

SPRINT _ 3

MANIPULACION DE TABLAS

NIVEL 1 _ EJERCICIO 1

Su 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 cada carta de una manera única y establecer una relación apropiada con las otras dos tablas ("transaction" y "company"). Después de crear la tabla, será necesario introducir la información del documento llamado "dades_introducir_credit". Recuerde mostrar el diagrama y hacer una breve descripción de él.

Creación de la tabla credit_card.

El datatype de todos los campos es definido como VARCHAR como valor generico al no tener información exacta del tipo de datos que usará cada campo.

Los campos "pin" y "cvv" podrían ser definidos como tipo INT ya que todos los datos conocidos actualmente corresponden a este tipo de datos.

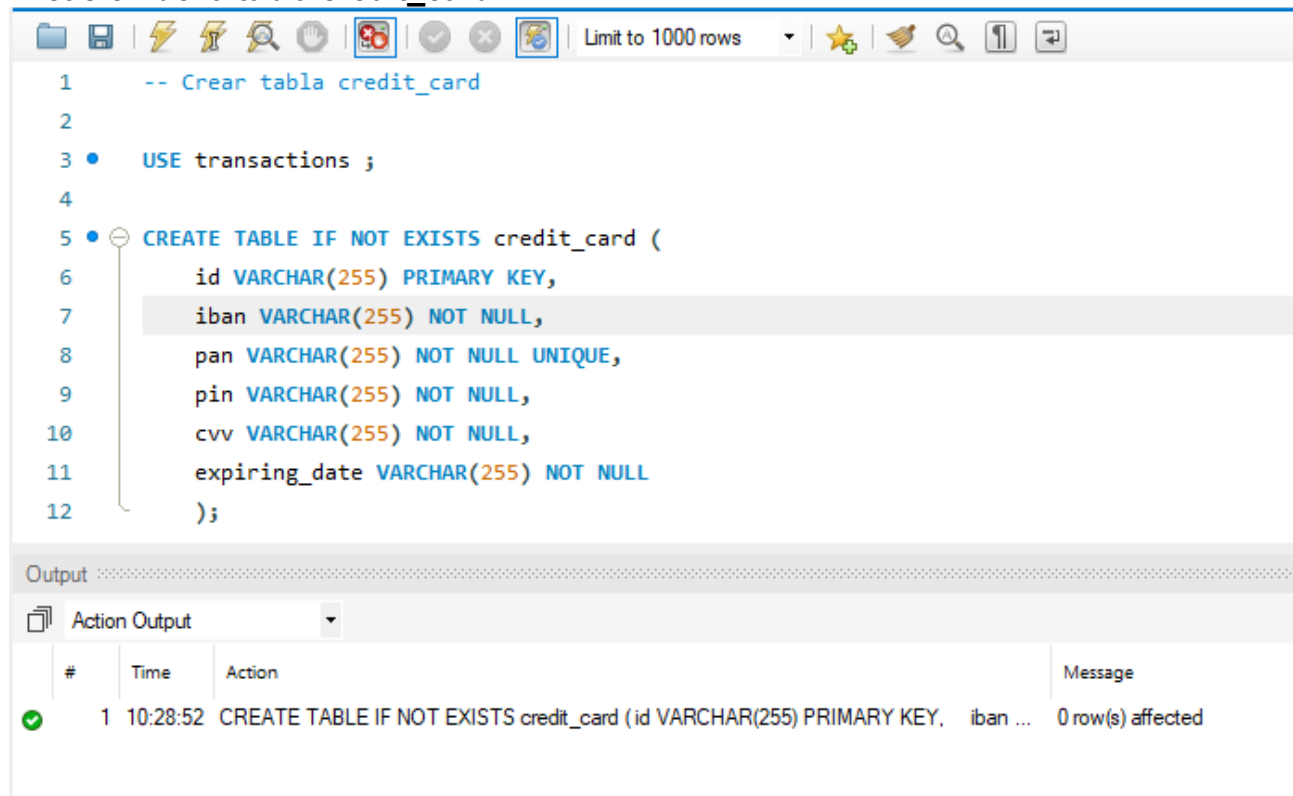
El dato "expiring_date" se define como VARCHAR para evitar posibles conflictos de formato en la introducción de datos si fuese definida como tipo DATE.

El tipo de datos puede ser modificado posteriormente para adaptarlo al formato DATE.

Se han definido todos los campos como NOT NULL considerando que todos los datos de la credit card son necesarios para la acreditación de la transacción.

El campo "pan" corresponde al número de la credit card, el cual identifica de forma univoca la credit card por lo que se definido como UNIQUE.

Creación de la tabla credit_card.



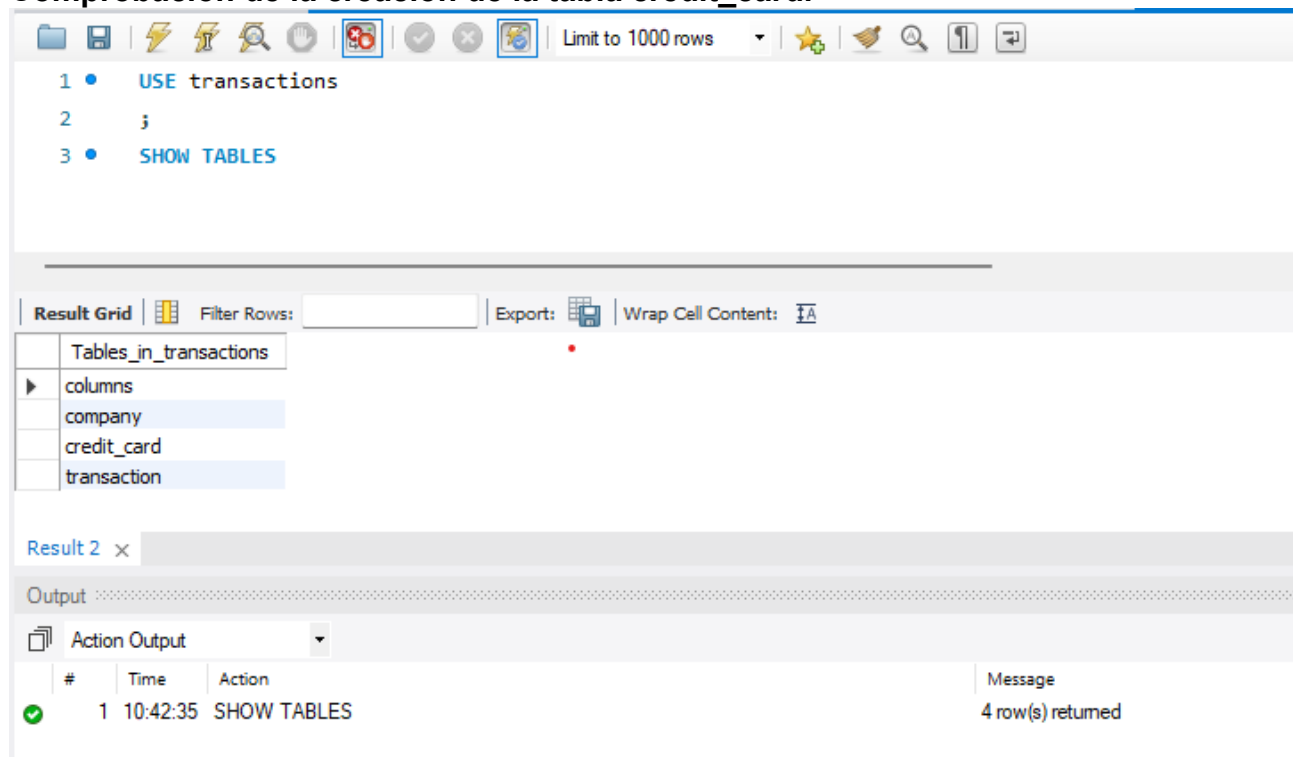
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search. The main editor contains the following SQL code:

```
1  -- Crear tabla credit_card
2
3  •  USE transactions ;
4
5  •  CREATE TABLE IF NOT EXISTS credit_card (
6      id VARCHAR(255) PRIMARY KEY,
7      iban VARCHAR(255) NOT NULL,
8      pan VARCHAR(255) NOT NULL UNIQUE,
9      pin VARCHAR(255) NOT NULL,
10     cvv VARCHAR(255) NOT NULL,
11     expiring_date VARCHAR(255) NOT NULL
12 );
```

Below the editor is the 'Output' pane, which is currently showing 'Action Output'. It contains a single entry:

#	Time	Action	Message
1	10:28:52	CREATE TABLE IF NOT EXISTS credit_card (id VARCHAR(255) PRIMARY KEY, iban ...	0 row(s) affected

Comprobación de la creación de la tabla credit_card.



The screenshot shows the same SQL IDE interface. The main editor contains the following SQL code:

```
1  •  USE transactions
2      ;
3  •  SHOW TABLES
```

Below the editor is the 'Result Grid' pane, which shows the output of the 'SHOW TABLES' command. It displays a table with the following data:

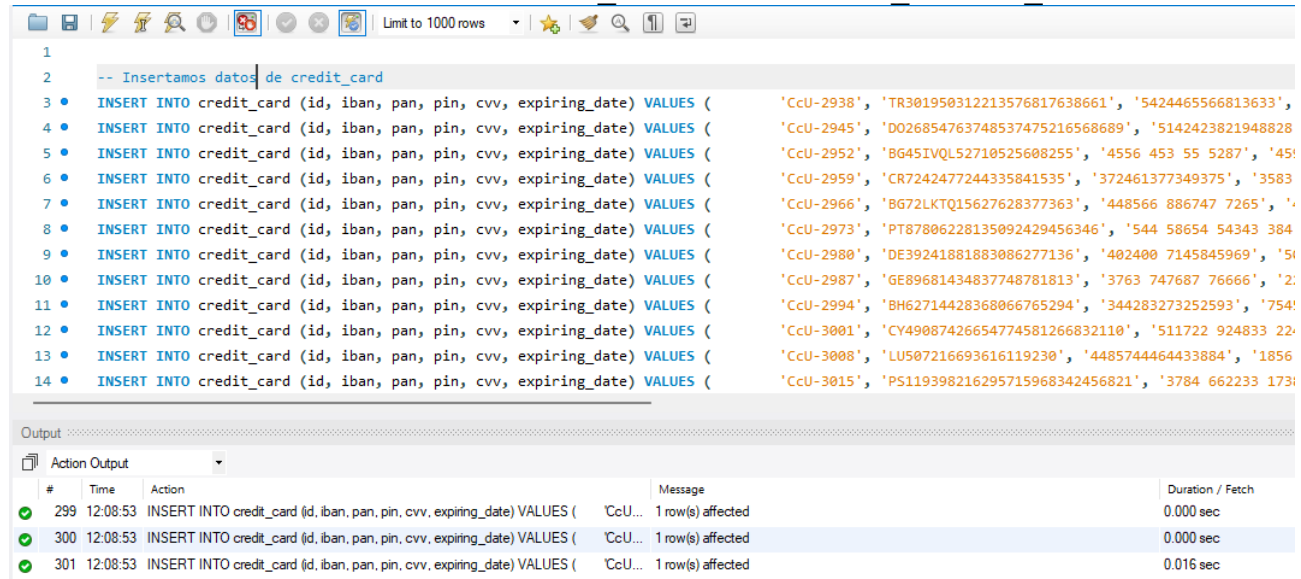
Tables_in_transactions
columns
company
credit_card
transaction

Below the Result Grid is the 'Output' pane, which is currently showing 'Action Output'. It contains a single entry:

#	Time	Action	Message
1	10:42:35	SHOW TABLES	4 row(s) returned

Introducción de datos en la tabla credit_card.

Se introducen los datos en la tabla “credit_card” desde “dades_introducir_credit”.



The screenshot shows a SQL IDE with a toolbar at the top. The main editor contains 14 SQL statements, all starting with a comment: `-- Insertamos datos de credit_card`. Each statement is an `INSERT INTO credit_card` command with columns `(id, iban, pan, pin, cvv, expiring_date)` and a `VALUES` clause containing a list of values. The statements are numbered 1 through 14. Below the editor, the 'Output' pane is visible, showing the 'Action Output' tab. It contains a table with 5 columns: '#', 'Time', 'Action', 'Message', and 'Duration / Fetch'. It shows the execution of the first three statements, each resulting in '1 row(s) affected' and a duration of '0.000 sec'.

```
1
2 -- Insertamos datos de credit_card
3 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2938', 'TR301950312213576817638661', '5424465566813633',
4 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2945', 'D026854763748537475216568689', '5142423821948828
5 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2952', 'BG45IVQL52710525608255', '4556 453 55 5287', '45
6 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2959', 'CR7242477244335841535', '372461377349375', '3583
7 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2966', 'BG72LKTQ15627628377363', '448566 886747 7265', '
8 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2973', 'PT87806228135092429456346', '544 58654 54343 384
9 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2980', 'DE39241881883086277136', '402400 7145845969', '5
10 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2987', 'GE89681434837748781813', '3763 747687 76666', '2
11 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2994', 'BH62714428368066765294', '344283273252593', '754
12 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3001', 'CY49087426654774581266832110', '511722 924833 22
13 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3008', 'LU507216693616119230', '4485744464433884', '1856
14 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-3015', 'PS119398216295715968342456821', '3784 662233 173
```

#	Time	Action	Message	Duration / Fetch
299	12:08:53	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU...	1 row(s) affected	0.000 sec
300	12:08:53	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU...	1 row(s) affected	0.000 sec
301	12:08:53	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU...	1 row(s) affected	0.016 sec

Creación de la foreign key en la tabla transaction referenciada a la tabla credit_card.

Se crea la foreign key.

Con esto se genera una relación de uno a muchos entre la tabla “credit_card” y la tabla “transaction”. Una credit_card puede ser usada en multiples transacciones y una transacción solo puede ser asignada a una credit_card.

Esta relación es entre los campos credit_card(id) PK y transaction(credit_card_id) FK.

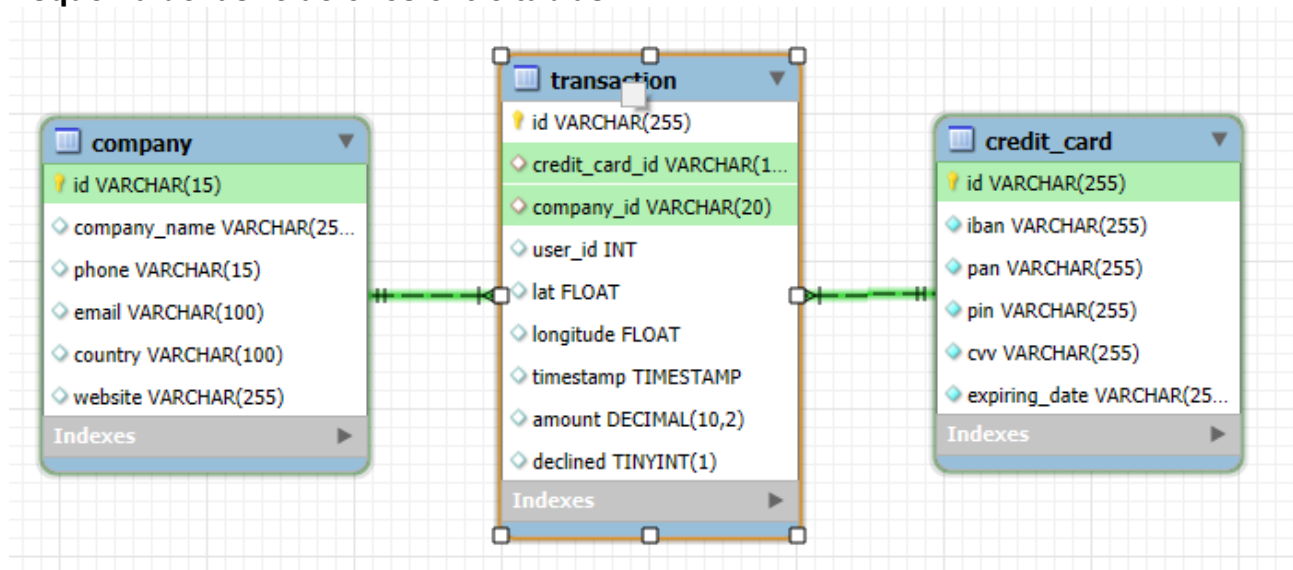
Al crear la relación se crean automaticamente los indices.

Creación de la relacion entre las tablas transaction y credit_card.

```
3 • ALTER TABLE transaction ADD foreign key (credit_card_id) REFERENCES credit_card(id)
4 ;
```

✓	2	11:15:02	ALTER TABLE transaction ADD foreign key (credit_card_id) REFERENCES credit_card(...	0 row(s) affected	Records: 0 Duplicates: 0 Warnings: 0
---	---	----------	---	-------------------	--------------------------------------

Esquema de las relaciones entre tablas.



Comprobación de los índices creados.

```

1 • show indexes from transaction
2 ;
  
```

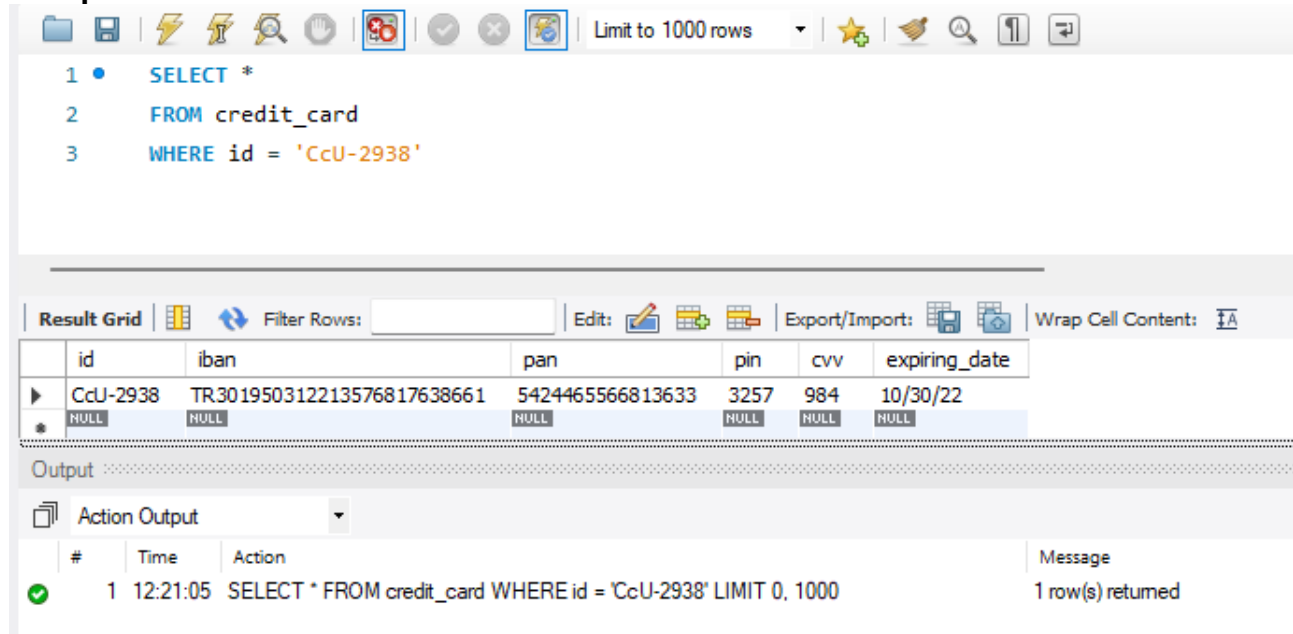
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
transaction	0	PRIMARY	1	id	A	587	NULL	NULL		BTREE
transaction	1	company_id	1	company_id	A	100	NULL	NULL	YES	BTREE
transaction	1	credit_card_id	1	credit_card_id	A	275	NULL	NULL	YES	BTREE

#	Time	Action	Message
✓ 1	11:43:46	show indexes from transaction	3 row(s) returned

NIVEL 1 _ 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 se mostrará para este registro es: R323456313122135717699999. Recuerde demostrar que el cambio se hizo.

Comprobación de los datos actuales del usuario CcU-2938.



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

```
1 • SELECT *
2 FROM credit_card
3 WHERE id = 'CcU-2938'
```

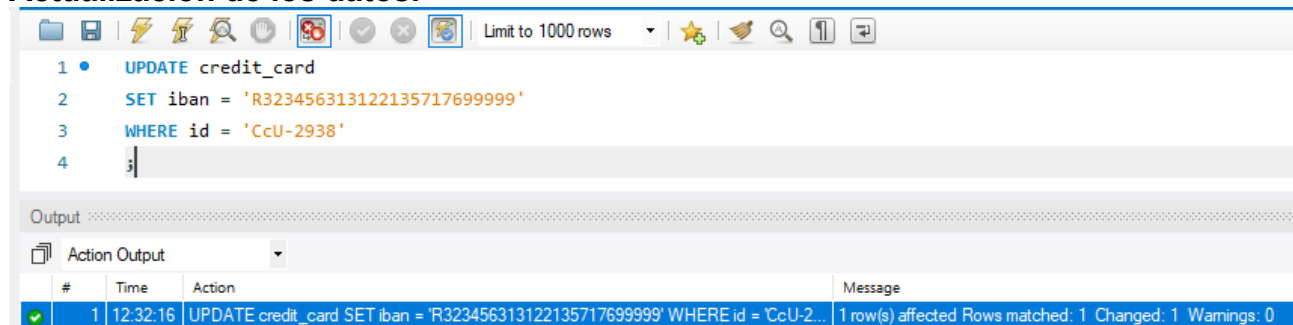
The results are displayed in a table with the following columns: id, iban, pan, pin, cvv, and expiring_date. The first row shows the data for the user CcU-2938.

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

The Output section shows the Action Output for the query:

#	Time	Action	Message
1	12:21:05	SELECT * FROM credit_card WHERE id = 'CcU-2938' LIMIT 0, 1000	1 row(s) returned

Actualización de los datos.



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

```
1 • UPDATE credit_card
2 SET iban = 'R323456313122135717699999'
3 WHERE id = 'CcU-2938'
```

The Output section shows the Action Output for the query:

#	Time	Action	Message
1	12:32:16	UPDATE credit_card SET iban = 'R323456313122135717699999' WHERE id = 'CcU-2938'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0

Comprobación del registro actualizado.

Limit to 1000 rows

```
1 SELECT *
2 FROM credit_card
3 WHERE id = 'CcU-2938'
4 ;
```

Result Grid

	id	iban	pan	pin	cvv	expiring_date
▶	CcU-2938	R323456313122135717699999	5424465566813633	3257	984	10/30/22
*	NULL	NULL	NULL	NULL	NULL	NULL

Output

Action Output

#	Time	Action	Message
✓ 1	12:38:56	SELECT * FROM credit_card WHERE id = 'CcU-2938' LIMIT 0, 1000	1 row(s) returned

NIVEL 1 _ EJERCICIO 3

En la tabla de "transacción", un nuevo usuario entra 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

Comprobación del registro con id 108B1D1D-5B23-A76C-55EF-C568E49A99DD.
Se comprueba que no existe un registro con este id.

The screenshot shows a database management interface. At the top, there's a toolbar with various icons. Below it, a SQL query is entered in a text area:

```
1  -- comprobación registro en tabla transaction
2  -- con id = 108B1D1D-5B23-A76C-55EF-C568E49A99DD
3  •  SELECT *
4  FROM transaction
5  WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD'
```

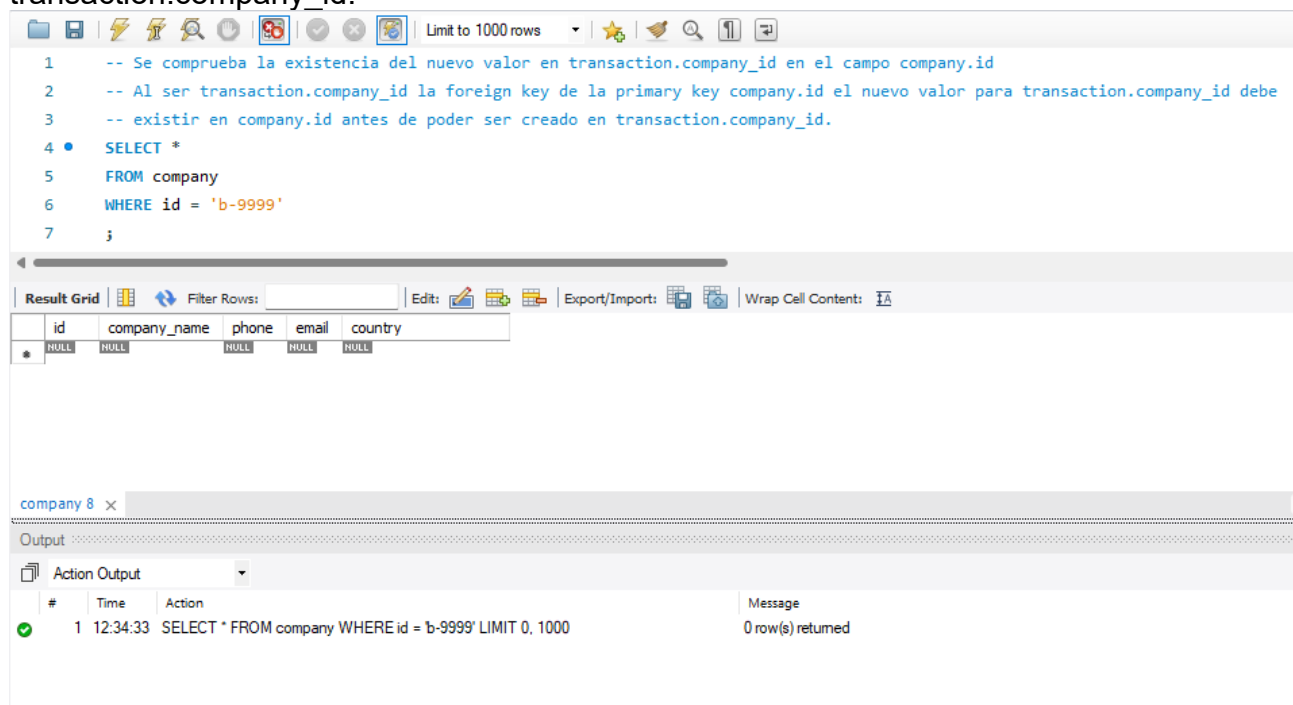
Below the query, there's a "Result Grid" section. It shows a table with 10 columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The first row shows all NULL values.

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

Below the result grid, there's a "transaction 4" tab. Underneath it, there's an "Output" section. It shows a table with 4 columns: #, Time, Action, and Message. The first row shows a successful query execution.

#	Time	Action	Message
1	13:08:55	SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD'	0 row(s) returned

Comprobación del nuevo valor de transaction.company_id en la tabla company.
Al ser transaction.company_id la foreign key del valor company.id el nuevo valor para transaction.company_id debe existir en company.id antes de ser creado en transaction.company_id.



The screenshot shows a database IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a "Limit to 1000 rows" dropdown. The SQL editor contains the following code:

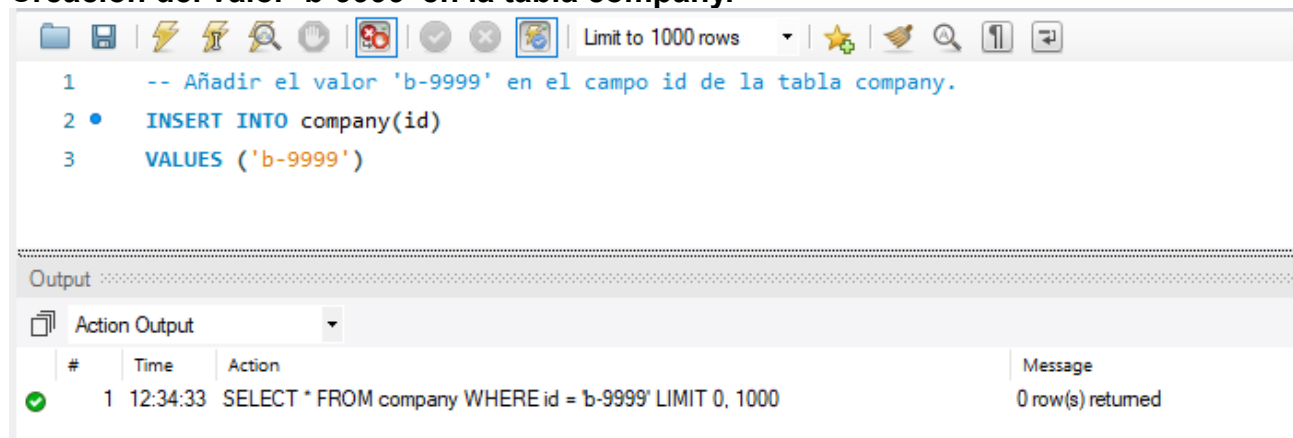
```
1 -- Se comprueba la existencia del nuevo valor en transaction.company_id en el campo company.id
2 -- Al ser transaction.company_id la foreign key de la primary key company.id el nuevo valor para transaction.company_id debe
3 -- existir en company.id antes de poder ser creado en transaction.company_id.
4 • SELECT *
5 FROM company
6 WHERE id = 'b-9999'
7 ;
```

Below the editor is a "Result Grid" section with a "Filter Rows:" input and buttons for "Edit", "Export/Import", and "Wrap Cell Content". A table with 5 columns (id, company_name, phone, email, country) is shown, with all cells containing "NULL".

At the bottom, the "Output" pane shows the "Action Output" for the executed query:

#	Time	Action	Message
1	12:34:33	SELECT * FROM company WHERE id = 'b-9999' LIMIT 0, 1000	0 row(s) returned

Creación del valor 'b-9999' en la tabla company.



The screenshot shows the same database IDE interface. The SQL editor contains the following code:

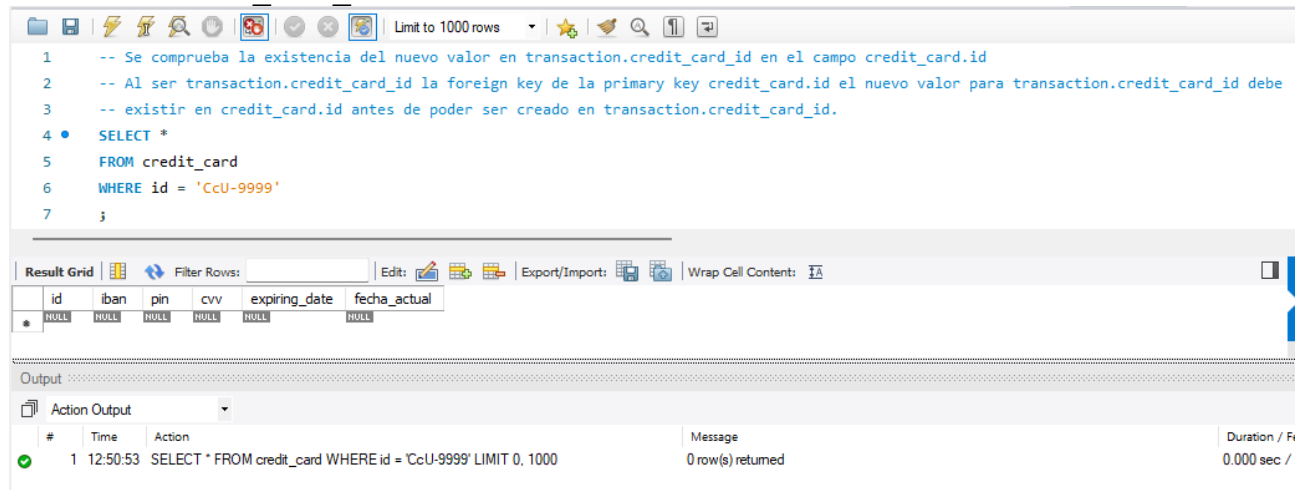
```
1 -- Añadir el valor 'b-9999' en el campo id de la tabla company.
2 • INSERT INTO company(id)
3 VALUES ('b-9999')
```

The "Output" pane shows the "Action Output" for the executed query:

#	Time	Action	Message
1	12:34:33	SELECT * FROM company WHERE id = 'b-9999' LIMIT 0, 1000	0 row(s) returned

Comprobación del nuevo valor de transaction.credit_card_id en la tabla credit_card.id

Al ser transaction.credit_card_id la foreign key del valor credit_card.id el nuevo valor para transaction.credit_card_id debe existir en credit_card.id antes de ser creado en transaction.credit_card_id.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search. The SQL editor contains the following query:

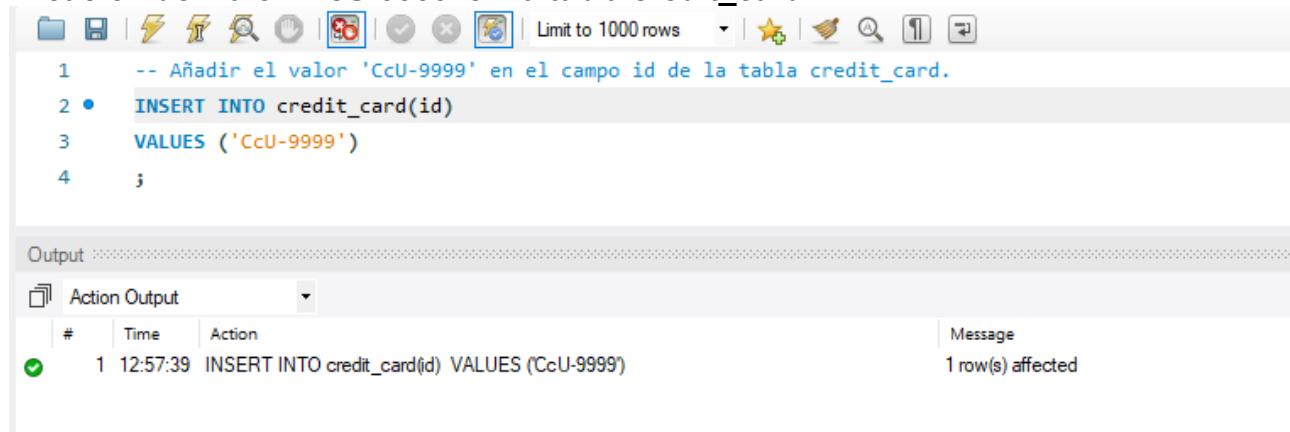
```
1 -- Se comprueba la existencia del nuevo valor en transaction.credit_card_id en el campo credit_card.id
2 -- Al ser transaction.credit_card_id la foreign key de la primary key credit_card.id el nuevo valor para transaction.credit_card_id debe
3 -- existir en credit_card.id antes de poder ser creado en transaction.credit_card_id.
4 • SELECT *
5 FROM credit_card
6 WHERE id = 'CcU-9999'
7 ;
```

Below the editor is the 'Result Grid' section, which is currently empty, showing columns: id, iban, pin, cvv, expiring_date, fecha_actual.

The 'Output' section at the bottom shows the 'Action Output' table:

#	Time	Action	Message	Duration / F
✓ 1	12:50:53	SELECT * FROM credit_card WHERE id = 'CcU-9999' LIMIT 0, 1000	0 row(s) returned	0.000 sec /

Creación del valor 'CcU-9999' en la tabla credit_card.



The screenshot shows the same SQL IDE interface. The SQL editor contains the following query:

```
1 -- Añadir el valor 'CcU-9999' en el campo id de la tabla credit_card.
2 • INSERT INTO credit_card(id)
3 VALUES ('CcU-9999')
4 ;
```

The 'Output' section at the bottom shows the 'Action Output' table:

#	Time	Action	Message
✓ 1	12:57:39	INSERT INTO credit_card(id) VALUES ('CcU-9999')	1 row(s) affected

Añadir la nueva fila de datos.

```
6 • INSERT INTO transaction
7   (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
8   VALUES
9   ( '108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999'
10  , '9999', '829.999', '-117.999', '111.11', '0' )
11  ;
```

Output			
Action Output			
#	Time	Action	Message
✓ 1	13:17:40	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amoun...	1 row(s) affected

Comprobación de la actualización.

Se ha añadido la nueva fila correctamente.

```
-- comprobación registro en tabla transaction
-- con id = 108B1D1D-5B23-A76C-55EF-C568E49A99DD
3 • SELECT *
4 FROM transaction
5 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

Output			
Action Output			
#	Time	Action	Message
✓ 1	13:20:43	SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99...	1 row(s) returned

NIVEL 1 _ EJERCICIO 4

De recursos humanos, se le pide que retire la columna "pan" de la tabla de crédito. Recuerda mostrar el cambio hecho.

Comprobación de las columnas existentes en la tabla credit_card.

The screenshot shows a database management tool interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1  -- comprobacion de las columnas de la tabla credit_card
2  •  SHOW COLUMNS
3    FROM credit_card
4    ;
```

Below the query, a "Result Grid" is displayed, showing the columns of the credit_card table:

Field	Type	Null	Key	Default	Extra
id	varchar(255)	NO	PRI	NULL	
iban	varchar(255)	NO		NULL	
pan	varchar(255)	NO	UNI	NULL	
pin	varchar(255)	NO		NULL	
cvv	varchar(255)	NO		NULL	
expiring_date	varchar(255)	NO		NULL	

Below the result grid, there is a "Result 3" tab and an "Output" section. The "Output" section shows the execution of the query:

Output

Action Output

#	Time	Action	Message
1	13:32:09	SHOW COLUMNS FROM credit_card	6 row(s) returned

Eliminación de la columna pan.

The screenshot shows the same database management tool interface. The SQL query is now:

```
1  -- Eliminación de la columna pan
2  -- en la tabla credit_card.
3  •  ALTER TABLE credit_card
4    DROP COLUMN pan
5    ;
```

Below the query, the "Output" section shows the execution of the query:

Output

Action Output

#	Time	Action	Message
1	13:37:39	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Comprobación de las columnas existentes despues de la actualización.

La columna pan se ha eliminado correctamente.

The screenshot shows a database management interface with a SQL editor at the top and a results panel below. The SQL editor contains the following code:

```
1  -- comprobacion de las columnas de la tabla credit_card
2  •  SHOW COLUMNS
3  FROM credit_card
4  ;
```

The results panel displays the output of the SQL command in a table format. The table has the following columns: Field, Type, Null, Key, Default, and Extra. The data is as follows:

Field	Type	Null	Key	Default	Extra
id	varchar(255)	NO	PRI	NULL	
iban	varchar(255)	NO		NULL	
pin	varchar(255)	NO		NULL	
cvv	varchar(255)	NO		NULL	
expiring_date	varchar(255)	NO		NULL	

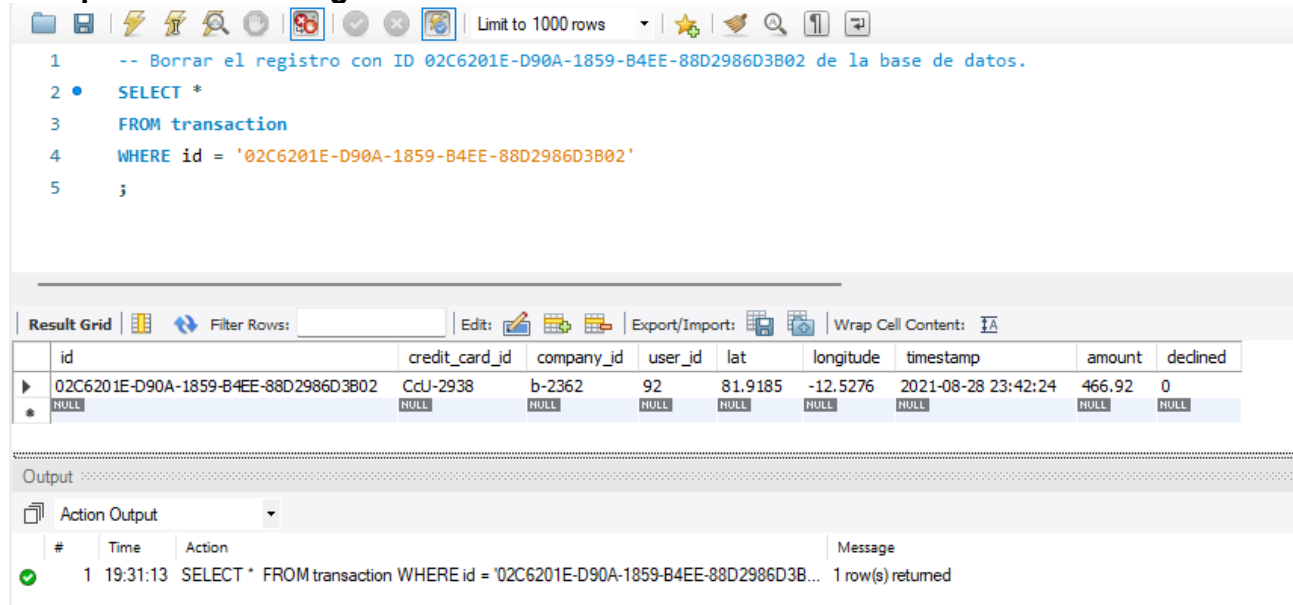
Below the table, the output of the SQL command is shown in a log format:

```
Result 4 x
Output :
Action Output
# Time Action Message
1 13:40:00 SHOW COLUMNS FROM credit_card 5 row(s) returned
```

NIVEL 2 _ EJERCICIO 1

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

Comprobación del registro a eliminar.



The screenshot shows a database management tool interface. At the top, there is a toolbar with various icons. Below the toolbar, a SQL query is entered in a text area:

```
1 -- Borrar el registro con ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de datos.
2 • SELECT *
3 FROM transaction
4 WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'
5 ;
```

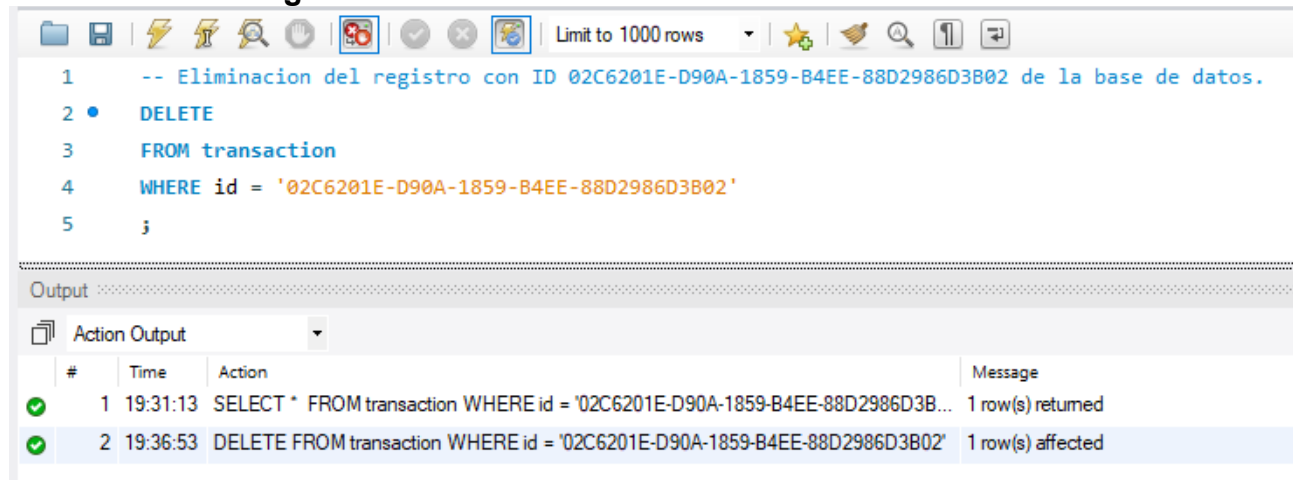
Below the query, there is a "Result Grid" section. It contains a table with the following data:

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

Below the result grid, there is an "Output" section. It shows the execution of the query:

#	Time	Action	Message
✓ 1	19:31:13	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B...	1 row(s) returned

Eliminación del registro.



The screenshot shows the same database management tool interface. The SQL query in the text area is now a DELETE statement:

```
1 -- Eliminación del registro con ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de datos.
2 • DELETE
3 FROM transaction
4 WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'
5 ;
```

Below the query, the "Output" section shows the execution of the query:

#	Time	Action	Message
✓ 1	19:31:13	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B...	1 row(s) returned
✓ 2	19:36:53	DELETE FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	1 row(s) affected

El registro se ha eliminado correctamente.

El registro se ha eliminado correctamente.

1 -- Comprobacion del registro con ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de datos.
 2 • SELECT *
 3 FROM transaction
 4 WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'
 5 ;

Result Grid

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

Output

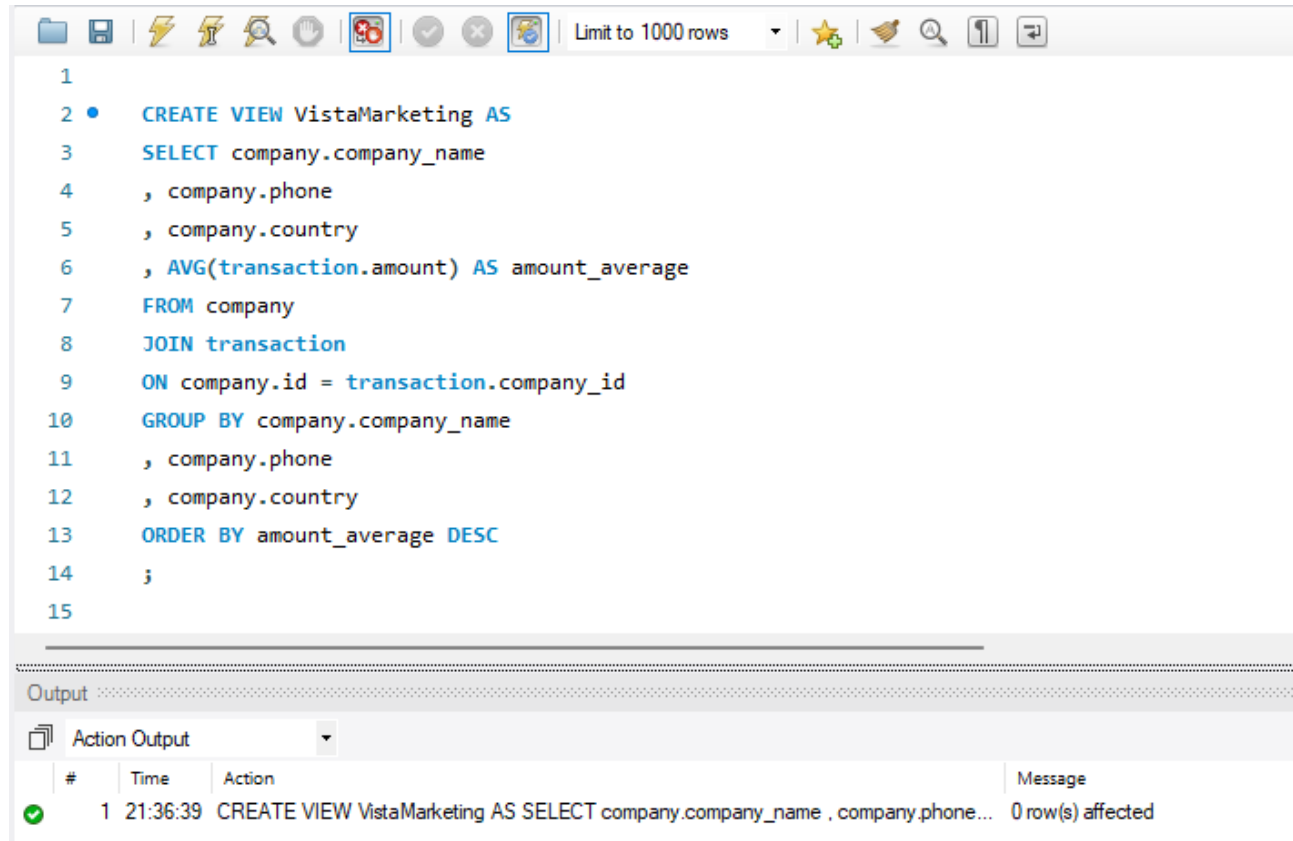
Action Output

#	Time	Action	Message
✓ 1	20:18:38	SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	0 row(s) returned

NIVEL 2 _ EJERCICIO 2

La sección de marketing quiere tener acceso a información específica para llevar a cabo análisis y estrategias eficaces. Se le ha pedido que cree una opinión que proporcione detalles clave sobre las empresas y sus transacciones. Usted tendrá que crear una vista llamada VistaMarketing que contiene la siguiente información: Nombre de la empresa. Número de teléfono de contacto. País de residencia. Compra media realizada por cada empresa. Presenta la imagen creada, ordenando los datos de mayor a menor de la compra media.

Creacion vista.




```
1
2 • CREATE VIEW VistaMarketing AS
3 SELECT company.company_name
4      , company.phone
5      , company.country
6      , AVG(transaction.amount) AS amount_average
7 FROM company
8 JOIN transaction
9 ON company.id = transaction.company_id
10 GROUP BY company.company_name
11      , company.phone
12      , company.country
13 ORDER BY amount_average DESC
14 ;
15
```


Output

#	Time	Action	Message
✓ 1	21:36:39	CREATE VIEW VistaMarketing AS SELECT company.company_name , company.phone...	0 row(s) affected

Presentación vista VistaMarketing.




Limit to 1000 rows





```
1 SELECT *
2 FROM VistaMarketing
3 ;
```

Result Grid



Filter Rows:


Export: 

Wrap Cell Content: 

	company_name	phone	country	amount_average
▶	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
	Viverra Donec Foundation	03 33 12 32 73	United Kingdom	442.280000
	Vestibulum Lorem PC	02 02 87 33 40	Belgium	434.060000

VistaMarketing 3 ×

Output

 Action Output

#	Time	Action	Message
✓ 1	10:21:41	SELECT * FROM VistaMarketing LIMIT 0, 1000	100 row(s) returned

NIVEL 2 _ EJERCICIO 3

Filtra la vista VistaMarketing mostrará únicamente las empresas que tienen su país de residencia en "Germany".

Filtro de VistaMarketing

The screenshot shows a database management tool interface. At the top, there's a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1 -- Filtra la vista VistaMarketing mostrará únicamente las empresas que tienen su país de residencia en "Germany".
2 • SELECT *
3 FROM transactions.vistamarketing
4 WHERE country = 'Germany'
5 ;
```

Below the query editor, there's a "Result Grid" section. It includes a "Filter Rows:" input field, an "Export:" button, and a "Wrap Cell Content:" checkbox. The grid displays the following data:

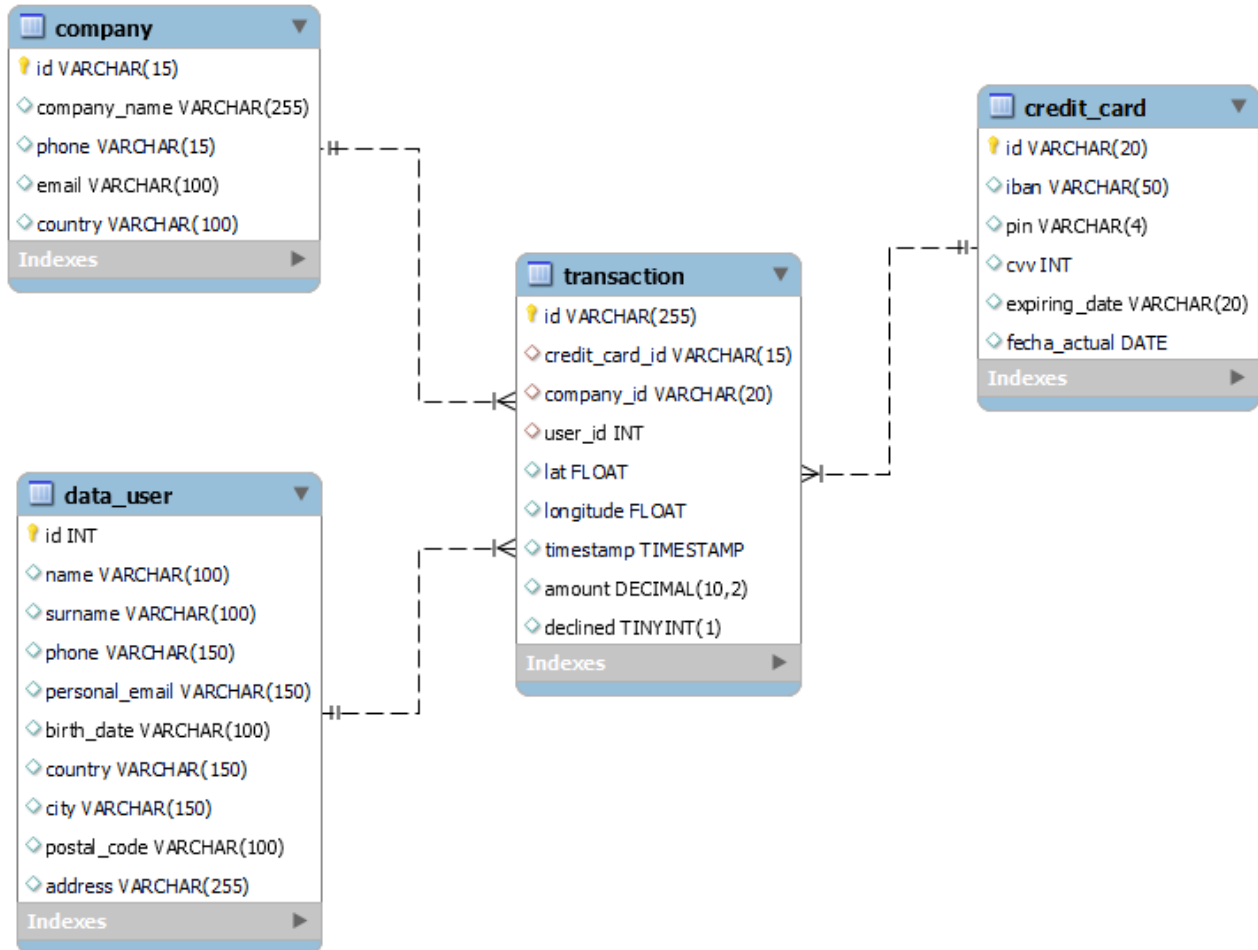
company_name	phone	country	amount_average
Aliquam PC	01 45 73 52 16	Germany	385.265000
Ac Industries	09 34 65 40 60	Germany	289.645000
Rutrum Non Inc.	02 66 31 61 09	Germany	266.900000
Nunc Interdum Incorporated	05 18 15 48 13	Germany	244.025238
Augue Foundation	06 88 43 15 63	Germany	240.800000
Ac Fermentum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

Below the result grid, there's an "Output" section. It includes a tab labeled "Action Output" and a table showing the execution details:

#	Time	Action	Message
1	21:51:31	SELECT * FROM transactions.vistamarketing WHERE country = 'Germany' LIMIT 0, 10...	8 row(s) returned

NIVEL 3 _ EJERCICIO 1

La próxima semana tendrá una nueva reunión con los gerentes de marketing. Un miembro del equipo hizo cambios en la base de datos, pero no recuerda cómo los hizo. Pídele que le ayude a dejar ejecutada los comandos para obtener el siguiente diagrama:



Se comprueban los datos de cada tabla existente y se modifican para adaptarlas al esquema.

Se crea la tabla "data_user", se comprueban los datos y se crean las relaciones correspondientes para adaptarla al esquema.

Comprobación de la tabla company.

Se realiza una comprobación de los datos de la tabla "company".

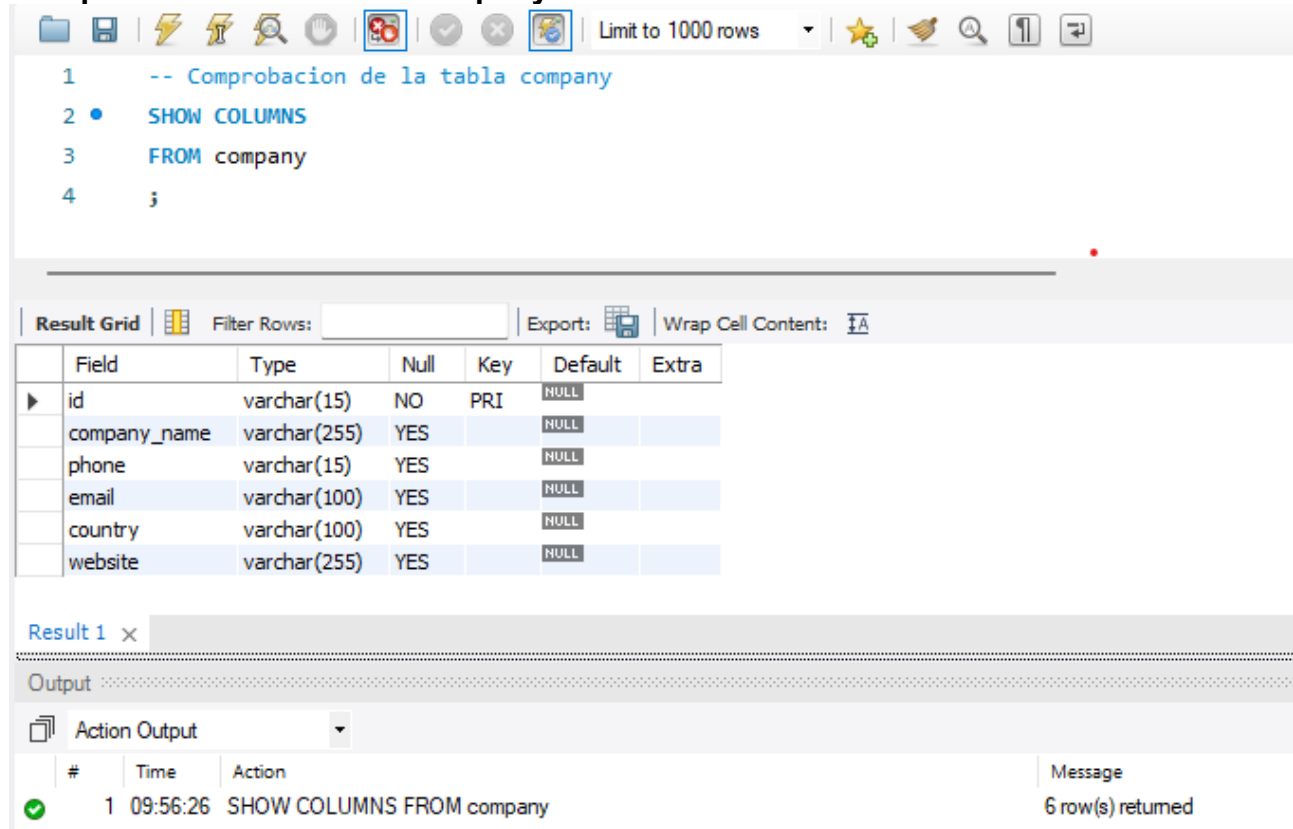
Los datatype de la tabla original son iguales a los de la tabla del esquema por lo que no es necesaria ninguna modificación de los datatype.

La tabla original contiene 6 columnas y la tabla del esquema solo 5.

Se borra la columna "website" de la tabla company para adaptarla a la tabla del esquema.

Se comprueban los datos de la tabla "company" despues de la actualización.

Comprobación de la tabla company.



The screenshot shows a database client interface with a SQL editor and a results pane. The SQL editor contains the following code:

```
1  -- Comprobacion de la tabla company
2  •  SHOW COLUMNS
3  FROM company
4  ;
```

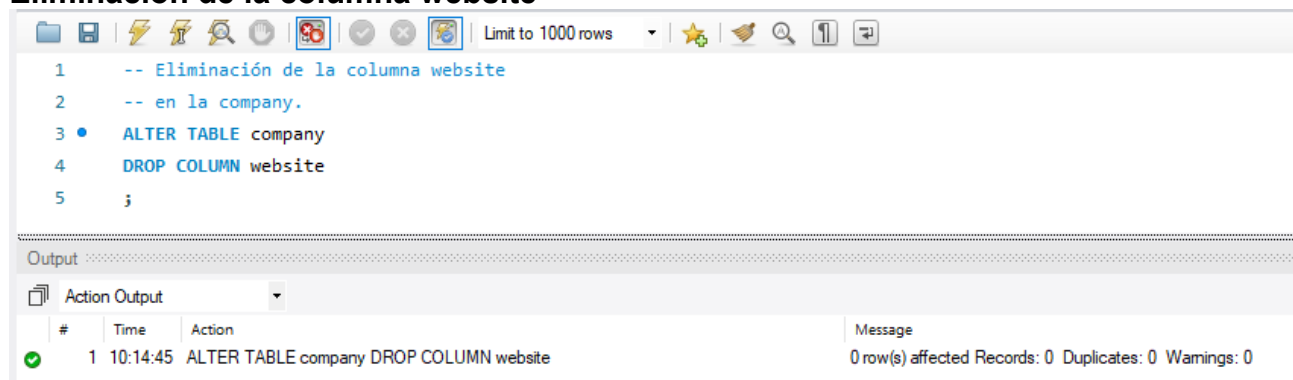
The results pane displays a table with the following columns: Field, Type, Null, Key, Default, and Extra. The data is as follows:

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

Below the table, the 'Output' section shows the execution details:

#	Time	Action	Message
1	09:56:26	SHOW COLUMNS FROM company	6 row(s) returned

Eliminación de la columna website



The screenshot shows a database client interface with a SQL editor and a results pane. The SQL editor contains the following code:

```
1  -- Eliminación de la columna website
2  -- en la company.
3  •  ALTER TABLE company
4  DROP COLUMN website
5  ;
```

The results pane displays the execution details:

#	Time	Action	Message
1	10:14:45	ALTER TABLE company DROP COLUMN website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Comprobación de la tabla company despues de la actualización.

Se comprueba que tanto el número de columnas como los datatype corresponden con los del esquema.

The screenshot shows a database management tool interface. At the top, there is a toolbar with various icons and a text input field containing the SQL command: `-- Comprobacion de la tabla company despues de la actualizacion.`. Below the toolbar, the command is executed, and the results are displayed in a table. The table has columns: Field, Type, Null, Key, Default, and Extra. The data rows are:

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

Below the table, there is an 'Output' section with a dropdown menu set to 'Action Output'. The output shows a single row with a green checkmark, indicating successful execution. The message is: '5 row(s) returned'.

Comprobación de la tabla credit_card.

Se realiza una comprobación de la tabla "credit_card".

Los datatype de la tabla original no corresponden con el esquema.

La tabla original contiene 5 columnas y la tabla del esquema 6.

Se modifican los datatype de la tabla para adaptarla al esquema.

Se añade la columna "fecha_actual" a la tabla para adaptarla al esquema.

Comprobación de la tabla credit_card.

Limit to 1000 rows

```
1 -- Comprobacion de la tabla credit_card.
2 • SHOW COLUMNS
3 FROM credit_card
4 ;
5
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	Field	Type	Null	Key	Default	Extra
▶	id	varchar(255)	NO	PRI	NULL	
	iban	varchar(255)	NO		NULL	
	pin	varchar(255)	NO		NULL	
	cvv	varchar(255)	NO		NULL	
	expiring_date	varchar(255)	NO		NULL	

Result 1 x

Output

Action Output

#	Time	Action	Message
✓ 1	11:00:46	SHOW COLUMNS FROM credit_card	5 row(s) returned

Modificación de los data_type y creación de la columna fecha_actual en la tabla credit_card.

Limit to 1000 rows

```
1 -- modificacion de los datatype y creacion de la columna fecha_actual.
2 • ALTER TABLE credit_card
3 MODIFY id VARCHAR(20),
4 MODIFY iban VARCHAR(50),
5 MODIFY pin VARCHAR(4),
6 MODIFY cvv INT,
7 MODIFY expiring_date VARCHAR(20),
8 ADD fecha_actual DATE
9 ;
```

Output

Action Output

#	Time	Action	Message
✓ 1	11:59:37	ALTER TABLE credit_card MODIFY id VARCHAR(20), MODIFY iban VARCHAR(50), M...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

Comprobación de la tabla credit_card despues de la actualización.

The screenshot shows a database management tool interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1  -- Comprobacion de la tabla credit_card despues de la actualizacion.
2  • SHOW COLUMNS
3  FROM credit_card
4  ;
5
```

Below the query, a "Result Grid" is displayed, showing the structure of the credit_card table. The grid has columns for Field, Type, Null, Key, Default, and Extra. The data is as follows:

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	

Below the result grid, there is a section for "Result 6" and an "Output" pane. The "Output" pane shows the execution of the SQL command "SHOW COLUMNS FROM credit_card" at 13:25:19, which returned 6 rows.

#	Time	Action	Message
1	13:25:19	SHOW COLUMNS FROM credit_card	6 row(s) returned

Creación de la tabla data_user.

Creación de la tabla user e introducción de los datos.

Comprobación de los datos.

El nombre de la tabla es 'user'. Se modifica el nombre de la tabla a 'data_user' para adaptarla al esquema.

Se modifica el nombre de la columna 'mail' a 'personal_mail' para adaptarla al esquema.

Creación de la tabla user

```
1      -- Creamos la tabla user
2
3  •   CREATE INDEX idx_user_id ON transaction(user_id);
4
5  •   CREATE TABLE IF NOT EXISTS user (
6      id INT PRIMARY KEY,
7      name VARCHAR(100),
8      surname VARCHAR(100),
9      phone VARCHAR(150),
10     email VARCHAR(150),
11     birth_date VARCHAR(100),
12     country VARCHAR(150),
13     city VARCHAR(150),
14     postal_code VARCHAR(100),
15     address VARCHAR(255),
16     FOREIGN KEY(id) REFERENCES transaction(user_id)
17 );
18
```

Insertar datos en la tabla user.

```
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 (
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 (
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 273	19:34:36	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	1 row(s) affected	0.000 sec
✓ 274	19:34:36	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	1 row(s) affected	0.000 sec
✓ 275	19:34:36	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	1 row(s) affected	0.000 sec
✓ 276	19:34:36	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)	1 row(s) affected	0.000 sec
✓ 277	19:34:36	SET foreign_key_checks = 1	0 row(s) affected	0.000 sec

Comprobación de las tablas de la base de datos transactions.

La tabla 'user' no corresponde al esquema.

Se modifica el nombre de la tabla a 'data_user'.

Comprobación de las tablas.

The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1  -- Comprobacion de las tablas
2  • SHOW TABLES
3  FROM transactions
4  ;
```

Below the query editor, the 'Result Grid' shows the output of the query:

Tables_in_transactions
company
credit_card
transaction
user
vistamarketing

The 'Output' pane shows the execution log:

#	Time	Action	Message
✓ 1	20:35:09	SHOW TABLES FROM transactions	5 row(s) returned

Modificación del nombre de la tabla user a data_user.

The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1  -- modificacion de la tabla user a data_user.
2  • ALTER TABLE user
3  RENAME TO data_user
4  ;
```

Below the query editor, the 'Output' pane shows the execution log:

#	Time	Action	Message
✓ 1	20:42:41	ALTER TABLE user RENAME TO data_user	0 row(s) affected

Comprobación de las tablas despues de la actualización.

The screenshot shows a database management tool interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1 -- Comprobacion de las tablas despues de la actualizacion.  
2 • SHOW TABLES  
3 FROM transactions  
4 ;
```

Below the query, there is a "Result Grid" section. It shows a table with one row and one column:

Tables_in_transactions
company
credit_card
data_user
transaction
vistamarketing

Below the result grid, there is a "Result 5" tab. It shows the "Output" section, which displays the "Action Output" for the query:

#	Time	Action	Message
1	20:46:59	SHOW TABLES FROM transactions	5 row(s) returned

Comprobación de los datos de la tabla data_user.

La columna 'email' no se corresponde con el esquema.

Se modifica el nombre de la columna 'email' a 'personal_email'.

The screenshot shows a database management tool interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1 • SHOW COLUMNS  
2 FROM data_user  
3 ;
```

Below the query, there is a "Result Grid" section. It shows a table with 10 rows and 7 columns:

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

Below the result grid, there is a "Result 2" tab. It shows the "Output" section, which displays the "Action Output" for the query:

#	Time	Action	Message
1	20:50:09	SHOW COLUMNS FROM data_user	10 row(s) returned

Modificar el nombre de la columna email a personal_email.

Limit to 1000 rows

```
1 -- Modificacion del nombre de la columna 'email' a 'personal_email'
2 • ALTER TABLE data_user
3   RENAME COLUMN email TO personal_email
4 ;
5
```

Output

Action Output

#	Time	Action	Message
✓ 1	21:45:45	ALTER TABLE data_user RENAME COLUMN email TO personal_email	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Comprobación de los datos de la tabla data_user despues de la actualización.

Limit to 1000 rows

```
1 -- Comprobacion de los datos de la tabla data_user.
2 • SHOW COLUMNS
3   FROM data_user
4 ;
```

Result Grid

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

Result 4 ×

Output

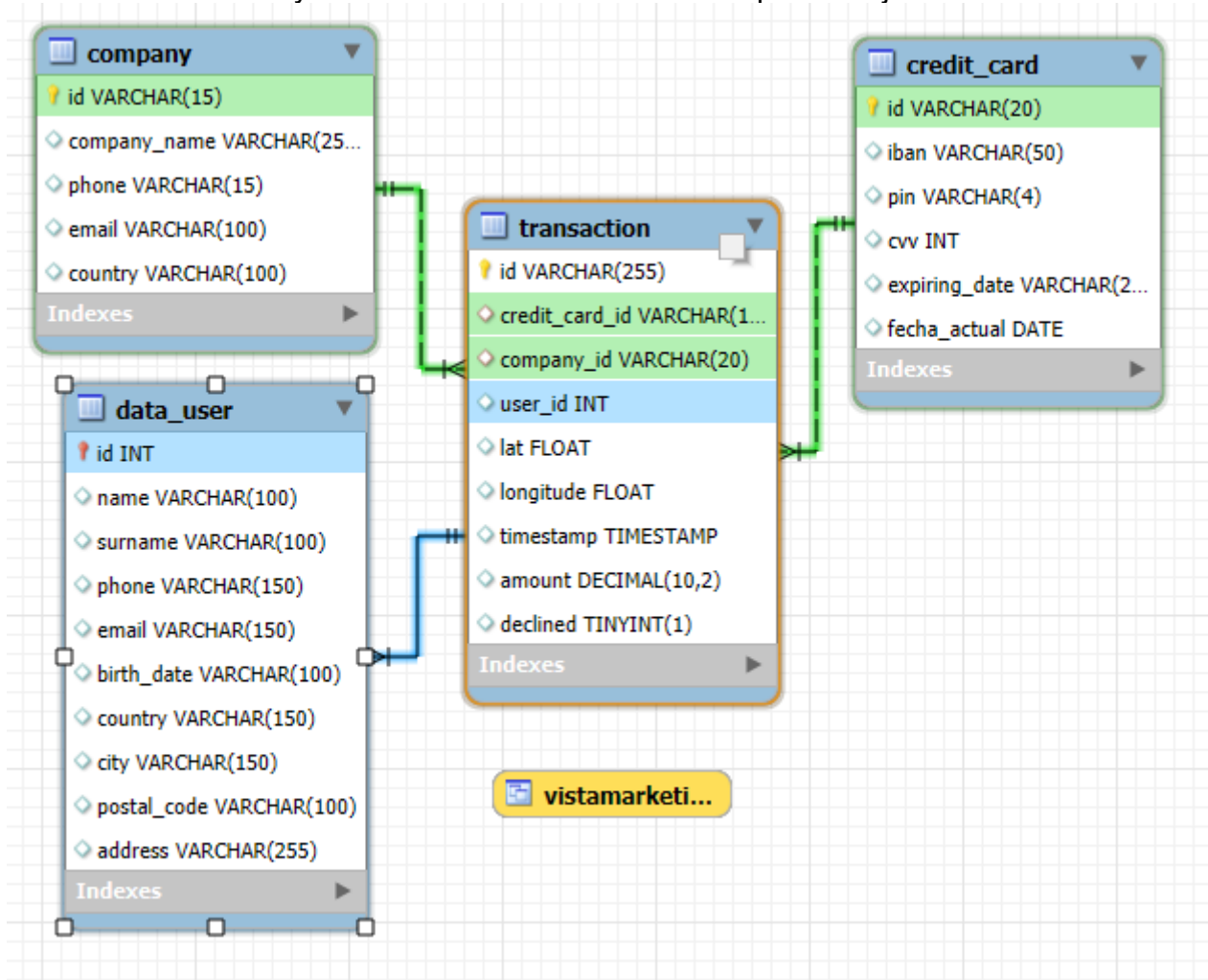
Action Output

#	Time	Action	Message
✓ 1	21:48:33	SHOW COLUMNS FROM data_user	10 row(s) returned

Comprobación del esquema de la base de datos transactions.

La relación entre las tablas data_user y transaction no se corresponde con las del esquema objetivo.

Se elimina la relación y se crea otra nueva acorde al esquema objetivo.



Eliminación de la relación entre las tablas data_user y transaction.

```
1  -- Eliminacion de la relacion data_user transaction
2  • SHOW CREATE TABLE data_user
3  ;
4  • ALTER TABLE data_user
5  DROP FOREIGN KEY data_user_ibfk_1
6  ;
7
```

Output

#	Time	Action	Message
1	22:13:41	ALTER TABLE data_user DROP FOREIGN KEY data_user_ibfk_1	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Creacion de la relación entre las tablas data_user y transaction.

No se puede crear directamente una la relación entre las tablas ya que hay un error de integridad referencial dado que en el ejercicio 3 del Nivel 1 se introdujeron datos de una transacción que incluía el dato transaction.user_id = '9999' que no existe en data_user.id. Al ser transaction.user_id la foreign key y data_user.id la primary key es necesario crear un id en la tabla data_user con valor '9999' antes de crear la relación.

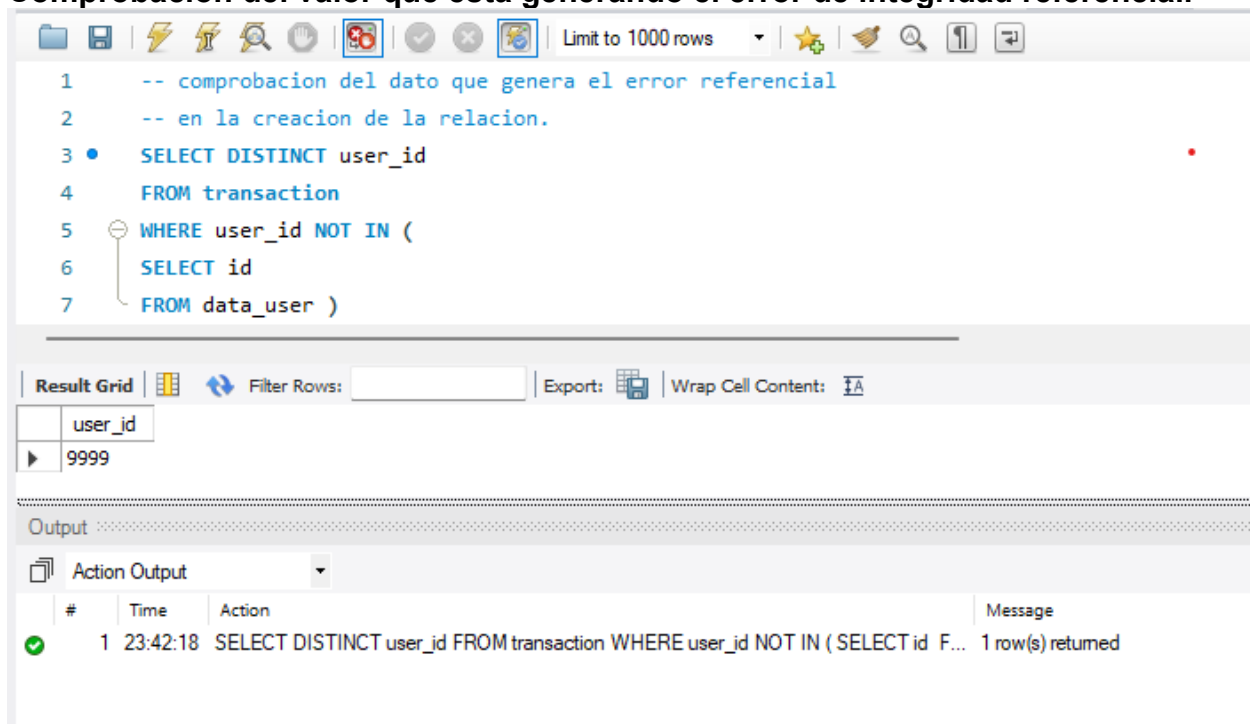
Se crea la relación entre las tablas transaction y data_user.

Esta relación es de uno a muchos.

Un usuario puede realizar varias transacciones y una transacción esta asignada a un único usuario.

Esta relación es entre los campos data_user(id) PK y transaction(user_id) FK.

Comprobación del valor que esta generando el error de integridad referencial.



The screenshot shows a database IDE with a SQL query editor and a result grid. The query is as follows:

```
1  -- comprobacion del dato que genera el error referencial
2  -- en la creacion de la relacion.
3  • SELECT DISTINCT user_id
4    FROM transaction
5    WHERE user_id NOT IN (
6      SELECT id
7      FROM data_user )
```

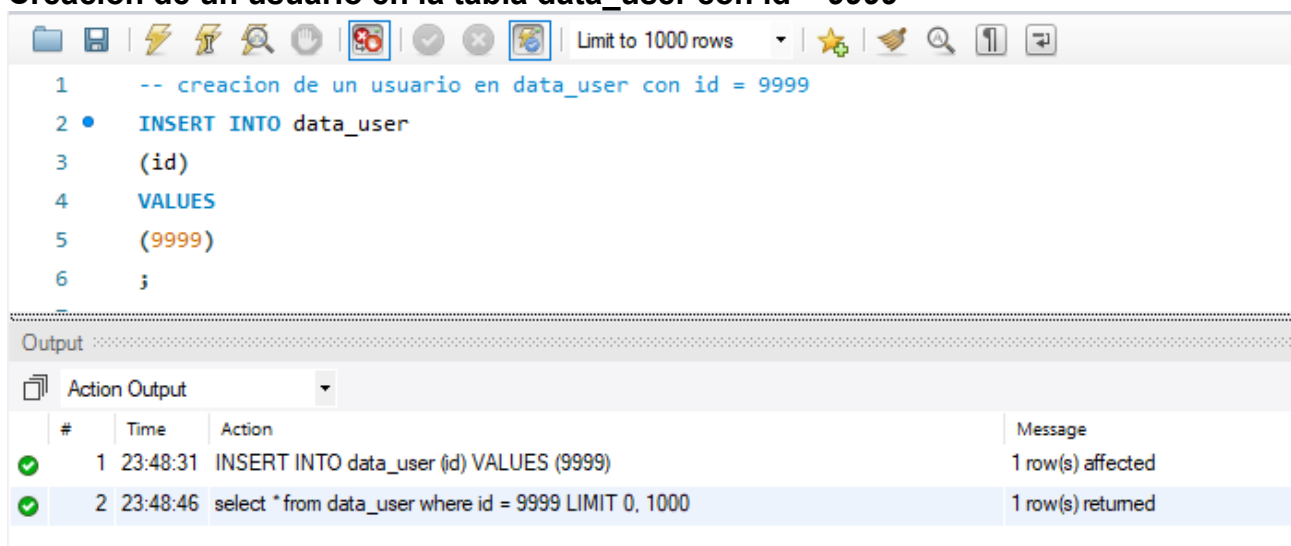
The result grid shows one row with the value 9999.

user_id
9999

The output section shows the following message:

#	Time	Action	Message
✓ 1	23:42:18	SELECT DISTINCT user_id FROM transaction WHERE user_id NOT IN (SELECT id F...	1 row(s) returned

Creación de un usuario en la tabla data_user con id = 9999



The screenshot shows a database IDE with a SQL query editor and an action output section. The query is as follows:

```
1  -- creacion de un usuario en data_user con id = 9999
2  • INSERT INTO data_user
3    (id)
4    VALUES
5    (9999)
6    ;
```

The action output section shows the following messages:

#	Time	Action	Message
✓ 1	23:48:31	INSERT INTO data_user (id) VALUES (9999)	1 row(s) affected
✓ 2	23:48:46	select * from data_user where id = 9999 LIMIT 0, 1000	1 row(s) returned

Creación de la relacion entre las tablas data_user y transaction.

```
1  -- Creacion de la relacion entre las tablas data_user y transaction.
2  • ALTER TABLE transaction
3  ADD foreign key (user_id) REFERENCES data_user(id)
4  ;
```

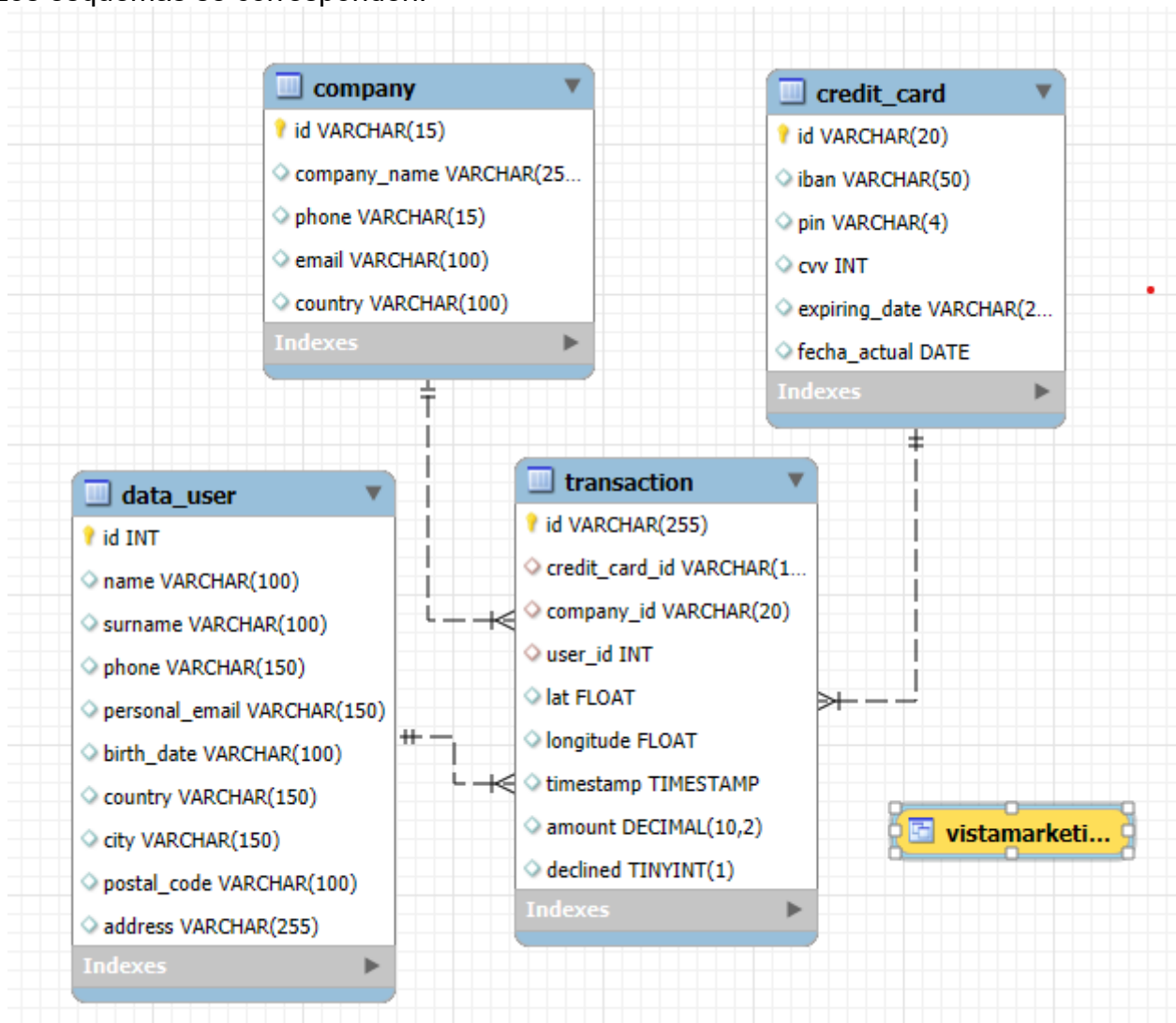
Output

Action Output

#	Time	Action	Message
1	23:51:45	ALTER TABLE transaction ADD foreign key (user_id) REFERENCES data_user(id)	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0

Comprobación del esquema de la base de datos transactions con el esquema objetivo.

Los esquemas se corresponden.



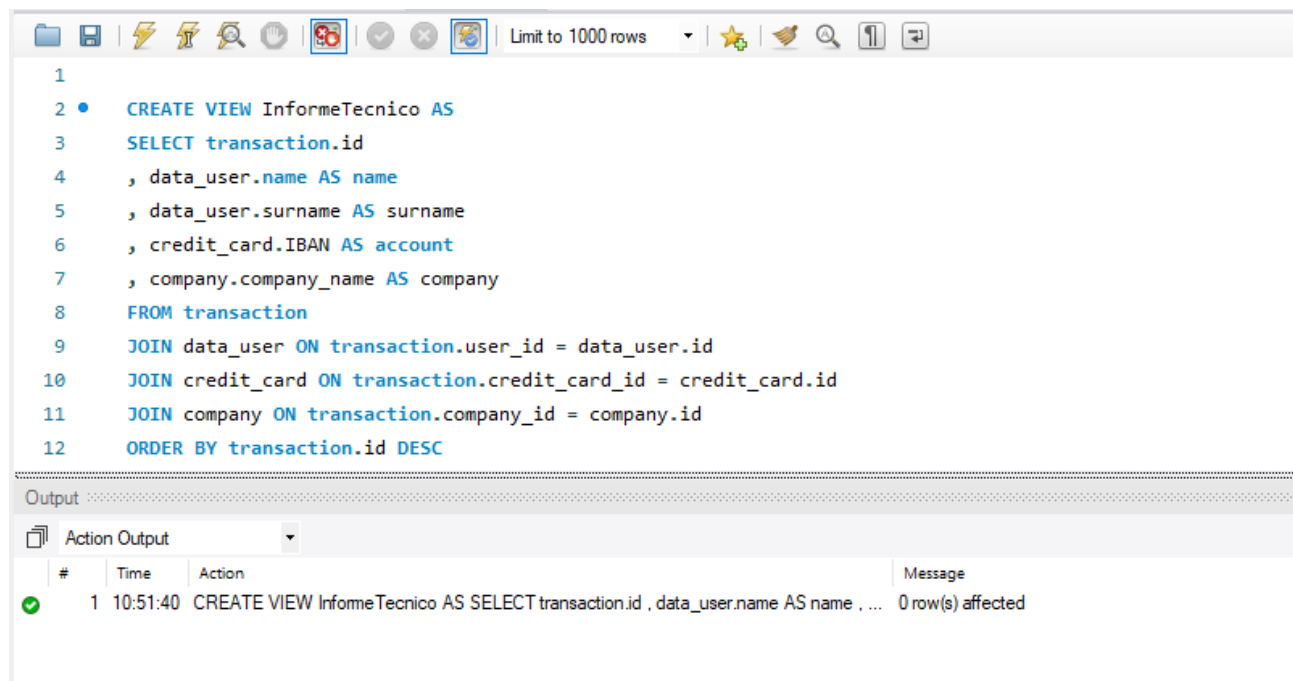
NIVEL 3 _ EJERCICIO 2

La compañía también le pide que cree una vista llamada "InformeTecnico" que contiene 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 empresa de la transacción realizada.
- Asegúrese de incluir información relevante de ambas tablas y utilizar alias para cambiar el nombre de las columnas según sea necesario.

Muestra los resultados de la vista, clasificando los resultados de forma descendente en función de la variable ID de transacción.

Creación de la vista InformeTecnico.



The screenshot shows a database management interface with a SQL editor and an output window. The SQL editor contains the following code:

```
1
2 • CREATE VIEW InformeTecnico AS
3   SELECT transaction.id
4     , data_user.name AS name
5     , data_user.surname AS surname
6     , credit_card.IBAN AS account
7     , company.company_name AS company
8   FROM transaction
9   JOIN data_user ON transaction.user_id = data_user.id
10  JOIN credit_card ON transaction.credit_card_id = credit_card.id
11  JOIN company ON transaction.company_id = company.id
12  ORDER BY transaction.id DESC
```

The output window, titled "Output", shows the execution results:

#	Time	Action	Message
✓ 1	10:51:40	CREATE VIEW InformeTecnico AS SELECT transaction.id , data_user.name AS name , ...	0 row(s) affected

Presentación de la vista InformeTecnico.

Limit to 1000 rows

```

928  -- Presentacion vista InformeTecnico.
929  •  SELECT *
930  FROM InformeTecnico
931  ;
932

```

Result Grid

Filter Rows:

Export:

Wrap Cell Content: [IA](#)

	id	name	surname	account	company
▶	FE96CE47-8D59-381C-4E18-E3CA3D44E8FF	Kenyon	Hartman	DO26854763748537475216568689	Magna A Neque Industries
	FE809ED4-2DB6-55AC-C915-929516E4646B	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
	FD2E8957-414B-BEEC-E9AD-59AA7A8A6290	Hedwig	Gilbert	GE84848451582810541526	Neque Tellus Imperdiet Corp.
	FCE2AB9A-271D-2BDC-9E49-8DD92A373391	Hakeem	Alford	MD1234119525145401270486	Nunc Interdum Incorporated
	FBD7E0D6-BA6B-F5BC-0CA9-EA4B8760100C	Hedwig	Gilbert	MU413233344453432541344788855	Mauris Id Inc.
	EAC7E498-8448-60AA-E802-A76C7E12631C	Glade	Dool	MT051WCE58869200575771634E82812	Acquid

InformeTecnico 1

×

Output

Action Output

#

Time

Action

Message

✓

1

11:00:09

SELECT * FROM InformeTecnico LIMIT 0, 1000

586 row(s) returned

