

En primer lugar he creado la base de datos, en este caso la llamaré ventas.

CREATE DATABASE ventas;

1 • CREATE DATABASE ventas;

Output			
Action Output			
#	Time	Action	Message
1	13:39:39	CREATE DATABASE ventas	1 row(s) affected

Mejor usar varchar's extensos, pero tampoco es necesario excederse y usar el máximo.

USE ventas;

```
CREATE TABLE companies(  
  company_ID varchar(7),  
  company_name varchar (100),  
  phone varchar(15),  
  email varchar (100),  
  country varchar (50),  
  website varchar (80),  
  primary key(company_ID)  
);
```

```
1  USE ventas;  
2  • CREATE TABLE companies(  
3    company_ID varchar(7),  
4    company_name varchar (100),  
5    phone varchar(15),  
6    email varchar (100),  
7    country varchar (50),  
8    website varchar (80),  
9    primary key(company_ID)  
10 );  
11
```

Output			
Action Output			
#	Time	Action	Message
1	13:40:06	CREATE TABLE companies(company_ID varchar(7), company_name varchar (100), phone varchar(15), email va...	Error Code: 1046. No database selected Select the defau
2	13:40:12	USE ventas	0 row(s) affected
3	13:40:34	CREATE TABLE companies(company_ID varchar(7), company_name varchar (100), phone varchar(15), email va...	0 row(s) affected

Hayamos el directorio donde alojar los archivos con el código:

```
SELECT @@secure_file_priv;
```

Una vez creada la tabla cargamos los datos con el siguiente código:

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

```
1 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv'
2 INTO TABLE companies
3 FIELDS TERMINATED BY ','
4 ENCLOSED BY '"'
5 LINES TERMINATED BY '\n'
6 IGNORE 1 ROWS;
7
```

Output			
Action Output			
#	Time	Action	Message
1	13:41:09	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv' INTO TABLE compa...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0

En la tabla credit_card uso la misma metodología anterior. Los nombres de las columnas serán diferentes, y el código para cargar la tabla cambiará el nombre de tabla y archivo.

```
create table credit_card (
ID varchar (10),
user_ID int ,
IBAN varchar (50),
PAN varchar (50),
PIN varchar (4),
CVV varchar (3),
TRACK1 varchar (100),
TRACK2 varchar (100),
expiring_date varchar(30),
primary key (ID)
);
```

```

1 • create table credit_card (
2     ID varchar      (10),
3     user_ID int      ,
4     IBAN varchar     (50),
5     PAN varchar      (50),
6     PIN varchar       (4),
7     CVV varchar       (3),
8     TRACK1 varchar   (100),
9     TRACK2 varchar   (100),
10    expiring_date varchar(30),
11    primary key (ID)
12 );

```

Output

Action Output

#	Time	Action	Message
1	13:44:31	create table credit_card (ID varchar (10), user_ID int , IBAN varchar (50), PAN varchar (50), PIN varchar (4)...	0 row(s) affected

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/credit_cards.csv'
INTO TABLE credit_card
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;

```

```

1 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv'
2 INTO TABLE credit_card
3 FIELDS TERMINATED BY ','
4 ENCLOSED BY '"'
5 LINES TERMINATED BY '\n'
6 IGNORE 1 ROWS;
7
8
9

```

Output

Action Output

#	Time	Action	Message
1	13:47:12	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv' INTO TABLE credit...	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0

Para la tabla products:

```
create table products (  
  ID varchar          (10),  
  product_name varchar (50),  
  price varchar       (10),  
  colour varchar      (20),  
  weight decimal (10,2),  
  warehouse_ID varchar (10),  
  primary key (ID)  
);
```

```
1 • create table products (  
2   ID varchar          (10),  
3   product_name varchar (50),  
4   price varchar       (10),  
5   colour varchar      (20),  
6   weight decimal (10,2),  
7   warehouse_ID varchar (10),  
8   primary key (ID)  
9   );  
10
```

Output				
Action Output				
#	Time	Action	Message	
✓ 1	17:57:48	create table products (ID varchar (10), product_name varchar (50), price varchar (10), colour varchar ...	0 row(s) affected	

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server  
8.0/Uploads/products.csv'  
INTO TABLE products  
FIELDS TERMINATED BY ','  
ENCLOSED BY ''''  
LINES TERMINATED BY '\n'  
IGNORE 1 ROWS;
```

```

1 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv'
2 INTO TABLE products
3 FIELDS TERMINATED BY ','
4 ENCLOSED BY '"'
5 LINES TERMINATED BY '\n'
6 IGNORE 1 ROWS;
7

```

Output			
Action Output			
#	Time	Action	Message
1	13:48:30	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv' INTO TABLE products ...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0

En este caso quiero que el tipo de dato de la columna price sea float con dos decimales.
Para eso he de borrar todos los valores “\$” de los registros de la columna price.

Primero desactivo el modo seguro para realizar cambios masivos:

```
SET SQL_safe_updates = 0;
```

```

1 • SET SQL_safe_updates = 0;
2

```

Output			
Action Output			
#	Time	Action	Message
1	13:49:10	SET SQL_safe_updates = 0	0 row(s) affected

A continuación hago el update

```
UPDATE products
SET
price = REPLACE (price, "$", " ");
```

```

1 • UPDATE products
2 SET
3 price = REPLACE (price, "$", " ");
4

```

Output			
Action Output			
#	Time	Action	Message
1	13:49:43	UPDATE products SET price = REPLACE (price, "\$", " ")	100 row(s) affected Rows matched: 100 Changed: 100 Warnings: 0

Activo el modo seguro de nuevo

```
SET SQL_safe_updates = 1;
```

1	•	SET SQL_safe_updates = 1;
2		

Output			
Action Output			
#	Time	Action	Message
1	13:50:17	SET SQL_safe_updates = 1	0 row(s) affected

Por último modifico la el tipo de dato de la columna con la siguiente query:

```
ALTER TABLE products  
MODIFY price float (10,2);
```

1	•	ALTER TABLE products
2		MODIFY price float (10,2);
3		

Output			
Action Output			
#	Time	Action	Message
1	13:50:39	ALTER TABLE products MODIFY price float (10,2)	100 row(s) affected, 1 warning(s): 1681 Specifying number of digits for floating point data types is deprecated and ...

La tabla users ha sido la más complicada, ya que en el comando LOAD DATA, en la sección de LINES TERMINATED BY hay que modificarlo por '\r\n' para que pueda leer el archivo correctamente.

Cargué los tres archivos de usuarios en una única tabla.

Las queries fueron las siguientes:

```
CREATE TABLE users (  
    ID INT PRIMARY KEY AUTO_INCREMENT,  
    name VARCHAR(50),  
    surname VARCHAR(50),  
    phone VARCHAR(15),  
    email VARCHAR(50),  
    birth_date VARCHAR(20),  
    country VARCHAR(20),  
    city VARCHAR(50),  
    postal_code VARCHAR(10),  
    address VARCHAR(100)  
);
```

```

1 CREATE TABLE users (
2     ID INT PRIMARY KEY AUTO_INCREMENT,
3     name VARCHAR(50),
4     surname VARCHAR(50),
5     phone VARCHAR(15),
6     email VARCHAR(50),
7     birth_date VARCHAR(20),
8     country VARCHAR(20),
9     city VARCHAR(50),
10    postal_code VARCHAR(10),
11    address VARCHAR(100)
12 );
13

```

Output				
Action Output				
#	Time	Action	Message	
✓ 1	18:00:37	CREATE TABLE users (ID INT PRIMARY KEY AUTO_INCREMENT, name VARCHAR(50), surname VARC...	0 row(s) affected	

Uso esta query tres veces, ya que hay tres archivos de usuarios diferentes.

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/users_usa.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(id, name, surname, phone, email, birth_date, country, city, postal_code, address);

```

```

1 LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv'
2 INTO TABLE users
3 FIELDS TERMINATED BY ','
4 OPTIONALLY ENCLOSED BY '"'
5 LINES TERMINATED BY '\r\n'
6 IGNORE 1 LINES
7 (id, name, surname, phone, email, birth_date, country, city, postal_code, address);
8

```

Output				
Action Output				
#	Time	Action	Message	
✓ 1	13:54:01	CREATE TABLE users (ID int, name varchar (50), sumame varchar (50), phone varchar (15), email varchar (50), ...	0 row(s) affected	
✗ 2	13:54:26	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv' INTO TABLE users_p...	Error Code: 1146. Table 'ventas.users_prueba' doesn't exist	
✓ 3	13:54:32	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv' INTO TABLE users FI...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0	

El siguiente paso es añadir en la tabla credit_card la foreign key con referencia a la tabla users. (PASO ANTERIOR)

```
ALTER TABLE credit_card  
ADD FOREIGN KEY (user_ID) REFERENCES users(ID);
```

Por ultimo, para crear la tabla transactions usé la siguiente query, modificando el campo FIELDS TERMINATED BY por ;

```
CREATE TABLE transaction (  
ID varchar (100),  
card_ID varchar (20),  
business_ID varchar (10),  
date timestamp,  
amount decimal (10,2),  
declined tinyint,  
products varchar (50),  
user_ID int,  
lat float,  
longitude float,  
PRIMARY KEY (ID)  
);
```

```
1 CREATE TABLE transaction (  
2 ID varchar (100),  
3 card_ID varchar (20),  
4 business_ID varchar (10),  
5 date timestamp,  
6 amount decimal (10,2),  
7 declined tinyint,  
8 products varchar (50),  
9 user_ID int,  
10 lat float,  
11 longitude float,  
12 PRIMARY KEY (ID)  
13 );  
14
```

Output			
Action Output			
#	Time	Action	Message
1	18:03:06	CREATE TABLE transaction (ID varchar (100), card_ID varchar (20), business_ID varchar (10), date timestamp...	0 row(s) affected


```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/transactions.csv'
INTO TABLE transaction
FIELDS TERMINATED BY ';'
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

```
1 • LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv'
2 INTO TABLE transaction
3 FIELDS TERMINATED BY ';'
4 ENCLOSED BY '"'
5 LINES TERMINATED BY '\n'
6 IGNORE 1 ROWS;
7
```

Output			
Action Output			
#	Time	Action	Message
1	14:11:52	CREATE TABLE transaction (ID varchar (100), card_ID varchar (20), business_ID varchar (10), date timesta...	0 row(s) affected
2	14:13:11	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv' INTO TABLE trans...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0

Añadir las siguientes foreign keys:

```
ALTER TABLE transaction
ADD FOREIGN KEY (card_ID) REFERENCES credit_card(id);
```

```
1 ALTER TABLE transaction
2 ADD FOREIGN KEY (card_ID) REFERENCES credit_card(id);
```

Output			
Action Output			
#	Time	Action	Message
1	14:21:05	ALTER TABLE transaction ADD FOREIGN KEY (card_ID) REFERENCES credit_card(id)	260 row(s) affected Records: 260 Duplicates: 0 Warnings: 0

ALTER TABLE transaction

ADD FOREIGN KEY (business_ID) REFERENCES companies(company_ID);

```
1 • ALTER TABLE transaction
2   ADD FOREIGN KEY (business_ID) REFERENCES companies(company_ID);
```

Output				
Action Output				
#	Time	Action	Message	
✓ 1	14:21:05	ALTER TABLE transaction ADD FOREIGN KEY (card_ID) REFERENCES credit_card(id)	260 row(s) affected Records: 260 Duplicates: 0 Warnings: 0	
✓ 2	14:21:29	ALTER TABLE transaction ADD FOREIGN KEY (business_ID) REFERENCES companies(company_ID)	260 row(s) affected Records: 260 Duplicates: 0 Warnings: 0	

Elimino la relación entre users y credit_card

ALTER table credit_card

drop constraint credit_card_ibfk_1;

```
1 • ALTER table credit_card
2   drop constraint credit_card_ibfk_1;
```

Output				
Action Output				
#	Time	Action	Message	
✗ 1	18:06:54	ALTER TABLE credit_card DROP foreign key user_id	Error Code: 1091. Can't DROP 'user_id'; check that column/key exists	
✓ 2	18:07:30	SHOW CREATE TABLE credit_card	1 row(s) returned	
✓ 3	18:08:09	ALTER table credit_card drop constraint credit_card_ibfk_1	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	

Añado la relación entre users y transaction para que finalmente quede como un modelo de estrella:

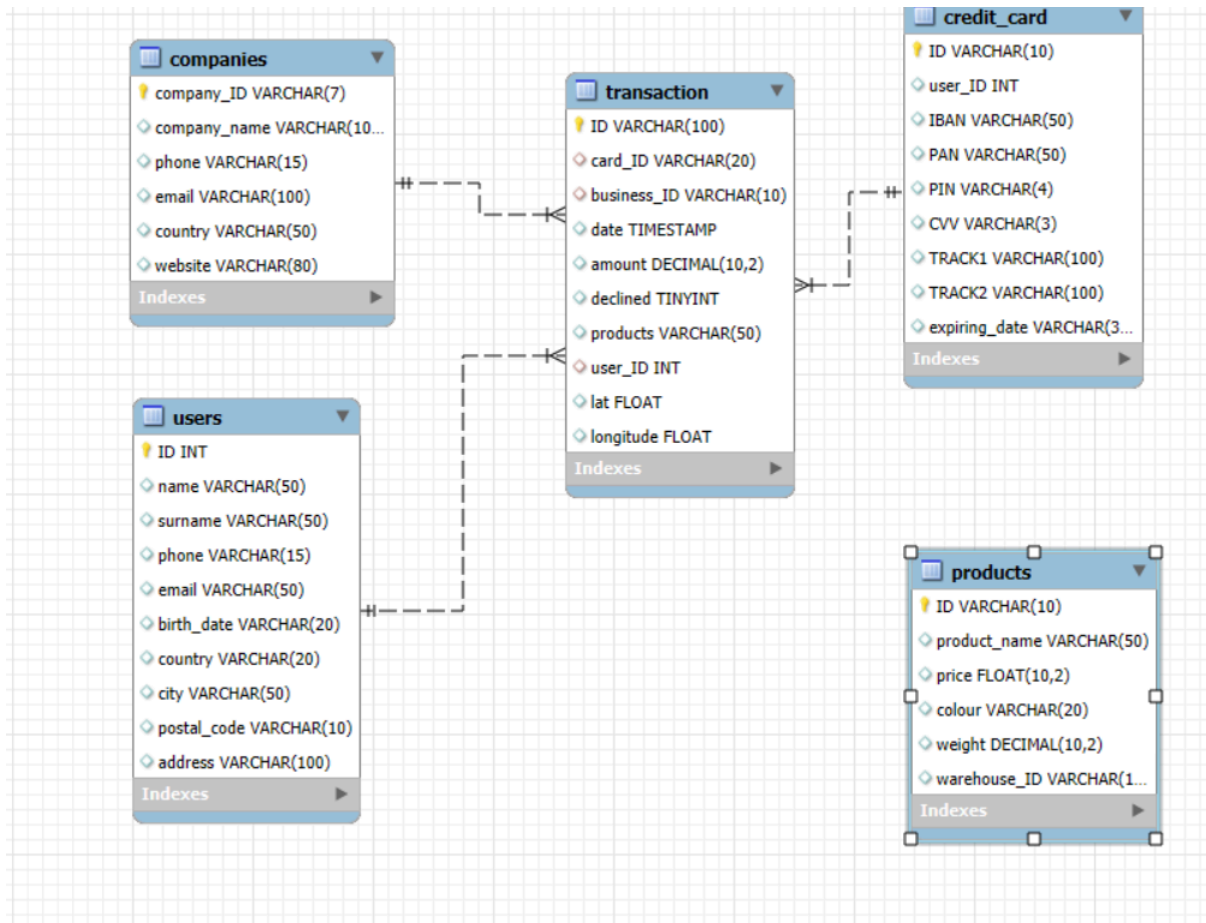
ALTER TABLE transaction

ADD FOREIGN KEY (user_id) REFERENCES users(ID);

```
1 • ALTER TABLE transaction
2   ADD FOREIGN KEY (user_id) REFERENCES users(ID);
3
```

Output				
Action Output				
#	Time	Action	Message	
✓ 1	18:10:30	ALTER TABLE transaction ADD FOREIGN KEY (user_id) REFERENCES users(ID)	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	

La tabla products no es posible relacionarla aún, debido a la disposición de los registros del campo products en la tabla transaction



Nivell 1

- Exercici 1

Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.

En la subconsulta realitzo el càlcul per a trobar els usuaris amb més de 30 transaccions, devolvent els dades dels mateixos en la consulta principal.

USE ventas;

```
SELECT *  
FROM users  
WHERE id IN (  
    SELECT user_id  
    FROM transaction  
    GROUP BY user_id  
    HAVING COUNT(id) > 30  
);
```

```
1 • SELECT *  
2 FROM users  
3 WHERE id IN (  
4     SELECT user_id  
5     FROM transaction  
6     GROUP BY user_id  
7     HAVING COUNT(id) > 30  
8 ) ;  
9
```

Result Grid										
Filter Rows:										
Edit: Export/Import: Wrap Cell Content:										
	ID	name	surname	phone	email	birth_date	country	city	postal_code	address
▶	92	Lynn	Riddle	1-387-885-4057	vitae.aliquet@outlook.edu	Sep 21, 1984	United States	Bozeman	61871	P.O. Box 712, 7907 Est St.
	267	Ocean	Nelson	079-481-2745	aenean@yahoo.com	Dec 26, 1991	Canada	Charlottetown	85X 3P4	Ap #732-8357 Pede, Rd.
	272	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	Apr 16, 1991	Canada	Tuktoyaktuk	Q4C 3G7	P.O. Box 496, 5145 Sapien I
	275	Kenyon	Hartman	082-871-7248	convallis.ante.lectus@yahoo.com	Aug 3, 1982	Canada	Richmond	R8H 2K2	8564 Facili. St.
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

users 15 x

Output

Action Output

#	Time	Action	Message
1	18:25:26	SELECT * FROM users WHERE id IN (SELECT user_id FROM transaction GROUP BY user_id HAVI...	4 row(s) returned

- Exercici 2

Mostra la mitjana d'amount per IBAN de les targetes de crèdit a la companyia Donec Ltd, utilitza almenys 2 taules.

En la siguiente query selecciono el IBAN de las tarjetas de credito. Realizo las joins necesarias para acceder a todos los datos. Filtro por la compañía Donec Ltd y también excluyo las operaciones declinadas.

```
SELECT IBAN, ROUND(AVG(amount),2) AS media
FROM credit_card cc
      JOIN transaction t
      ON cc.id = t.card_id
      JOIN companies c
      ON t.business_ID = c.company_ID
WHERE company_name = 'Donec Ltd'
AND declined = 0
GROUP BY IBAN;
```

The screenshot shows a database query interface. The top section displays the SQL query: `SELECT IBAN, ROUND(AVG(amount),2) AS media FROM credit_card cc JOIN transaction t ON cc.id = t.card_id JOIN companies c ON t.business_ID = c.company_ID WHERE company_name = 'Donec Ltd' AND declined = 0 GROUP BY IBAN;`. Below the query, there is a 'Result Grid' section with a table showing the results. The table has two columns: 'IBAN' and 'media'. The first row shows the IBAN 'PT87806228135092429456346' and the media value '42.82'. Below the 'Result Grid' section, there is a 'Result 16' section with a tab labeled 'Output'. The 'Output' section shows a log of actions. The first action is a query that returns 4 rows. The second action is the query shown in the 'Result Grid' section, which returns 1 row.

IBAN	media
PT87806228135092429456346	42.82

#	Time	Action	Message
1	18:25:26	SELECT * FROM users WHERE id IN (SELECT user_id FROM transaction GROUP BY user_id HAVI...	4 row(s) returned
2	18:27:13	SELECT IBAN, ROUND(AVG(amount),2) AS media FROM credit_card cc JOIN transaction t ON cc.id = t.card...	1 row(s) returned

Nivell 2

Crea una nova taula que reflecteixi l'estat de les targetes de crèdit basat en si les últimes tres transaccions van ser declinades i genera la següent consulta:

Para esta query utilizo caso para que cuando la suma de los declined sea tres, devuelta el estado como “tarjeta inactiva”. Con row number y partition creo subgrupos o particiones las ultimas 3 fechas de cada card id.

```
CREATE TABLE credit_card_estado AS
SELECT card_id, SUM(declined) AS suma_declinados,
       CASE
         WHEN SUM(declined) >= 3 THEN 'Tarjeta inactiva'
         ELSE 'Tarjeta activa'
       END AS tarjeta_estado
FROM (
  SELECT card_id, business_ID, date, amount, declined,
         ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY date DESC) AS
numeroFila
  FROM
    transaction
) AS ranked_transactions
WHERE
  numeroFila <= 3
GROUP BY
  card_id;
```

```
1 • CREATE TABLE credit_card_estado AS
2   SELECT card_id, SUM(declined) AS suma_declinados,
3     CASE
4       WHEN SUM(declined) = 3 THEN 'Tarjeta inactiva'
5       ELSE 'Tarjeta activa'
6     END AS tarjeta_estado
7   FROM (
8     SELECT card_id, business_ID, date, amount, declined,
9           ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY date DESC) AS numeroFila
10    FROM
11      transaction
12  ) AS ranked_transactions
13  WHERE
14    numeroFila <= 3
15  GROUP BY
16    card_id;
17
```

Output

#	Time	Action	Message
1	14:54:54	CREATE TABLE credit_card_estado AS SELECT card_id, SUM(declined) AS suma_declinados, CASE WHEN...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

Exercici 1

Quantes targetes estan actives?

Con un contejo de los id de las tarjetas activas obtengo el resultado. No hay ninguna tarjeta desactivada porque ninguna de ella tiene las últimas tres transacciones declinadas.

```
SELECT Count(card_id)
FROM credit_card_estado
WHERE tarjeta_Estado = 'Tarjeta activa';
```

```
1 • SELECT Count(card_id)
2 FROM credit_card_estado
3 WHERE tarjeta_Estado = 'Tarjeta activa';
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid has two columns: 'Count(card_id)' and a value '275'. Above the grid, there are controls for 'Filter Rows', 'Export', and 'Wrap Cell Content'.

Count(card_id)
275

Para añadir la relacion entre las tablas credit card y la resultante con el estado de las mismas (son la misma tabla así que es una relacion 1 -1) uso las siguientes queries:

```
ALTER TABLE credit_card_estado
```

```
ADD PRIMARY KEY (card_id);
```

```
ALTER TABLE credit_card_estado
```

```
ADD FOREIGN KEY (card_id) REFERENCES credit_card(ID);
```

```

18 • ALTER TABLE credit_card_estado
19   ADD PRIMARY KEY (card_id);
20
21 • ALTER TABLE credit_card_estado
22   ADD FOREIGN KEY (card_id) REFERENCES credit_card(ID);
23

```

Output			
Action Output			
#	Time	Action	Message
✓ 16	10:26:02	ALTER TABLE credit_card_estado ADD PRIMARY KEY (card_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
✓ 17	10:28:39	ALTER TABLE credit_card_estado ADD FOREIGN KEY (card_id) REFERENCES credit_card(ID)	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

Nivell 3

Crea una taula amb la qual puguem unir les dades del nou arxiu products.csv amb la base de dades creada, tenint en compte que des de transaction tens product_ids. Genera la següent consulta:

Exercici 1

Necessitem conèixer el nombre de vegades que s'ha venut cada producte.

Desactivo modo seguro

```
SET SQL_safe_updates = 0;
```

Actualizo con UPDATE y REPLACE los espacios después de la coma que separa los id de la columna products.

UPDATE transaction

```
SET products = REPLACE(products, ',', '');
```

```

1  UPDATE transaction
2  SET products = REPLACE(products, ',', '');

```

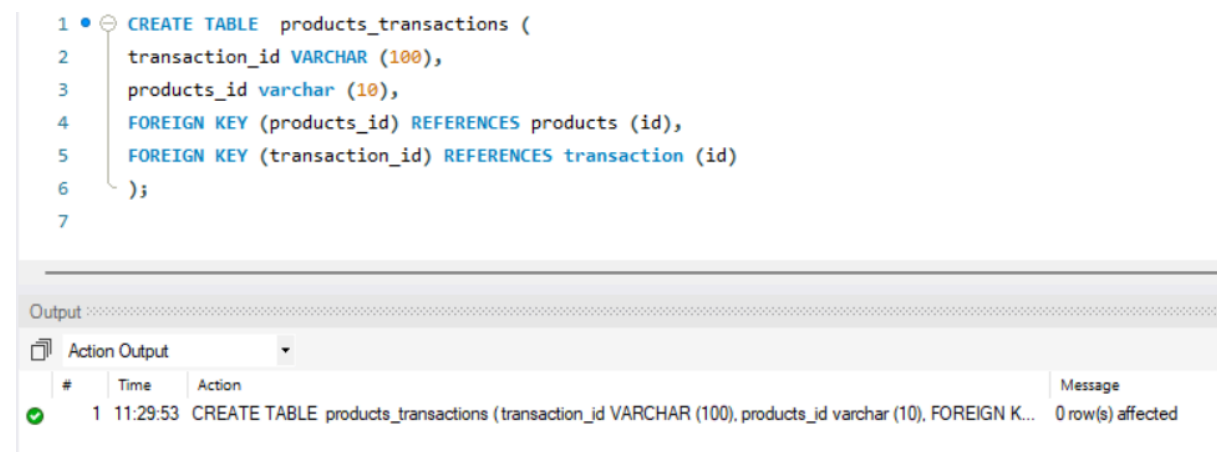
Output			
Action Output			
#	Time	Action	Message
✓ 1	10:36:29	UPDATE transaction SET products = REPLACE(products, ',', '')	0 row(s) affected Rows matched: 587 Changed: 0 Warnings: 0

Activo modo seguro nuevamente

```
SET SQL_safe_updates = 1;
```

Creo la nueva tabla donde en el paso siguiente separaré los ID de cada producto en los ID de las transacciones.

```
CREATE TABLE products_transactions (  
transaction_id VARCHAR (100),  
products_id varchar (10),  
FOREIGN KEY (products_id) REFERENCES products (id),  
FOREIGN KEY (transaction_id) REFERENCES transaction (id)  
);
```



The screenshot shows a SQL IDE with a query editor and an output window. The query editor contains the following SQL code:

```
1 • CREATE TABLE products_transactions (  
2   transaction_id VARCHAR (100),  
3   products_id varchar (10),  
4   FOREIGN KEY (products_id) REFERENCES products (id),  
5   FOREIGN KEY (transaction_id) REFERENCES transaction (id)  
6 );  
7
```

The output window shows the execution result:

#	Time	Action	Message
✓ 1	11:29:53	CREATE TABLE products_transactions (transaction_id VARCHAR (100), products_id varchar (10), FOREIGN K...	0 row(s) affected

Con el siguiente código inserto los datos mencionados anteriormente. Por cada product ID habrá un ID de transacción, siempre y cuando haya más de un product ID:

```
INSERT INTO products_transactions (transaction_id, products_id)  
SELECT transaction.id AS transaction_id, products.id AS product_id  
FROM transaction  
JOIN products  
ON FIND_IN_SET(products.id, transaction.products) > 0;
```

```

20 • INSERT INTO products_transactions (transaction_id, products_id)
21 SELECT transaction.id AS transaction_id, products.id AS product_id
22 FROM transaction
23 JOIN products
24     ON FIND_IN_SET(products.id, transactions.products) > 0;
25

```

Output			
Action Output			
#	Time	Action	Message
8	11:16:36	INSERT INTO products_transactions SELECT transaction.id AS transaction_id, products.id AS product_id FR...	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0
9	11:16:44	SELECT * FROM ventas_products_transactions	1457 row(s) returned

```

7 • ALTER TABLE products_transactions
8 ADD FOREIGN KEY (transaction_id) REFERENCES transaction (id);

```

Output			
Action Output			
#	Time	Action	Message
1	11:20:51	ALTER TABLE products_transactions ADD FOREIGN KEY (transaction_id) REFERENCES transaction (id)	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0

Con la siguiente query hago un recuento de las veces que se ha vendido cada producto, y una join con products para conocer el nombre de cada artículo. Ordeno por el recuento ascendente para mejor visibilidad.

```

SELECT product_name, COUNT(transaction_id) AS recuento_ventas_producto
FROM products_transactions pt
JOIN products p
ON pt.products_id = p.id
GROUP BY 1
ORDER BY recuento_ventas_producto asc;

```

```

1 • SELECT product_name, COUNT(transaction_id) AS recuento_ventas_producto
2 FROM products_transactions pt
3 JOIN products p
4 ON pt.products_id = p.id
5 GROUP BY 1
6 ORDER BY recuento_ventas_producto asc;

```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	product_name	recuento_ventas_producto
▶	Lannister	47
	Dorne bastard	47
	Karstark Dorne	48
	dooku solo	49
	Tully maester Tarly	49

Result 3 x

Output




 Action Output

#	Time	Action	Message
✓ 1	11:34:42	SELECT product_name, COUNT(transaction_id) AS recuento_ventas_producto FROM products_transactions pt...	24 row(s) returned

```

4 • SELECT product_name, COUNT(transaccion) AS recuento_ventas_producto
5 FROM productos2 p2
6 JOIN products p1
7 ON p2.producto = p1.id
8 GROUP BY 1
9 ORDER BY recuento_ventas_producto asc;

```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	product_name	recuento_ventas_producto
▶	Lannister	47
	Dorne bastard	47
	Karstark Dorne	48
	dooku solo	49
	Tully maester Tarly	49
	duel tourney Lannister	51
	Direwolf Littlefinger	51
	Lannister Barratheon Direwolf	53
	Tully Dorne	54
	north of Casterly	54

Result 13 x

Output

 Action Output

#	Time	Action	Message
✓ 1	10:57:53	SELECT product_name, COUNT(transaccion) AS recuento_ventas_producto FROM productos2 p2 JOIN produ...	24 row(s) returned
✓ 2	10:59:12	SELECT product_name, COUNT(transaccion) AS recuento_ventas_producto FROM productos2 p2 JOIN produ...	24 row(s) returned

