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 què es va realitzar la transacció
    - amount: import
    - declined: 0 = transacció acceptada, 1 = transacció declinada
    - product_ids Podria ser una FK que enllaçés 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

- **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 🙂

Comencem amb la creació de la base de dades:

```
CREATE DATABASE Transactions4;
```

I seguim creant les taules:

```sql
-- Creació de la taula companies
CREATE TABLE IF NOT EXISTS companies
(
    company_id VARCHAR(10) PRIMARY KEY,
    company_name VARCHAR(100),
    phone VARCHAR(20),
    email VARCHAR(50),
    country VARCHAR(50),
    website VARCHAR(255)
);
```

```sql
-- primer creo la taula users per després poder crear la FK de credit_cards
CREATE TABLE IF NOT EXISTS users
(
    id INT PRIMARY KEY,
    name VARCHAR(100),
    surname VARCHAR(255),
    phone VARCHAR(20),
    email VARCHAR(50),
    birth_date VARCHAR(50),
    country VARCHAR(100),
    city VARCHAR(100),
    postal_code VARCHAR(30),
    address VARCHAR(255)
);
```

```sql
-- Creació de la taula credit_cards
CREATE TABLE IF NOT EXISTS credit_cards
(
    id VARCHAR(10) PRIMARY KEY,
    user_id INT,
    iban VARCHAR(50),
    pan VARCHAR(30),
    pin VARCHAR(4),
    cvv INT,
    track1 VARCHAR(255),
    track2 VARCHAR(255),
    expiring_date DATE,
    FOREIGN KEY(user_id) REFERENCES users(id)
);
```

```sql
-- Creació de la taula products
CREATE TABLE IF NOT EXISTS products
(
    id INT PRIMARY KEY,
    product_name VARCHAR(50),
    price FLOAT,
    colour VARCHAR(50),
    weight FLOAT,
    warehouse_id VARCHAR(10)
);
```

```sql
-- Creació de la taula transacctions
CREATE TABLE IF NOT EXISTS transactions
(
    id VARCHAR(100) PRIMARY KEY,
    card_id VARCHAR(10),
    business_id VARCHAR(10),
    timestamp TIMESTAMP,
    amount FLOAT,
    declined BOOLEAN,
    product_ids VARCHAR(50),
    user_id INT,
    lat VARCHAR(50),
    longitude VARCHAR(50),
    FOREIGN KEY(card_id) REFERENCES credit_cards(id),
    FOREIGN KEY(business_id) REFERENCES companies(company_id),
    FOREIGN KEY(user_id) REFERENCES users(id)
);
```

```sql
-- Per últim creo la taula nexe: products_transactions
CREATE TABLE IF NOT EXISTS products_transactions
(
    id VARCHAR(100),
    product_ids INT,
    FOREIGN KEY(id) REFERENCES transactions(id),
    FOREIGN KEY(product_ids) REFERENCES products(id)
);
```

Totes les importacions dels arxius .csv es van fer amb el **Wizard**: Al panell de l'esquerra clic botó dret sobre la taula o la mateixa base de dades → Table Data Import Wizard i cal triar la taula que ja està creada.

Un cop feta la correcció Peer to peer amb la companya Carla Lupión, em comenta que seria interessant que provés de fer com a mínim una importació manual per aprendre com va així que esborro el contingut de la taula **companies**:

```sql
-- Intento esborrar el contingut de la taula companies per procedir a la importació de dades "manual"
DELETE FROM companies;

-- No em deixa pq estic en mode segur. Desactivo.
SET SQL_SAFE_UPDATES = 0;

-- Rebo un altre error per les Foreing Keys:
SET FOREIGN_KEY_CHECKS = 0;

-- Esborro i torno a activar:
DELETE FROM companies;
SET FOREIGN_KEY_CHECKS = 1;
SET SQL_SAFE_UPDATES = 1;
```

Output

Action Output

#	Time	Action	Message
1	22:08:36	SET SQL_SAFE_UPDATES = 0	0 row(s) affected
2	22:08:39	DELETE FROM companies	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`transacti...
3	22:11:16	SET FOREIGN_KEY_CHECKS = 0	0 row(s) affected
4	22:11:20	DELETE FROM companies	100 row(s) affected
5	22:11:46	SET FOREIGN_KEY_CHECKS = 1	0 row(s) affected
6	22:11:48	SET SQL_SAFE_UPDATES = 1	0 row(s) affected

Ja no hi ha res:

```
1  SELECT * FROM transactions4.companies;
```

Result Grid | Filter Rows: | Edit:

company_id	company_name	phone	email	country	website
NULL	NULL	NULL	NULL	NULL	NULL

I importo l'arxiu .csv desde la consola:

```
-- Importo els arxius de companies "manualment"
LOAD DATA INFILE 'G:/Mi unidad/ESTUDIS/TECH/IT ACADEMY/DATA ANALISIS/2. SQL/S4.01. Creació de Base de Dades/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES; -- per ignorar els títols de les columnes

-- Em dona error --secure-file-priv i haig de buscar la carpeta segura al meu PC
SHOW VARIABLES LIKE 'secure_file_priv';
```

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

Variable_name	Value
secure_file_priv	C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\

```
-- Intento importar els arxius de companies "manualment" desde la carpeta segura
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES; -- per ignorar els títols de les columnes
```

Output

Action Output

#	Time	Action	Message
1	22:24:26	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv' IN...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0

Si l'arxiu .csv es veiés per columnes des de excel, com a la taula transactions, s'entendria que aquest està separat per punt i coma (;) i no per comes (,) i caldria explicitar-ho a la query.

FIELDS TERMINATED BY ';'

Ja tornem a tenir dades de companies:

```
1  SELECT * FROM transactions4.companies;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

company_id	company_name	phone	email	country	website
b-2222	Ac Fermentum Incorporated	06 85 56 52 33	donec.porttitor.tellus@yahoo.net	Germany	https://instagram.com/site
b-2226	Magna A Neque Industries	04 14 44 64 62	risus.donec.nibh@icloud.org	Australia	https://whatsapp.com/group/9
b-2230	Fusce Corp.	08 14 97 58 85	risus@protonmail.edu	United States	https://pinterest.com/sub/cars
b-2234	Convallis In Incorporated	06 66 57 29 50	mauris.ut@aol.couk	Germany	https://cnn.com/user/110
b-2238	Ante Iaculis Nec Foundation	08 23 04 99 53	sed.dictum.proin@outlook.ca	New Zealand	https://netflix.com/settings
b-2242	Donec Ltd	01 25 51 37 37	at.iaculis@hotmail.couk	Norway	https://nytimes.com/user/110
b-2246	Sed Nunc Ltd	02 62 64 73 48	nibh@yahoo.org	United Kingdom	https://cnn.com/one
b-2250	Amet Nulla Donec Corporation	07 15 25 14 74	mattis.integer.eu@protonmail.net	Italy	https://netflix.com/sub/cars
b-2254	Nascetur Ridiculus Mus Inc.	06 26 87 61 84	suspendisse.dui@icloud.net	United States	https://ebay.com/sub
b-2258	Vestibulum Lorem PC	02 02 87 33 40	aenean.massa.integer@aol.net	Belgium	https://pinterest.com/sub/cars
b-2262	Gravida Sagittis LLP	03 81 28 33 97	turpis.vitae@google.ca	Sweden	https://naver.com/site
b-2266	Mus Aenean Eget Foundation	06 25 15 52 43	mi.duis@hotmail.net	Sweden	https://instagram.com/group/9
b-2270	Dis Parturient Institute	05 36 29 78 74	purus@protonmail.org	Ireland	https://google.com/one
b-2274	Sed LLC	01 63 16 26 52	at@outlook.com	Belgium	https://reddit.com/fr
b-2278	Arcu LLP	06 46 04 41 45	dui@aol.ca	Norway	https://yahoo.com/sub
b-2282	Pretium Neque Corp.	07 77 48 55 28	eleifend.nec.malesuada@proton...	Australia	https://netflix.com/sub
b-2286	Fringilla LLC	08 29 15 93 57	gravida@protonmail.couk	New Zealand	https://reddit.com/user/110
b-2290	Quisque Libero LLC	01 45 48 71 11	sapien.molestie.orci@hotmail.couk	China	https://baidu.com/group/9

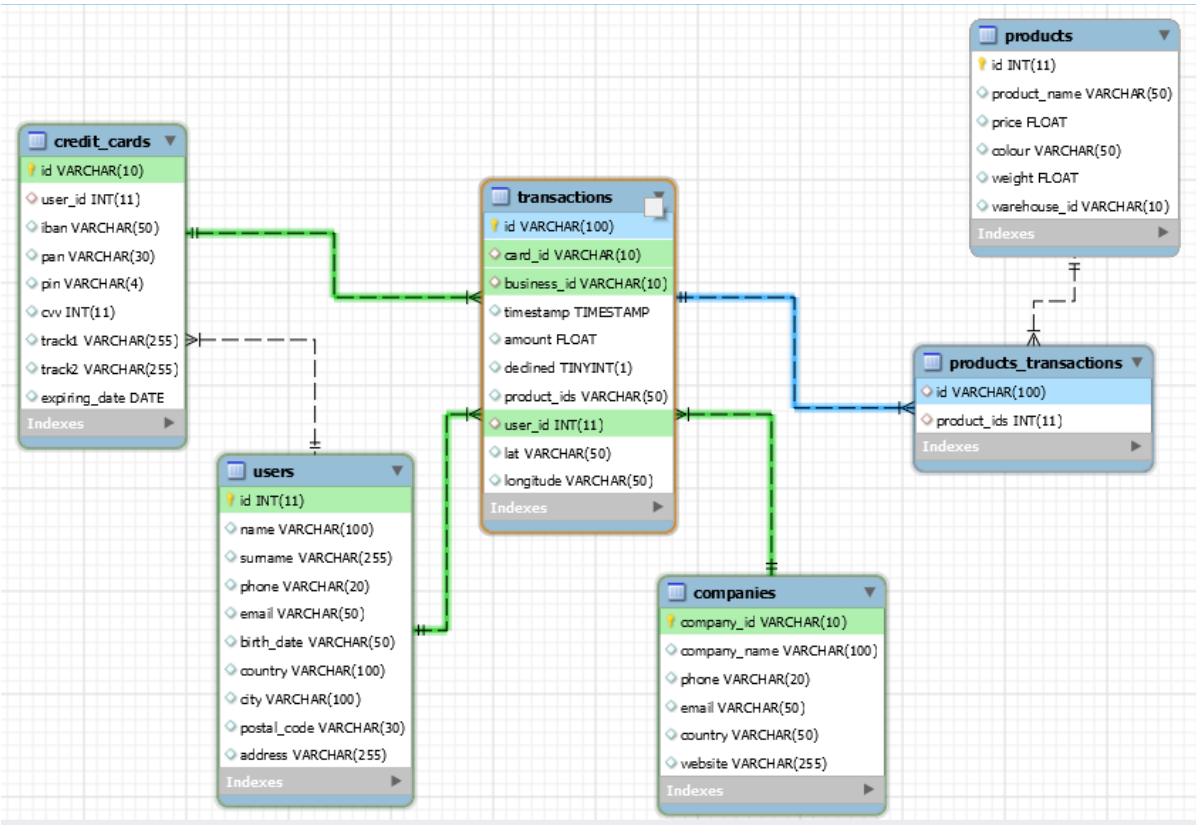
companies 1

Output

Action Output

#	Time	Action	Message
1	22:24:26	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv' I...	100 row(s) affected Records: 100 Deleted: 0
2	22:28:00	SELECT * FROM transactions4.companies LIMIT 0, 5000	100 row(s) returned

I el diagrama queda així:



Tot i que pugui semblar que existeix una relació circular entre les taules **transactions**, **users** i **credit_cards** això no és així.

Una relació circular es donaria si:

- No es pogués afegir una nova transacció (amb un id d'usuari nou), sense afegir un nou usuari i
- No es pogués afegir un nou usuari sense afegir una nova targeta de crèdit i
- No es pogués afegir una nova targeta de crèdit sense afegir una nova transacció.

Llavors sí estaríem parlant d'una relació circular, que ens tindria bloquejats alhora d'afegir nous registres.

Però les relacions NO estan creades així. Totes les relacions de la taula **users** són de 1 a Molts. I totes les relacions de la taula **transactions** amb aquestes dues: **users** i **credit_cards**; són de Molts a 1. Així que no podem dir que la relació s'ha generat de manera circular.

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.
He afegit la ciutat i el país per donar una mica més d'informació dels usuaris de la consulta, tot i que l'enunciat no ho demana.

```
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.

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, cc.iban, ROUND(AVG(t.amount),2) AS Mitja
124 FROM transactions t
125 JOIN companies c
126 ON t.business_id = c.company_id
127 JOIN credit_cards cc
128 ON t.card_id = cc.id
129 WHERE c.company_name = 'Donec Ltd'
130 GROUP BY t.business_id, t.card_id;
```

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

company_name	iban	Mitja
Donec Ltd	PT87806228135092429456346	203.71

Si el que es vol mirar és l'amount gastat real, s'hauria de filtrar amb el declined = 0. En tot cas s'hauria de preguntar a la persona que ens ho ha demanat com ho vol o presentar les dues opcions. Quedaria aquest import:

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

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

company_name	iban	Mitja
Donec Ltd	PT87806228135092429456346	42.82

★★ 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).

```
-- 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

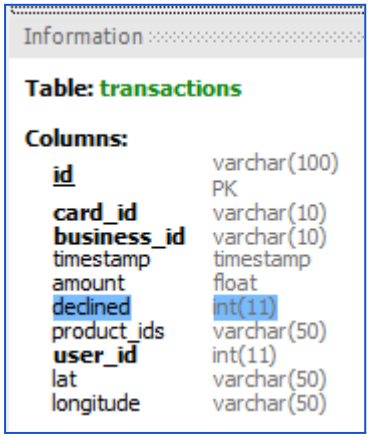
Action Output

#	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

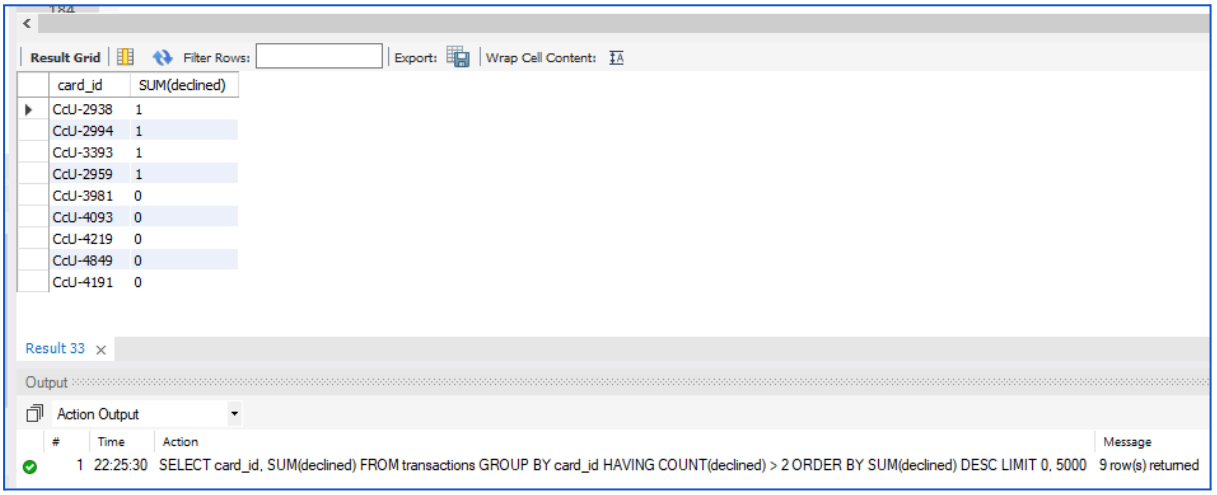
D'aquestes 9 targetes que tenen 3 moviments o més, em disposo a contar quants d'aquests ha estat rebutjats.

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:

```
-- 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;
```




```
-- 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;
```



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.

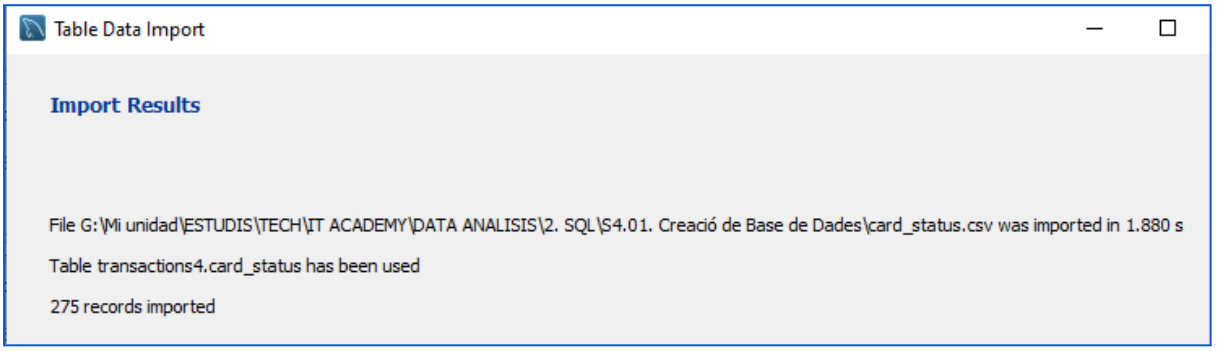
```sql
-- Creant la taula que es demana a l'enunciat
CREATE TABLE IF NOT EXISTS card_status
(
    id VARCHAR(10) PRIMARY KEY,
    status BOOLEAN,
    FOREIGN KEY(id) REFERENCES credit_cards(id)
);
```

```
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
```

```sql
-- 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:



Comprovem que les dades s'han importat bé:

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

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	status
CcU-2938	1
CcU-2945	1
CcU-2952	1
CcU-2959	1
CcU-2966	1
CcU-2973	1
CcU-2980	1
CcU-2987	1
CcU-2994	1

card_status 2

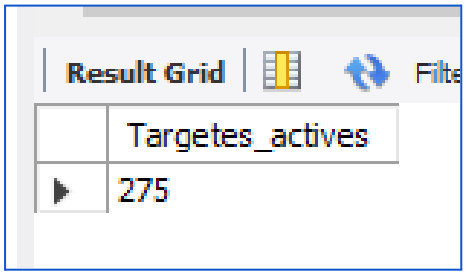
```
Output
Action Output
#  Time      Action                                                  Message
1  23:26:17  SELECT * FROM transactions4.card_status LIMIT 0, 5000  275 row(s) returned
```

2.1. Exercici 1

Quantes targetes estan actives?

* Totes les targetes estan actives.

```sql
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.

Es va crear en el moment inicial al veure que la columna *product_ids* de la taula **transactions** contenia diversos registres per cel·la.

```sql
CREATE TABLE IF NOT EXISTS products_transactions
(
    id VARCHAR(100),
    product_ids INT,
    FOREIGN KEY(id) REFERENCES transactions(id),
    FOREIGN KEY(product_ids) REFERENCES products(id)
);
```

```
1 •  SELECT * FROM transactions4.products_transactions;
```

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

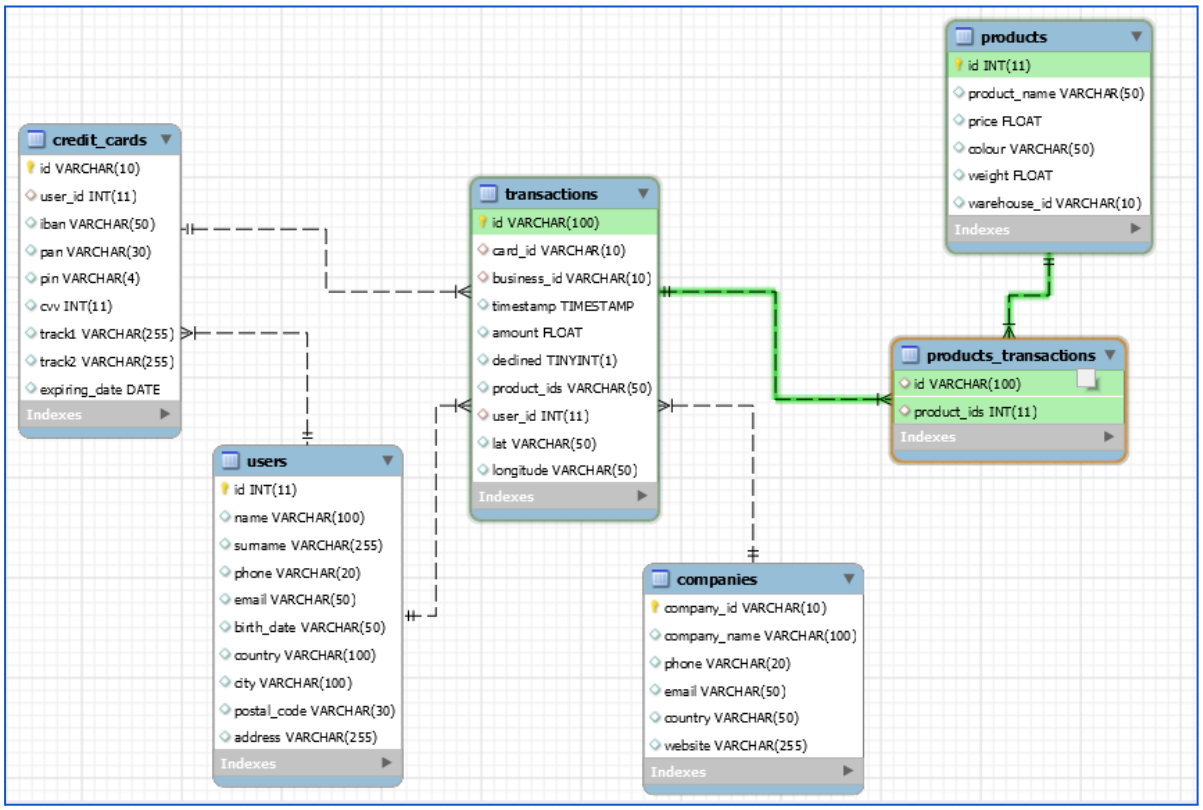
id	product_ids
02C6201E-D90A-1859-B4EE-88D2986D3B02	71
02C6201E-D90A-1859-B4EE-88D2986D3B02	1
02C6201E-D90A-1859-B4EE-88D2986D3B02	19
0466A42E-47CF-8D24-FD01-C0B689713128	47
0466A42E-47CF-8D24-FD01-C0B689713128	97
0466A42E-47CF-8D24-FD01-C0B689713128	43
063FBA79-99EC-66FB-29F7-25726D1764A5	47
063FBA79-99EC-66FB-29F7-25726D1764A5	67
063FBA79-99EC-66FB-29F7-25726D1764A5	31
063FBA79-99EC-66FB-29F7-25726D1764A5	5
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	89
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	83
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	79
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	43
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	31
07A46D48-31A3-7E87-65B9-0DA902AD109F	47
07A46D48-31A3-7E87-65B9-0DA902AD109F	23
09DE92CE-6F27-2BB7-13B5-9385B2B3B8E2	67
09DE92CE-6F27-2BB7-13B5-9385B2B3B8E2	7
0A476ED9-0C13-1962-F87B-D3563924B539	29

products_transactions 2

Output

Action Output

#	Time	Action	Message
1	21:31:36	SELECT * FROM transactions4.products_transactions LIMIT 0, 5000	1457 row(s) returned



3.1. Exercici 1

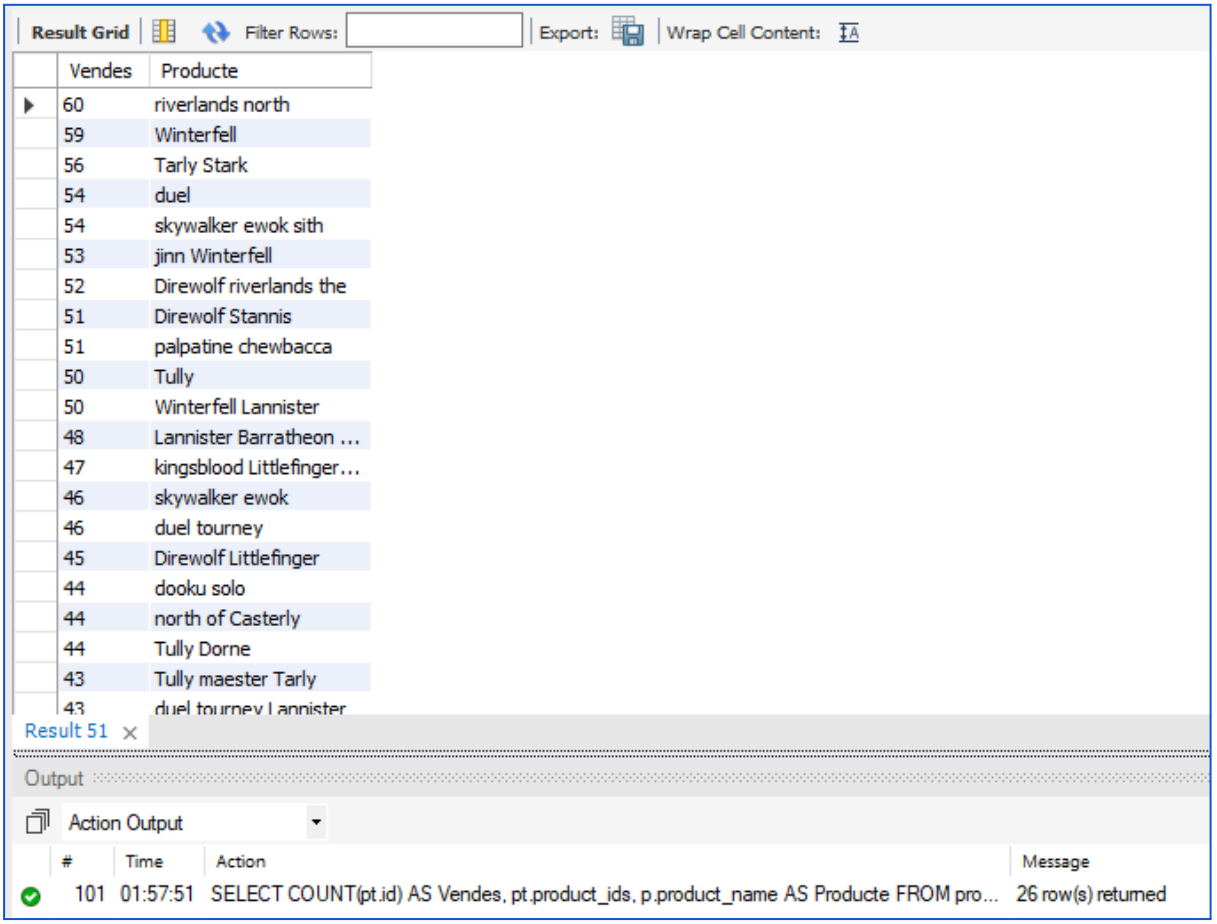
Necessitem conèixer el nombre de vegades que s'ha venut cada producte.

S'aplica el filtre declined = 0 (només les transaccions acceptades) perquè l'enunciat demana vendes. Si l'enunciat demanés l'amount total no caldria posar el filtre.

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

```sql
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