

Sprint 3: Manipulación de tablas

Santiago Álvarez Salvado

Contenido

Nivel 1..... 3

 Ejercicio 1 3

 Ejercicio 2 4

 Ejercicio 3 5

 Ejercicio 4 7

Nivel 2..... 8

 Ejercicio 1 8

 Ejercicio 2 9

 Ejercicio 3 10

Nivel 3..... 11

 Ejercicio 1 11

 Ejercicio 2 18

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 forma 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_cred". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

Para crear la tabla credit_card se ha utilizado un ejemplo del archivo adjunto datos_introducir_sprint3_credit.sql, la longitud del tipo de dato es estimada en base a los datos que hay que introducir luego, posteriormente se introducen los datos del archivo datos_introducir_sprint3_credit.sql y se define la foreign key para unir las tablas con credit_card_id de transaction e id de credit_card.

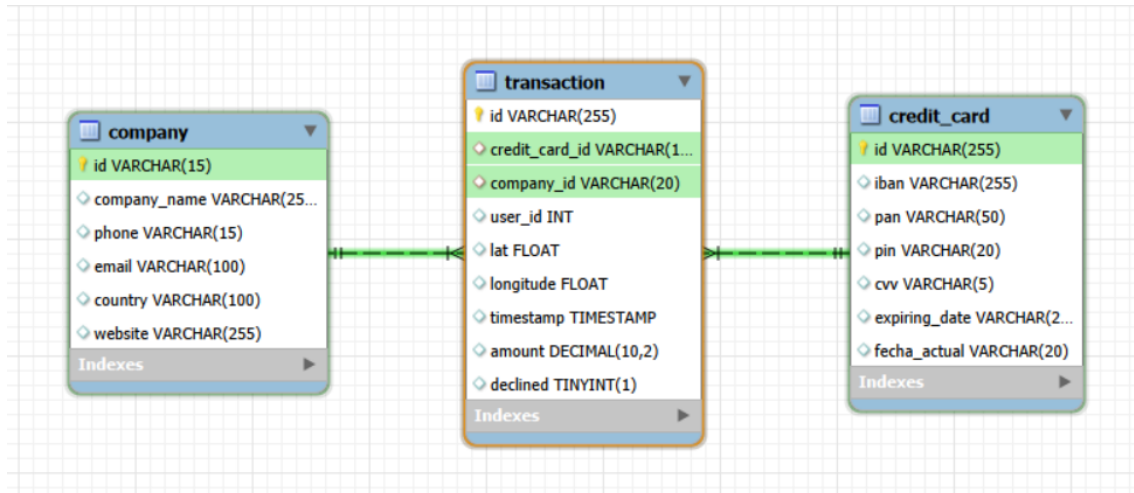
```
7 • create table if not exists credit_card (  
8     id varchar(255) primary key not null,  
9     iban varchar(255),  
10    pan varchar(50),  
11    pin varchar(20),  
12    cvv varchar(5),  
13    expiring_date varchar(20)  
14 );  
  
11 -- Insertamos datos de credit_card  
12 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2938', 'TR301950312213576817638661', '5424465566813633', '3257', '904', '10/30/22');  
13 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2945', 'D026854763748537475216568609', '5142423821948828', '9080', '887', '08/24/23');  
14 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('BG45IVQL52710525608255', '4556 453 55 5287', '4598', '438', '06/29/21');  
15 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2952', 'CR7242477244335841535', '372461377349375', '3583', '667', '02/24/23');  
16 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2966', 'BG72LKTQ15627628377363', '448566 886747 7265', '4900', '130', '10/29/24');  
17 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2973', 'PT87806228135092429456346', '544 58654 54343 384', '8760', '887', '01/30/25');  
18 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2980', 'DE39241881883086277136', '402400 7145845969', '5075', '596', '07/24/22');  
19 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2987', 'GE89681434837748781813', '3763 747687 76666', '2298', '797', '10/31/23');  
20 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-2994', 'BH62714428368066765294', '344283273252593', '7545', '595', '02/28/22');  
21 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3001', 'CY49087426654774581266832110', '511722 924833 2244', '9562', '867', '09/16/22');  
22 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3008', 'LU507216693616119230', '4485744464433884', '1856', '740', '04/05/25');  
23 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3015', 'PS119398216295715968342456821', '3784 662233 17389', '3246', '822', '01/31/22');  
24 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3022', 'GT91695162850556977423121857', '5164 1379 4842 3951', '5610', '342', '04/25/25');  
25 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3029', 'AZ62317413982441418123739746', '3429 279566 77631', '9708', '505', '09/02/23');  
26 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3036', 'AZ39336002925842865843941994', '3768 451556 48766', '2232', '565', '10/27/25');  
27 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3043', 'TW6488143310514852179535', '455676 6437463635', '5909', '196', '06/07/25');  
28 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3050', 'FR5167744369175836831854477', '4024007123722', '4834', '126', '10/09/23');  
29 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3057', 'LU931822574697545215', '3484 621767 21237', '6085', '848', '09/14/25');  
30 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3064', 'PS146965545449253377627273133', '3467 732741 26810', '3865', '498', '06/03/25');  
31 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3071', 'NO8923814763512', '3464 789562 23352', '6625', '661', '12/20/23');  
32 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3078', 'IS025127145884623279548733', '4539 322 74 2377', '9405', '720', '03/08/23');  
33 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3085', 'BE63114723972437', '5266 3346 1135 1687', '7241', '413', '05/10/23');  
34 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-3092', 'RO65L5001166122125447487', '3488 754223 46253', '9417', '594', '12/19/22');
```

Output

#	Time	Action	Message	Duration / Fetch
5067	16:34:11	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-9580', 'XX78125888985195...	1 row(s) affected	0.000 sec
5068	16:34:11	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CCU-9581', 'XX91567051640538...	1 row(s) affected	0.016 sec

```
1 • alter table transaction  
2 add constraint fk_credit_card_id  
3 foreign key (credit_card_id)  
4 references credit_card(id);
```

Para generar el diagrama entidad relación he utilizado la herramienta reverse engineer de workbench; como podemos observar, tenemos las tablas anteriores (transaction y company), y la nueva tabla credit_card, la cual se relaciona con transaction a través de credit_card_id de transaction e id de credit card, la relación entre ambas tablas es 1-N de credit_card a transaction, ya que una tarjeta de crédito puede tener muchas transacciones, pero una transacción solo puede pertenecer a una tarjeta de créditos.



Ejercicio 2

El departamento de Recursos Humanos ha identificado un error en el número de cuenta asociado a su tarjeta de crédito con ID CcU-2938. La información que debe mostrarse para este registro es: TR323456312213576817699999. Recuerda mostrar que el cambio se realizó.

Para este apartado primero se verificamos los datos para asegurar que realmente hay un problema con el numero de cuenta asociado a esa tarjeta, para ello se realiza una selección toda la información de la tabla credit_card y se filtra con el id que esta causando el posible problema, como se puede observar, efectivamente el numero de cuenta no coincide.

```

5028 • select * from credit_card
5029 where id = "CcU-2938";
  
```

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

Para poder solucionar este problema se actualiza la tabla con un update y el nombre de la tabla, y se declara el nuevo valor, para que surta efecto habrá que poner un filtro en el que se especifique a que id se le hará el cambio de número de cuenta, posteriormente al ejecutar la query anterior se podrá observar que ahora el número de cuenta coincide con el que identifico el departamento de recursos humanos.

```

5031 • update credit_card
5032 set iban = "TR323456312213576817699999"
5033 where id = "CcU-2938";
5034
5035 • select * from credit_card
5036 where id = "CcU-2938";

```

	id	iban	pan	pin	cvv	expiring_date	fecha_actual
▶	CcU-2938	TR323456312213576817699999	5424465566813633	3257	984	10/30/22	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Ejercicio 3

En la tabla "transaction" ingresa una nueva transacción con la siguiente información:

Id	108B1D1D-5B23-A76C-55EF-C568E49A99DD
credit_card_id	CcU-9999
company_id	b-9999
user_id	9999
lato	829.999
longitud	-117.999
amunt	111.11
declined	0

HAY ALGUNOS CAMPOS MAL ESCRITOS EN LA TABLA (lato , longitud y amunt, serían lat, longitude y amount) Y FALTA timestamp, además, al hacer esto ocurriría un error como el siguiente: Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`transactions`.`transaction`, CONSTRAINT `transaction_ibfk_1` FOREIGN KEY (`company_id`) REFERENCES `company` (`id`)), el error es causado porque no existen los datos en las demás tablas.

```

5049 • insert into transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) values
5050 ("108B1D1D-5B23-A76C-55EF-C568E49A99DD", "CcU-9999", "b-9999", "9999", "829.999", "-117.999", "111.11", "0");

```

#	Time	Action	Message
5087	17:29:44	insert into transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) values ("108...	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`transactions`.`transaction`...
5088	17:30:50	insert into transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) values ("108...	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`transactions`.`transaction`...

Para solucionar este problema hay que insertar un registro nuevo en credit_card con el id CcU-9999 ya que no existe.

```

5052 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES
5053 ('CcU-9999', 'TR357681763013866195031221', '5424646668135533', '3585', '458', '09/10/23');
5054
5055 • select * from credit_card
5056 where id = "CcU-9999";
5057

```

Result Grid							
Filter Rows: <input type="text"/>							
Edit: Export/Import: Wrap Cell Content:							
	id	iban	pan	pin	cvv	expiring_date	fecha_actual
▶	CcU-9999	TR357681763013866195031221	5424646668135533	3585	458	09/10/23	NULL
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL

credit_card 7 x				
Output				
Action Output				
#	Time	Action	Message	
✓ 12	10:40:20	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-9999', 'TR3576817630138...	1 row(s) affected	
✓ 13	10:42:23	select * from credit_card where id = "CcU-9999"	1 row(s) returned	

También hay que agregar un nuevo registro en Company con el id b-9999 ya que tampoco existe.

```

5058 • INSERT INTO company (id, company_name, phone, email, country, website) VALUES
5059 ('b-9999', 'Garden Smith Corp.', '07 45 87 32 28', 'garden.smith@protonmail.couk', 'United States', 'https://gardensmith.com');
5060
5061 • select * from company
5062 where id = "b-9999";

```

Result Grid						
Filter Rows: <input type="text"/>						
Edit: Export/Import: Wrap Cell Content:						
	id	company_name	phone	email	country	website
▶	b-9999	Garden Smith Corp.	07 45 87 32 28	garden.smith@protonmail.couk	United States	https://gardensmith.com
•	NULL	NULL	NULL	NULL	NULL	NULL

company 10 x				
Output				
Action Output				
#	Time	Action	Message	
✓ 16	12:15:15	INSERT INTO company (id, company_name, phone, email, country, website) VALUES ('b-9999', 'Garden S...	1 row(s) affected	
✓ 17	12:15:26	select * from company where id = "b-9999"	1 row(s) returned	

Una vez añadidos los nuevos registros en las tablas Company y credit_card se podrá agregar el registro nuevo en transaction.

```

5049 • insert into transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined) values
5050 ("10881D1D-5B23-A76C-55EF-C568E49A99DD", "CcU-9999", "b-9999", "9999", "829.999", "-117.999", "2024-08-14 12:24:25", "111.11", "0");
5051

```

Output				
Action Output				
#	Time	Action	Message	
✓ 17	12:15:26	select * from company where id = "b-9999"	1 row(s) returned	
✓ 18	12:33:54	insert into transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined) v...	1 row(s) affected	

Una vez realizado lo anterior, hay que comprobar si existe algún registro con el identificador id de la tabla anterior, se podrá observar que aparecen los datos registrados anteriormente.

```
5052 • select *
5053 from transaction
5054 where id = "10881D1D-5B23-A76C-55EF-C568E49A99DD";
5055
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
10881D1D-5B23-A76C-55EF-C568E49A99DD	CcU-9999	b-9999	9999	829.999	-117.999	2024-08-14 12:24:25	111.11	0

transaction 11 x

Output

Action Output

#	Time	Action	Message
18	12:33:54	insert into transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined) v...	1 row(s) affected
19	12:38:27	select * from transaction where id = "10881D1D-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.

Para este apartado primero se hace una selección de todo para ver que el campo pan existe en la tabla credit_card.

```
5075 • select *
5076 from credit_card;
```

id	iban	pan	pin	cvv	expiring_date	fecha_actual
CcS-4857	XX4857591835292505850771	2314242385113924	1819	467	09/27/25	NULL
CcS-4858	XX8581768137002436094025	6582720299715533	3964	817	12/28/28	NULL
CcS-4859	XX7826930491423553609370	8861684536289642	4983	277	11/26/26	NULL
CcS-4860	XX5559590368835304645299	2481155515498459	6876	661	07/27/27	NULL
CcS-4861	XX2035182877195191627307	1308930301149557	5710	398	04/25/26	NULL

credit_card 12 x

Output

Action Output

#	Time	Action	Message
19	12:38:27	select * from transaction where id = "10881D1D-5B23-A76C-55EF-C568E49A99DD"	1 row(s) returned
20	12:47:24	select * from credit_card	5001 row(s) returned

Para eliminar el campo pan será necesario utilizar alter table para modificar la tabla, definir el nombre de la tabla en la que queremos eliminar, con drop column podremos eliminar la columna que sea necesaria, en este caso pan.

```
5078 • alter table credit_card drop column pan;
5079
```

Output

Action Output

#	Time	Action	Message
20	12:47:24	select * from credit_card	5001 row(s) returned
21	13:44:05	alter table credit_card drop column pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Si se vuelve a verificar si sigue existiendo el campo pan en la tabla credit_card se podrá comprobar que ya no existe.

```
5080 • select *
5081 from credit_card;
```

The screenshot shows a database interface with a query result grid for the 'credit_card' table. The grid has columns: id, iban, pin, cvv, expiring_date, and fecha_actual. It displays 6 rows of data. Below the grid, the 'credit_card 13 x' tab is visible. The 'Output' section shows an 'Action Output' log with two entries: a successful 'alter table credit_card drop column pan' at 13:44:05 and a successful 'select * from credit_card' at 13:44:55, which returned 5001 rows.

id	iban	pin	cvv	expiring_date	fecha_actual
CcS-4857	XX4857591835292505850771	1819	467	09/27/25	NULL
CcS-4858	XX8581768137002436094025	3964	817	12/28/28	NULL
CcS-4859	XX7826930491423553609370	4983	277	11/26/26	NULL
CcS-4860	XX559590368835304645299	6876	661	07/27/27	NULL
CcS-4861	XX2035182877195191627307	5710	398	04/25/26	NULL

#	Time	Action	Message
21	13:44:05	alter table credit_card drop column pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
22	13:44:55	select * from credit_card	5001 row(s) returned

Nivel 2

Ejercicio 1

Elimina de la tabla transacción el registro con ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD de la base de datos.

Para este apartado seleccionaremos todo de la tabla transaction y se pondrá un filtro para buscar el registro en el campo id y comprobar que existe.

```
5086 • select *
5087 from transaction
5088 where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD";
5089
```

The screenshot shows a database interface with a query result grid for the 'transaction' table. The grid has columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. It displays 2 rows of data. Below the grid, the 'transaction 14 x' tab is visible. The 'Output' section shows an 'Action Output' log with three entries: a successful 'select * from credit_card' at 13:44:55 (5001 rows returned), a successful 'select * from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"' at 13:52:11 (1 row returned), and a successful 'delete from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"' at 13:52:33 (1 row affected).

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
000447FE-B650-4DCF-85DE-C7ED0EE1CAAD	CcS-5019	b-2370	438	41.5972	12.2218	2016-12-21 20:07:18	155.63	0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#	Time	Action	Message
22	13:44:55	select * from credit_card	5001 row(s) returned
23	13:52:11	select * from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) returned
24	13:52:33	delete from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) affected

Una vez verificado será necesario utilizar delete from, el nombre de la tabla y un filtro para buscar el registro del campo id, de esta manera se borrara el registro que coincida con el que se ha puesto en el filtro.

```
5090 • delete from transaction
5091 where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD";
```

The screenshot shows the 'Output' section of the database interface, specifically the 'Action Output' log. It contains two entries: a successful 'select * from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"' at 13:52:11 (1 row returned) and a successful 'delete from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"' at 13:52:33 (1 row affected).

#	Time	Action	Message
23	13:52:11	select * from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) returned
24	13:52:33	delete from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) affected

Al volver a ejecutar la query que se utilizó para comprobar si el registro existía podremos observar que ya no existe y ha sido eliminado.

```
5093 • select *
5094 from transaction
5095 where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD";
```

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

transaction 15 x

Output

Action Output

#	Time	Action	Message
✓ 24	13:52:33	delete from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) affected
✓ 25	13:52:56	select * from transaction where id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	0 row(s) returned

Ejercicio 2

La sección 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. Media de compra realizada por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor promedio de compra.

Para realizar este apartado será necesario utilizar `create view` para crear la vista, ponerle un nombre (en este caso VistaMarketing) y hacer la query necesaria, la cual tiene la selección de los campos requeridos por la sección de marketing, un join entre la tabla Company y la tabla transaction, esta todo agrupado por el nombre de la compañía, el teléfono y el país, por ultimo se ordena de mayor a menor por la media de amount a la que se ha renombrado como mediaCompras.

```
5102 • create view VistaMarketing as
5103 select company_name, phone, country, avg(amount) as mediaComparas
5104 from company
5105 join transaction
5106 on company.id = transaction.company_id
5107 group by company_name, phone, country
5108 order by mediaComparas desc;
```

5109

Output

Action Output

#	Time	Action	Message
✓ 32	14:04:37	create view VistaMarketing as select company_name, phone, country, avg(amount) as mediaComparas from ...	0 row(s) affected

Para ver los resultados de la vista será necesario seleccionar todo y en este caso en el from en vez de poner una tabla se pondrá el nombre de la vista.

```
5110 • select * from vistamarketing;
```

company_name	phone	country	mediaComparas
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.867160
Pretium Neque Corp.	07 77 48 55 28	Australia	276.158330
Urna Convallis Associates	06 01 24 77 04	United States	274.235011
At Associates	09 56 61 10 65	New Zealand	272.214870
Metus Vitae Associates	08 25 44 40 66	Australia	270.080965

vistamarketing 20 x

Output

Action Output

#	Time	Action
33	14:06:36	select * from vistamarketing
34	14:08:21	select * from vistamarketing

Ejercicio 3

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

Para realizar este apartado será necesario seleccionar todo para ver toda la información de la vista, y un filtro en el que especificaremos que el país (country) sea Alemania, de esta manera el resultado será todo la información de las empresas ubicadas en Alemania.

```
5114 • select *
5115 from vistamarketing
5116 where country = "Germany";
```

company_name	phone	country	mediaComparas
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.867160
Nunc Interdum Incorporated	05 18 15 48 13	Germany	259.319156
Convallis In Incorporated	06 66 57 29 50	Germany	257.745376
Ac Industries	09 34 65 40 60	Germany	255.147288
Rutrum Non Inc.	02 66 31 61 09	Germany	255.136927
Auctor Mauris Corp.	05 62 87 14 41	Germany	254.765518
Augue Foundation	06 88 43 15 63	Germany	253.505000
Aliquam PC	01 45 73 52 16	Germany	253.136923

vistamarketing 21 x

Output

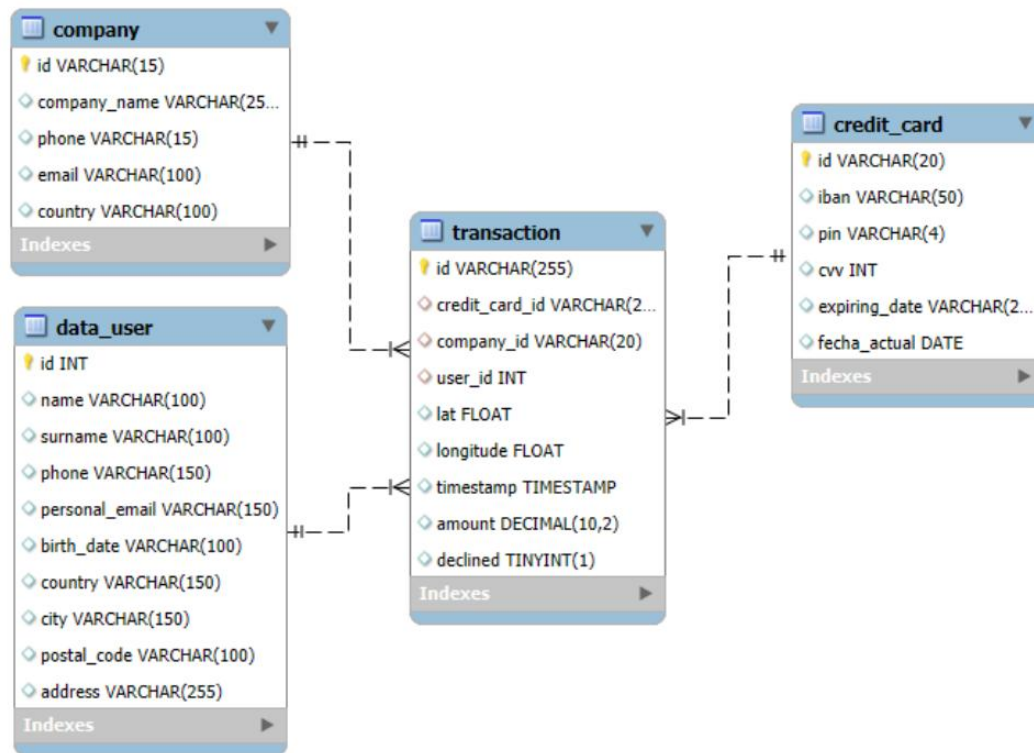
Action Output

#	Time	Action	Message
34	14:08:21	select * from vistamarketing	101 row(s) returned
35	16:04:18	select * from vistamarketing where country = "Germany"	8 row(s) returned

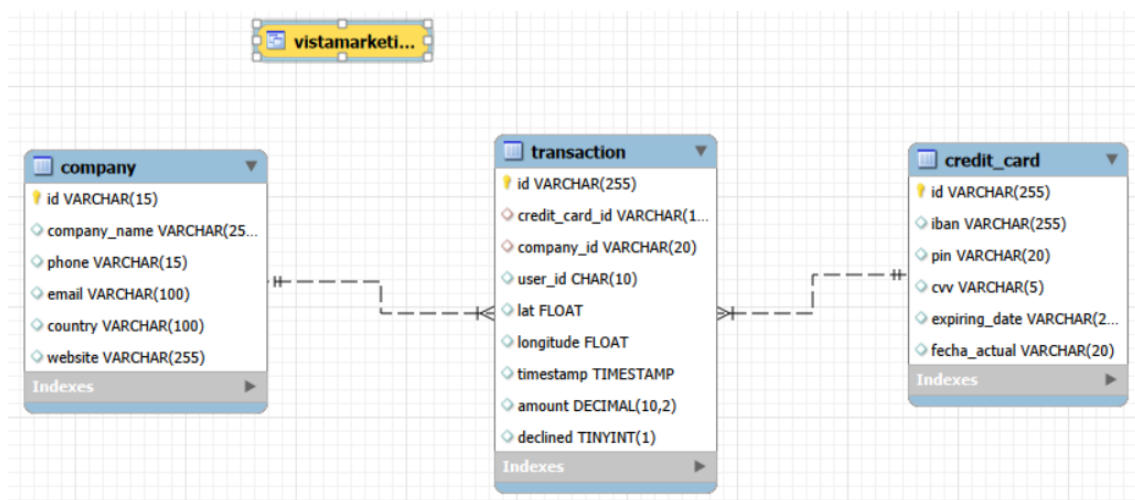
Nivel 3

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 realizó. Te pide que le ayudes a dejar los comandos ejecutados para obtener el siguiente diagrama:



El diagrama inicial estaría así



Primero eliminamos la columna website de la tabla Company, para ello utilizamos alter table, el nombre de la tabla de la que queremos borrar la columna (company) y drop column con el nombre de la columna.

```
5064 • alter table company
5065 drop column website;
5066
```

Output

#	Time	Action	Message
29	18:35:33	DROP TABLE 'transactions'. 'user'	0 row(s) affected
30	18:39:47	alter table company drop column website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

company

Columns

- id
- company_name
- phone
- email
- country
- website

company

Columns

- id
- company_name
- phone
- email
- country

Luego, en la tabla credit_card modificamos los siguientes campos:

Id varchar(255) a id varchar(20):

Para poder modificarlo hay que borrar la clave foránea, si no dará error, para ello ejecutamos las siguientes queries, primero borramos la clave foránea, y luego modificamos el dato

```
5067 • ALTER TABLE transaction
5068 DROP FOREIGN KEY fk_credit_card_id;
5069
5070 • alter table credit_card
5071 modify id varchar(20);
5072
```

Output

#	Time	Action	Message
34	18:55:24	ALTER TABLE transaction DROP FOREIGN KEY fk_credit_card_id	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
35	18:55:33	alter table credit_card modify id varchar(20)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

Information

Table: credit_card

Columns:

Column	Type
id	varchar(20) PK
iban	varchar(255)
pin	varchar(20)
cvv	varchar(5)
expiring_date	varchar(20)
fecha_actual	varchar(20)

Iban varchar(255) a iban varchar(50)

Information

Table: credit_card

Columns:

Column	Type
id	varchar(20) PK
iban	varchar(50)
pin	varchar(20)
cvv	varchar(5)
expiring_date	varchar(20)
fecha_actual	varchar(20)

```
5072
5073 • alter table credit_card
5074 modify iban varchar(50);
5075
```

Output

#	Time	Action	Message
35	18:55:33	alter table credit_card modify id varchar(20)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0
36	19:45:55	alter table credit_card modify iban varchar(50)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

Pin varchar(20) a pin varchar(4)

Information

Table: credit_card

Columns:

- id: varchar(20) PK
- iban: varchar(50)
- pin: varchar(4)
- cvv: varchar(5)
- expiring_date: varchar(20)
- fecha_actual: varchar(20)

Object Info Session

```
5075
5076 • alter table credit_card
5077   modify pin varchar(4);
5078
```

Output

#	Time	Action	Message
36	19:45:55	alter table credit_card modify iban varchar(50)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0
37	19:50:37	alter table credit_card modify pin varchar(4)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

Cvv varchar(5) a cvv int

Information

Table: credit_card

Columns:

- id: varchar(20) PK
- iban: varchar(50)
- pin: varchar(4)
- cvv: int
- expiring_date: varchar(20)
- fecha_actual: varchar(20)

Object Info Session

```
5078
5079 • alter table credit_card
5080   modify cvv int;
5081
```

Output

#	Time	Action	Message
37	19:50:37	alter table credit_card modify pin varchar(4)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0
38	19:52:22	alter table credit_card modify cvv int	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

Fecha_actual varchar(20) fecha_actual DATE

Information

Table: credit_card

Columns:

- id: varchar(20) PK
- iban: varchar(50)
- pin: varchar(4)
- cvv: int
- expiring_date: varchar(20)
- fecha_actual: date

Object Info Session

```
5081
5082 • alter table credit_card
5083   modify fecha_actual date;
5084
```

Output

#	Time	Action	Message
38	19:52:22	alter table credit_card modify cvv int	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0
39	19:54:06	alter table credit_card modify fecha_actual date	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

Corrección (antes salía todo null porque no se le había asignado la fecha actual)

```
136 • alter table credit_card
137   add fecha_actual date default (current_date); -- correccion
138
```

Output

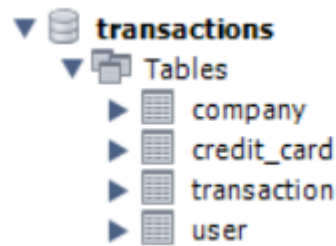
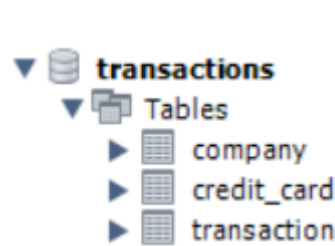
#	Time	Action	Message
19	12:33:50	alter table credit_card add fecha_actual date default (current_date)	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0

Ahora creamos la tabla user, para ello utilizaremos la query del archivo estructura datos user.sql

```
5085 • CREATE TABLE IF NOT EXISTS user (
5086   id CHAR(10) PRIMARY KEY,
5087   name VARCHAR(100),
5088   surname VARCHAR(100),
5089   phone VARCHAR(150),
5090   email VARCHAR(150),
5091   birth_date VARCHAR(100),
5092   country VARCHAR(150),
5093   city VARCHAR(150),
5094   postal_code VARCHAR(100),
5095   address VARCHAR(255)
5096 );
5097
```

Output

#	Time	Action	Message
39	19:54:06	alter table credit_card modify fecha_actual date	5001 row(s) affected Records: 5001 Duplicates: 0 Warnings: 0
40	20:01:48	CREATE TABLE IF NOT EXISTS user (id CHAR(10) PRIMARY KEY, name VARCHAR(100), surname VAR...	0 row(s) affected



Comprobamos que la tabla user esta vacía.

5002 • `select *from user;`

id	name	surname	phone	email	birth_date	country	city	postal_code	address
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

user 1 x

Output

Action Output

#	Time	Action	Message
40	20:01:48	CREATE TABLE IF NOT EXISTS user (id CHAR(10) PRIMARY KEY, name VARCHAR(100), surname VAR...	0 row(s) affected
41	20:04:19	select *from user	0 row(s) returned

Insertamos las queries del archivo datos introducir sprint3 user.sql y volvemos a comprobar si se han introducido todos los datos correctamente.

4992 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4945", "Pimdet", "Zvagjirk", "+91-

4993 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4947", "Ypyafn", "Tfiilwnz", "+95-

4994 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4950", "Akxgbk", "Fxtbrlub", "+96-

4995 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4952", "Fufhmu", "Sbgtpsp", "+61-

4996 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4961", "Phwjfi", "Hyxbvzkf", "+34-

4997 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4977", "Gjldyg", "Egqecsc", "+62-

4998 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4980", "Asgaxi", "Btardzti", "+39-

4999 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4983", "Eremdc", "Orekkpbr", "+59-

5000 • `INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES (` "4999", "Omjnoy", "Rmuqvngx", "+32-

5001

5002 • `select *from user;`

id	name	surname	phone	email	birth_date	country	city	postal_code	address
1	Zeus	Gamble	1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	New York	10001	348-7818 Sagittis St.
10	Robert	McCarthy	(324) 746-6771	fermentum@protonmail.com	Apr 30, 1984	United States	San Jose	95101	P.O. Box 773
100	Melodie	Mclean	1-677-221-7152	risus.varius@google.ca	Sep 15, 1989	United States	San Jose	95101	Ap #644-8492 Sagittis St.
1000	Amigrv	Qbulxpb	+48-258-9936	amigrv.qbulxpb@example.com	May 17, 1970	Germany	Stuttgart	70173	215 Qbulxpb St
1001	Nfvlrb	Oydaibvg	+94-121-2522	nfvlrb.oydaibvg@example.com	Mar 4, 1994	Germany	Cologne	50667	121 Oydaibvg St

user 2 x

Output

Action Output

#	Time	Action	Message
5041	20:05:06	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALU...	1 row(s) affected
5042	20:05:25	select *from user	5000 row(s) returned

Ahora se modificaran algunos valores de la tabla transaction y de la tabla user:

User_id char(10) a user_id int (tabla transaction)

Information	
Table: transaction	
Columns:	
id	varchar(255) PK
credit_card_id	varchar(15) PK
company_id	varchar(20)
user_id	char(10)
lat	float
longitude	float

Information	5097	
	5098	alter table transaction
	5099	modify user_id int;
	5100	
Output		
Action Output		
#	Time	Action
5045	20:17:07	select * from user where id = "9999"
5046	20:23:25	alter table transaction modify user_id int
		Message
		0 row(s) returned
		100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0

User_id char(10) a user_id int (tabla user)

Information	
Table: user	
Columns:	
id	char(10) PK
name	varchar(100)
surname	varchar(100)
phone	varchar(150)
email	varchar(150)

Information	5100	
	5101	alter table user
	5102	modify id int;
	5103	
Output		
Action Output		
#	Time	Action
5046	20:23:25	alter table transaction modify user_id int
5047	20:26:03	alter table user modify id int
		Message
		100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0
		5000 row(s) affected Records: 5000 Duplicates: 0 Warnings: 0

Credit_card_id varchar(15) a credit_card_id(20) (tabla transaction)

Information	5103	
	5104	alter table transaction
	5105	modify credit_card_id varchar(20);
	5106	
Output		
Action Output		
#	Time	Action
5047	20:26:03	alter table user modify id int
5048	20:59:41	alter table transaction modify credit_card_id varchar(20)
		Message
		5000 row(s) affected Records: 5000 Duplicates: 0 Warnings: 0
		0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Renombramos la tabla user por data_user

Information	5105	modify credit_card_id varchar(20);
	5106	
	5107	alter table user rename to data_user;
	5108	
Output		
Action Output		
#	Time	Action
5049	21:03:09	alter table transaction add constraint fk_credit_card_id foreign key (credit_card_id) references credit_card(id)
5050	21:07:12	alter table user rename to data_user
		Message
		100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0
		0 row(s) affected

Renombramos la columna email por personal_email (tabla data_user)

Information

Table: **data_user**

Columns:

Column	DataType	PK
id	int	PK
name	varchar(100)	
surname	varchar(100)	
phone	varchar(150)	
personal_email	varchar(150)	
birth_date	varchar(100)	
country	varchar(150)	
city	varchar(150)	

Object Info Session

```

5108
5109 • alter table data_user
5110 rename column email to personal_email;
5111

```

Output

#	Time	Action	Message
5059	21:22:16	alter table transaction add constraint fk_data_user_id foreign key (user_id) references data_user(id)	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0
5060	21:27:51	alter table data_user rename column email to personal_email	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Declaramos la clave foránea de la tabla credit_card en transaction

```

5107 • alter table transaction
5108 add constraint fk_credit_card_id
5109 foreign key (credit_card_id)
5110 references credit_card(id);
5111

```

Output

#	Time	Action	Message
5048	20:59:41	alter table transaction modify credit_card_id varchar(20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
5049	21:03:09	alter table transaction add constraint fk_credit_card_id foreign key (credit_card_id) references credit_card(id)	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0

Declaramos la clave foránea de la tabla data_user en transaction

Da un error (1452. Cannot add or update a child row: a foreign key constraint fails)

Este error ocurre porque hay datos que no concuerdan en ambas tablas, en este caso el problema está en el registro que añadimos en el ejercicio 3 del nivel 1, añadimos un registro en el que el campo user_id tenía como usuario el 9999, el cual no existe en la tabla user

```

5116 • alter table transaction
5117 add constraint fk_data_user_id
5118 foreign key (user_id)
5119 references data_user(id);
5120
5121 • select *
5122 from transaction
5123 where user_id = "9999";
5124

```

Result Grid

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B ID ID-SB23-A76C-55EF-C568E49A99DD	CcJ-9999	b-9999	9999	829.999	-117.999	2024-08-14 12:24:25	111.11	0

transaction 5

Output

#	Time	Action	Message
5053	21:10:23	alter table transaction add constraint fk_data_user_id foreign key (user_id) references data_user(id)	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (transactions.`#sql-1860...
5054	21:11:14	select * from transaction where user_id = "9999"	1 row(s) returned

```

5125 • select *
5126 from data_user
5127 where id = "9999";
5128

```

Result Grid

id	name	surname	phone	email	birth_date	country	city	postal_code	address
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Como se aprecia en la imagen anterior, el usuario 9999 no existe en la tabla data_user, se crea con la siguiente query

```

5129 • insert into data_user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES
5130 ("9999", "lzxjnvlj", ",mxsfknw", "+39-478-3548", "asgaxisdgsi@example.com", "Jan 15, 1989", "Canada", "Winnipeg", "R2C 0A1", "284 Btardzti St");
5131

```

Output

#	Time	Action	Message
5056	21:18:53	insert into data_user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VAL...	1 row(s) affected

Al comprobar de nuevo el registro en la tabla data_user podremos comprobar que ahora si existe

```

123 • select *
124 from data_user
125 where id = "9999";
126

```

Result Grid

	id	name	surname	phone	personal_email	birth_date	country	city	postal_code	address
▶	9999	lzxjnvlj	,mxsfknw	+39-478-3548	asgaxisdgsi@example.com	Jan 15, 1989	Canada	Winnipeg	R2C 0A1	284 Btardzti St
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

data_user 1 x

Output

#	Time	Action	Message
24	11:51:15	select company.company_name, company.phone, company.country, date(timestamp) as fecha, round(transaction.a...	8 row(s) returned
25	16:21:23	select * from data_user where id = "9999"	1 row(s) returned

Si ejecutamos la cuery para declarar la clave foránea ahora podremos comprobar que se ejecuta sin problema y no da ningún error

```

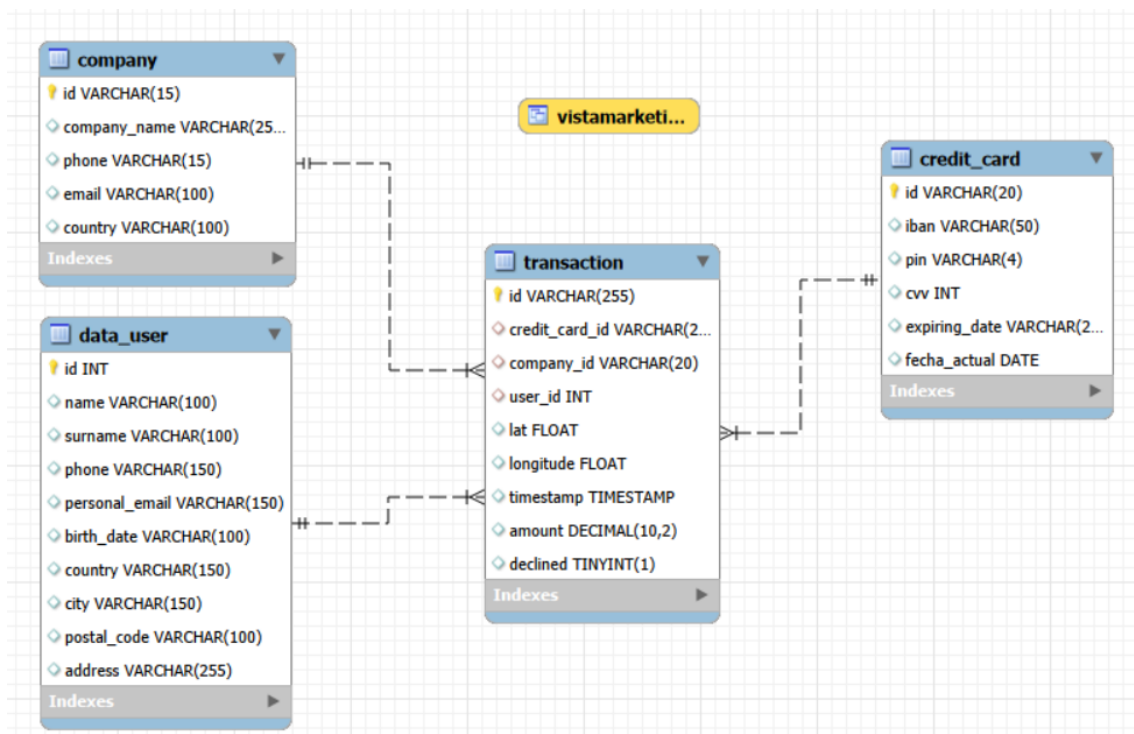
5136 • alter table transaction
5137 add constraint fk_data_user_id
5138 foreign key (user_id)
5139 references data_user(id);

```

Output

#	Time	Action	Message
5058	21:19:58	select * from data_user where id = "9999"	1 row(s) returned
5059	21:22:16	alter table transaction add constraint fk_data_user_id foreign key (user_id) references data_user(id)	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0

El diagrama entidad-relación final quedaría de la siguiente manera:



La relación entre la tabla data_user y transaction es 1-N de data_user a transaction, esto quiere decir que un usuario puede tener muchas transacciones, pero una transacción solo puede pertenecer a un usuario.

Ejercicio 2

La empresa también le pide 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 usada.

Nombre de la compañía de la transacción realizada.

Asegúrese de incluir información relevante de las tablas que conocerá y utilice alias para cambiar de nombre columnas según sea necesario.

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

Para este apartado se crea una vista con create view y se le pone InformeTecnico, en la selección se ponen todos los campos solicitados, se renombrar para que se mas legible y unimos la tabla credit_card, transaction, data_user y Company con join, luego se selecciona todo de la vista informetecnico y se ordena de mayor a menor por el campo identificador (que es el id de la tabla transaction).

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Tables' folder is expanded, showing the 'transaction' table. The 'Columns' section for 'transaction' lists: id (varchar(255), PK), credit_card_id (varchar(20)), user_id (int), lat (float), longitude (float), and timestamp (timestamp). The main pane shows the SQL script for creating the view 'InformeTecnico' and selecting from it. The script is as follows:

```
5153
5154 * create view InformeTecnico as
5155 select transaction.id as identificador, data_user.name as nombre_usuario, data_user.surname as apellido_usuario, credit_card.iban, company.company_name as nombre_compania
5156 from credit_card
5157 join transaction
5158 on credit_card.id = transaction.credit_card_id
5159 join data_user
5160 on transaction.user_id = data_user.id
5161 join company
5162 on transaction.company_id = company.id;
5163
5164 * select * from informetecnico
5165 order by identificador desc;
```

The 'Result Grid' shows the output of the view. The columns are: identificador, nombre_usuario, apellido_usuario, iban, and nombre_compania. The results are as follows:

identificador	nombre_usuario	apellido_usuario	iban	nombre_compania
FFFD31D6-94D5-47CE-854A-7C68E1CC274B	Bmrgl	Tprvmtc	XX794814451211289182409922	Turpe Company
FFFCF76D-ECF9-4985-42D0-42A7B79998FC	Dlflcd	Vllogdl	XX636251701647892036676034	Amet Nalla Donec Corporation
FFFC9E8D-27C7-4ADE-98F2-7533EF4DF126	Securp	FaeFvqfy	XX162677143304223631437567	Nunc Interdum Incorporated
FFFB270D-F53A-4D3D-9666-E5307C33CCB4	Ggdpq	Ulrzqyh	XX395114267082019952567052	Viverra Donec Foundation
FFFE3CE-234E-408C-48EF-F9CAD577224A	Yshmq	Zpsleed	XX8845462156537570367941	Convallo In Incorporated

The 'Output' pane shows the execution of the view. The first action is 'select * from vistamarketing order by transaction id desc' (Error Code: 1054, Unknown column 'transaction id' in 'order clause'). The second action is 'select * from informetecnico order by identificador desc' (100000 row(s) returned).