Tasca S4.01. Creació de Base de Dades

Descripció

Partint d'alguns arxius CSV dissenyaràs i crearàs la teva base de dades.

⭐ Nivell 1

Descarrega els arxius CSV, estudia'ls i dissenya una base de dades amb un esquema d'estrella que contingui, almenys 4 taules de les quals puguis realitzar les següents consultes:

Revisant els arxius csv trobem la següent disposició:

- **companies**: És doncs una taula descriptiva amb dades d'empreses.
  - company_id (PK)
  - company_name
  - phone
  - email
  - country
  - website

- **credit_cards**: És una altra taula de dimensions amb dades de targetes de crèdit.
  - id (PK): entenc que serà l'identificació de la targeta de crèdit
  - user_id: i aquesta l'identificació del titular de la targeta. FK amb la taula **users**.
  - iban: compte bancari internacional
  - pan: núm. targeta
  - pin
  - cvv
  - track1: informació comprimida de seguretat de les targetes
  - track2: més informació, encara més comprimida.
  - expiring_date: data límit (arreglarem el format abans d'importar)

- **products**: Una altra taula qualitativa amb dades descriptives sobre diferents productes.
  - id (PK)
  - product_name
  - price (elimino el signe de dolar abans d'importar)
  - colour
  - weight
  - warehouse_id: Identificació del magatzem

- **transactions**: Taula de fets
    - id (PK)
    - card_id (FK) que connecta amb la taula **credit_card**
    - business_id (FK) que connecta amb la taula **companies**
    - timestamp: moment en que es va realitzar la transacció
    - amount: import
    - declined: 0 = transacció acceptada, 1 = transacció declinada
    - product_ids Podria ser una FK que connectés amb la taula **products** **ATENCIÓ PERQUÈ AQUESTA COLUMNA POT CONTENIR MÉS D'UN PRODUCTE!** Però no la farem FK.
    - user_id (FK) que connecta amb la taula **users**
    - lat: Latitud des d'on s'ha efectuat la transacció?
    - longitude: Longitud des d'on s'ha efectuat la transacció?

No m'agrada gens la disposició de la columna products_ids amb les separacions per comes, així que he generat una taula intermitja entre **products** i **transactions**, que es dirà **products_transactions** que només tindrà 2 columnes:

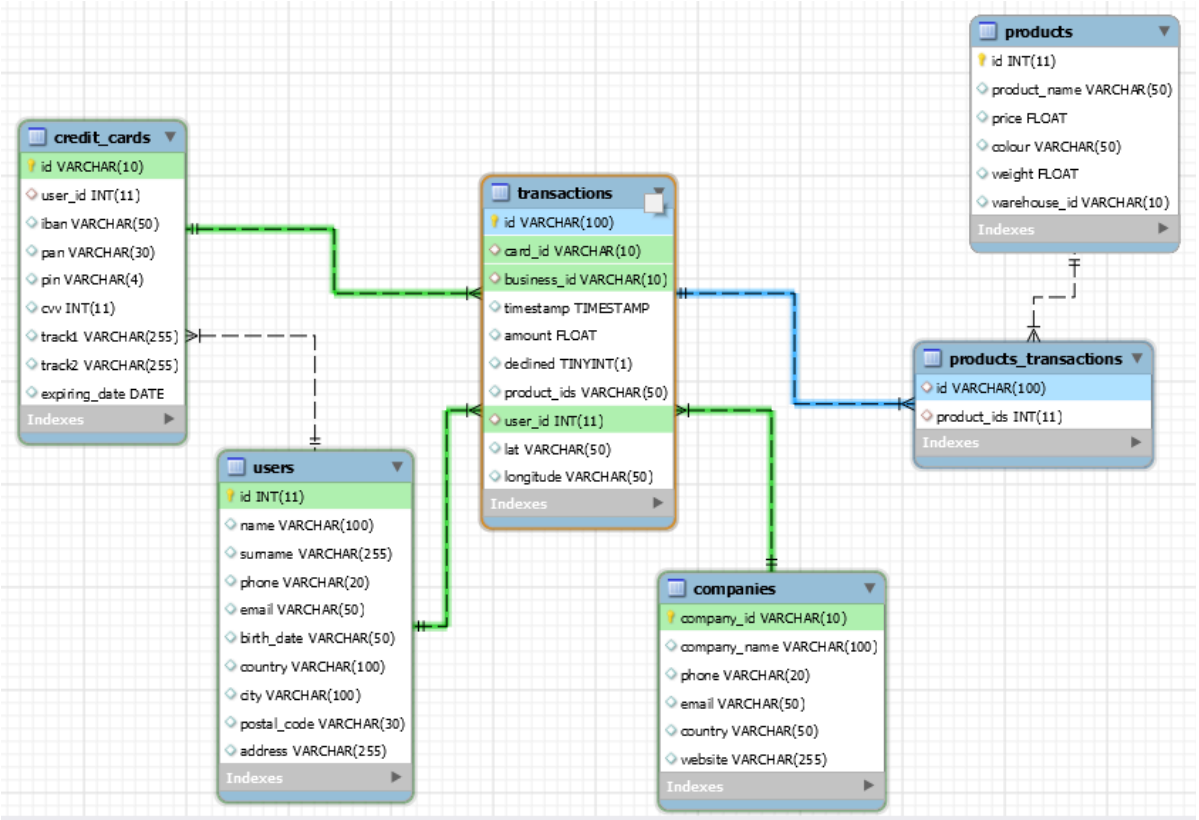
- id: número de la transacció
- product_ids

En un cas normal eliminaria la columna product_ids de la taula **transactions** però de moment la deixaré per explorar si més endevant la puc transformar dins del propi SQL.

- **users_ca**: Taula dimensional d'usuaris de Canadà
- **users_uk**: Taula dimensional d'usuaris de United Kingdom
- **users_usa**: Taula dimensional d'usuaris dels Estats Units
    - id (PK)
    - name
    - surname
    - phone
    - email
    - birth_date: No m'agrada el format però donat que en aquest sprint no treballarem amb aquestes dades no les modificaré.
    - country
    - city
    - postal_code
    - address

Com que aquestes tres taules són idèntiques en el seu format i ja tenen la columna o atribut "country" que les diferencia, prefereixo ajuntar-les en una sola taula **users** que les agrupi a les 3. En agrupar-les me n'adono que fins hi tot l'id és correlatiu 🙂

I el diagrama queda així:



1.1. Exercici 1

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

Faig una query amb subconsulta que mostri tots els usuaris amb més de 30 transaccions:

```
94  SELECT name, surname, city, country
95  FROM users
96  WHERE id IN
97  (
98      SELECT user_id
99      FROM transactions
100     GROUP BY user_id
101     HAVING COUNT(id) > 30
102 );
```

sult Grid | Filter Rows: | Export:

name	surname	city	country
Lynn	Riddle	Bozeman	United States
Ocean	Nelson	Charlottetown	Canada
Hedwig	Gilbert	Tuktoyaktuk	Canada
Kenyon	Hartman	Richmond	Canada

1.2. Exercici 2

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

He filtrat amb el declined = 0 perquè dedueixo que s'està buscant l'amount gastat:

```
123  SELECT c.company_name, ROUND(AVG(t.amount),2) AS Mitja
124  FROM transactions t
125  JOIN companies c
126  ON t.business_id = c.company_id
127  WHERE c.company_name = 'Donec Ltd' AND declined = 0
128  GROUP BY t.business_id;
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

company_name	Mitja
Donec Ltd	42.82

En tot cas, si mirem la mitja de l'amount de totes les transaccions, sense tenir en compte si han estat rebutjades (declinades) o no la consulta queda així:

```
123  SELECT c.company_name, ROUND(AVG(t.amount),2) AS Mitja
124  FROM transactions t
125  JOIN companies c
126  ON t.business_id = c.company_id
127  WHERE c.company_name = 'Donec Ltd'
128  GROUP BY t.business_id;
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

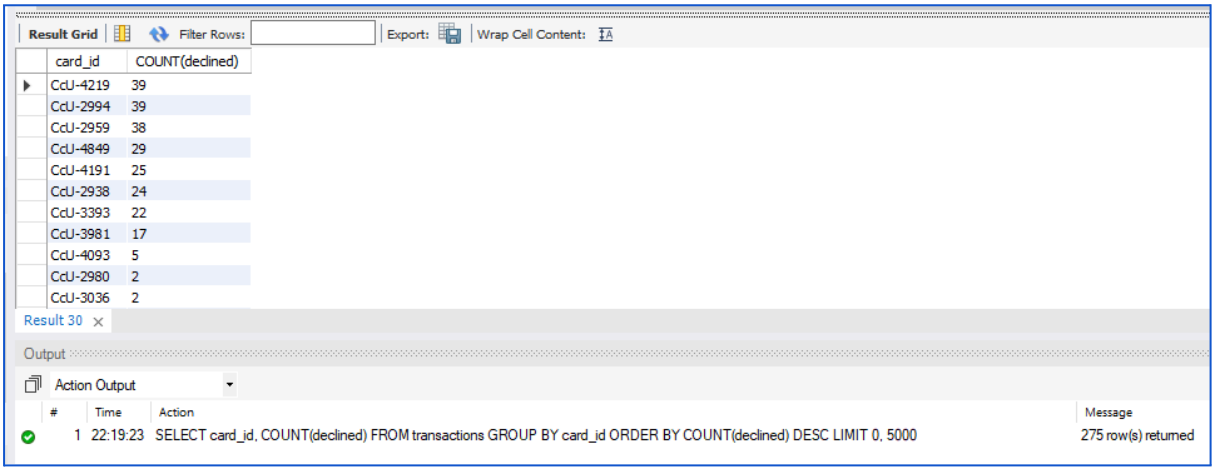
company_name	Mitja
Donec Ltd	203.71

⭐⭐ 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:

Tenint en compte que per considerar que una **targeta està inactiva** ha de tenir les últimes 3 transaccions rebutjades (declined = 1) mirarem quantes targetes tenen 3 moviments o més (ja que les que només tinguin 1 o 2 moviments, encara que tots fossin rebutjats, no complirien la condició de 3 transaccions rebutjades).

```sql
-- veiem que només 9 targetes tenen més de 3 transaccions. La resta 2 o menys
SELECT card_id, COUNT(declined)
FROM transactions
GROUP BY card_id
ORDER BY COUNT(declined) DESC;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

card_id	COUNT(declined)
CcU-4219	39
CcU-2994	39
CcU-2959	38
CcU-4849	29
CcU-4191	25
CcU-2938	24
CcU-3393	22
CcU-3981	17
CcU-4093	5
CcU-2980	2
CcU-3036	2

Result 30

Output

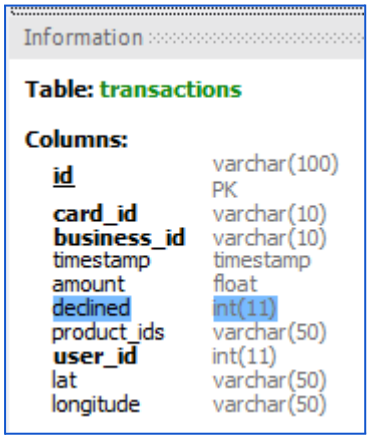
Action Output

#	Time	Action	Message
1	22:19:23	SELECT card_id, COUNT(declined) FROM transactions GROUP BY card_id ORDER BY COUNT(declined) DESC LIMIT 0, 5000	275 row(s) returned

D'aquestes 9 targetes que tenen 3 moviments o més, em disposo a contar quants d'aquests ha estat rebutjats.

Per fer això he canviat el format de la columna declined de BOOLEAN a INT. Tot i que crec que no era necessari. Ha sigut per si de cas:

```sql
-- Canvio la columna declined de BOOLEAN a INT per poder sumar (tot i que crec que pot ser no calia)
ALTER TABLE transactions MODIFY COLUMN declined INT;
```



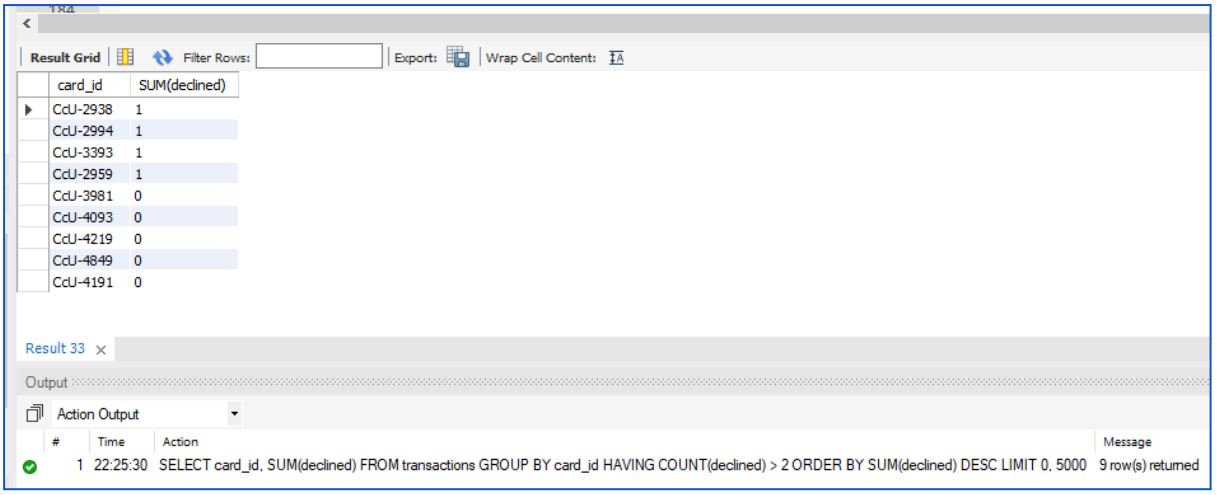
Information

Table: transactions

Columns:

id	varchar(100) PK
card_id	varchar(10)
business_id	varchar(10)
timestamp	timestamp
amount	float
declined	int(11)
product_ids	varchar(50)
user_id	int(11)
lat	varchar(50)
longitude	varchar(50)

```sql
-- Cap targeta amb 3, o més, transaccions té 3 transaccions rebutjades. Ja no cal mirar les 3 últimes
SELECT card_id, SUM(declined)
FROM transactions
GROUP BY card_id
HAVING COUNT(declined) > 2
ORDER BY SUM(declined) DESC;
```



card_id	SUM(declined)
CcU-2938	1
CcU-2994	1
CcU-3393	1
CcU-2959	1
CcU-3981	0
CcU-4093	0
CcU-4219	0
CcU-4849	0
CcU-4191	0

Result 33

Output

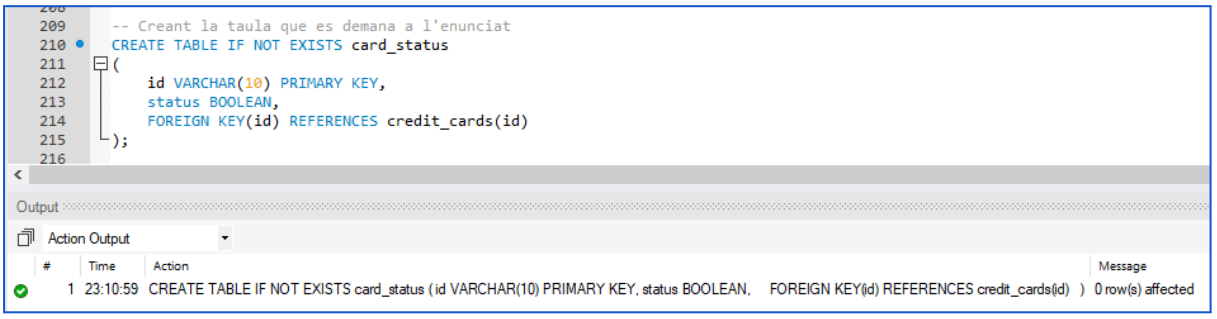
Action Output

#	Time	Action	Message
1	22:25:30	SELECT card_id, SUM(declined) FROM transactions GROUP BY card_id HAVING COUNT(declined) > 2 ORDER BY SUM(declined) DESC LIMIT 0, 5000	9 row(s) returned

Veient que la quantitat de “declineds” per targeta mai arriba a 3 ja no cal filtrar pels 3 últims moviments, ja que cap sumarà 3.

Per tant, responent a la pregunta 2.1: totes les targetes estan actives *

Ara caldrà crear la taula.

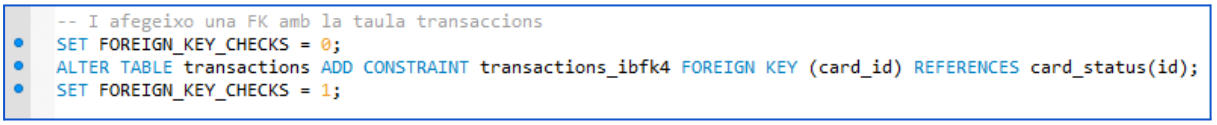


```
209  -- Creant la taula que es demana a l'enunciat
210  CREATE TABLE IF NOT EXISTS card_status
211  (
212      id VARCHAR(10) PRIMARY KEY,
213      status BOOLEAN,
214      FOREIGN KEY(id) REFERENCES credit_cards(id)
215  );
216
```

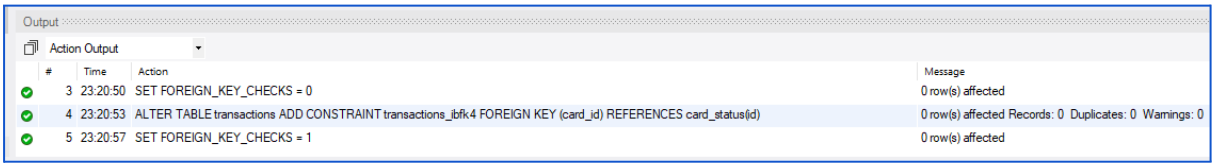
Output

Action Output

#	Time	Action	Message
1	23:10:59	CREATE TABLE IF NOT EXISTS card_status (id VARCHAR(10) PRIMARY KEY, status BOOLEAN, FOREIGN KEY(id) REFERENCES credit_cards(id))	0 row(s) affected



```
-- I afegeixo una FK amb la taula transaccions
SET FOREIGN_KEY_CHECKS = 0;
ALTER TABLE transactions ADD CONSTRAINT transactions_ibfk4 FOREIGN KEY (card_id) REFERENCES card_status(id);
SET FOREIGN_KEY_CHECKS = 1;
```



Output

Action Output

#	Time	Action	Message
3	23:20:50	SET FOREIGN_KEY_CHECKS = 0	0 row(s) affected
4	23:20:53	ALTER TABLE transactions ADD CONSTRAINT transactions_ibfk4 FOREIGN KEY (card_id) REFERENCES card_status(id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
5	23:20:57	SET FOREIGN_KEY_CHECKS = 1	0 row(s) affected

Un cop creada la taula importo les dades amb el Wizard:

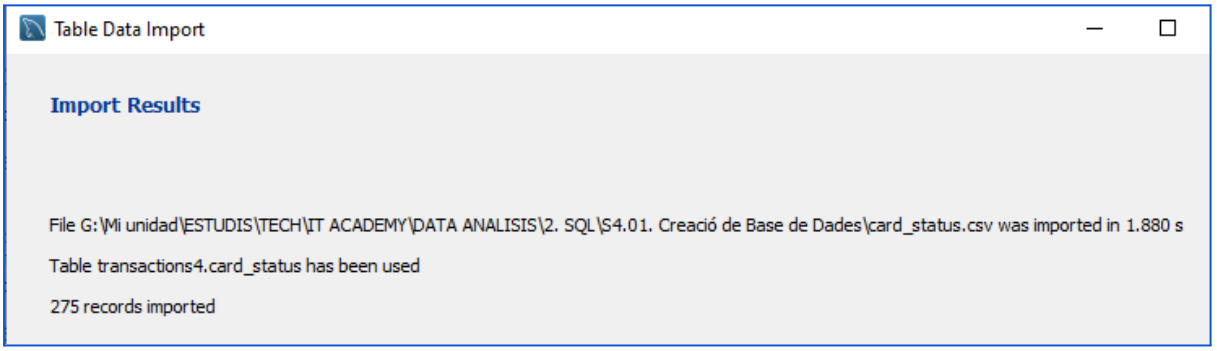


Table Data Import

Import Results

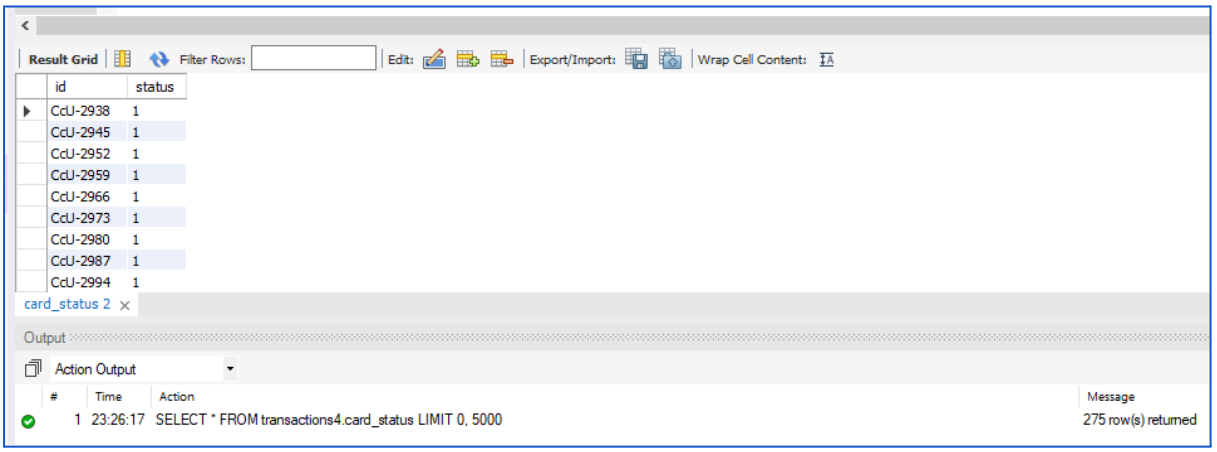
File G:\Mi unidad\ESTUDIS\TECH\IT ACADEMY\DATA ANALISIS\2. SQL\S4.01. Creació de Base de Dades\card_status.csv was imported in 1.880 s

Table transactions4.card_status has been used

275 records imported

Comprovem que les dades s'han importat bé:

```
SELECT *
FROM transactions4.card_status;
```

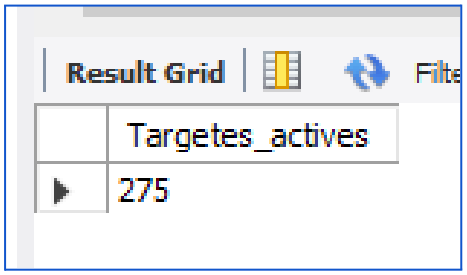


2.1. Exercici 1

Quantes targetes estan actives?

* Totes les targetes estan actives.

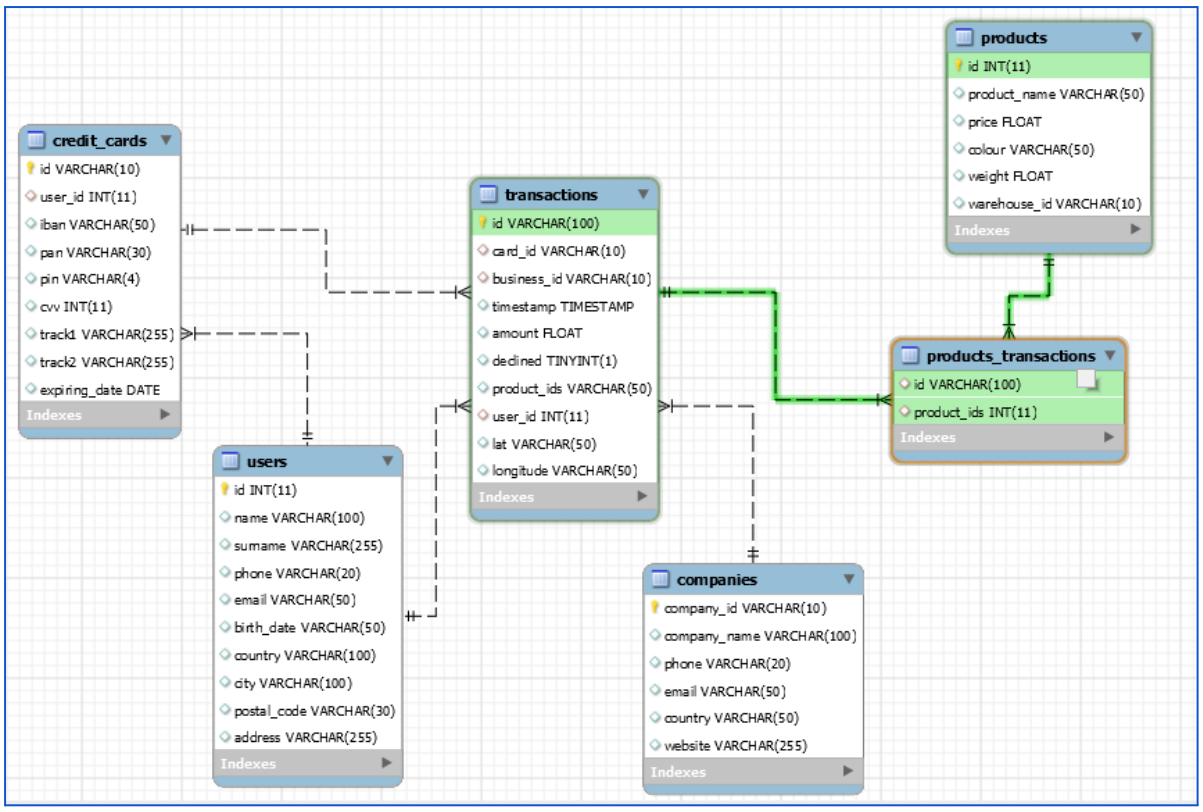
```
SELECT COUNT(*) AS Targetes_actives
FROM card_status
WHERE status = 1;
```



★★★ 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:

La taula enllaç creada s'anomena **products_transactions** i relaciona els números de transacció de la taula **transactions** amb els ids de la taula **products**, però separats per cel·les per màxima comoditat:



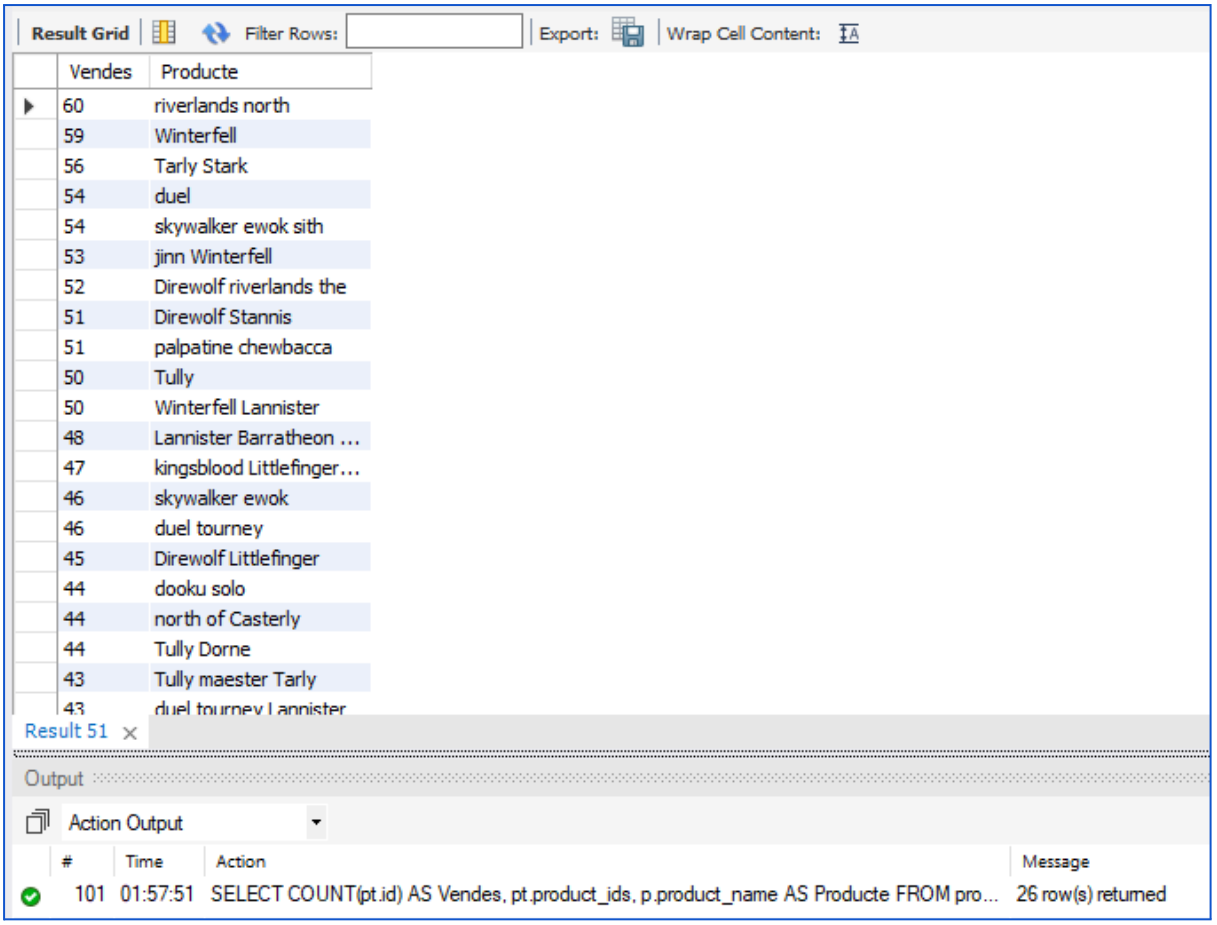
3.1. Exercici 1

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

Aquesta seria la query, només amb les vendes: transaccions acceptades = (declined = 0).

```
SELECT COUNT(pt.id) AS Vendes, p.product_name AS Producte
FROM products_transactions pt
LEFT JOIN products p
ON pt.product_ids = p.id
WHERE pt.id IN
(
    SELECT t.id
    FROM transactions t
    WHERE t.declined = 0
)
GROUP BY pt.product_ids
ORDER BY Vendes DESC;
```

I el resultat:


Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Vendes
Producte
60 riverlands north
59 Winterfell
56 Tarly Stark
54 duel
54 skywalker ewok sith
53 jinn Winterfell
52 Direwolf riverlands the
51 Direwolf Stannis
51 palpatine chewbacca
50 Tully
50 Winterfell Lannister
48 Lannister Barratheon ...
47 kingsblood Littlefinger...
46 skywalker ewok
46 duel tourney
45 Direwolf Littlefinger
44 dooku solo
44 north of Casterly
44 Tully Dorne
43 Tully maester Tarly
43 duel tourney Lannister
Result 51
Output
Action Output
Time Action Message
101 01:57:51 SELECT COUNT(pt.id) AS Vendes, pt.product_ids, p.product_name AS Producte FROM pro... 26 row(s) returned