

Tarea S3.01. Manipulación de tablas

Descripción

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, tendrás que 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 online. En esta tarea, empezarás a trabajar con información relacionada con tarjetas de crédito.



Nivell 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_credit". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

Para empezar en este caso de tener que crear una tabla que se llame 'credit_card', pensé: ¿Que campos llevaría?

Y para ello, como contaba con "datos_introducir_credit" con los datos a ingresar, preste atención a los campos que contenían esa información.

INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date)

Esos iban a ser los campos que iba a necesitar para crear la tabla credit_card.

Entonces procedo a la creación de la tabla.

```
CREATE TABLE IF NOT EXISTS credit_card (  
  id VARCHAR(100) PRIMARY KEY,  
  iban VARCHAR(50) NOT NULL,  
  pan VARCHAR(100),  
  pin INT,  
  cvv SMALLINT,  
  expiring_date VARCHAR(10)  
);
```

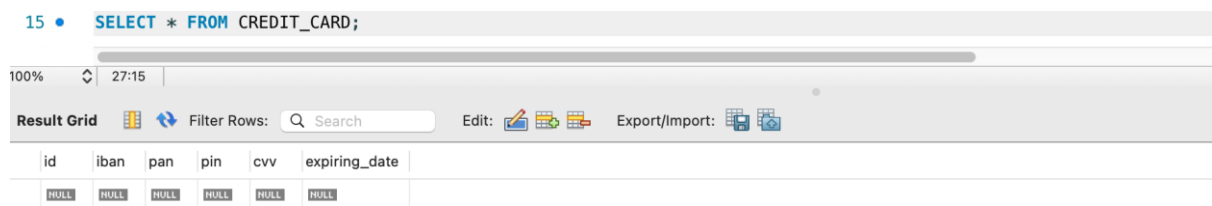
Luego le indico la foreign key que se va a relacionar con el campo credit_card_id de la tabla transaction.

```
ALTER TABLE transaction
ADD CONSTRAINT fk_credit_card
FOREIGN KEY (credit_card_id)
REFERENCES credit_card(id);
```

De la tabla transaction le agrego la cláusula 'constraint' que asegura la integridad y exactitud de la información.

Realizo la consulta y tenemos la tabla creada.

Hice los mismos pasos con la tabla **user**.



Después de que tenga la tabla creada, pase al siguiente paso que fue cargar los datos

del archivo `datos_introducir_credit.sql`

```
2 -- Insertamos datos de credit_card
3 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2938', 'TR301950312213576817638661', '5424465566813633', '3257', '984', '10/30/22');
4 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2945', 'D026854763748537475216568689', '5142423821948828', '9080', '887', '08/24/23');
5 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2952', 'BG45IVQL52710525608255', '4556 453 55 5287', '4598', '438', '06/29/21');
6 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ( 'CcU-2959', 'CR7242477244335841535', '372461377349375', '3583', '667', '02/24/23');
```

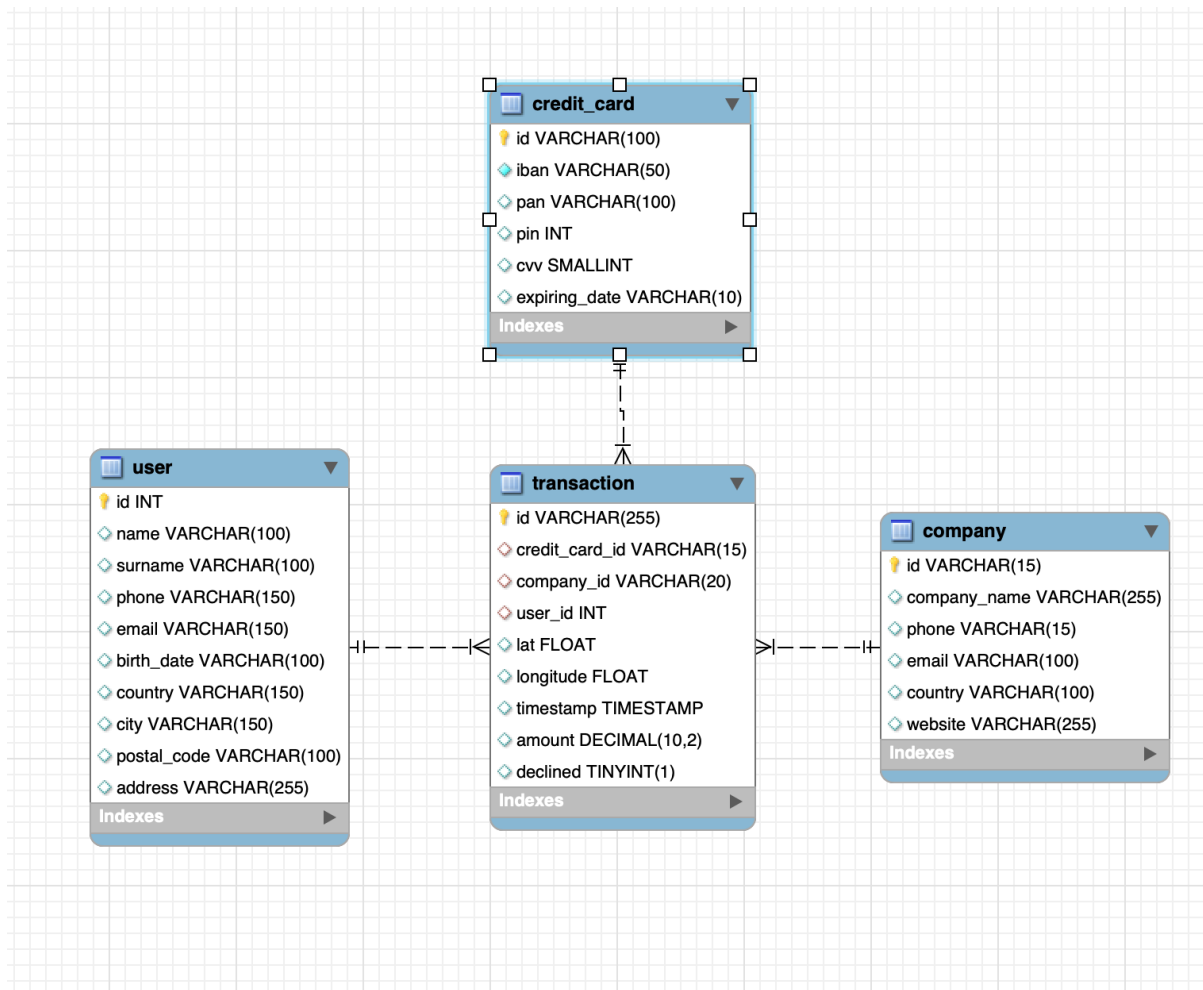
Ejecuté la script, y luego hice la comprobación.

id	iban	pan	pin	cvv	expiring_date
CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22
CcU-2945	D026854763748537475216568689	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	BG72LKQT15627628377363	448566 886747 7265	4900	130	10/29/24
CcU-2973	PT8780628135092429456346	544 58654 54343 384	8760	887	01/30/25
CcU-2980	DE39241881883086277136	402400 7145845969	5075	596	07/24/22
CcU-2987	GE89681434837748781813	3763 747687 76666	2298	797	10/31/23
CcU-2994	BH62714426368066765294	344283273252593	7545	595	02/28/22
CcU-3001	CY48087426654774581266832110	511722 924833 2244	9562	867	09/16/22
CcU-3008	LU507216693616119230	4485744464433884	1856	740	04/05/25
CcU-3015	PS119398216295715968342456821	3784 662233 17389	3246	822	01/31/22
CcU-3022	GT91695162850556977423121857	5164 1379 4842 3951	5610	342	04/25/25
CcU-3029	AZ62317413982441418123739746	3429 279566 77631	9708	505	09/02/23

credit_card 7

Action Output

Time	Response
1127 11:08:23	1 row(s) affected
1128 11:08:23	1 row(s) affected
1129 11:08:23	1 row(s) affected
1130 11:08:23	1 row(s) affected
1131 11:08:23	1 row(s) affected
1132 11:08:23	1 row(s) affected
1133 11:08:23	1 row(s) affected
1134 11:08:23	1 row(s) affected
1135 11:08:23	1 row(s) affected
1136 11:08:27	275 row(s) returned



Así quedaría el diagrama de relación. Donde tenemos la tabla principal que es **transaction** que es de muchas a una, es decir, una compañía puede tener varias transacciones, pero no puede haber 1 transacción para 2 compañías. Lo mismo sucede con la tabla user, que un usuario puede hacer varias transacciones, y que una tarjeta de crédito puede usarse para varias transacciones también.

- 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.

Lo primero que hice fue hacer la consulta correspondiente para encontrar y verificar esta cuenta:

```

26 • Select * from credit_card
27 where id = 'CcU-2938';

```

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

Luego procedo a efectuar el cambio que Recursos Humanos solicito:
Utilizando la cláusula *update*.

La sintaxis es:

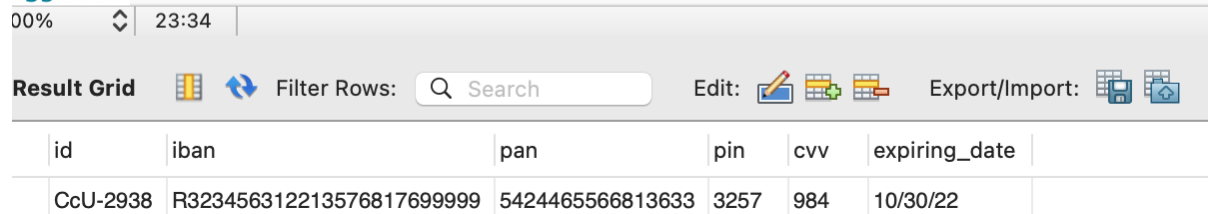
UPDATE 'tabla'

SET 'Campo' = 'valor'

Y en este caso, he utilizado el filtro para identificarlo

WHERE 'campo' = 'valor'

```
29 • Update credit_card
30 set iban = 'R323456312213576817699999'
31 where id = 'CcU-2938';
32
33 • Select * from credit_card
34 where id = 'CcU-2938';
35
```



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

Compruebo que se ha efectuado la actualización satisfactoriamente.

- 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

late 829.999

longitud -117.999

amount 111.11

declined 0

Para ingresar un nuevo usuario debo hacer un INSERT en la tabla transaction.

Sintaxis:

Insert into transaction (campos, campos, campos) ya hecho en el primer ejercicio.

Values ('información del campo 1', 'información del campo 2', ...) así sucesivamente según los campos que tenga.

```
INSERT INTO TRANSACTION(id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined)
VALUE ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', 82.999, -117.999, '2024-07-22 14:30:00', 111.11, 0);
```

Cuando intento agregar esta nueva cuenta en la tabla transaction, da un error y es porque estoy agregando datos como por ejemplo 'CcU-9999' que es la ID (PK) de la tabla CREDIT_CARD, 'b-9999' que a su vez es la ID de COMPANY_ID, y '9999' que es la ID de la tabla USER. Y este error es a causa de que estos datos no existen en sus respectivas tablas y entonces no se podrán agregar por son foreign keys de transaction, y los datos que estamos intentando agregar no existen en sus respectivas tablas.

Entonces se me ocurrieron dos maneras de hacerlo. Pienso en teoría que si tengo una transacción que agregar y no tengo esos datos debería consultarlos y en ese caso los agregaría para que en todas sus tablas tengan el valor, la información completa.





```
INSERT INTO CREDIT_CARD VALUE ('CcU-9999', 'x', 'x', '0', '0', 'x');
```

```
INSERT INTO COMPANY VALUE ('b-9999', 'x', 'x', 'x', 'x', 'x');
```

```
INSERT INTO USER VALUE ('9999', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x');
```

Una vez agregados:

```
+
; • INSERT INTO TRANSACTION(id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined)
;   VALUE ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', 82.999, -117.999, '2024-07-22 14:30:00', 111.11, 0);
7 • SELECT * FROM TRANSACTION
}   WHERE CREDIT_CARD_ID = 'CcU-9999';
}
}
```

80:63									
ult Grid  Filter Rows: <input type="text"/> Search  Edit:  Export/Import: 									
id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined	
108B1D1D-5B23-A76C-55EF-C568E49A99DD	CcU-9999	b-9999	9999	82.999	-117.999	2024-07-22 14:30:00	111.11	0	

Así efectivamente se pudo agregar los valores en la tabla transaction.

Luego otra forma es deshabilitando las foreign key

Set foreign_key_checks = 0

Realizar el Insert

Y luego:

Set foreign_key_checks = 1

Porque la función de la foreign key es encargarse de que a la hora de ingresar valores se asegure de que estos estén en las tablas, es decir, existan.

Y en este caso la utilice para deshabilitarla momentáneamente.

```
Set Foreign_key_checks = 0;  
Insert into transaction(id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined)  
VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', 82.999, -117.999, '2024-07-22 14:30:00', 111.11, 0);  
Set foreign_key_checks = 1;
```

Esta forma no es recomendable, porque no estás haciendo efectiva la relación.

La estas desactivando y activando momentáneamente.

- Ejercicio 4

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

Así es la tabla CREDIT_CARD ahora.

75 • **SELECT * FROM CREDIT_CARD;**

76

77

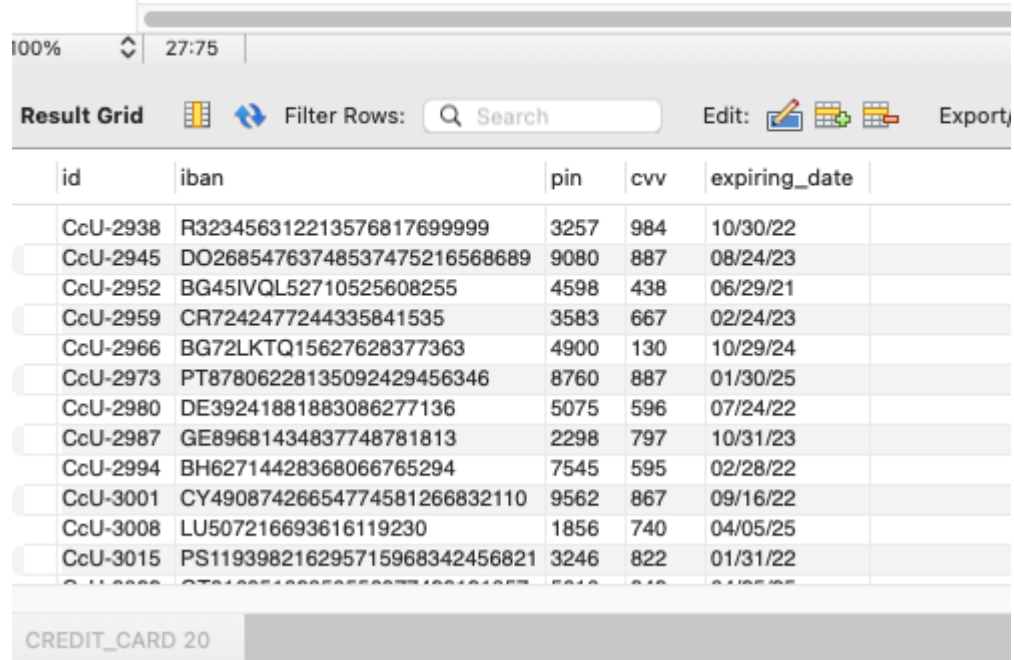
id	iban	pan	pin	cvv	expiring_date
CcU-2938	R323456312213576817699999	5424465566813633	3257	984	10/30/22
CcU-2945	DO26854763748537475216568689	5142423821948828	9080	887	08/24/23
CcU-2952	BG45IVQL52710525608255	4556 453 55 5287	4598	438	06/29/21
CcU-2959	CR7242477244335841535	372461377349375	3583	667	02/24/23
CcU-2966	BG72LKTQ15627628377363	448566 886747 7265	4900	130	10/29/24
CcU-2973	PT87806228135092429456346	544 58654 54343 384	8760	887	01/30/25
CcU-2980	DE39241881883086277136	402400 7145845969	5075	596	07/24/22
CcU-2987	GE89681434837748781813	3763 747687 76666	2298	797	10/31/23
CcU-2994	BH62714428368066765294	344283273252593	7545	595	02/28/22
CcU-3001	CY49087426654774581266832110	511722 924833 2244	9562	867	09/16/22
CcU-3008	LU507216693616119230	4485744464433884	1856	740	04/05/25
CcU-3015	PS119398216295715968342456821	3784 662233 17389	3246	822	01/31/22

CREDIT_CARD 19

Como en recursos humanos deciden que el campo 'pan' ya no es necesario, procedo a eliminarlo de la siguiente manera.

Para eliminar he utilizado la siguiente clausula:

```
76 • ALTER TABLE CREDIT_CARD DROP COLUMN pan;  
77
```



The screenshot shows a database management tool interface. At the top, there's a command editor with the SQL command: `ALTER TABLE CREDIT_CARD DROP COLUMN pan;`. Below the command editor, there's a toolbar with options like "Result Grid", "Filter Rows", "Search", "Edit", and "Export". The main area displays a table with the following columns: `id`, `iban`, `pin`, `cvv`, and `expiring_date`. The table contains 15 rows of data. The status bar at the bottom indicates "CREDIT_CARD 20".

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
CcU-2973	PT87806228135092429456346	8760	887	01/30/25
CcU-2980	DE39241881883086277136	5075	596	07/24/22
CcU-2987	GE89681434837748781813	2298	797	10/31/23
CcU-2994	BH62714428368066765294	7545	595	02/28/22
CcU-3001	CY49087426654774581266832110	9562	867	09/16/22
CcU-3008	LU507216693616119230	1856	740	04/05/25
CcU-3015	PS119398216295715968342456821	3246	822	01/31/22

Verificamos y comprobamos que se ha eliminado esa columna correctamente.



Nivell 2

Ejercicio 1

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

```
85 • DELETE FROM TRANSACTION
86 WHERE ID = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
87 • SELECT * FROM TRANSACTION
88 WHERE ID = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
89
```

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

60... 19:12:49 DELETE FROM TRANSACTION... 1 row(s) affected 0.0081 sec
60... 19:13:35 SELECT * FROM TRANSACTION... 0 row(s) returned 0.0011 sec / 0.00002...

Para eliminar el registro de la tabla transaction utilice la cláusula DELETE utilizando un filtro para identificar el ID proporcionado que es el campo ID de la tabla TRANSACTION.

Efectivamente se ha eliminado.

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 necesaria que crees una vista llamada VistaMarketing que contenga la siguiente información: Nombre de la compañía. Teléfono de contacto. País de residencia. Media de compra realizado por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor promedio de compra.

Para crear una vista, usaremos la sentencia CREATE 'nombre_de_la_vista' AS. En este caso es CREATE VistaMarketing AS que es el nombre se nos ha solicitado.

103 • CREATE VIEW VistaMarketing AS

Luego en la consulta se solicita:

Nombre de la compañía(company_name) Teléfono de contacto(phone) País de residencia(country) de la tabla COMPANY.

Y luego se nos pide la media de compra realizada(avg(amount)) de la tabla TRANSACTION.

```
104 SELECT COMPANY_NAME as Nombre_de_la_compañía, PHONE as Telefono_de_contacto, COUNTRY as Pais_de_residencia, avg(amount) as Media_de_compra
105 FROM COMPANY
```


Haremos un JOIN en la primary key de company (company.id) y la foreign key de transaction (transaction.company_id) para manipular lo que necesitamos de ambas tablas.

```
106 JOIN TRANSACTION
```

```
107 on company.id = transaction.company_id
```

En este caso usaremos una función agregada agrupada por GROUP BY de la información que provee cada compañía.

y a su vez como necesitamos ordenar el promedio de compra realizada de mayor a menor, usaremos la cláusula ORDER BY Media_de_compra (nombre que le cambie para que se entienda mejor) DESC (descendiente).

```
108 GROUP BY Nombre_de_la_compañía, Telefono_de_contacto, Pais_de_residencia
```

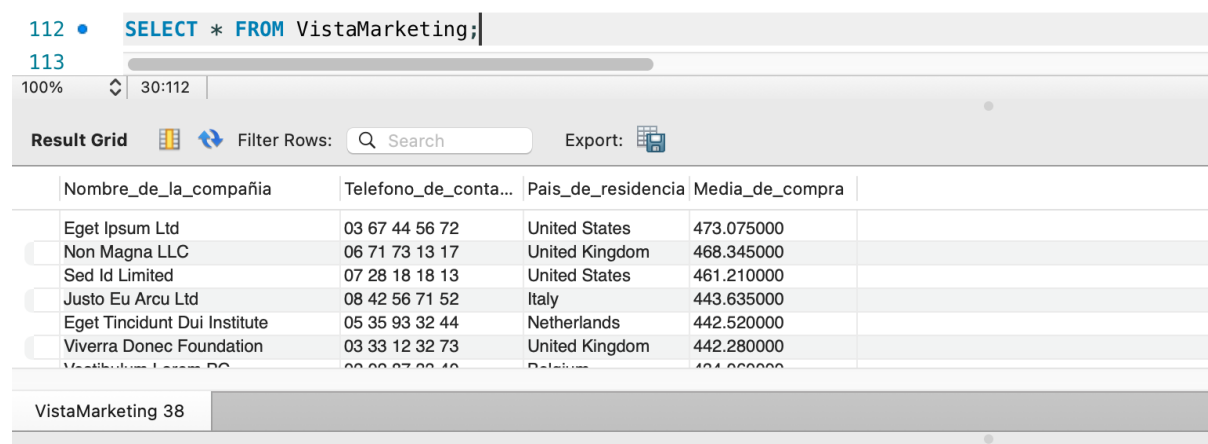
```
109 ORDER BY Media_de_compra DESC;
```

Procedemos a crear la vista:

```
104 • CREATE VIEW VistaMarketing AS
105 SELECT COMPANY_NAME as Nombre_de_la_compañía, PHONE as Telefono_de_contacto, COUNTRY as Pais_de_residencia, avg(amount) as Media_de_compra
106 FROM COMPANY
107 JOIN TRANSACTION
108 on company.id = transaction.company_id
109 GROUP BY Nombre_de_la_compañía, Telefono_de_contacto, Pais_de_residencia
110 ORDER BY Media_de_compra DESC;
111
```

Y efectuamos la comprobación

```
112 • SELECT * FROM VistaMarketing;
113
```



Nombre_de_la_compañía	Telefono_de_conta...	Pais_de_residencia	Media_de_compra
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
Medi...

VistaMarketing 38

Ejercicio 3

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

```

118 • SELECT * FROM VistaMarketing
119 WHERE Pais_de_residencia = 'germany';
120

```

100% 38:119

Result Grid



Filter Rows:

Search

Export:



	Nombre_de_la_compañia	Telefono_de_conta...	Pais_de_residencia	Media_de_compra	
	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 88 87 44 44	Germany	184.810000	

VistaMarketing 39