

ENTREGABLE: sprint 03

ALUMNO: Ignacio Soldevilla Royo

FECHA: 23-10-2025

NIVEL 1

***** Antes de empezar los ejercicios, añado la tabla user. *****

Primero voy a añadir la estructura de la tabla `user` y cargar sus datos ejecutando los ficheros sql que me proporciona el ejercicio y así la BD `transactions` constará de las tablas `company`, `transaction` y `user`

3 • SHOW TABLES;

Result Grid | Filter Rows:

Tables_in_transactions
company
transaction

Output

#	Time	Action	Message	Duration / Fetch
2	17:36:42	SHOW TABLES	2 row(s) returned	0.000 sec / 0.000 sec

```
1 • CREATE TABLE IF NOT EXISTS user (  
2     id CHAR(10) PRIMARY KEY,  
3     name VARCHAR(100),  
4     surname VARCHAR(100),  
5     phone VARCHAR(150),  
6     email VARCHAR(150),  
7     birth_date VARCHAR(100),  
8     country VARCHAR(150),  
9     city VARCHAR(150),  
10    postal_code VARCHAR(100),  
11    address VARCHAR(255)  
12 );
```

Output

#	Time	Action	Message	Duration / Fetch
4	17:39:06	CREATE TABLE IF NOT EXISTS user (id CHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARCHAR(150), email...	0 row(s) affected	0.094 sec

```
1 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "151", "Meghan", "Hayden", "0888 746 6747", "arcu.vel@hotmail.ca", "Jul 2, 1988", "United Kingdom", "London", "E  
2 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "152", "Hakeem", "Alford", "(0111) 367 0184", "adipiscing.ligula@google.edu", "Sep 30, 1979", "United Kingdom", "  
3 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "153", "Keegan", "Pugh", "(016977) 3851", "sodales.nisi@aol.org", "Jul 27, 1994", "United Kingdom", "London", "E  
4 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "154", "Cooper", "Bullock", "(021) 2521 6627", "et@outlook.net", "Nov 2, 1986", "United Kingdom", "Manchester", "  
5 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "155", "Joshua", "Russell", "055 4409 5286", "justo.nec.ante@outlook.edu", "Jan 23, 1984", "United Kingdom", "War  
6 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "156", "Remedios", "Case", "055 3114 1566", "mollis.phasellus.libero@aol.com", "Oct 9, 1994", "United Kingdom", "  
7 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "157", "Philip", "Carey", "0800 640 6251", "phasellus@yahoo.net", "Oct 10, 1992", "United Kingdom", "Manchester", "  
8 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "158", "Fatima", "Dyer", "0800 1111", "adipiscing@google.org", "Dec 24, 1988", "United Kingdom", "Birmingham", "E  
9 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "159", "Kyllynn", "Acevedo", "056 5727 9602", "dignissim.lacus.aliquam@google.org", "May 10, 2000", "United Kingd  
10 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "160", "Lael", "Woody", "07123 850737", "nunc.sed@aol.org", "Nov 19, 1987", "United Kingdom", "London", "EC1A 1BE
```

Output

#	Time	Action	Message	Duration / Fetch
5037	17:48:55	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ("4983", "Eremdo", "Orekkpbr", "...	1 row(s) affected	0.000 sec

3 • SHOW TABLES;

Result Grid	Filter Rows:
Tables_in_transactions	
company	
transaction	
user	

Añado la relación entre la tabla user (id) con la tabla transaction (user_id) creando una FK en la tabla transaction.

Pero, para crear una relación PK - FK ambos campos deben ser del mismo tipo y el campo user(id) es CHAR(10) y el transaction(user_id) es INT, por lo que primero voy a cambiar en la tabla user el user(id) de CHAR(10) a INT

```
20 • ALTER TABLE user
21 MODIFY COLUMN id INT;
--
```

#	Time	Action	Message	Duration / Fetch
5031	10:37:46	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_user FOREIGN KEY (user_id) REFERENCES user(id)	Error Code: 3780. Referencing column 'user_id' and referenced column 'id' in foreign key constraint 'fk_transaction_user' are incompatible.	0.000 sec
5032	11:02:11	ALTER TABLE user MODIFY COLUMN id INT	5000 row(s) affected Records: 5000 Duplicates: 0 Warnings: 0	0.110 sec

```
23 -- Añado la relacion entre la tabla user (id) con la tabla transaction (user_id) creando una FK en la tabla transaction
24 • ALTER TABLE transaction
25 ADD CONSTRAINT fk_transaction_user
26 FOREIGN KEY (user_id)
27 REFERENCES user(id);
28
```

#	Time	Action	Message	Duration / Fetch
5033	11:06:25	ALTER TABLE user MODIFY COLUMN id INT NULL	Error Code: 1171. All parts of a PRIMARY KEY must be NOT NULL: if you need NULL in a key, use UNIQUE instead	0.000 sec
5034	11:06:38	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_user FOREIGN KEY (user_id) REFERENCES user(id)	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0	2.063 sec

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 ingrese la información del documento denominado "datos_introducir_credit". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

Longitudes de datos clave en una tarjeta bancaria:

Dato	Descripción	Longitud típica
IBAN	Código internacional de cuenta bancaria. Incluye país, control y número de cuenta.	Hasta 34 caracteres alfanuméricos (en España: 24)
PAN	Número de cuenta principal en la tarjeta (número largo en el anverso).	12 a 19 dígitos
PIN	Código personal secreto para autorizar operaciones en cajeros o TPV.	4 dígitos (a veces 6)
CVV / CVC / CID	Código de seguridad para compras online o sin presencia física.	3 dígitos (Visa/Mastercard), 4 dígitos (Amex)
Fecha de expiración	Fecha en que vence la tarjeta, en formato MM/AA o MM/AAAA.	4 a 6 caracteres

1. Con estos datos y los que veo en el fichero "datos_introducir_sprint3_credit.sql" creo la tabla "credit_card"

```

23 • USE transactions;
24
25 -- Creamos la tabla credit_card
26 • CREATE TABLE IF NOT EXISTS credit_card (
27     id VARCHAR(20) PRIMARY KEY,
28     iban VARCHAR(50),
29     pan VARCHAR(50),
30     pin VARCHAR(4),
31     cvv INT,
32     expiring_date VARCHAR(20)
33 );

```

#	Time	Action	Message	Duration / Fetch
2	09:44:43	CREATE TABLE IF NOT EXISTS credit_card (id VARCHAR(20) PRIMARY KEY, iban VARCHAR(50), pan VARCHAR(50), pin V...	0 row(s) affected	0.109 sec

2. Compruebo que esta creada con sus columnas:

```

37 • SHOW COLUMNS FROM credit_card;

```

Field	Type	Null	Key	Default	Extra
id	varchar(20)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pan	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	int	YES		NULL	
expiring_date	varchar(20)	YES		NULL	

#	Time	Action	Message	Duration / Fetch
5	09:50:40	SHOW COLUMNS FROM credit_card	6 row(s) returned	0.000 sec / 0.000 se

3. Cargo los datos de la tabla con el fichero facilitado en el ejercicio:

```

1 -- Insertamos datos de credit_card
2 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2938', 'TR301950312213576817638661', '5424465566813633', '3257', '984', '10/30/22');
3 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2945', 'D026854763748537475216568689', '5142423821948828', '9080', '887', '08/24/23');
4 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2952', 'BG45IVQL52710525608255', '4556 453 55 5287', '4598', '438', '06/29/21');
5 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2959', 'CR7242477244335841535', '372461377349375', '3583', '667', '02/24/23');
6 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2966', 'BG72LKTQ15627628377363', '448566 886747 7265', '4900', '130', '10/29/24');
7 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2973', 'PT87806228135092429456346', '544 58654 54343 384', '8760', '887', '01/30/25');
8 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2980', 'DE39241881883086277136', '402400 7145845969', '5075', '596', '07/24/22');
9 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2987', 'GE89681434837748781813', '3763 747687 76666', '2298', '797', '10/31/23');
10 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-2994', 'BH62714428368066765294', '344283273252593', '7545', '595', '02/28/22');
11 • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-3001', 'CY49087426654774581266832110', '511722 924833 2244', '9562', '867', '09/16/22');

```

#	Time	Action	Message	Duration / Fetch
5019	10:20:43	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (CcU-9580', 'XX78125889851950806677358', '5541182364498931', '927...	1 row(s) affected	0.000 sec

4. Añado la relación entre la tabla credit_card (id) con la tabla transaction (credit_card_id) creando una FK en la tabla transaction

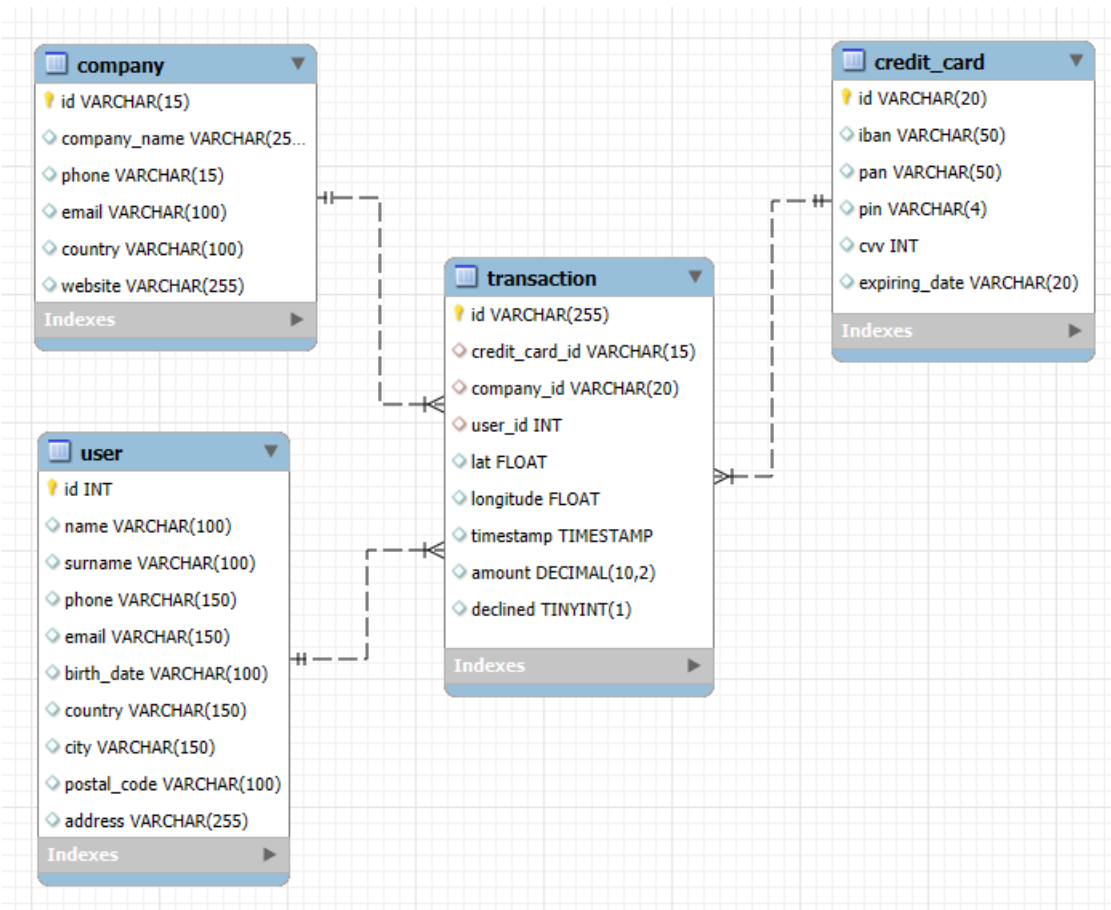
```

41 -- Añado la relacion entre la tabla credit_card (id) con la tabla transaction (credit_card_id) creando una FK en la tabla transaction
42 • ALTER TABLE transaction
43     ADD CONSTRAINT fk_transaction_credit_card
44     FOREIGN KEY (credit_card_id)
45     REFERENCES credit_card(id);

```

#	Time	Action	Message	Duration / Fetch
5024	10:23:39	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card FOREIGN KEY (credit_card_id) REFERENCES credit_card(id)	100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0	1.671 sec

5. Quedándome el siguiente diagrama



Donde vemos una BD en estrella, donde la tabla de hechos es la “transaction” que se relaciona mediante FKs con el resto de tablas de dimensiones. La cardinalidad es en todas 1:N con la tabla de hechos.

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.

1. Visualizo los datos iniciales

```

69  -- Visualizo los datos originales de este ID
70  • SELECT *
71  FROM credit_card
72  WHERE id = "CcU-2938"
73  ;
  
```

Result Grid						
Filter Rows:						
	id	iban	pan	pin	cvv	expiring_date
▶	CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22
*	NULL	NULL	NULL	NULL	NULL	NULL

Output			
#	Time	Action	Message
5039	11:29:38	SELECT * FROM credit_card WHERE id = "CcU-2938"	1 row(s) returned
5040	12:00:30	SELECT * FROM credit_card WHERE id = "CcU-2938"	1 row(s) returned

2. Modifico los datos al valor que me dice el ejercicio (TR323456312213576817699999)

```

79      -- Modifico los datos del registro con ID= "CcU-2938" para el nuevo valor de iban
80 •    UPDATE credit_card
81      SET iban = "TR323456312213576817699999"
82      WHERE id = "CcU-2938"
83      ;

```

Output				
#	Time	Action	Message	Duration / Fetch
5040	12:03:30	SELECT * FROM credit_card WHERE id = "CcU-2938"	1 row(s) returned	0.047 sec / 0.000 sec
5041	12:03:30	UPDATE credit_card SET iban = "TR323456312213576817699999" WHERE id = "CcU-2938"	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.031 sec

3. Visualizo que se ha realizado correctamente el cambio

```

85      -- Visualizo los datos despues del cambio de este ID
86 •    SELECT *
87      FROM credit_card
88      WHERE id = "CcU-2938"
89      ;

```

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

Output				
#	Time	Action	Message	Duration / Fetch
5041	12:03:30	UPDATE credit_card SET iban = "TR323456312213576817699999" WHERE id = "CcU-2938"	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.031 sec
5042	12:05:04	SELECT * FROM credit_card WHERE id = "CcU-2938"	1 row(s) returned	0.000 sec / 0.000 sec

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

1. Compruebo si ya existe o no un registro con el id a crear (108B1D1D-5B23-A76C-55EF-C568E49A99D)

```

103      -- Compruebo que no exista un registro con el ID = 108B1D1D-5B23-A76C-55EF-C568E49A99D nuevo a crear
104 •    SELECT *
105      FROM transaction
106      WHERE id = "108B1D1D-5B23-A76C-55EF-C568E49A99D";

```

Result Grid									
Filter Rows:									
Edit:									
Export/Import:									
Wrap Cell Content:									
id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	

Output			
#	Time	Action	Message
5046	12:15:14	SELECT * FROM transaction WHERE id = "00043A49-2949-4948-A5DD-A5BAE38B190D"	1 row(s) returned
5047	12:15:26	SELECT * FROM transaction WHERE id = "108B1D1D-5B23-A76C-55EF-C568E49A99D"	0 row(s) returned

No existe. Pero antes de crearlo debo comprobar que están creados en las tablas company, user y credit_card la compañía, el usuario y la tarjeta de crédito con las identidades que me da en los datos del ejercicio, y veo que no están definidos ninguno de los tres:

```

108 -- Compruebo que exista el 'credit_card_id' que voy a añadir en la tabla credit_card
109 • SELECT *
110 FROM credit_card
111 WHERE id = "CcU-9999";

```

Result Grid						
Filter Rows:						
Edit:						
id	iban	pan	pin	cvv	expiring_date	
NULL	NULL	NULL	NULL	NULL	NULL	

Output			
#	Time	Action	Message
5054	13:03:11	SELECT * FROM transaction WHERE id = "108B1D1D-5B23-A76C-55EF-C568E49A99D"	0 row(s) returned
5055	13:03:29	SELECT * FROM credit_card WHERE id = "CcU-9999"	0 row(s) returned

```

113 -- Compruebo que exista el 'company_id' que voy a añadir en la tabla company
114 • SELECT *
115 FROM company
116 WHERE id = "b-9999";

```

Result Grid					
Filter Rows:					
Edit:					
id	iban	pan	pin	cvv	expiring_date
NULL	NULL	NULL	NULL	NULL	NULL

Output			
#	Time	Action	Message
5051	12:38:56	SELECT * FROM credit_card WHERE id = "CcU-9999"	0 row(s) returned
5052	12:41:02	SELECT * FROM company WHERE id = "b-9999"	0 row(s) returned

```

118 -- Compruebo que exista el 'user_id' que voy a añadir en la tabla user
119 • SELECT *
120 FROM user
121 WHERE id = "9999";

```

Result Grid										
Filter Rows:										
Edit:										
Export/Import:										
Wrap Cell Content:										
id	name	surname	phone	email	birth_date	country	city	postal_code	address	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	

Output			
#	Time	Action	Message
5067	13:08:06	SELECT * FROM company WHERE id = "b-9999"	0 row(s) returned
5068	13:08:25	SELECT * FROM user WHERE id = "9999"	0 row(s) returned

Por lo tanto, la respuesta al ejercicio es que no se puede crear esta transacción al no estar creados en las tablas relacionadas ni la tarjeta de crédito ni el usuario ni la compañía. También podría inventarme los datos y crear previamente en las tablas correspondientes el usuario, la compañía y la tarjeta de crédito, y así luego si podría introducir la transacción con el comando:

```

INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99D', 'CcU-9999', 'b-9999', '9999', '829.999', '-117.999', '111.11', '0');

```

Ejercicio 4

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

1. Visualizo que existe esta columna 'pan' en la tabla credit_car

```
132 -- Visualizo las columnas de la tabla para comprobar que existe la columna "pan"
133 • SHOW COLUMNS FROM credit_card;
---
```

Field	Type	Null	Key	Default	Extra
id	varchar(20)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pan	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	int	YES		NULL	
expiring_date	varchar(20)	YES		NULL	

#	Time	Action	Message	Duration / Fetch
5070	13:22:08	SHOW COLUMNS FROM credit_card	6 row(s) returned	0.062 sec / 0.000 sec

2. Como si existe, la borro

```
126 -- Borrado de columna pan
127 • ALTER TABLE credit_card DROP COLUMN pan;
```

#	Time	Action	Message	Duration / Fetch
5071	13:29:15	SHOW COLUMNS FROM credit_card	6 row(s) returned	0.016 sec / 0.000 sec
5072	13:33:49	ALTER TABLE credit_card drop column pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.109 sec

3. Visualizo que ya no existe la columna 'pan' para asegurarme que se ha borrado

```
138 -- Visualizo las columnas de la tabla para comprobar que ya NO existe la columna "pan"
139 • SHOW COLUMNS FROM credit_card;
```

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	

#	Time	Action	Message	Duration / Fetch
5072	13:33:49	ALTER TABLE credit_card drop column pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.109 sec
5073	13:35:20	SHOW COLUMNS FROM credit_card	5 row(s) returned	0.016 sec / 0.000 sec

Ejercicio 1

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

1. Primero lo busco para asegurarme que existe y también que la condición de búsqueda es correcta y así aplicarla luego al borrado para borrar lo que quiero borrar.

```
151 -- Primero lo visualizo para asegurarme que esta
152 • SELECT *
153 FROM transaction
154 WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD";
155
```

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	Duration / Fetch
4	18:06:21	SHOW COLUMNS FROM credit_card	5 row(s) returned	0.016 sec / 0.000 sec
5	18:06:31	SELECT * FROM transaction WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) returned	0.000 sec / 0.000 sec

2. Una vez visto que si esta, lo elimino utilizando el WHERE para asegurar q solo se borra este:

```
156 -- Elimino el registro
157 • DELETE FROM transaction
158 WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD";
159
```

#	Time	Action	Message	Duration / Fetch
7	18:16:22	SELECT * FROM transaction WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) returned	0.000 sec / 0.000 sec
8	18:16:41	DELETE FROM transaction WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) affected	0.031 sec

3. Compruebo que ya no existe para asegurarme que se ha borrado

```
160 -- Compruebo que ya no existe para asegurarme que se ha borrado
161 • SELECT *
162 FROM transaction
163 WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD";
164
```

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

#	Time	Action	Message	Duration / Fetch
8	18:16:41	DELETE FROM transaction WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	1 row(s) affected	0.031 sec
9	18:19:18	SELECT * FROM transaction WHERE id = "000447FE-B650-4DCF-85DE-C7ED0EE1CAAD"	0 row(s) returned	0.000 sec / 0.000 sec

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.

1. Creo la vista, entendiendo que el orden descendente NO debe estar incluido dentro de la vista


```

173 -- Creo la vista probando la consulta primero y luego la pongo en la vista.
174 -- Entiendo que la ordenacion descendente NO tiene que estar incluida en la vista
175 • CREATE VIEW VistaMarketing AS
176     SELECT
177         c.company_name AS compañía,
178         c.phone AS telefono_contacto,
179         c.country AS pais_residencia,
180         ROUND(AVG(t.amount),2) AS media_compras
181     FROM company AS c
182     JOIN transaction AS t
183     ON c.id = t.company_id
184     GROUP BY c.company_name, c.phone, c.country
185 ;

```

#	Time	Action	Message	Duration / Fetch
43	12:35:53	SHOW FULL TABLES IN transactions WHERE TABLE_TYPE = 'VIEW'	1 row(s) returned	0.000 sec / 0.000 sec
44	12:37:03	CREATE VIEW VistaMarketing AS SELECT c.company_name AS compañía, c.phone AS telefono_contacto, c.country AS pais_residencia, ROUN...	0 row(s) affected	0.016 sec

2. Visualizo la lista de vistas definidas en la base de datos transactions para ver que esta la que acabo de definir

```

187 -- visualizo la lista de vistas para ver que esta creada
188 • SHOW FULL TABLES IN transactions WHERE TABLE_TYPE = 'VIEW';

```

Result Grid	Filter Rows:
Tables_in_transactions	Table_type
vistamarketing	VIEW

#	Time	Action	Message	Duration / Fetch
38	19:41:26	CREATE VIEW VistaMarketing AS SELECT c.company_name AS compañía, c.phone AS telefono_contacto, c.country AS pais_residencia, ROUN...	0 row(s) affected	0.015 sec
39	19:43:04	SHOW FULL TABLES IN transactions WHERE TABLE_TYPE = 'VIEW'	1 row(s) returned	0.016 sec / 0.000 sec

3. Ejecuto una llamada a esta vista para ver que funciona correctamente

```

190 -- pruebo que funciona la llamada
191 • SELECT *
192 FROM vistamarketing
193 ORDER BY media_compras DESC;

```

compañía	telefono_contacto	pais_residencia	media_compras
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.87
Pretium Neque Corp.	07 77 48 55 28	Australia	276.16
Urna Convallis Associates	06 01 24 77 04	United States	274.24
At Associates	09 56 61 10 65	New Zealand	272.21
Metus Vitae Associates	08 25 44 40 66	Australia	270.08

#	Time	Action	Message	Duration / Fetch
46	12:39:19	SELECT * FROM vistamarketing ORDER BY media_compras	100 row(s) returned	0.312 sec / 0.000 sec
47	12:39:54	SELECT * FROM vistamarketing ORDER BY media_compras DESC	100 row(s) returned	0.329 sec / 0.000 sec

Ejercicio 3

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

Simplemente es ejecutar la consulta poniendo el filtro de 'Germany' en el país de residencia

```

189 • SELECT *
190 FROM vistamarketing
191 WHERE pais_residencia = "Germany"
192 ;

```

Result Grid				
Filter Rows:		Export:	Wrap Cell Content:	
	compañía	telefono_contacto	pais_residencia	media_compras
▶	Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.87
	Nunc Interdum Incorporated	05 18 15 48 13	Germany	259.32
	Convallis In Incorporated	06 66 57 29 50	Germany	257.75
	Ac Industries	09 34 65 40 60	Germany	255.15
	Rutrum Non Inc.	02 66 31 61 09	Germany	255.14
	Auctor Mauris Corp.	05 62 87 14 41	Germany	254.77
	Augue Foundation	06 88 43 15 63	Germany	253.51
	Aliquam PC	01 45 73 52 16	Germany	253.14

Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
12	13:28:35	SELECT * FROM vistamarketing ORDER BY media_compras DESC	100 row(s) returned	0.609 sec / 0.016 sec
13	13:29:20	SELECT * FROM vistamarketing WHERE pais_residencia = "Germany"	8 row(s) returned	0.046 sec / 0.000 sec

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:

Esta estructura de BD ya la he ido dejando así, ejecutando unos previos antes de iniciar los ejercicios y otras en el proceso de los ejercicios. A continuación, describo el orden que he ido ejecutando:

1. Parto inicialmente de una BD que contine las tablas "company" y "transaction", relacionadas entre con cardinalidad 1:N y unidas por la PK comany(id) y la FK transaction(company_id).
2. Defino la estructura de la tabla user ejecutando el fichero [estructura datos user.sql](#) que me da el enunciado del ejercicio.
3. Cargo los datos en la estructura creada ejecutando el fichero [datos introducir sprint3 user.sql](#) que me da el enunciado del ejercicio.
4. Para crear la relación entre la tabla "user" y la "transaction" los campos a utilizar deben ser del mismo tipo, estos campos son user(id) que es CHAR(10) y el transaction(user_id) es INT, por lo que modifico en la tabla "user" el campo "id" y lo paso den tipo CHAR(10) y así será del mismo tipo que el campo "company_id" de la tabla "transaction" y lo hago con este comando

ALTER TABLE user

MODIFY COLUMN id INT;

5. Ahora ya puedo crear la FK en la tabla "transaction" relacionada con la PK de la tabla "user" con este comando:

ALTER TABLE transaction

ADD CONSTRAINT fk_transaction_user

FOREIGN KEY (user_id)

REFERENCES user(id);

6. Defino la estructura de la tabla “credit_card”. Para ello edito el fichero [datos_introducir_sprint3_credit.sql](#) que me da el enunciado del ejercicio y de allí obtengo todos los nombres exactos de las columnas que debe tener la tabla. Con estos datos monto el fichero que definirá la tabla “credit_card” y le asigno la PK al campo o columna “id” de la siguiente manera:


```
CREATE TABLE IF NOT EXISTS credit_card (  
    id VARCHAR(20) PRIMARY KEY,  
    iban VARCHAR(50),  
    pan VARCHAR(50),  
    pin VARCHAR(4),  
    cvv INT,  
    expiring_date VARCHAR(20)  
);
```

7. Cargo los datos en la estructura creada ejecutando el fichero [datos_introducir_sprint3_credit.sql](#) que me da el enunciado del ejercicio.
8. Hay que definir la relación entre las tablas “credit_card” (id) y PK con la tabla “transaction” (credit_card_id) creando una FK en la tabla “transaction” que las relacione, llamándola “fk_transaction_credit_card”. Lo hago así:

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

9. Y con todo lo anterior ya me queda la tabla de la imagen, que se trata de una BD, siendo la tabla de hechos la “transaction” y las tablas de dimensiones “company”, “user” y “credit_card”. Todas tienen una cardinalidad de 1:N con la tabla de hechos.
10. Me doy cuenta que hay una columna o campo nuevo en la tabla “credit_card” llamado “fecha_actual” del tipo DATE. Así, que lo debo añadir así:
- a. Visualizo los campos actuales antes de definir el nuevo

```
213 -- Visualizo los campos de la tabla antes de añadir el nuevo campo  
214 • DESCRIBE credit_card;
```

Result Grid						
Filter Rows:						
Export:  Wrap Cell Cor						
	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	

Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
8	10:31:38	DESCRIBE credit_card	5 row(s) returned	0.016 sec / 0.000 sec

- b. Añado a la tabla “credit_card” la comuna “fecha_actual” del tipo DATE

```
216 -- Añado el nuevo campo o columna  
217 • ALTER TABLE credit_card ADD fecha_actual DATE;  
218
```

Output			
#	Time	Action	Message
8	10:31:38	DESCRIBE credit_card	5 row(s) returned
9	10:34:12	ALTER TABLE credit_card ADD fecha_actual DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

c. Visualizo que se ha creado correctamente

```
219 -- Visualizo los campos de la tabla para comprobar que se ha añadido el nuevo campo
220 • DESCRIBE credit_card;
```

Field	Type	Null	Key	Default	Extra
id	varchar(20)	NO	PRI	NONE	
iban	varchar(50)	YES		NONE	
pin	varchar(4)	YES		NONE	
cvv	int	YES		NONE	
expiring_date	varchar(20)	YES		NONE	
fecha_actual	date	YES		NONE	

Output			
#	Time	Action	Message
9	10:34:12	ALTER TABLE credit_card ADD fecha_actual DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
10	10:38:27	DESCRIBE credit_card	6 row(s) returned

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.

1. Hago primero la consulta que contendrá la lista para asegurarme que contine lo que me piden, y una vez la tengo, creo la vista que contendrá esta consulta. Entendiendo que en la vista no debe estar incluida la ordenación y que esta se hará con la llamada a ella:

```

228 -- Creo la vista probando la consulta primero y luego la pongo en la vista.
229 -- Entiendo que la ordenacion descendente NO tiene que estar incluida en la vista
230 • CREATE VIEW InformeTecnico AS
231     SELECT
232         t.id AS id_transaccion,
233         u.name AS nombre_usuario,
234         u.surname AS apellido_usuario,
235         cc.iban AS iban_tarjeta,
236         c.company_name AS compañía
237     FROM transaction AS t
238     JOIN user AS u
239     ON t.user_id = u.id
240     JOIN credit_card AS cc
241     ON t.credit_card_id = cc.id
242     JOIN company AS c
243     ON t.company_id = c.id
244 ;

```

#	Time	Action	Message	Duration / Fetch
32	12:18:06	SHOW FULL TABLES IN transactions WHERE TABLE_TYPE = 'VIEW'	1 row(s) returned	0.016 sec / 0.000 sec
33	12:20:32	CREATE VIEW InformeTecnico AS SELECT t.id AS id_transaccion, u.name AS nombre_usuario, u.surname AS apellido_usuario, cc.iban AS iban_tarjeta, c.company_name AS compañía	0 row(s) affected	0.047 sec

2. Visualizo la lista de vistas para ver que ya está incluida la vista “InformeTecnico” que acabo de crear

```

243 -- visualizo la lista de vistas para ver que SI esta creada la vista InformeTecnico
244 • SHOW FULL TABLES IN transactions WHERE TABLE_TYPE = 'VIEW';

```

Result Grid	Filter Rows:	Export:
Tables_in_transactions	Table_type	
informetecnico	VIEW	
vistamarketing	VIEW	

#	Time	Action	Message	Duration / Fetch
33	12:20:32	CREATE VIEW InformeTecnico AS SELECT t.id AS id_transaccion, u.name AS nombre_usuario, u.surname AS apellido_usuario, cc.iban AS iban_tarjeta, c.company_name AS compañía	0 row(s) affected	0.047 sec
34	12:23:42	SHOW FULL TABLES IN transactions WHERE TABLE_TYPE = 'VIEW'	2 row(s) returned	0.000 sec / 0.000 sec

3. Llamo a la vista para ver que funciona y ordeno los resultados de forma descendente según el id de la tabla “transaction” que lo he llamado en la vista “id_transaccion”

```

248 -- pruebo que funciona la llamada y que la ordena descendente por el id de la tabla transaction
249 • SELECT *
250 FROM informetecnico
251 ORDER BY id_transaccion;
--

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
id_transaccion	nombre_usuario	apellido_usuario	iban_tarjeta	compañía
00043A49-2949-494B-A5DD-A5BAE3BB19DD	Gxnmjn	Fjycossj	XX691799023254713153661899	Eget Tincidunt Dui Institute
00045D6B-ED2E-4F2F-8186-CEE074D875D0	Wolake	Ukynumly	XX715219809262791935275891	Neque Tellus Imperdiet Corp.
000481C3-1C26-4FEF-83A0-4CD0EB004BBD	Umhwioi	Ubmdxvmz	XX785022829713140880782522	Fusce Corp.
00051AA4-9CBE-4268-B070-C38062A1B3E2	Qpfyfa	Mtnpfdfq	XX634626564536875934323485	Mus Aenean Eget Foundation
0008A312-EDFE-4A4F-BC99-E9C92EC3CA4D	Keegan	Watson	RO76DAFO6583348580208155	Aliquam Iaculis Lacus Corp.
0009A151-9BCF-4E31-9053-A468FF77FAAB	Rreqrc	Jfdvcgne	XX986535778187638923658336	Lorem Ipsum Dolor Corp.
0009D494-6245-4DF9-955D-2C084191CFFB	Novmmv	Boznhdmn	XX738074164311515717218547	Amet Luctus Vulputate Foundation
000A1DEC-CDB6-4AB2-A619-71DA88D4A262	Hxzzri	Mzchxymz	XX714741860105067529287700	Nulla Integer Vulputate Corp.
000A1E64-1414-40B0-9D92-5678A4D958E2	Ghsngl	Gkwhelrz	XX343833941783938653185254	Auctor Mauris Corp.
000A5879-3472-41D9-AF60-42D3503B543C	Lucy	Branch	DK3216331269627227	Euismod Mauris Institute

#	Time	Action	Message	Duration / Fetch
37	12:28:24	SELECT * FROM informetecnico ORDER BY id_transaccion	99999 row(s) returned	0.610 sec / 0.031 sec
38	12:29:31	SELECT * FROM informetecnico ORDER BY id_transaccion	99999 row(s) returned	0.625 sec / 0.047 sec

