta bancaria, es único. No puede aparecer en blanco, por ello NOT NULL. Es de tipo VARCHAR y no numérico para que en el caso de contener ceros iniciales no sean anulados y tampoco realizaremos operaciones matemáticas con él. Su longitud es de 50

Tarea S3.01. Manipulación de tablas

Pin: Código de acceso a la tarjeta y de seguridad. No puede aparecer en blanco, por ello NOT NULL. Es de tipo VARCHAR para que pueda ser variable su longitud ya que puede ser variable por diferentes causas. La longitud óptima de un pin es de 4 a 6 dígitos pero indico una longitud máxima de 10 dígitos por si hubiese alguna excepción en algún país. No puede ser un tipo de dato numérico para que en el caso de contener ceros iniciales no sean anulados y tampoco realizaremos operaciones matemáticas con él.

Cvv: Código de seguridad de la tarjeta para verificar que el usuario tiene la tarjeta en su poder a la hora de hacer transacciones online. No puede aparecer en blanco, por ello NOT NULL. Es de tipo VARCHAR para que pueda ser variable su longitud ya que aunque normalmente es de 3 dígitos hay casos muy concretos en el que lo es de 4 indico una longitud de 5 por precaución.

Expiring_date: Es la fecha de caducidad de la tarjeta de esta manera se valida que la tarjeta es operativa en la fecha de realizar la transacción. Es NOT NULL para que no aparezca en blanco y VARCHAR con un límite de 15 dígitos. Se podría haber introducido también como dato de tipo DATE. *Si lo pusiéramos como DATE, habría que modificar los insert para donde se especifica el valor de la fecha '10/30/24', ya que no coincide con el formato Date, habría que hacerlo indicando el formato para que no falle STR_TO_DATE('10/30/24','%m/%d/%y'). Otra opción es mantenerlo como varchar, no tocamos los insert, y si al hacer una consulta queremos tratar el resultado como fecha para algo especial (como sacar el mes), en la consulta, en vez de hacer select expiring_date, habría que hacer select STR_TO_DATE(expiring_date, '%m/%d/%y'). Esta es la opción que vamos a utilizar.*

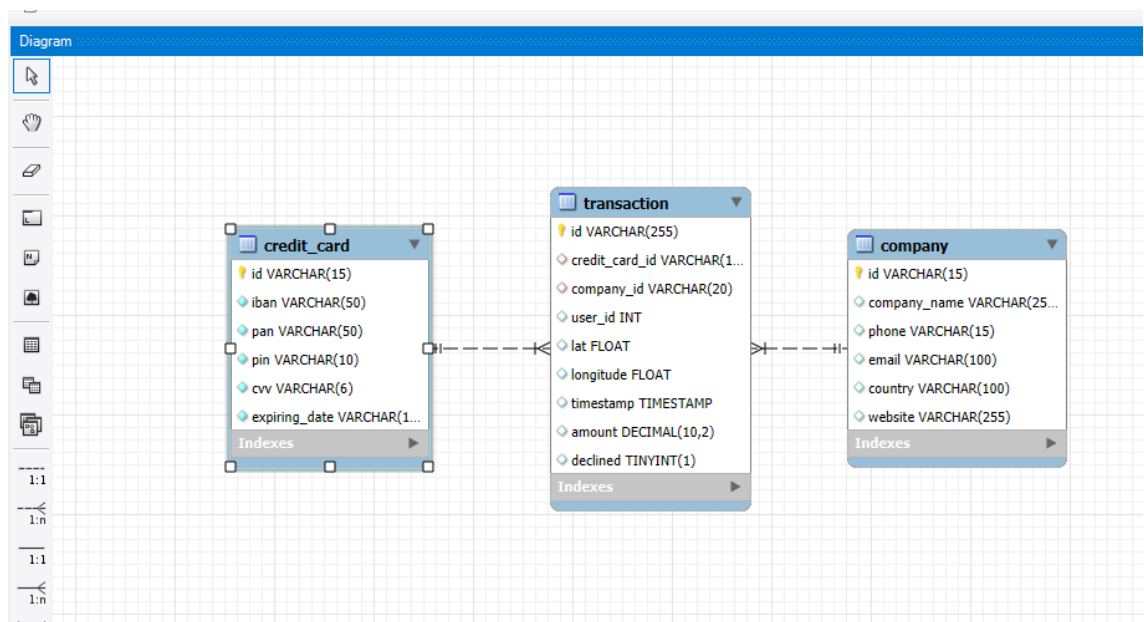
La nueva tabla credit_card se compone de 275 líneas y 7 columnas.

Las tablas forman parte de una base de datos relacional que siguen el modelo estrella siendo la tabla Transaction la tabla de hechos y las tablas Company y Credit_card las tablas de dimensiones.

Tipos de relaciones en el modelo son N:1 siendo muchos en la tabla de hechos Transactions y de 1 en las tablas de dimensiones Company y Credit_card. Una empresa puede tener muchas transacciones pero una transacción solo puede pertenecer a una empresa. En el caso de una 1 a tabla credit_card, una tarjeta puede ser usada en muchas transacciones pero en una transacción solo se utiliza un tarjeta.

Tarea S3.01. Manipulación de tablas

El diagrama EER sería el siguiente:



Comprobamos cómo nos queda la tabla:

task_sprint3* x datos_introducir_credit

16

17 • `SELECT * FROM credit_card;`

Result Grid

	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	BG72LKTO15627628377363	448566 886747 7265	4900	130	10/29/24

credit_card 4 x

Output

Action Output

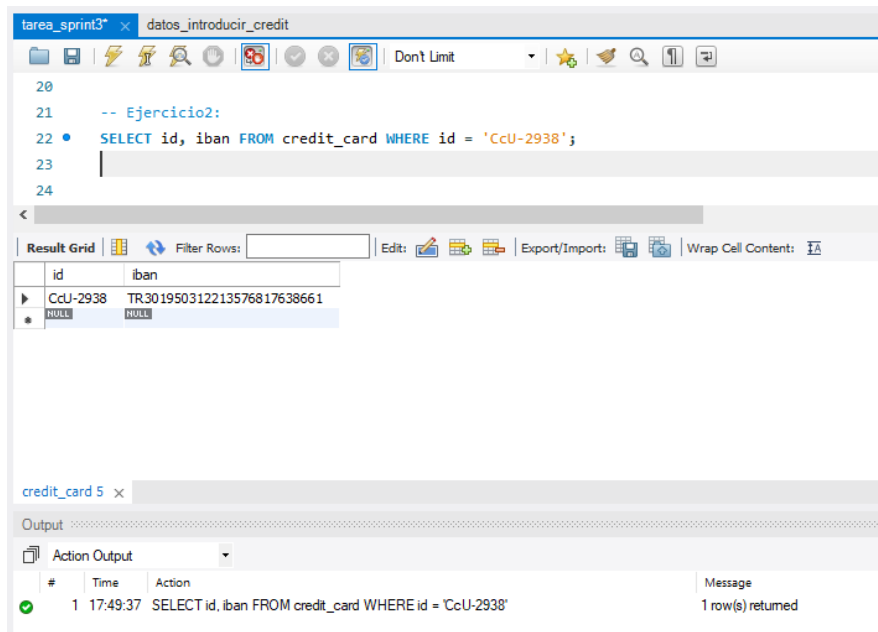
#	Time	Action	Message
✓ 1	17:47:14	SELECT * FROM credit_card	275 row(s) returned

Tarea S3.01. Manipulación de tablas

• Ejercicio 2

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

Compruebo los datos que había en el registro a modificar, el que pertenecía al usuario CcU -2938.



The screenshot shows a SQL IDE window titled 'datos_introducir_credit'. The SQL editor contains the following code:

```
20
21 -- Ejercicio2:
22 • SELECT id, iban FROM credit_card WHERE id = 'CcU-2938';
23
24
```

Below the editor, the 'Result Grid' displays the results of the query:

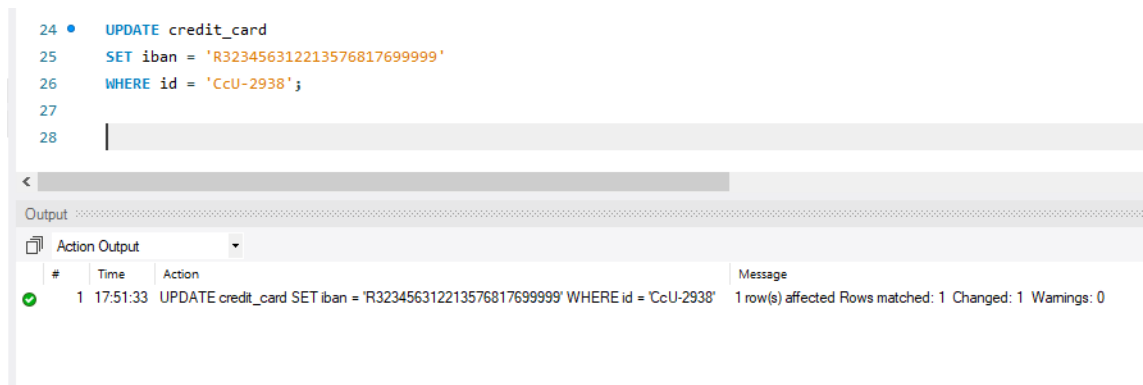
id	iban
CcU-2938	TR301950312213576817638661

The 'Output' pane shows the 'Action Output' for the query:

#	Time	Action	Message
1	17:49:37	SELECT id, iban FROM credit_card WHERE id = 'CcU-2938'	1 row(s) returned

Con el comando UPDATE indico la tabla donde voy a realizar los cambios.

Seguidamente con el comando SET indico el nuevo valor para el campo Iban ya que este es el dato a modificar y por último con la condición WHERE señalo el número de Id.



The screenshot shows the same SQL IDE window. The SQL editor now contains the following code:

```
24 • UPDATE credit_card
25 SET iban = 'R323456312213576817699999'
26 WHERE id = 'CcU-2938';
27
28
```

The 'Output' pane shows the 'Action Output' for the update query:

#	Time	Action	Message
1	17:51:33	UPDATE credit_card SET iban = 'R323456312213576817699999' WHERE id = 'CcU-2938'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0

Tarea S3.01. Manipulación de tablas

Compruebo que se ha hecho el cambio:

The screenshot shows a database management interface. At the top, a SQL query is entered: `SELECT id, iban FROM credit_card WHERE id = 'CcU-2938';`. Below the query, the 'Result Grid' displays the results of the query. The grid has two columns: 'id' and 'iban'. The first row shows 'CcU-2938' and 'R323456312213576817699999'. The second row shows 'NULL' and 'NULL'. Below the result grid, the 'Output' pane shows the 'Action Output' for the query. It indicates that the query was executed successfully at 17:52:28 and returned 1 row(s).

id	iban
CcU-2938	R323456312213576817699999
NULL	NULL

credit_card 7 x

Output

Action Output

#	Time	Action	Message
1	17:52:28	SELECT id, iban FROM credit_card WHERE id = 'CcU-2938'	1 row(s) returned

• Ejercicio 3

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

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

Tarea S3.01. Manipulación de tablas

Primero he intentado añadir directamente a Transaction la información que figura en el enunciado:

```
32
33 • INSERT INTO transaction
34 (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
35 VALUES
36 ('10881D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', 829.999, -117.999, 111.11, 0);
37
```

Output

#	Time	Action	Message	Duration
1	18:05:04	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, ...	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (Transacti...	0.000 s

Y me ha resultado un error, indicando que fallan las claves foráneas

Tras ver que el problema venía por claves foráneas ya que antes de insertar un registro nuevo en la tabla de hechos, este caso la Transaction, se han de ingresar en la tabla de dimensiones company.

Creo en la tabla company b-9999 con el comando insert into

```
38 • INSERT INTO company (id) VALUES ('b-9999');
39 • SELECT * FROM company WHERE id = 'b-9999';
```

Result Grid

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

18:06:27	INSERT INTO company (id) values (b-9999)	1 row(s) affected
18:07:13	SELECT * FROM company WHERE id = 'b-9999'	1 row(s) returned

Luego creo la tarjeta de crédito:

```
41 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date)
42 VALUES ('CcU-9999', '0', '0', '0', '0', '2025-01-01');
43
```

Output

#	Time	Action	Message
1	18:16:26	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-9999', '0', '0', '0', '0', '2025-01-01')	1 row(s) affected

Tarea S3.01. Manipulación de tablas

Ahora ya sí que con el código anterior del comando INSERT INTO podemos añadir el nuevo registro a la tabla de hechos Transactions.

```
38
39 • INSERT INTO transaction
40     (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
41     VALUES
42     ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', 829.999, -117.999, 111.11, 0);
43
```

Output

Action Output

#	Time	Action	Message
1	18:18:01	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, ...	1 row(s) affected

Comprobamos que el registro se ha añadido correctamente:

```
43
44 • SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';
45
```

Result Grid

	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 12 x

Output

Action Output

#	Time	Action	Message
1	18:19:18	SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD'	1 row(s) returned

• Ejercicio 4

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

Con el comando DROP COLUMN, eliminamos la columna que necesitamos:

```
45
46 -- Ejercicio 4
47 • ALTER TABLE credit_card
48   DROP COLUMN pan;
```

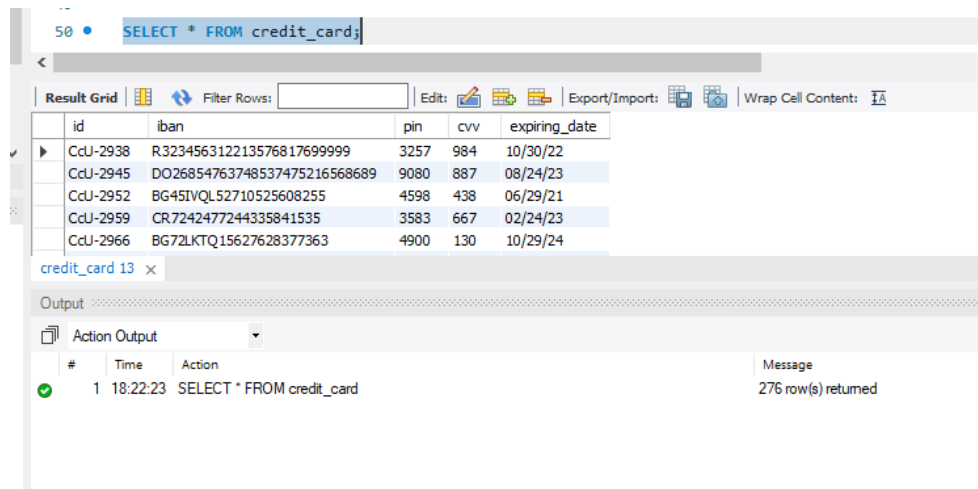
Output

Action Output

#	Time	Action	Message
1	18:21:39	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Tarea S3.01. Manipulación de tablas

Comprobamos que se ha eliminado correctamente la columna pan:



50 • `SELECT * FROM credit_card;`

Result Grid

	id	iban	pin	cvv	expiring_date
▶	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

credit_card 13 x

Output

Action Output

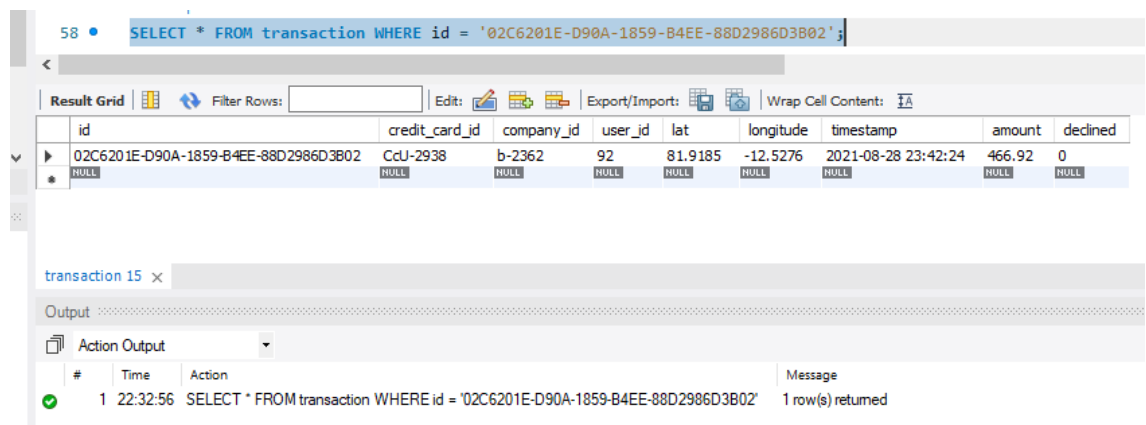
#	Time	Action	Message
✓ 1	18:22:23	SELECT * FROM credit_card	276 row(s) returned

NIVEL 2

• Ejercicio 1

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

Hago la comprobación antes de ejecutar el comando DELETE del registro en la tabla Transaction.



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

Result Grid

	id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
▶	02C6201E-D90A-1859-B4EE-88D2986D3B02	CcU-2938	b-2362	92	81.9185	-12.5276	2021-08-28 23:42:24	466.92	0
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

transaction 15 x

Output

Action Output

#	Time	Action	Message
✓ 1	22:32:56	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	1 row(s) returned

Ejecuto el comando DELETE:



60 • `DELETE FROM transaction`

61 `WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';`

Output

Action Output

#	Time	Action	Message
✓ 1	22:50:03	DELETE FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	1 row(s) affected

Tarea S3.01. Manipulación de tablas

Compruebo que se ha eliminado el registro:

```
62
63 -- Compruebo que se ha eliminado el registro
64 • SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
65
```

The screenshot shows a SQL IDE interface. At the top, a query is executed. Below it, a 'Result Grid' is displayed with columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. All cells in the grid contain 'NULL'. Below the grid, there is a tab labeled 'transaction 19'. At the bottom, an 'Output' window shows the 'Action Output' for the query, indicating '0 row(s) returned'.

• Ejercicio 2

El departamento de marketing desea tener acceso a información específica para realizar análisis y estrategias efectivas. Se ha solicitado crear una vista que proporcione detalles clave sobre las compañías y sus transacciones. Será necesario que crees una vista llamada VistaMarketing que contenga la siguiente información: Nombre de la compañía, teléfono de contacto, país de residencia y el promedio de compra realizado por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor promedio de compra.

Creo la vista con el comando CREATE VIEW

```
66 -- Ejercicio2
67 • CREATE VIEW VistaMarketing AS
68 SELECT
69     c.company_name,
70     c.phone,
71     c.country,
72     AVG(t.amount) AS average_sales
73 FROM
74     company AS c
75 INNER JOIN
76     transaction AS t
77 ON c.id = t.company_id
78 GROUP BY company_name, phone, country
79 ORDER BY average_sales DESC;
80
```

The screenshot shows a SQL IDE interface. A query to create a view is executed. Below it, an 'Output' window shows the 'Action Output' for the query, indicating '0 row(s) affected'.

Tarea S3.01. Manipulación de tablas

Compruebo la vista creada

The screenshot shows the SQL Server Enterprise Manager interface. The query editor at the top contains the following SQL statement:

```
81 • SELECT * FROM VistaMarketing;
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query. The results are shown in a table with the following columns: company_name, phone, country, and average_sales. The table contains 101 rows of data.

company_name	phone	country	average_sales
Eget Ipsum 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

Below the result grid, the 'Output' tab is active, showing the 'Action Output' for the query. The output indicates that the query was executed successfully and returned 101 rows.

#	Time	Action	Message
1	22:59:15	SELECT * FROM VistaMarketing	101 row(s) returned

• Ejercicio 3

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

En la vista aplico un filtro con WHERE. No hace falta volver a introducir el código de la vista si no que cómo ya está creada sólo es necesario llamarla cómo si de cualquier otra tabla se tratara

The screenshot shows the SQL Server Enterprise Manager interface. The query editor at the top contains the following SQL statement:

```
103 • SELECT * FROM vistamarketing  
104 WHERE country = 'Germany';  
105
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query. The results are shown in a table with the following columns: company_name, phone, country, and average_sales. The table contains 8 rows of data, all of which are from Germany.

company_name	phone	country	average_sales
Aliquam PC	01 45 73 52 16	Germany	385.265000
Ac Industries	09 34 65 40 60	Germany	289.645000
Rutrum Non Inc.	02 66 31 61 09	Germany	266.900000
Nunc Interdum Incorporated	05 18 15 48 13	Germany	244.025238
Augue Foundation	06 88 43 15 63	Germany	240.800000
Ac Fermentum Incorporated	06 85 56 52 33	Germany	206.465000
Auctor Mauris Corp.	05 62 87 14 41	Germany	184.310000
Convallis In Incorporated	06 66 57 29 50	Germany	156.730000

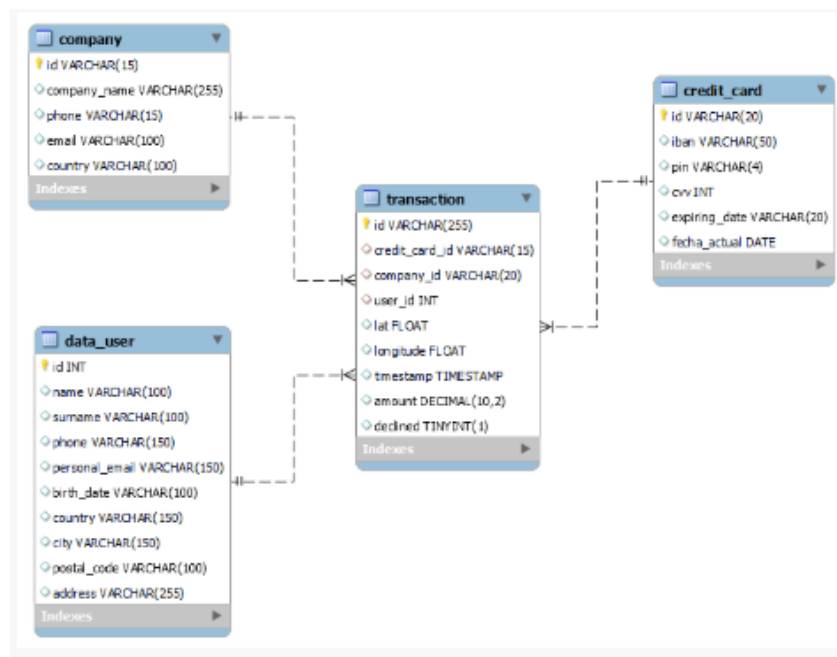
Below the result grid, the 'Output' tab is active, showing the 'Action Output' for the query. The output indicates that the query was executed successfully and returned 8 rows.

#	Time	Action	Message
1	23:03:31	SELECT * FROM vistamarketing WHERE ...	8 row(s) returned

NIVEL 3

- Ejercicio 1

La próxima semana tendrás una nueva reunión con los gerentes de marketing. Un compañero de tu equipo realizó modificaciones en la base de datos, pero no recuerda cómo las hizo. Te pide que lo ayudes a dejar los comandos ejecutados para obtener el siguiente diagrama:



Recordatorio

En esta actividad, es necesario que describas el "paso a paso" de las tareas realizadas. Es importante realizar descripciones sencillas, simples y fáciles de comprender. Para realizar esta actividad, deberás trabajar con los archivos denominados "estructura_dades_user" y "dades_introducir_user".

Tarea S3.01. Manipulación de tablas

Antes de nada, ejecuto el script estructura_datos_user para crear el índice idx_user_id sobre transaction(user_id) y crear la tabla user si no existe. Quito la Fk del script original porque no tiene sentido y no daría la cardinalidad de un modelo de estrella en el diagrama.

The screenshot shows a database IDE with a search bar at the top containing 'data_user'. The main editor displays a SQL script for creating a table named 'user'. The script is as follows:

```
88  
89 CREATE TABLE IF NOT EXISTS user (  
90     id INT PRIMARY KEY,  
91     name VARCHAR(100),  
92     surname VARCHAR(100),  
93     phone VARCHAR(150),  
94     email VARCHAR(150),  
95     birth_date VARCHAR(100),  
96     country VARCHAR(150),  
97     city VARCHAR(150),  
98     postal_code VARCHAR(100),  
99     address VARCHAR(255)  
100 );
```

Below the editor, the 'Output' pane shows the 'Action Output' for the executed script:

#	Time	Action	Message
1	21:00:40	CREATE TABLE IF NOT EXISTS user (id INT PRIMARY KEY, name VARCHAR...	0 row(s) affected

Ejecuto el script para insertar los datos de usuarios datos_introducir_user

The screenshot shows a database IDE with multiple tabs open: 'tarea_sprint3*', 'datos_introducir_credit', 'estructura_datos_user', and 'datos_introducir_user (1)'. The active tab 'datos_introducir_user (1)' displays a SQL script for inserting data into the 'user' table. The script starts with setting foreign key checks to 0 and then inserts 13 rows of user data. The script is as follows:

```
1 SET foreign_key_checks = 0;  
2  
3 -- Insertamos datos de user  
4 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "1", "Zeus", "Gamb"  
5 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "2", "Garrett", "H"  
6 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "3", "Ciaran", "Ha"  
7 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "4", "Howard", "St"  
8 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "5", "Hayfa", "Pie"  
9 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "6", "Joel", "Tysc"  
10 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "7", "Rafael", "Ji"  
11 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "8", "Nissim", "Fr"  
12 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "9", "Mannix", "Mc"  
13 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "10", "Robert", "H"  
14 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "11", "Joan", "Bai"  
15 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "12", "Benedict", "  
16 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "13", "Allegra", "
```

Below the editor, the 'Output' pane shows the 'Action Output' for the executed script:

#	Time	Action	Message	Du
273	17:55:14	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_co...	1 row(s) affected	0.0
274	17:55:14	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_co...	1 row(s) affected	0.0
275	17:55:14	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_co...	1 row(s) affected	0.0
276	17:55:14	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_co...	1 row(s) affected	0.0
277	17:55:14	SET foreign_key_checks = 1	0 row(s) affected	0.0

Tarea S3.01. Manipulación de tablas

Hago un select count para comprobar cuantos datos se han cargado

The screenshot shows a SQL IDE interface. At the top, a query is entered: `SELECT COUNT(*) FROM user;`. Below the query editor, a 'Result Grid' is displayed with a single row containing the value '275'. Below the result grid, there is an 'Output' section with a dropdown menu set to 'Action Output'. The 'Action Output' log shows a single entry with a green checkmark, indicating a successful execution of the query.

#	Time	Action	Message
1	23:02:18	SELECT COUNT(*) FROM user	1 row(s) returned

Realizo todos los cambios necesarios para dejar la base de datos como se indica en el enunciado:

- Renombro la tabla user como data_user
- Renombro la columna email de la tabla data_user
- Elimino la columna website de la tabla company
- Añado la columna fecha_actual a la tabla credit-card.

The screenshot shows a SQL IDE interface with a list of SQL commands and their execution results. The commands are as follows:

```
105 • ALTER TABLE user RENAME to data_user;
106 • ALTER TABLE data_user RENAME COLUMN email to personal_email;
107 -- Hago este insert porque sino al agregar la FK no se podria realizar..., ya que saldria foreign key violada
108 • INSERT INTO data_user (id) VALUES (9999);
109 • ALTER TABLE transaction
110     ADD CONSTRAINT
111     FOREIGN KEY (user_id) REFERENCES data_user(id);
112 • ALTER TABLE company DROP COLUMN website;
113 • ALTER TABLE credit_card ADD COLUMN fecha_actual DATE;
114 • ALTER TABLE credit_card MODIFY id VARCHAR(20);
115 • ALTER TABLE credit_card MODIFY pin VARCHAR(4);
116 • ALTER TABLE credit_card MODIFY cvv INT;
117 • ALTER TABLE credit_card MODIFY expiring_date VARCHAR(20);
```

The 'Action Output' log shows the results of these commands:

#	Time	Action	Message
1	21:18:34	ALTER TABLE user RENA...	0 row(s) affected
2	21:18:34	ALTER TABLE data_user ...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
3	21:18:35	INSERT INTO data_user (...)	1 row(s) affected
4	21:18:35	ALTER TABLE transaction...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0
5	21:18:35	ALTER TABLE company ...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
6	21:18:35	ALTER TABLE credit_card...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
7	21:18:35	ALTER TABLE credit_card...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Tarea S3.01. Manipulación de tablas

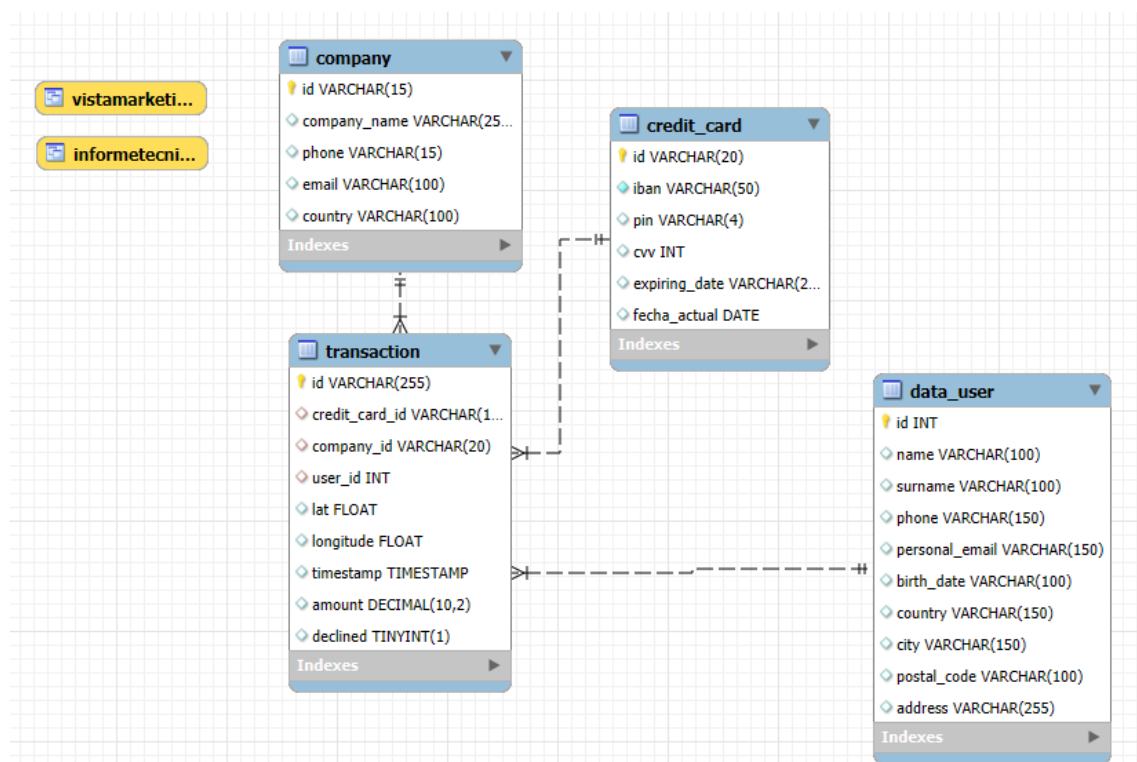
Hago modificaciones de los tipos de datos y longitud de algunos de los campos de la tabla credit_card

```
120 • ALTER TABLE credit_card MODIFY id VARCHAR(20);
121 • ALTER TABLE credit_card MODIFY pin VARCHAR(4);
122 • ALTER TABLE credit_card MODIFY cvv INT;
123 • ALTER TABLE credit_card MODIFY expiring_date VARCHAR(20);
```

Output

#	Time	Action	Message
✓ 1	23:21:19	ALTER TABLE credit_card MODIFY id VARCHAR(20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
✓ 2	23:21:22	ALTER TABLE credit_card MODIFY pin VARCHAR(4)	276 row(s) affected Records: 276 Duplicates: 0 Warnings: 0
✓ 3	23:21:25	ALTER TABLE credit_card MODIFY cvv INT	276 row(s) affected Records: 276 Duplicates: 0 Warnings: 0
✓ 4	23:21:30	ALTER TABLE credit_card MODIFY expiring_date VARCHAR(20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Saco el Diagrama ER para ver que tras los cambios queda igual que el diagrama que se presentaba en el enunciado



Tarea S3.01. Manipulación de tablas

• Ejercicio 2

La empresa también te solicita crear una vista llamada "InformeTecnico" que contenga la siguiente información:

- ID de la transacción
- Nombre del usuario/a
- Apellido del usuario/a
- IBAN de la tarjeta de crédito utilizada
- Nombre de la compañía de la transacción realizada

Asegúrate de incluir información relevante de ambas tablas y utiliza alias para renombrar las columnas según sea necesario. Muestra los resultados de la vista, ordenando los resultados de manera descendente en función de la variable ID de la transacción.

En primer lugar creo la vista:

```
128 • CREATE VIEW InformeTecnico AS
129 SELECT
130     t.id AS id_transaction,
131     du.name AS nombre_usuario,
132     du.surname AS apellido_usuario,
133     cc.iban,
134     c.company_name AS compañía
135 FROM transaction as t
136 INNER JOIN data_user AS du ON t.user_id = du.id
137 INNER JOIN credit_card AS cc ON t.credit_card_id = cc.id
138 INNER JOIN company AS c ON t.company_id = c.id
139 ORDER BY id_transaction DESC;
140
```

Output

Action Output

#	Time	Action	Message
✓ 1	23:27:32	CREATE VIEW InformeTecnico AS SELECT t.id AS id_transaction, du...	0 row(s) affected

Comprobamos que la vista se ha creado correctamente:

```
143
144 • SELECT * FROM InformeTecnico;
145
```

Result Grid

id_transaction	nombre_usuario	apellido_usuario	iban	compañía
FE96CE47-8D59-381C-4E18-E3CA3D-4E8FF	Kenyon	Hartman	DO26854763748537475216568689	Magna A Neque Industries
FE809ED4-2DB6-55AC-C915-929516E46468	Molly	Gilliam	SE2813123487163628531121	Nunc Interdum Incorporated
FD9CBCCD-8E1E-8DA1-4606-7E3A6F3A5A65	Linus	Willis	KW9485332754781757886242955643	Nunc Interdum Incorporated
FD89D51B-AE8D-77DC-E450-B8083FBD3187	Hilda	Levy	LT053237077744561475	Malesuada PC
FD2E8957-4148-BEEC-E9AD-59AA7A8A6290	Hedwig	Gilbert	GE84848451582810541526	Neque Tellus Imperdiet Corp.

InformeTecnico 26

Output

Action Output

#	Time	Action	Message
✓ 1	23:28:43	SELECT * FROM InformeTecnico	586 row(s) returned