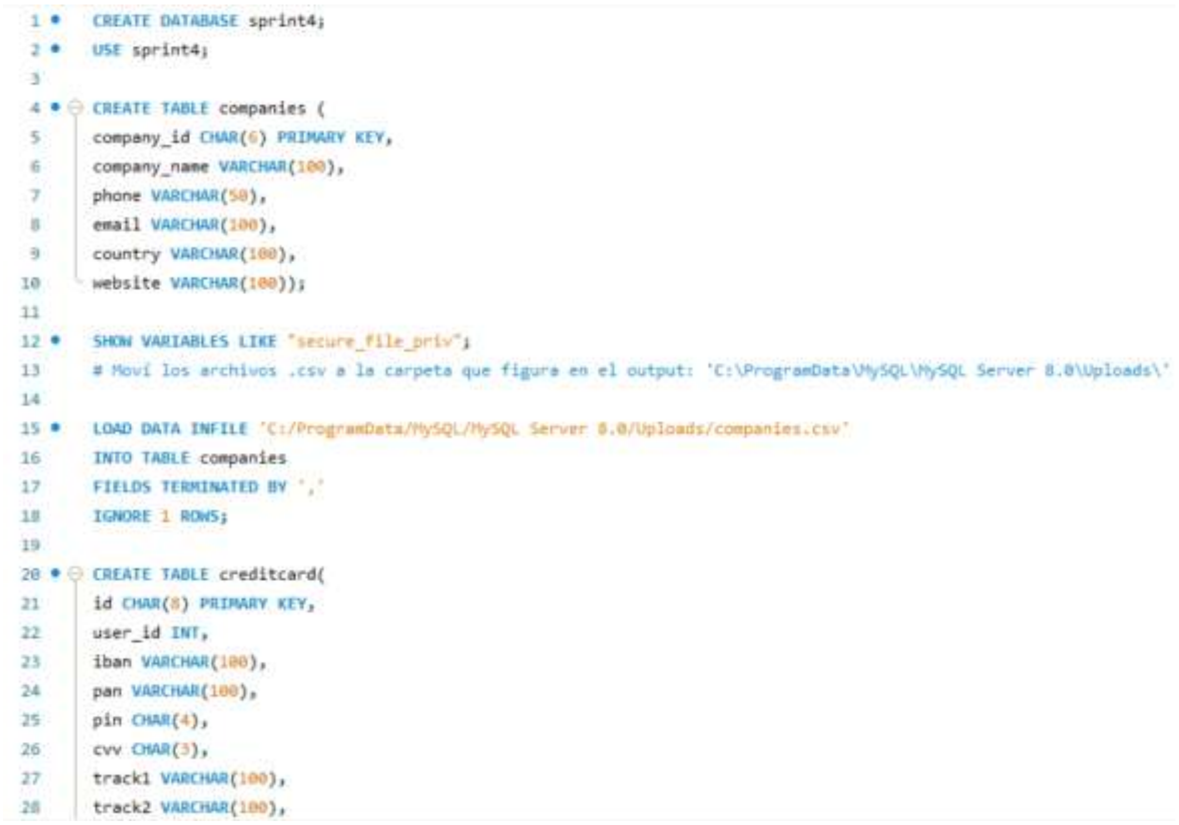
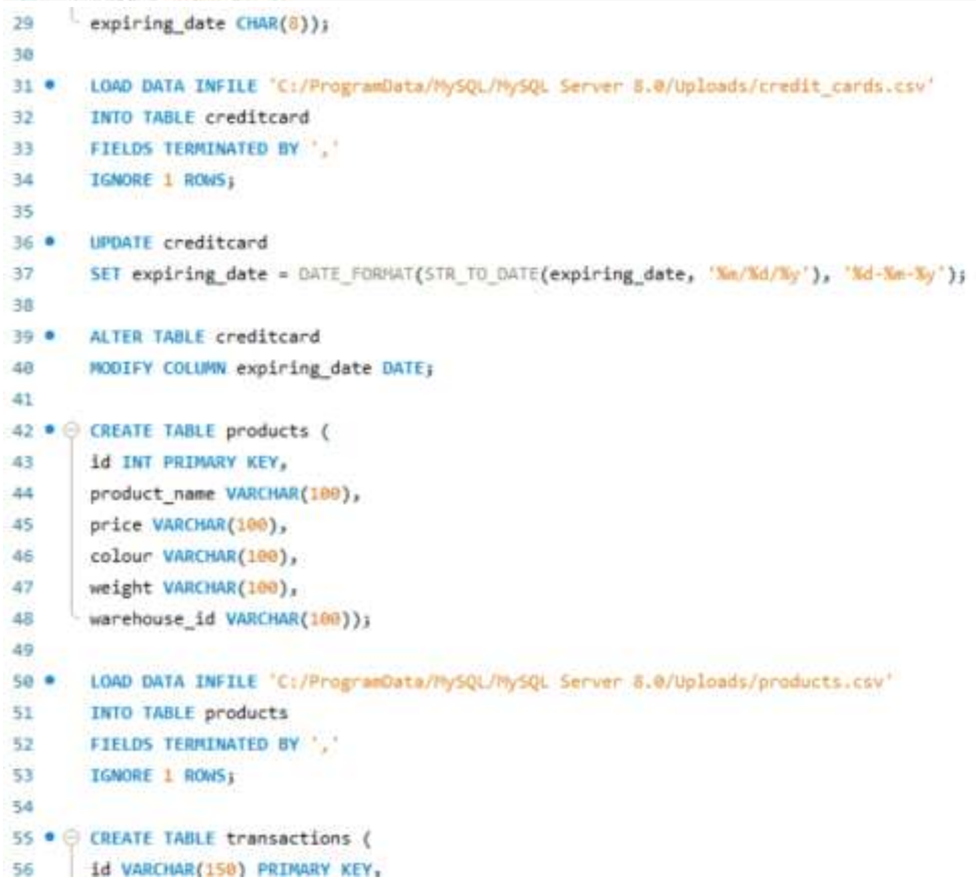
IT Academy – Datos
Sprint 4 – 25/2/25
Lucía Cané

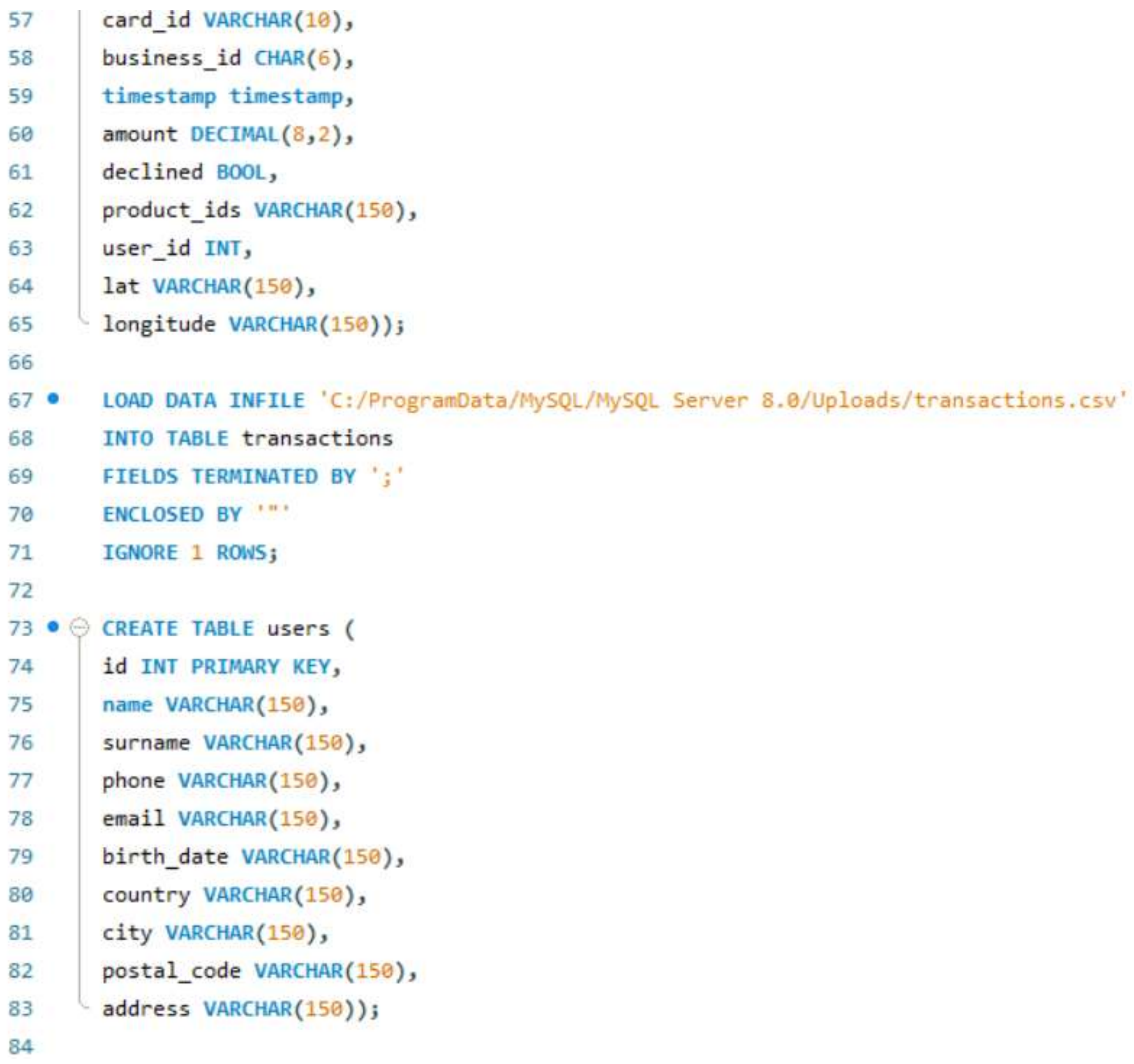
Descarga los archivos CSV, estudiales y diseña una base de datos con un esquema de estrella que contenga, al menos 4 tablas de las que puedas realizar las siguientes consultas:



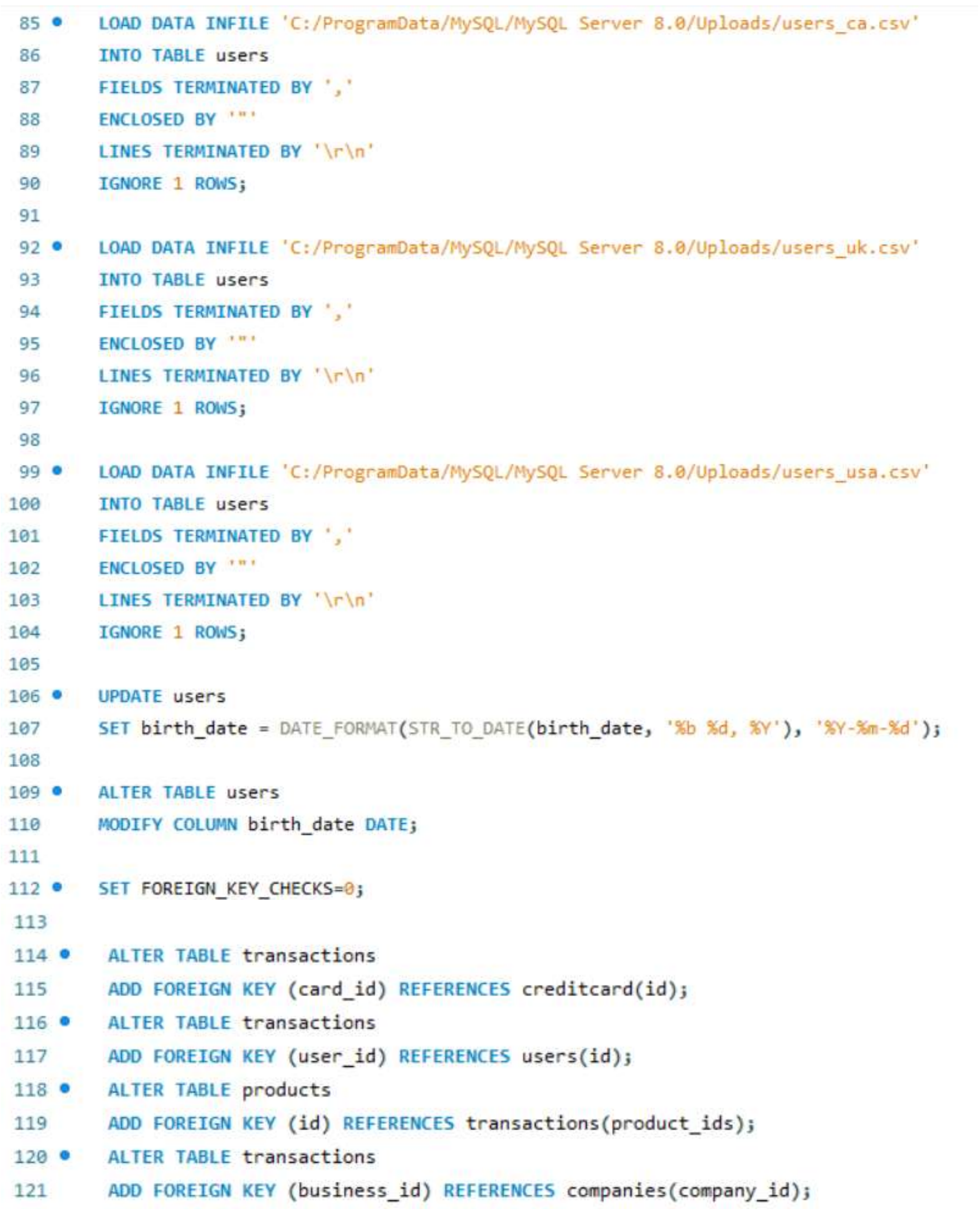
```sql
CREATE DATABASE sprint4;
USE sprint4;

CREATE TABLE companies (
company_id CHAR(6) PRIMARY KEY,
company_name VARCHAR(100),
phone VARCHAR(50),
email VARCHAR(100),
country VARCHAR(100),
website VARCHAR(100));

SHOW VARIABLES LIKE "secure_file_priv";
# Moví los archivos .csv a la carpeta que figura en el output: 'C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\'

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

CREATE TABLE creditcard(
id CHAR(8) PRIMARY KEY,
user_id INT,
iban VARCHAR(100),
pan VARCHAR(100),
pin CHAR(4),
cvv CHAR(3),
track1 VARCHAR(100),
track2 VARCHAR(100),
```



```sql
expiring_date CHAR(8));

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv'
INTO TABLE creditcard
FIELDS TERMINATED BY ','
IGNORE 1 ROWS;

UPDATE creditcard
SET expiring_date = DATE_FORMAT(STR_TO_DATE(expiring_date, '%m/%d/%y'), '%d-%m-%y');

ALTER TABLE creditcard
MODIFY COLUMN expiring_date DATE;

CREATE TABLE products (
id INT PRIMARY KEY,
product_name VARCHAR(100),
price VARCHAR(100),
colour VARCHAR(100),
weight VARCHAR(100),
warehouse_id VARCHAR(100));

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

CREATE TABLE transactions (
id VARCHAR(150) PRIMARY KEY,
```



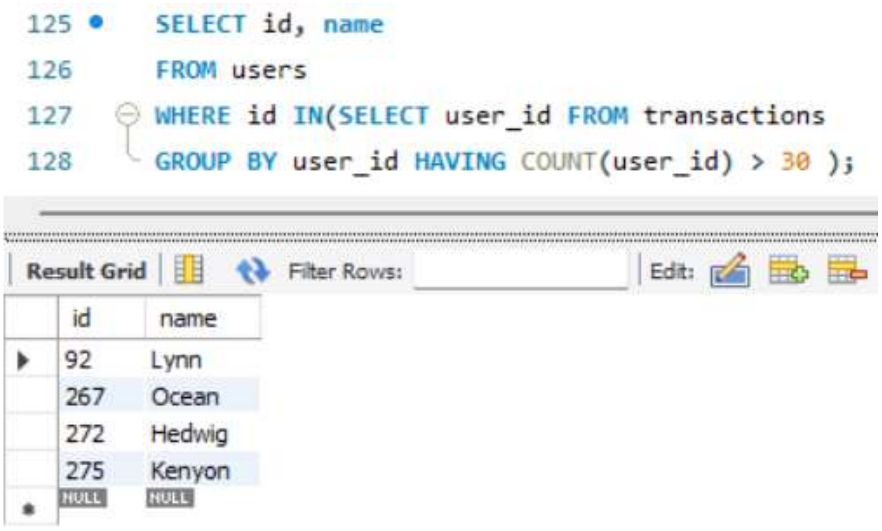
```sql
card_id VARCHAR(10),
business_id CHAR(6),
timestamp timestamp,
amount DECIMAL(8,2),
declined BOOL,
product_ids VARCHAR(150),
user_id INT,
lat VARCHAR(150),
longitude VARCHAR(150));

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

CREATE TABLE users (
id INT PRIMARY KEY,
name VARCHAR(150),
surname VARCHAR(150),
phone VARCHAR(150),
email VARCHAR(150),
birth_date VARCHAR(150),
country VARCHAR(150),
city VARCHAR(150),
postal_code VARCHAR(150),
address VARCHAR(150));
```



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

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

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

UPDATE users
SET birth_date = DATE_FORMAT(STR_TO_DATE(birth_date, '%b %d, %Y'), '%Y-%m-%d');

ALTER TABLE users
MODIFY COLUMN birth_date DATE;

SET FOREIGN_KEY_CHECKS=0;

 ALTER TABLE transactions
 ADD FOREIGN KEY (card_id) REFERENCES creditcard(id);
 ALTER TABLE transactions
 ADD FOREIGN KEY (user_id) REFERENCES users(id);
 ALTER TABLE products
 ADD FOREIGN KEY (id) REFERENCES transactions(product_ids);
 ALTER TABLE transactions
 ADD FOREIGN KEY (business_id) REFERENCES companies(company_id);
```

Nivel 1 - Ejercicio 1

Realiza una subconsulta que muestre a todos los usuarios con más de 30 transacciones utilizando al menos 2 tablas.



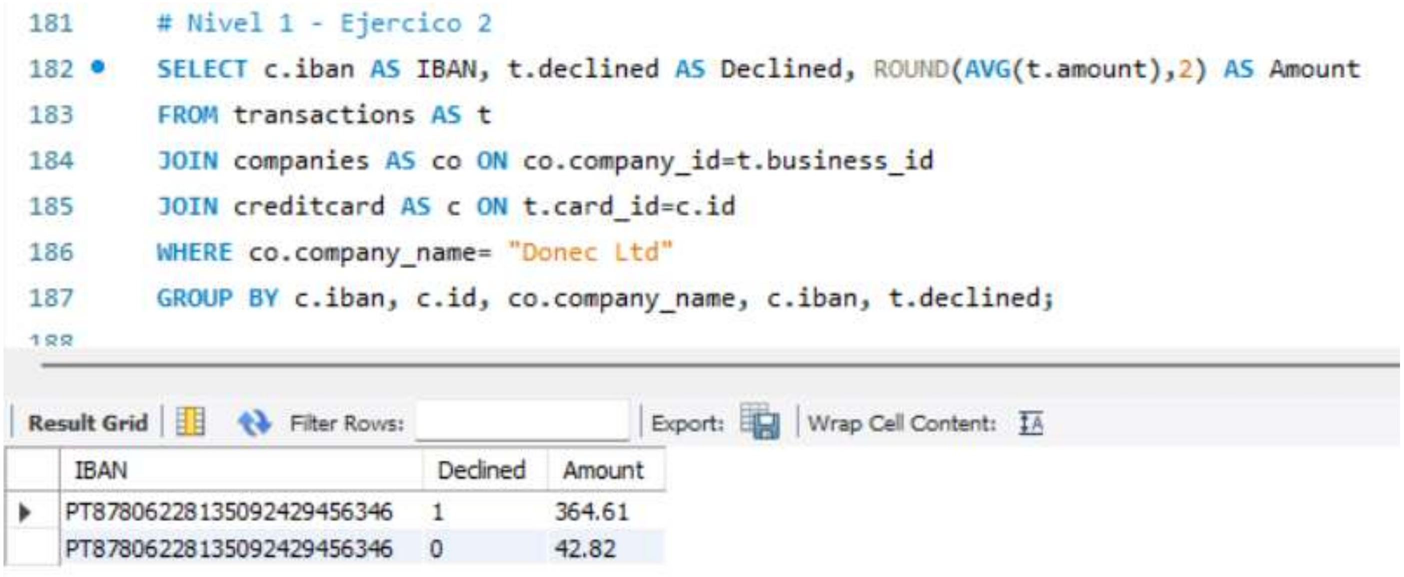
```sql
SELECT id, name
FROM users
WHERE id IN(SELECT user_id FROM transactions
GROUP BY user_id HAVING COUNT(user_id) > 30 );
```

Result Grid

id	name
92	Lynn
267	Ocean
272	Hedwig
275	Kenyon
NULL	NULL

Nivel 1 - Ejercicio 2

Muestra la media de amount por IBAN de las tarjetas de crédito en la compañía Donec Ltd., utiliza por lo menos 2 tablas.



```sql
# Nivel 1 - Ejercico 2
SELECT c.iban AS IBAN, t.declined AS Declined, ROUND(AVG(t.amount),2) AS Amount
FROM transactions AS t
JOIN companies AS co ON co.company_id=t.business_id
JOIN creditcard AS c ON t.card_id=c.id
WHERE co.company_name= "Donec Ltd"
GROUP BY c.iban, c.id, co.company_name, c.iban, t.declined;
```

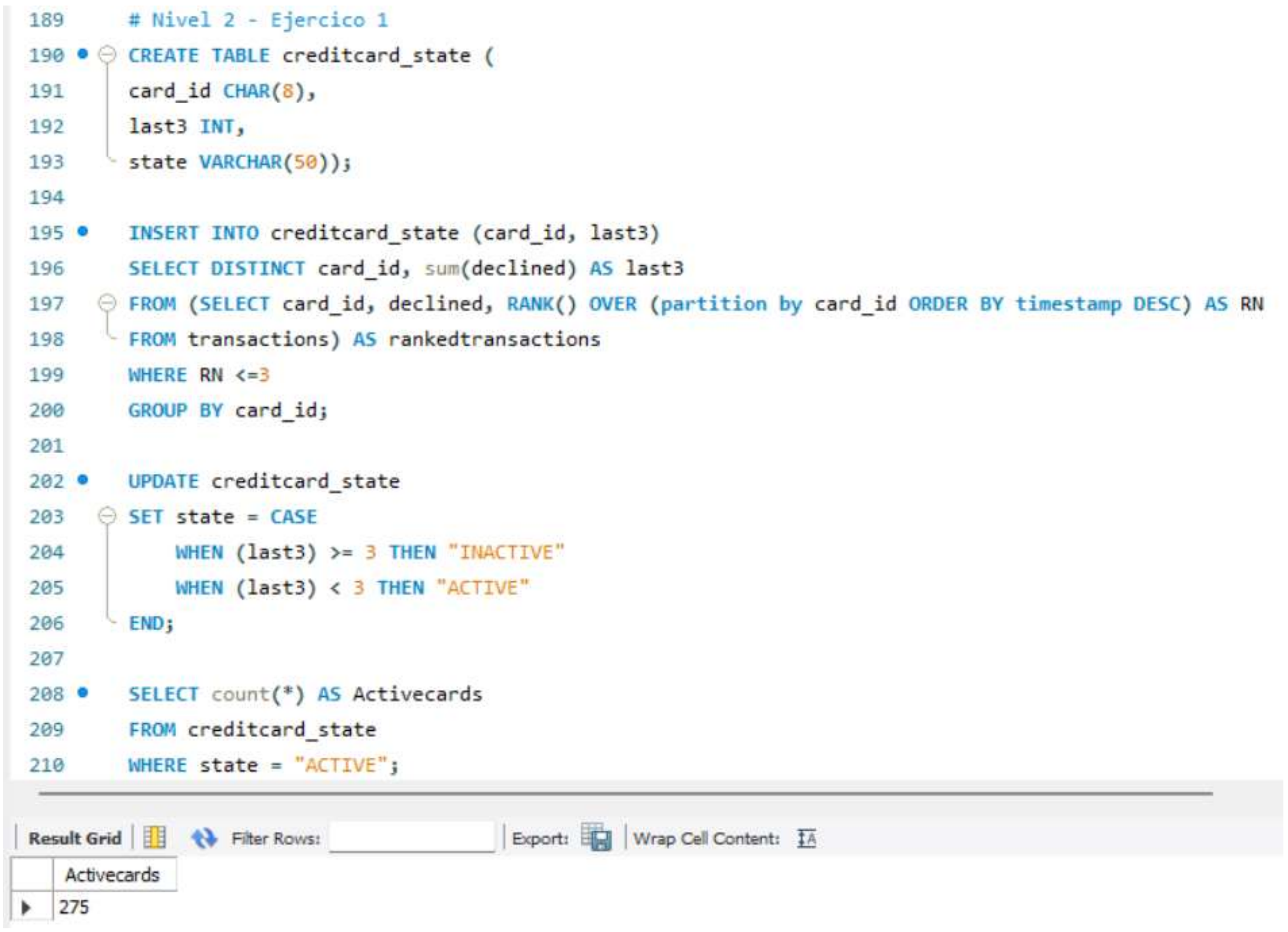
Result Grid

IBAN	Declined	Amount
PT87806228135092429456346	1	364.61
PT87806228135092429456346	0	42.82

Nivel 2 - Ejercicio 1

Crea una nueva tabla que refleje el estado de las tarjetas de crédito basado en si las últimas tres transacciones fueron declinadas y genera la siguiente consulta:

¿Cuántas tarjetas están activas?

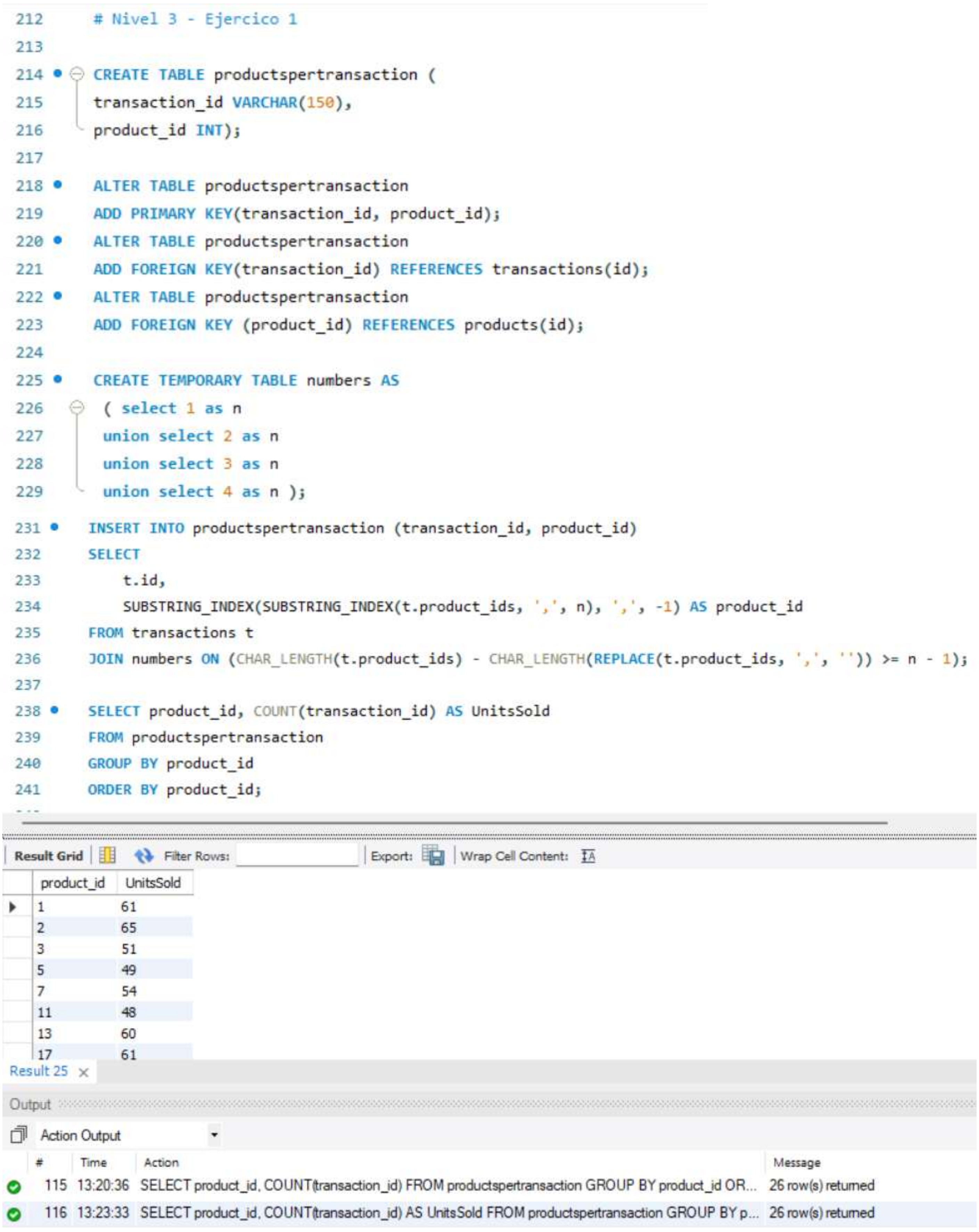


```sql
# Nivel 2 - Ejercico 1
CREATE TABLE creditcard_state (
card_id CHAR(8),
last3 INT,
state VARCHAR(50));

INSERT INTO creditcard_state (card_id, last3)
SELECT DISTINCT card_id, sum(declined) AS last3
FROM (SELECT card_id, declined, RANK() OVER (partition by card_id ORDER BY timestamp DESC) AS RN
FROM transactions) AS rankedtransactions
WHERE RN <=3
GROUP BY card_id;

UPDATE creditcard_state
SET state = CASE
    WHEN (last3) >= 3 THEN "INACTIVE"
    WHEN (last3) < 3 THEN "ACTIVE"
END;

SELECT count(*) AS Activecards
FROM creditcard_state
WHERE state = "ACTIVE";
```

Activecards
275

Nivel 3 - Ejercicio 1

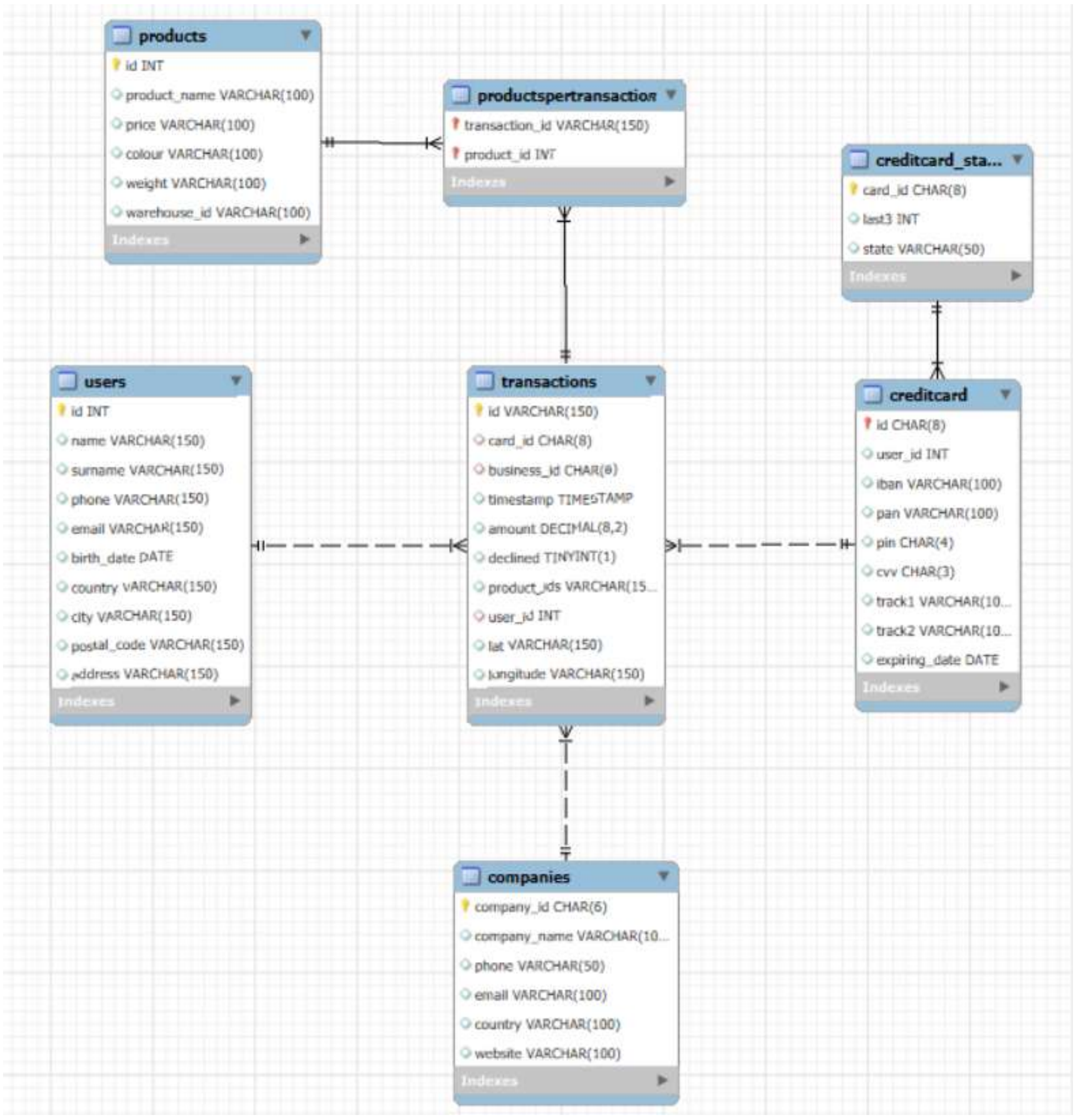
Crea una tabla con la que podamos unir los datos del nuevo archivo products.csv con la base de datos creada, teniendo en cuenta que desde transaction tienes product_ids. Genera la siguiente consulta:
Necesitamos conocer el número de veces que se ha vendido cada producto.



```sql
# Nivel 3 - Ejercico 1

CREATE TABLE productspertransaction (
transaction_id VARCHAR(150),
product_id INT);

ALTER TABLE productspertransaction
ADD PRIMARY KEY(transaction_id, product_id);
ALTER TABLE productspertransaction
ADD FOREIGN KEY(transaction_id) REFERENCES transactions(id);
ALTER TABLE productspertransaction
ADD FOREIGN KEY (product_id) REFERENCES products(id);

CREATE TEMPORARY TABLE numbers AS
 ( select 1 as n
 union select 2 as n
 union select 3 as n
 union select 4 as n );

INSERT INTO productspertransaction (transaction_id, product_id)
SELECT
    t.id,
    SUBSTRING_INDEX(SUBSTRING_INDEX(t.product_ids, ',', n), ',', -1) AS product_id
FROM transactions t
JOIN numbers ON (CHAR_LENGTH(t.product_ids) - CHAR_LENGTH(REPLACE(t.product_ids, ',', ''')) >= n - 1);

SELECT product_id, COUNT(transaction_id) AS UnitsSold
FROM productspertransaction
GROUP BY product_id
ORDER BY product_id;
```

product_id	UnitsSold
1	61
2	65
3	51
5	49
7	54
11	48
13	60
17	61

#	Time	Action	Message
115	13:20:36	SELECT product_id, COUNT(transaction_id) FROM productspertransaction GROUP BY product_id OR...	26 row(s) returned
116	13:23:33	SELECT product_id, COUNT(transaction_id) AS UnitsSold FROM productspertransaction GROUP BY p...	26 row(s) returned


products
id INT
product_name VARCHAR(100)
price VARCHAR(100)
colour VARCHAR(100)
weight VARCHAR(100)
warehouse_id VARCHAR(100)
Indexes
productspertransaction
transaction_id VARCHAR(150)
product_id INT
Indexes
creditcard_sta...
card_id CHAR(8)
last3 INT
state VARCHAR(50)
Indexes
users
id INT
name VARCHAR(150)
surname VARCHAR(150)
phone VARCHAR(150)
email VARCHAR(150)
birth_date DATE
country VARCHAR(150)
city VARCHAR(150)
postal_code VARCHAR(150)
address VARCHAR(150)
Indexes
transactions
id VARCHAR(150)
card_id CHAR(8)
business_id CHAR(6)
timestamp TIMESTAMP
amount DECIMAL(8,2)
declined TINYINT(1)
product_ids VARCHAR(15...
user_id INT
lat VARCHAR(150)
longitude VARCHAR(150)
Indexes
creditcard
id CHAR(8)
user_id INT
iban VARCHAR(100)
pan VARCHAR(100)
pin CHAR(4)
cvv CHAR(3)
track1 VARCHAR(10...
track2 VARCHAR(10...
expiring_date DATE
Indexes
companies
company_id CHAR(6)
company_name VARCHAR(10...
phone VARCHAR(50)
email VARCHAR(100)
country VARCHAR(100)
website VARCHAR(100)
Indexes