Nivell 1

Descàrrega 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:

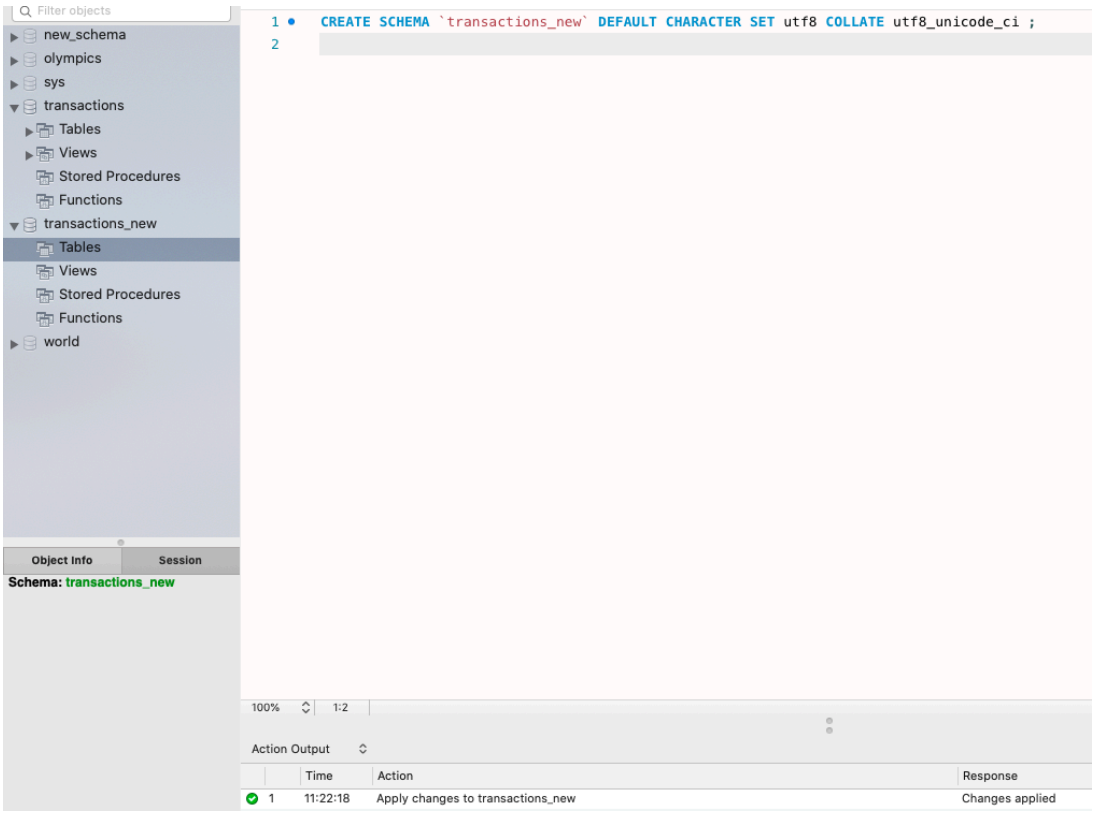
Primero analizamos los archivos CSV. Los campos y los tipos de datos. Las relaciones entre las tablas.

Importante: el campo “product_ids” de la tabla “transactions” tiene una lista de “id” de productos separados con coma. Y hay que crear una tabla intermedia para conectar a las tablas “transactions” y “products”. Para esto usaré una tabla “product_sold”.

Tres tablas de los usuarios tienen la misma estructura y las podemos unir en una misma tabla “users”.

					users	
transactions	companies	products	credit_cards	users_ca	users_uk	users_usa
id	company_id	id	id	id	id	id
card_id	company_name	product_name	user_id	name	name	name
business_id	phone	price	iban	surname	surname	surname
timestamp	email	colour	pan	phone	phone	phone
amount	country	weight	pin	email	email	email
declined	website	warehouse_id	cvv	birth_date	birth_date	birth_date
product_ids			track1	country	country	country
user_id			track2	city	city	city
lat			expiring_date	postal_code	postal_code	postal_code
longitude				address	address	address

Primero creamos una nueva base de datos con el nombre ¨transactions_new¨.
Codificando con estandar utf8 que nos permite usar unos caracteres con acentos (como Québec City).



Luego creamos las tablas para cargarlas con los archivos CSV.

```
CREATE TABLE IF NOT EXISTS  companies (
        company_id VARCHAR(6) PRIMARY KEY ,
        company_name VARCHAR(50),
        phone VARCHAR(15),
        email VARCHAR(50),
        country VARCHAR(50),
        website varchar(50)
    );

-- permitimos data loading
SET GLOBAL local_infile = 1;

LOAD DATA   INFILE '/Volumes/GoogleDrive/Mi unidad/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
IGNORE 1 LINES;
```

'local_infile'; [enabling local importing on the server]

Sale un error:
Error Code: 1290. The MySQL server is running with the --secure-file-priv option so it cannot execute this statement

Comprobamos donde esta la carpeta con los datos de mysql

```
1 • show variables like "secure_file_priv";
2 • show variables like "datadir"
```

100% 1:1

Result Grid Filter Rows: Search Export:

Variable_name	Value
datadir	/usr/local/mysql/data/

Tenemos que meter los archivos CSV a mysql a la carpeta de la nueva base de datos.
Para esto primero en el Terminal de MAC ejecutamos los comandos para dar el acceso:
sudo chmod -R 755 /usr/local/mysql/data/transactions_new

Nombre	Fecha de modificación	Tamaño	Clase
companies.csv	1 oct 2024 13:01	10 KB	valores...com
credit_cards.csv	1 oct 2024 13:03	41 KB	valores...com
products.csv	1 oct 2024 13:04	4 KB	valores...com
transactions.csv	1 oct 2024 13:04	73 KB	valores...com
users_ca.csv	1 oct 2024 13:04	9 KB	valores...com
users_uk.csv	1 oct 2024 13:04	6 KB	valores...com
users_usa.csv	1 oct 2024 13:04	19 KB	valores...com
companies.ibd	hoy 10:07	115 KB	Documento
credit_cards.ibd	hoy 10:07	164 KB	Documento

Macintosh HD › usr › local › mysql-8.0.21-macos10.15-x86_64 › data › transactions_new

Al intentar load sale otro error

```
LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
IGNORE 1 LINES;
```

-- Error Code: 29. File '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/companies.csv' not found (OS errno 13 - Permission denied)

En Terminal (repito para cada archivo CSV)
sudo chown mysql:mysql /usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/companies.csv
sudo chown mysql:mysql /usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/*.csv

Ahora podemos ejecutar los loads.

Tabla ¨companies¨

```
-- Creamos las tablas y insertamos en ellas datos de los archivos .csv
CREATE TABLE IF NOT EXISTS  companies (
        company_id VARCHAR(6) PRIMARY KEY ,
        company_name VARCHAR(50),
        phone VARCHAR(15),
        email VARCHAR(50),
        country VARCHAR(50),
        website varchar(50)
    );

-- permitimos data loading
SET GLOBAL local_infile = 1;

LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
IGNORE 1 LINES;
```

Tabla ¨transaction¨

```
CREATE TABLE IF NOT EXISTS  transactions (

        id VARCHAR(50) PRIMARY KEY ,
        card_id VARCHAR(15),
        business_id VARCHAR(6),
        timestamp  TIMESTAMP,
        amount  DECIMAL(10, 2),
        declined  BOOLEAN,
        product_ids VARCHAR(25),
        user_id VARCHAR(6),
        lat  VARCHAR(25),
        longitude VARCHAR(25)
    );

LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/transactions.csv'
INTO TABLE transactions
FIELDS TERMINATED BY ';' -- separador diferente de las otras tablas
ENCLOSED BY '"'
IGNORE 1 LINES;
```

Tabla ¨credit_cards¨

```
CREATE TABLE IF NOT EXISTS  credit_cards (
        id VARCHAR(15) PRIMARY KEY ,
        user_id VARCHAR(6),
        iban VARCHAR(50),
        pan VARCHAR(50),
        pin CHAR(4),
        cvv CHAR(3),
        track1 VARCHAR(50),
        track2 VARCHAR(50),
        expiring_date varchar(10)
        );

LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/credit_cards.csv'
INTO TABLE credit_cards
FIELDS TERMINATED BY ',' -- separador coma
ENCLOSED BY '"'
IGNORE 1 LINES;
```

Tabla ¨products¨

```
CREATE TABLE IF NOT EXISTS  products (
        id VARCHAR(5) PRIMARY KEY ,
        product_name VARCHAR(50),
        price VARCHAR(25),
        colour VARCHAR(25),
        weight FLOAT,
        warehouse_id varchar(6)
        );

LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/products.csv'
INTO TABLE products
FIELDS TERMINATED BY ',' -- separador coma
ENCLOSED BY '"'
IGNORE 1 LINES;
```

Tabla ¨users¨

```
CREATE TABLE IF NOT EXISTS  users (
        id VARCHAR(5) PRIMARY KEY ,
        name VARCHAR(50),
        surname VARCHAR(50),
        phone VARCHAR(25),
        email VARCHAR(50),
        birth_date VARCHAR(20),
        country VARCHAR(25),
        city VARCHAR(25),
        postal_code VARCHAR(15),
        address VARCHAR(100)
        );

-- el campo address en algunos casos contiene coma.
-- Tenemos la  condicion ENCLOSED BY '"' y añadimos un marcador del fin del registro LINES TERMINATED BY '\r\n'
-- \r\n is a newline and a carriage return. Used as a new line character in Windows

LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/users_ca.csv'
INTO TABLE users
FIELDS TERMINATED BY ',' -- separador coma
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
;
LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/users_usa.csv'
INTO TABLE users
FIELDS TERMINATED BY ',' -- separador coma
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
;
LOAD DATA   INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/users_uk.csv'
INTO TABLE users
FIELDS TERMINATED BY ',' -- separador coma
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
;
```

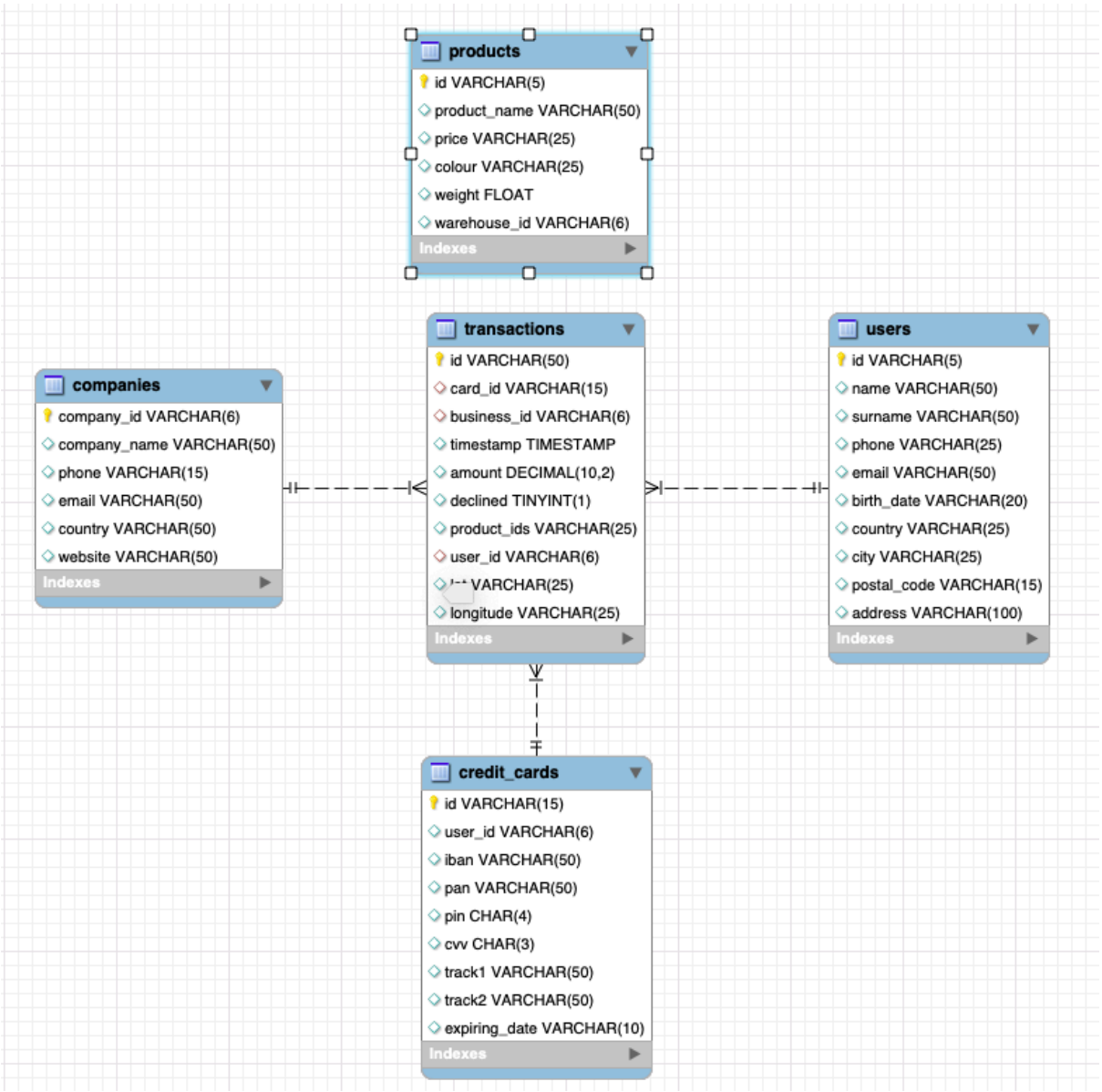
Diagrama EER despues de ejecutar el codigo de arriba

Establecemos las relaciones entre las tablas

```
ALTER TABLE transactions
add foreign key(card_id) references credit_cards(id),
add foreign key(business_id) references companies(company_id),
add foreign key(user_id) references users(id);
```

✓	1	00:09:49	USE transactions_new	0 row(s) affected
✓	2	00:09:49	CREATE TABLE IF NOT EXISTS companies (company_id VARCHAR(6) PRIMARY KEY , com...	0 row(s) affected
✓	3	00:09:49	SET GLOBAL local_infile = 1	0 row(s) affected
✓	4	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/companies....	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0
✓	5	00:09:49	CREATE TABLE IF NOT EXISTS transactions (id VARCHAR(50) PRIMARY KEY , card_id VARCHAR...	0 row(s) affected
✓	6	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/transaction...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0
✓	7	00:09:49	CREATE TABLE IF NOT EXISTS credit_cards (id VARCHAR(15) PRIMARY KEY , user_id VARCHAR(...	0 row(s) affected
✓	8	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/credit_cards...	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0
✓	9	00:09:49	CREATE TABLE IF NOT EXISTS products (id VARCHAR(5) PRIMARY KEY , product_name VARCHA...	0 row(s) affected
✓	10	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/products.cs...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0
✓	11	00:09:49	CREATE TABLE IF NOT EXISTS users (id VARCHAR(5) PRIMARY KEY , name VARCHAR(50),...	0 row(s) affected
✓	12	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/users_ca.cs...	75 row(s) affected Records: 75 Deleted: 0 Skipped: 0 Warnings: 0
✓	13	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/users_usa.c...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0
✓	14	00:09:49	LOAD DATA INFILE '/usr/local/mysql-8.0.21-macos10.15-x86_64/data/transactions_new/users_uk.cs...	50 row(s) affected Records: 50 Deleted: 0 Skipped: 0 Warnings: 0
✓	15	00:09:49	ALTER TABLE transactions add foreign key(card_id) references credit_cards(id), add foreign key(busi...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Diagrama EER despues de ejecutar el codigo de arriba



- Exercici 1

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

La solución utilizando subconsulta, como solicitado

```
126 •  SELECT *, (select count(user_id) from transactions where user_id = users.id) as num_transactions
127    FROM users
128    WHERE id in(
129            select user_id
130            from transactions
131            group by user_id
132            having count(user_id) > 30
133            order by count(user_id) desc
134    )
135    order by num_transactions desc;
136    ;
137
```

100% 1:138

Result Grid Filter Rows: Search Export:

id	name	surname	phone	email	birth_date	country	city	postal_code	address	num_transa
272	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	Apr 16, 1991	Canada	Tuktoyaktuk	Q4C 3G7	P.O. Box 496, 5145 Sapien Road	76
267	Ocean	Nelson	079-481-2745	aenean@yahoo.com	Dec 26, 1991	Canada	Charlottetown	85X 3P4	Ap #732-8357 Pede, Rd.	52
275	Kenyon	Hartman	082-871-7248	convallis.ante.lectus@yahoo.com	Aug 3, 1982	Canada	Richmond	R8H 2K2	8564 Facilisi. St.	48
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.	39

16 00:26:13 SELECT *, (select count(user_id) from transactions where transactions.user_id = users.id) as num_tr... 4 row(s) returned

No está especificado si tomamos en cuenta las transacciones declinadas. Considero tanto las transacciones declinadas como las no declinadas.

- Exercici 2

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

```
1 •  SELECT c.company_name, cc.iban, round(avg(t.amount),2) as avg_transaction
2    FROM transactions t
3            left outer join companies c on t.business_id=c.company_id
4            left outer join credit_cards cc on t.card_id=cc.id
5    WHERE c.company_name='Donec Ltd'
6    GROUP BY c.company_name,cc.iban
7    ;
```

100% 32:6

Result Grid Filter Rows: Search Export:

company_name	iban	avg_transaction
Donec Ltd	PT87806228135092429456346	203.72

158 12:00:20 SELECT c.company_name, cc.iban, round(avg(t.amount),2) as avg_transaction FROM transactions t... 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:

```sql
CREATE TABLE cc_state AS
-- depende de si la targeta tiene tres ultimas transacciones declinadas le ponemos 'No activa' si no 'Activa'
SELECT  card_id,
        if (sum(declined)>=3,'No activa', 'Activa') as cc_state
FROM (

-- esta subquery es el primer paso
-- para cada tarjeta de credito numeramos sus transaciones de mas reciente (numero 1) a mas antigua (numero n)
-- para eso usamos dos funcciones ROW_NUMBER y PARTITION BY
SELECT  card_id,
        CAST(timestamp AS DATE) ,
        declined, ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp desc) as 'RN_OrderDate'
FROM transactions
    ) a

-- aqui elegimos solo las tres transaciones mas recientes
WHERE RN_OrderDate<4
GROUP BY card_id
```

```
237  13:21:14  CREATE TABLE cc_state AS  -- depende de si la targeta tiene tres ultimas transacciones declinadas l...  275 row(s) affected Records: 275  Duplicates: 0  War...
```

Hacemos las conexiones

```sql
ALTER TABLE cc_state
ADD FOREIGN KEY (card_id) references credit_cards(id);
```

```sql
SELECT * FROM cc_state;
```

100% 15:1

Result Grid Filter Rows:

card_id	cc_state
CcU-2952	Activa
CcU-2959	Activa
CcU-2966	Activa
CcU-2973	Activa
CcU-2980	Activa
CcU-2987	Activa
CcU-2994	Activa
CcU-3001	Activa
CcU-3008	Activa
CcU-3015	Activa
CcU-3022	Activa
CcU-3029	Activa
CcU-3036	Activa
CcU-3043	Activa
CcU-3050	Activa
CcU-3057	Activa
CcU-3064	Activa
CcU-3071	Activa
CcU-3078	Activa

Exercici 1

Quantes targetes estan actives?

```
1  SELECT cc_state,
2      count(card_id) as num_credit_cards
3  FROM cc_state
4  WHERE cc_state='Activa'
```

100% 24:4

Result Grid Filter Rows: Search Export:

cc_state	num_credit_cards
Activa	275

37 11:37:56 SELECT cc_state, count(card_id) as num_credit_cards FROM cc_state WHERE cc_state='Activa' 1 row(s) returned

El resultado de la query es que todas las tarjetas están activas.
Sin embargo con el cliente se puede trabajar más los requisitos de una tarjeta inactiva. Por ejemplo, considerar una tarjeta inactiva si tiene todas las transacciones ¨declined¨ aunque son menos de 3, etc.

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:

El campo ¨product_ids¨ es un string donde se guardan los ids de productos vendidos en cada transacción separados por coma.

Para poder usar esta información hay que crear una tabla intermedia “product_sold” para conectar la tabla de dimensión “productos” con la tabla de hechos “trancactions”.

Solo tiene dos campos el número de transacción (transaction_id) y el producto vendido (product_id).

```
CREATE TABLE IF NOT EXISTS product_sold
(transaction_id VARCHAR(50),
product_id VARCHAR(5),
PRIMARY KEY (transaction_id, product_id),
FOREIGN KEY (transaction_id) references transactions(id),
FOREIGN KEY (product_id) references products(id));
```

Opcion 1 con substring_index

Paso 1

n es el máximo de los productos vendidos en cada transacción

```
-- creamos una tabla temporal que consiste de un campo con los numeros de 1 a 10
-- CTE (common table expression) create temporary result set
-- RECURSIVE for generating serie of numbers
CREATE TEMPORARY TABLE numbers  WITH RECURSIVE  cte AS
( SELECT 1 as n
  UNION ALL
  SELECT n +1
  FROM cte -- refers to itself to repeatedly add new values
  WHERE n < 10 -- termination condition
)
SELECT * FROM cte;
```

```sql
select *
from numbers
```

100% 9:1

Result Grid Filter Rows

n
1
2
3
4
5
6
7
8
9
10

Paso 2

Creamos la tabla “product_sold”

Con la función substring_index sacamos los id de cada producto vendido

Join con la tabla temporal ¨numbers¨nos permite hacerlo tantas veces cuantos productos están registrados para cada transacción.

```sql
INSERT INTO product_sold (transaction_id, product_id)
SELECT  id as transaction_id,
    trim(substring_index( substring_index(product_ids, ',', n), ',', -1 )) as product_id
FROM transactions join numbers
    on  char_length(product_ids) - char_length(replace(product_ids, ',', ''))+1 >= n
;
```

```
1 • SELECT * FROM transactions_new.product_sold;
```

100% 1:1

Result Grid Filter Rows: Search Edit:

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

Opcion 2 con find_in_set

```
INSERT INTO product_sold (transaction_id, product_id)
SELECT t.id, p.id
FROM transactions t
inner join products p on find_in_set(p.id, replace(t.product_ids,', ',','))
where declined = 0;
```

Comprobamos

```
1 • SELECT * FROM transactions_new.product_sold;
```

100% 1:1

Result Grid Filter Rows: Search Edit: Ex

transaction_id	product_id
02C6201E-D90A-1859-B4EE-88D2986D3B02	1
02C6201E-D90A-1859-B4EE-88D2986D3B02	19
02C6201E-D90A-1859-B4EE-88D2986D3B02	71
0466A42E-47CF-8D24-FD01-C0B689713128	43
0466A42E-47CF-8D24-FD01-C0B689713128	47
0466A42E-47CF-8D24-FD01-C0B689713128	97
063FBA79-99EC-66FB-29F7-25726D1764A5	31
063FBA79-99EC-66FB-29F7-25726D1764A5	47
063FBA79-99EC-66FB-29F7-25726D1764A5	5
063FBA79-99EC-66FB-29F7-25726D1764A5	67
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	79
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	83
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	89
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	31
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	43

Exercici 1

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

```
202     -- el numero de ventas de cada producto
203 •   SELECT p.id, product_name, count(ps.transaction_id) as num_sold
204     from  product_sold ps
205     inner join products p on product_id = p.id
206     inner join transactions t on transaction_id=t.id
207     where declined=0
208     group by p.id, product_name
209     order by num_sold desc;
210
```

100% 1:210

Result Grid Filter Rows: Search Export:

id	product_name	num_sold
23	riverlands north	68
67	Winterfell	68
79	Direwolf riverlands the	66
2	Tarly Stark	65
43	duel	65
47	Tully	62
1	Direwolf Stannis	61
17	skywalker ewok sith	61
97	jinn Winterfell	61
13	palpatine chewbacca	60

Result 4 Result 5 Result 6

Action Output

	Time	Action	Response
43	15:32:13	SELECT p.id, product_name, count(ps.transaction_id) as num_sold from product_sold ps inner join p...	26 row(s) returned

Diagrama

El diagrama final que tenemos para nuestra base de datos