

## 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:**

### 1 - Tenemos que crear una base de datos que llamaremos Transacciones

```
CREATE DATABASE transacciones
```

### 2- Creamos la primera tabla maestro llamada companies e importamos los datos.

```
CREATE TABLE companies (  
company_id VARCHAR(15) PRIMARY KEY,  
company_name VARCHAR(255),  
phone VARCHAR(15),  
email VARCHAR(100),  
country VARCHAR(100),  
Website VARCHAR(100) );
```

Procedemos a cargar el fichero .csv con los datos con la siguiente instrucción LOAD DATA. Me da el siguiente error:

"Error code: 1290. the MYSQL server is running with the --secure-file-priv option so it cannot execute this statement".

En internet, he encontrado que eso significa que sólo podemos cargar los datos que están en una determinada carpeta por seguridad.

Con la siguiente instrucción, obtenemos la carpeta en la que debemos guardar el archivo.

```
SHOW VARIABLES LIKE "secure_file_priv"
```

Resultado de la anterior búsqueda es:

```
C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\
```

Sólo podemos cargar ficheros CSV que estén en ese directorio.

LOAD DATA

INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv"

INTO TABLE companies

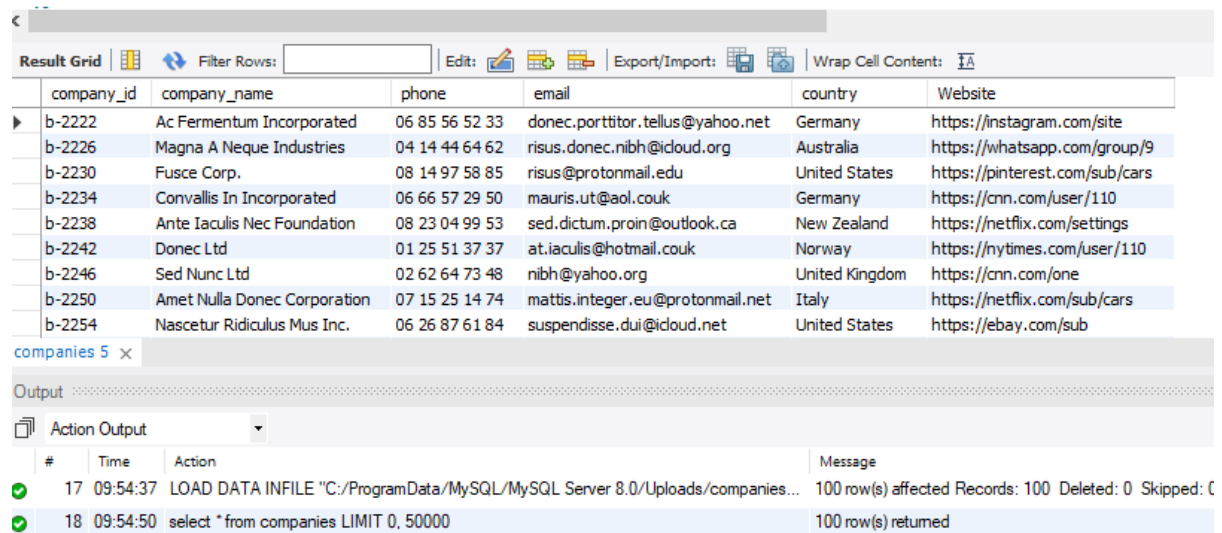
FIELDS TERMINATED BY ','

IGNORE 1 ROWS;

Miramos que se haya cargado todo

SELECT \* FROM companies;

41 • `select * from companies;`



	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

companies 5 x

Output

Action Output

#	Time	Action	Message
✓ 17	09:54:37	LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0
✓ 18	09:54:50	select * from companies LIMIT 0, 50000	100 row(s) returned

### 3 –Creamos la tabla maestro credit\_cards e importamos los datos:

```
CREATE TABLE credit_cards (  
  
    id VARCHAR(15) PRIMARY KEY,  
  
    user_id INT,  
  
    iban VARCHAR(50),  
  
    pan VARCHAR(30),  
  
    pin VARCHAR(4),  
  
    cvv INT,  
  
    track1 VARCHAR(200),  
  
    track2 VARCHAR(200),
```

```
expiring_date VARCHAR(10));
```

Comandos para cargar los datos:

```
LOAD DATA INFILE
```

```
"C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv"
```

```
INTO TABLE credit_cards
```

```
FIELDS TERMINATED BY ','
```

```
IGNORE 1 ROWS;
```

Miramos que se haya cargado todo.

73 • `SELECT * FROM credit_cards;`

	id	user_id	iban	pan	pin	cvv	track1	track2
▶	CcU-2938	275	TR301950312213576817638661	5424465566813633	3257	984	%88383712448554646^WovsxejDpwiev^8604...	%87653863056044187=800716333673
	CcU-2945	274	DO26854763748537475216568689	5142423821948828	9080	887	%84621311609958661^UftuyfsSeimxn^06106...	%84149568437843501=510714033071
	CcU-2952	273	BG45IVQL52710525608255	4596 453 55 5287	4598	438	%82183285104307501^CddyttUxwfdq^5907...	%86778580257827162=69068597400?
	CcU-2959	272	CR7242477244335841535	372461377349375	3583	667	%87281111956795320^XocddjBdecd^09016...	%84246154489281853=280522391678
	CcU-2966	271	BG72LKTQ15627628377363	448566 886747 7265	4900	130	%847289332322756223^JhlgvsuFbmwgj^7202...	%82318571115599881=890821578475
	CcU-2973	270	PT87806228135092429456346	544 58654 54343 384	8760	887	%84761405253275637^HjnnipoBlejrl^7108515...	%87816169831446746=1310277279
	CcU-2980	269	DE39241881883086277136	402400 7145845969	5075	596	%87320483593870549^OokzqxrtHpsed^4901...	%82474313962214151=041221913175
	CcU-2987	268	GE89681434837748781813	3763 747687 76666	2298	797	%84750646345146674^Pjnlrfgwwtrf^83051...	%85441935173418615=410370453677

credit\_cards 9 x

Output

Action Output

#	Time	Action	Message
✓ 26	10:03:25	LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_card...	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0
✓ 27	10:03:32	SELECT * FROM credit_cards LIMIT 0, 50000	275 row(s) returned

#### 4 \_Creamos la tabla maestro users e importamos los datos:

```
CREATE TABLE users (
```

```
Id INT PRIMARY KEY,
```

```
name VARCHAR(100),
```

```
surname VARCHAR(100),
```

```
phone VARCHAR(100),
```

```
Email VARCHAR(150),
```

```
birth_date VARCHAR(100),
```

```
country VARCHAR(150),  
city VARCHAR(150),  
postal_code VARCHAR(100),  
address VARCHAR(125));
```

Importamos los datos con el fichero con la instrucción LOAD FILE

Cargamos los datos de users\_ca:

```
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;
```

Cargamos los datos de users\_uk:

```
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;
```

Cargamos los datos de users\_usa:

## 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;
```

Las tablas usuarios nos ha dado problemas de carga porque hay saltos de páginas y algunos campos están incluido entre comillas. Por eso, añadimos ENCLOSED BY Y LINES TERMINATED BY

También había diferentes formatos de postal code, pero como hemos definido la variable postcode como VARCHAR no hemos tenido problema

Revisamos los campos cargados.

The screenshot shows a database management tool interface. At the top, a SQL query is entered: `SELECT * FROM users;`. Below the query, a table titled "Result Grid" displays the data. The table has 10 columns: Id, name, surname, phone, Email, birth\_date, country, city, postal\_code, and address. It contains 9 rows of user data. Below the table, there is an "Output" section with a table showing the execution log. The log has 4 columns: #, Time, Action, and Message. It shows two actions: a successful "LOAD DATA INFILE" operation at 11:32:07 and a successful "SELECT \* FROM users LIMIT 0, 50000" operation at 11:36:50.

Id	name	surname	phone	Email	birth_date	country	city	postal_code	address
1	Zeus	Gamble	1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	Lowell	73544	348-7818 Sagittis St.
2	Garrett	Mcconnell	(718) 257-2412	integer.vitae.nibh@protonmail.org	Aug 23, 1992	United States	Des Moines	59464	903 Sit Ave
3	Ciaran	Harrison	(522) 598-1365	interdum.feugiat@aol.org	Apr 29, 1998	United States	Columbus	56518	736-2063 Tellus St.
4	Howard	Stafford	1-411-740-3269	ornare.egestas@idoud.edu	Feb 18, 1989	United States	Kailua	77417	Ap #545-2244 Erat. Rd.
5	Hayfa	Pierce	1-554-541-2077	et.malesuada.fames@hotmail.org	Sep 26, 1998	United States	Sandy	31564	341-2821 Ultrices Av.
6	Joel	Tyson	(718) 288-8020	gravida.nunc.sed@yahoo.ca	Oct 15, 1989	United States	Nashville	96838	888-2799 Amet Street
7	Rafael	Jimenez	(817) 689-0478	eget@outlook.ca	Dec 4, 1981	United States	Hillsboro	29874	8627 Malesuada Rd.
8	Nissim	Franks	(692) 157-3469	egestas.aliquam.fringilla@google.ca	Aug 1, 1993	United States	Jackson	61750	Ap #251-7144 Integer St.
9	Mannix	Mcclain	(590) 883-2184	aliquam.nisl@outlook.com	Jan 24, 1987	United States	Richmond	35987	647-3080 Lacus. St.

#	Time	Action	Message
57	11:32:07	LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Users_usa...."	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0
58	11:36:50	SELECT * FROM users LIMIT 0, 50000	275 row(s) returned

## 5 – Creamos la tabla de hechos transactions:

```
CREATE TABLE transactions (
```

```
Id VARCHAR(255),
```

```
card_id VARCHAR(15),
```

```

business_id VARCHAR(25),

Timestamp TIMESTAMP,

Amount DECIMAL (10,2),

Declined TINYINT(1),

product_ids VARCHAR(20),

user_id INT,

Lat FLOAT,

Longitude FLOAT,

CONSTRAINT FK_user_id FOREIGN KEY (user_id) REFERENCES users(id),

CONSTRAINT FK_card_id FOREIGN KEY (card_id) REFERENCES credit_cards(id),

CONSTRAINT FK_business_id FOREIGN KEY (business_id) REFERENCES companies(company_id)

);

```

Hemos olvidado decir que el campo ID de la tabla transactions será la PRIMARY KEY.

La añadiremos con ALTER TABLE

```

ALTER TABLE transactions

ADD PRIMARY KEY (id);

```

Cargamos los datos en la tabla transactions.

```

LOAD DATA

INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv"

INTO TABLE transactions

FIELDS TERMINATED BY ','

IGNORE 1 ROWS;

```

176 • `SELECT * FROM transactions;`

	Id	card_id	business_id	Timestamp	Amount	Declined	product_ids	user_id	Lat	Longitude
▶	108B1D1D-5B23-A76C-55EF-C568E49A05DD	CcU-2938	b-2222	2021-07-07 17:43:16	293.57	0	59	275	83.7839	-178.86
	7DC26247-20EC-53FE-E555-B6C2E55CA5D5	CcU-2945	b-2226	2022-02-04 15:52:56	312.50	0	71, 41	275	58.9367	-76.8171
	72997E96-DC2C-A4D7-7C24-66C302F8AE5A	CcU-2952	b-2230	2022-01-30 15:16:36	239.87	0	97, 41, 3	275	43.3584	-17.658
	A8069F53-965E-A2A8-CE06-CA8C4FD92501	CcU-2959	b-2234	2021-04-15 13:37:18	60.99	0	11, 13, 61, 29	275	1.64819	-158.007
	2F3B6AB6-147D-EB0B-FE8D-9A4E2EA9DBD5	CcU-2966	b-2238	2021-10-18 06:12:03	33.81	0	47, 37, 11, 1	275	-43.4811	16.6025
	5B0EEF86-B8A1-EFAA-5EE1-27E7DC8F54A4	CcU-2973	b-2242	2022-01-06 01:44:48	42.82	0	23, 19, 71	275	-64.1136	85.2491
	2B928E1C-EC14-A760-0A75-871477649D6A	CcU-2980	b-2246	2021-08-10 08:14:49	383.73	0	59, 13, 23	275	-41.0496	161.685
	063FBA79-99EC-66FB-29F7-25726D1764A5	CcU-2987	b-2250	2022-01-06 21:25:27	92.61	0	47, 67, 31, 5	275	-81.2227	-129.05
	D33B21EB-A61E-BD8C-SDE7-1DA4AC210AE6	CcU-2994	b-2254	2022-03-02 13:29:50	20.35	0	17, 13, 73	275	5.27302	5.06131

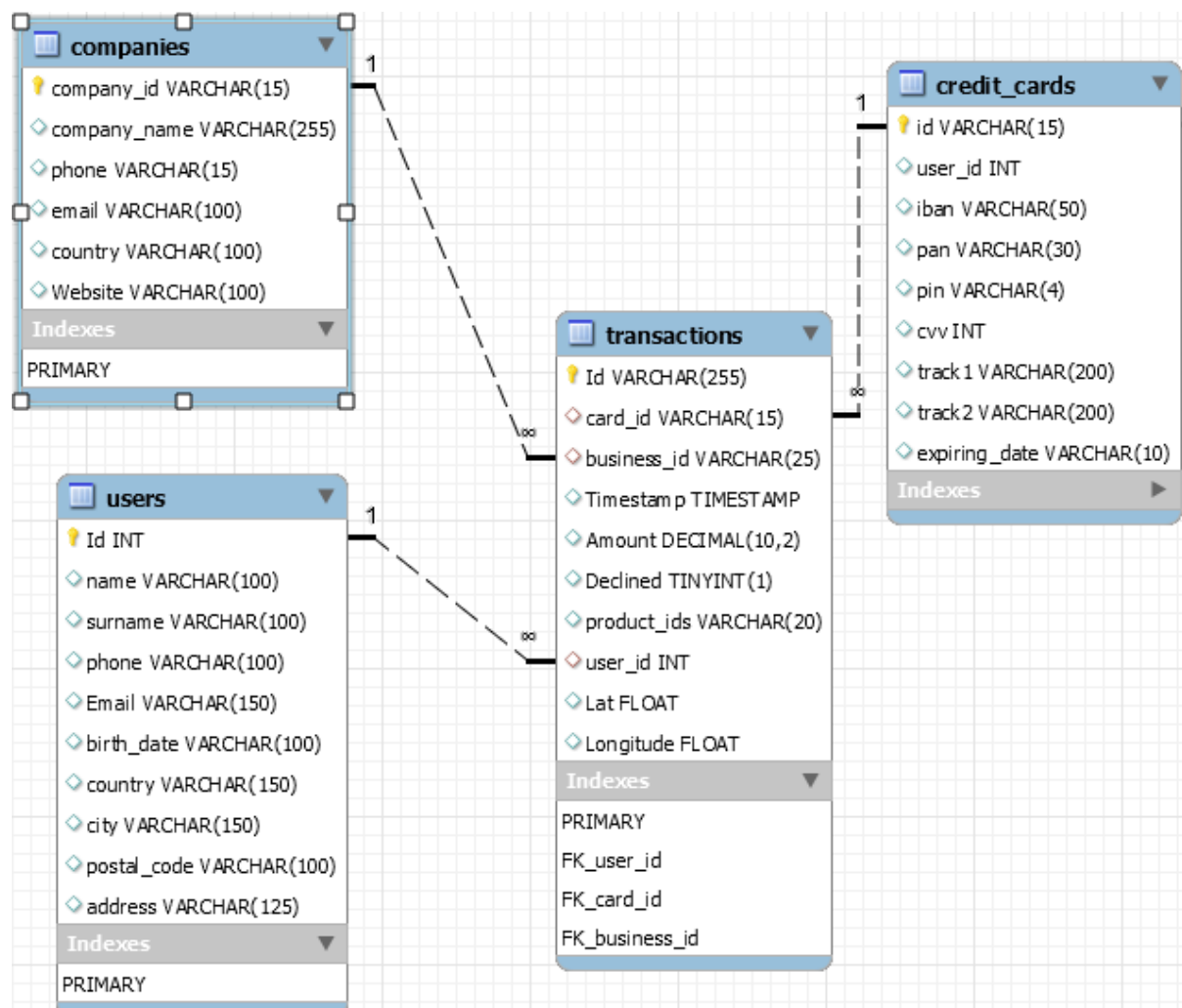
transactions 17 x

Output

Action Output

#	Time	Action	Message
60	11:40:53	LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transaction...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0
61	11:41:22	SELECT * FROM transactions LIMIT 0, 50000	587 row(s) returned

La unión de las cuatro tablas quedará así:



## Exercici 1

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

### Subconsulta:

Unimos las tablas transactions y users con un JOIN y luego agrupamos las transacciones por usuarios (contamos el número transacciones) y ordenamos por campo suma transacciones orden descendiente

```
SELECT users.Id, users.name, users.surname, count(transactions.id) AS transacciones
FROM transacciones.transactions
JOIN transacciones.users
ON transactions.user_id = users.Id
GROUP BY users.Id, users.name, users.surname
HAVING transacciones > 30
ORDER BY 3 DESC;
```

### Query final.

Con la consulta anterior haremos un derived table de la cual extraeremos los datos que necesitamos:

```
SELECT subquery.ID, subquery.Name, subquery.Surname, subquery.transacciones
from ( SELECT users.Id as ID, users.name AS Name, users.surname AS
Surname, count(transactions.id) AS transacciones
FROM transacciones.transactions
JOIN transacciones.users
ON transactions.user_id = users.Id
GROUP BY users.Id, users.name, users.surname
HAVING transacciones > 30
```



ORDER BY 3 DESC) subquery;

Resultados de la consulta:

```
196 • SELECT subquery.ID, subquery.Name, subquery.Surname, subquery.transacciones
197 from ( SELECT users.Id as ID, users.name AS Name, users.surname AS Surname, count(transactions.id) AS transacciones
198 FROM transacciones.transactions
199 JOIN transacciones.users
200 ON transactions.user_id = users.Id
201 GROUP BY users.Id, users.name, users.surname
202 HAVING transacciones > 30
203 ORDER BY 3 DESC) subquery;
204
```

	ID	Name	Surname	transacciones
▶	92	Lynn	Riddle	39
	267	Ocean	Nelson	52
	275	Kenyon	Hartman	48
	272	Hedwig	Gilbert	76

Result 45 x

Output

#	Time	Action	Message
✖ 101	13:02:37	SELECT subquery.ID, subquery.Name, subquery.Surname, subquery.transacciones from ...	Error Code: 1054. Unknown column 'subquery.transacciones' in 'field list'
✔ 102	13:03:12	SELECT subquery.ID, subquery.Name, subquery.Surname, subquery.transacciones from...	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.**

Tenemos que unir las tablas transactions, credit\_cards y companies, para obtener los campos IBAN, Amount y Company\_name. Después agrupamos por iban, hacemos el promedio del amount y añadimos una condición en el WHERE para indicar que la compañía ha de ser la Donec LTD.

```
SELECT credit_cards.iban, ROUND(AVG(transactions.Amount),2)

FROM transacciones.transactions

JOIN transacciones.credit_cards

ON transactions.card_id = credit_cards.id

JOIN transacciones.companies

ON transactions.business_id= companies.company_id

WHERE companies.company_name = "Donec Ltd"

GROUP BY credit_cards.iban;
```

Resultados de la consulta:

```
220 • SELECT credit_cards.iban, ROUND(AVG(transactions.Amount),2)
221 FROM transacciones.transactions
222 JOIN transacciones.credit_cards
223 ON transactions.card_id = credit_cards.id
224 JOIN transacciones.companies
225 ON transactions.business_id= companies.company_id
226 WHERE companies.company_name = "Donec Ltd"
227 GROUP BY credit_cards.iban;
```

Result Grid

iban	ROUND(AVG(transactions.Amount),2)
PT87806228135092429456346	203.72

Result 3 x

Output

Action Output

#	Time	Action	Message
✓ 1	22:28:37	SELECT credit_cards.iban, ROUND(AVG(transactions.Amount),2) FROM transacciones.tr...	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 a ser declinades i genera la següent consulta:**

Tenemos que empezar creando la tabla que nos piden

### Subquery

Con esta subquery sacaremos las 3 últimas transacciones que tiene cada tarjeta. Usaremos el comando ROW\_NUMBER().

con el cual haremos una partición de la columna card\_id y seleccionaremos las tres últimas transacciones de cada tarjeta.

```

SELECT card_id, declined, timestamp,

ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS rownum

FROM transactions;

```

Usamos la anterior subquery para hacer una "derived table", de la cual cogeremos la tarjeta de crédito, declined y timestamp. Agrupamos por card\_id y sumamos por declined

Finalmente haremos usando el comando CASE crearemos una nueva columna en la que incluiremos el comentario si está activa o no.

### Query final

```

SELECT card_id AS id, SUM(declined) AS declined,

CASE

    WHEN SUM(declined) >=3 THEN "INACTIVA"

    ELSE "ACTIVA"

END AS comments

FROM (SELECT card_id AS card_id, declined AS declined, timestamp,

        ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS
        rownum

    FROM transactions) as subquery

WHERE subquery.rownum <=3

GROUP BY card_id ;

```

Resultado de la query:

Me sale que todas están activas

```

7 • SELECT card_id AS id, SUM(declined) AS declined,
8 CASE
9     WHEN SUM(declined) >=3 THEN "INACTIVA"
10    ELSE "ACTIVA"
11 END AS comments
12 FROM (SELECT card_id AS card_id, declined AS declined, timestamp, ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS rownum
13 FROM transactions) as subquery
14 WHERE subquery.rownum <=3
15 GROUP BY card_id ;

```

id	declined	comments
CcJ-2938	0	ACTIVA
CcJ-2945	1	ACTIVA
CcJ-2952	1	ACTIVA
CcJ-2959	0	ACTIVA
CcJ-2966	1	ACTIVA
CcJ-2973	1	ACTIVA

#	Time	Action	Message	Duration / Fetch
30	09:42:40	SELECT credit_cards.iban, AVG(transactions.Amount) FROM transacciones.transaction...	1 row(s) returned	0.094 sec / 0.000
31	09:51:26	SELECT card_id AS id, SUM(declined) AS declined, CASE WHEN SUM(declined) >=...	275 row(s) returned	0.000 sec / 0.000

Creamos una tabla en la para la tarjeta: card\_status

```

CREATE TABLE Card_Status(

id VARCHAR(15) PRIMARY KEY,

declined TINYINT(1),

comments VARCHAR(25));

```

Insertamos los datos de la la busqueda en nueva tabla

```

INSERT INTO card_status(id,declined,comments)

SELECT card_id AS id, SUM(declined) AS declined,

CASE

    WHEN SUM(declined) >=3 THEN "INACTIVA"

    ELSE "ACTIVA"

END AS comments

FROM (SELECT card_id AS card_id, declined AS declined, timestamp, ROW_NUMBER()

OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS rownum

FROM transactions) as subquery

WHERE subquery.rownum <=3

GROUP BY card_id;

```

Consulta para ver que todos los datos se han cargado bien

```
SELECT * FROM CARD_STATUS;
```

The screenshot shows a database interface with a SQL query editor at the top containing the query: `SELECT * FROM CARD_STATUS;`. Below the editor is a toolbar with options like 'Result Grid', 'Filter Rows', 'Edit', 'Export/Import', and 'Wrap Cell Content'. The main area displays a table with the following data:

	id	declined	comments
▶	CcU-2938	0	ACTIVA
	CcU-2945	1	ACTIVA
	CcU-2952	1	ACTIVA
	CcU-2959	0	ACTIVA
	CcU-2966	1	ACTIVA
	CcU-2973	1	ACTIVA

Below the table, there is a tab labeled 'CARD\_STATUS 20'. At the bottom, an 'Output' section shows a log of actions:

#	Time	Action	Message
✓ 31	09:51:26	SELECT card_id AS id, SUM(declined) AS declined, CASE WHEN SUM(declined) >=...	275 row(s) returned
✓ 32	09:56:03	SELECT * FROM CARD_STATUS LIMIT 0, 50000	275 row(s) returned

Ahora tenemos que enlazar esta nueva tabla card\_status con la tabla credit\_cards. Lo haremos con el ALTER TABLE

```
ALTER TABLE card_status
```

```
ADD CONSTRAINT FK_id FOREIGN KEY (id) REFERENCES credit_cards(id);
```

\*Ejercicio 1 ¿Cuántas tarjetas están activas?\*

```
SELECT COUNT(id)
```

```
FROM card_status
```

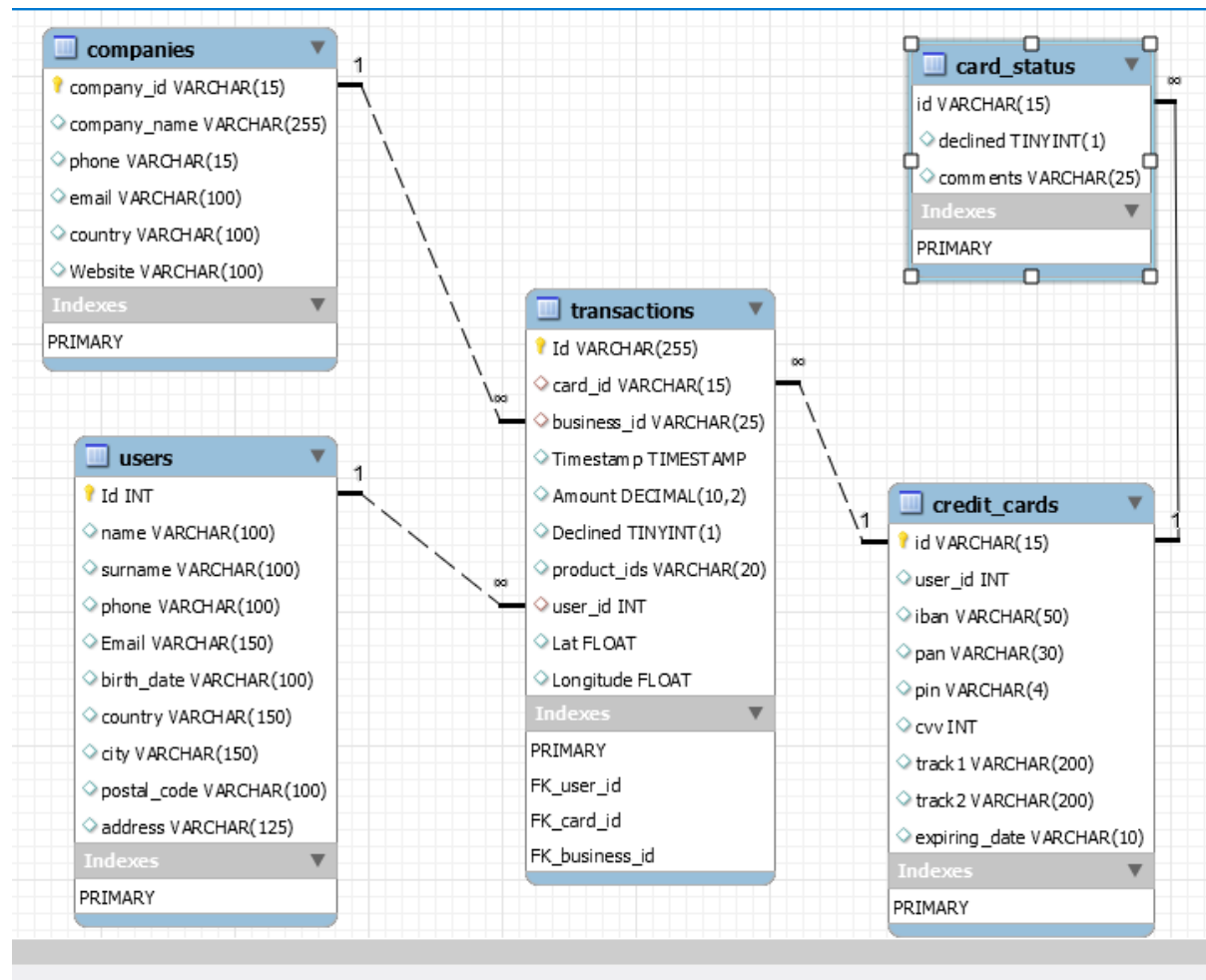
```
WHERE comments = "ACTIVA";
```

Me sale que todas las tarjetas están activas:

The screenshot shows a database interface with a SQL query editor containing the query: `SELECT COUNT(id) FROM card_status WHERE comments = "ACTIVA";`. Below the editor is a toolbar with options like 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. The main area displays a single-row table with the following data:

COUNT(id)
275

El nuevo EER diagrama quedará así:



### 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.

#### 1 - Creamos la tabla product e insertamos los datos.

```

CREATE TABLE products (
  Id INT PRIMARY KEY,
  product_name VARCHAR(255),
  price DECIMAL (10,2),
  colour VARCHAR(100),
  weight DECIMAL (10,2),

```

```
warehouse_id VARCHAR(25)

);
```

Cargamos los datos en la tabla products.

LOAD DATA

```
INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv"
```

```
INTO TABLE products
```

```
FIELDS TERMINATED BY ','
```

```
IGNORE 1 ROWS;
```

Result Grid										
Filter Rows:										
Edit: Export/Import: Wrap Cell Content:										
	Id	card_id	business_id	Timestamp	Amount	Declined	product_ids	user_id	Lat	Longitude
▶	02C6201E-D90A-1859-B4EE-88D2986D3B02	CdU-2938	b-2362	2021-08-28 23:42:24	466.92	0	71, 1, 19	92	81.9185	-12.5276
	0466A42E-47CF-8D24-FD01-C0B689713128	CdU-4219	b-2302	2021-07-26 07:29:18	49.53	0	47, 97, 43	170	-43.9695	-117.525
	063FBA79-99EC-66FB-29F7-25726D1764A5	CdU-2987	b-2250	2022-01-06 21:25:27	92.61	0	47, 67, 31, 5	275	-81.2227	-129.05
	0668296C-CDB9-A883-76BC-2E4C4F8C8AE	CdU-3743	b-2618	2022-01-26 02:07:14	394.18	0	89, 83, 79	265	-34.3593	-100.556
	06CD9AA5-9B42-D684-DDDD-A5E394FEB499	CdU-2959	b-2346	2021-10-26 23:00:01	279.93	0	43, 31	92	33.7381	158.298
	07A46D48-31A3-7E87-65B9-0DA902AD109F	CdU-3225	b-2386	2021-06-28 21:11:42	340.87	1	47, 23	272	38.8342	92.1905

Tenemos un problema. En la columna product\_id de la tabla Transactions hay varios valores en un mismo campo. Tenemos que separarlos. Primero, haremos una tabla donde separaremos los valores separados por comas en columnas mediante la funcion substring\_index(). Renombraremos cada columna un product\_id diferente (P1, P2, P3 y p4).

```
SELECT id,product_ids,
substring_index(product_ids,',',1) AS P1,
substring_index(substring_index(product_ids,',',2), ',', -1) AS P2,
```

Con el primer string cogeremos los dos primeros valores, y con el segundo Substring nos quedaremos con el segundo valor

```
substring_index(substring_index(product_ids,',',3), ',', -1) AS P3,
```

# Al igual que arriba, sacamos tres valores, y nos quedaremos con el tercero

```
substring_index(product_ids,',',-1) AS P4
```

# Nos quedamos con el último valor.

```
FROM transactions;
```

Con la formula anterior, obtendremos algunas columnas con product\_ids repetidos. Pero eso no es problema. Más adelante, cuando unamos las tablas, usaremos un "UNION" y no un "UNION ALL". Eso hará que, si hay filas repetidas, las elimine.

Ahora haremos utilizaremos la tabla anterior como si fuera una “derived table” y haremos cuatros SELECTS que uniremos con UNION.

```
SELECT ID, P1 FROM DERIVED TABLE
UNION
SELECT ID, P2 FROM DERIVED TABLE
UNION
SELECT ID, P3 FROM DERIVED TABLE
UNION
SELECT ID, P4 FROM DERIVED TABLE
```

Pero este planteamiento tiene un problema y es que se trata de una consulta estática. Si el día de mañana, hubiera una transacción con más de cuatro productos esta consulta se quedaría insuficiente. Tenemos que automatizar la consulta.

Para automatizar la función SUBSTRING\_INDEX (string, delimiter, **number**) tendremos que hacer que el argumento number de la función Substring\_index sea un elemento cambiante, es decir, que vaya cambiando según el número de productos que contiene cada registro.

```
SELECT
transactions.id,
SUBSTRING_INDEX(SUBSTRING_INDEX(TRANSACTIONS.PROducT_IDS, ',', numero.num ), ',', -1) AS
Product_id
FROM transactions;
```

Para ello, definiremos una “derived-table” llamada número con una columna que se llame Num.



Esta nueva tabla números será la unión de 6 SELETCS ya que está pensada para que tenga un máximo de seis productos. Si se quiere incluir más de seis productos en la transacción, habrá que ampliarlo.

Para crear la "derived table" llamada número.

```
Select 1 as Num Union all Select 2 Union all Select 3 Union all Select 4 Union all Select 5 Union all Select 6 Union all
```

Ahora, haremos un JOIN de esta tabla con la consulta anterior

La consulta final quedará así:

#### **SELECT**

```
transactions.id,
```

```
SUBSTRING_INDEX(SUBSTRING_INDEX(TRANSACTIONS.PROducT_IDS, ',', numero.num ), ',', -1) AS  
Product_id
```

**FROM** transactions

#### **JOIN**

```
(Select 1 as Num Union all Select 2 Union all Select 3 Union all Select 4 Union all Select 5 Union all Select 6) AS numero
```

```
ON LENGTH(TRANSACTIONS.PROducT_IDS) -LENGTH(REPLACE(TRANSACTIONS.PROducT_IDS, ',')) +  
1 >= Numero.num;
```

Con la SELECT anterior, los Num de la tabla Numero irán sustituyendo el **numero.num** de la función Substring\_index cuando se vaya haciendo el JOIN.

Ahora bien, hay transacciones que tienen dos productos, otros tres. ¿Cómo instruimos para sepan cuantos productos tiene cada transacción?

Lo haremos mediante la condición ON del JOIN

1 - `LENGTH(TRANSACTIONS.PROducT_IDS)` --- Indica cuántos dígitos tiene el cada campo incluyendo las comas y los espacios.

2 - `LENGTH(REPLACE(TRANSACTIONS.PROducT_IDS, ',', ''))` --- Primero le decimos que elimine las comas. Y luego pedimos que sume los dígitos.

3 - Si a la resta de los dos anterior le sumamos 1, nos da cuántos product\_ids tenemos. Le sumamos + 1 porque el último producto no de cada transacción no va seguida de una coma. De no sumar uno, nos dejaríamos un producto en cada transacción.

`LENGTH(TRANSACTIONS.PROducT_IDS)`

menos

`LENGTH(REPLACE(TRANSACTIONS.PROducT_IDS, ',', ''))`

4 - En el ON indicamos que el número de productos que hayen cada transacción, ha de ser mayor al número de la tabla numero.Num.

`LENGTH(TRANSACTIONS.PROducT_IDS)`

menos

`LENGTH(REPLACE(TRANSACTIONS.PROducT_IDS, ',', '')) + 1`

Ha de ser

`>= Numero.num`

Resultado de la consulta:

```

391 • SELECT
392     transactions.id,
393     SUBSTRING_INDEX(SUBSTRING_INDEX(TRANSACTIONS.PRODUCT_IDS, ',', numero.num ), ',', -1) AS Product_id
394 FROM transactions
395 JOIN
396 ( Select 1 as Num Union all Select 2 Union all Select 3 Union all Select 4 Union all Select 5 Union all Select 6) AS numero
397 ON LENGTH(TRANSACTIONS.PRODUCT_IDS) - LENGTH(REPLACE(TRANSACTIONS.PRODUCT_IDS, ',', '')) + 1 >= Numero.num;

```

Result Grid

id	Product_id
02C6201E-D90A-1859-B4EE-88D2986D3802	19
02C6201E-D90A-1859-B4EE-88D2986D3802	1
02C6201E-D90A-1859-B4EE-88D2986D3802	71
0466A42E-47CF-8D24-FD01-C0B689713128	43
0466A42E-47CF-8D24-FD01-C0B689713128	97
0466A42E-47CF-8D24-FD01-C0B689713128	47

Result 1 x

Output

Action Output

#	Time	Action	Message
1	23:27:32	SELECT transactions.id, SUBSTRING_INDEX(SUBSTRING_INDEX(TRANSACTIONS.PRODUCT_IDS, ',', numero.num ), ',', -1) AS Product_id	1457 row(s) returned

## Ahora creamos una tabla id\_productid

```

CREATE TABLE Transaction_product (
    Transaction_id VARCHAR(255),
    product_id INT NOT NULL,
    CONSTRAINT Transaction_product PRIMARY KEY(Transaction_id,product_id)
);

```

## Añadimos los datos a la Transaction\_product

```

INSERT INTO Transaction_product (Transaction_id, product_id)

SELECT     transactions.id as Transaction_id,

            SUBSTRING_INDEX(SUBSTRING_INDEX(TRANSACTIONS.PRODUCT_IDS, ',', numero.num ), ',',
            -1) AS Product_id

FROM transactions

JOIN

( Select 1 as Num

Union all

```

Select 2

Union all

Select 3

Union all

Select 4

Union all

Select 5

Union all

Select 6) AS numero

```
ON LENGTH(TRANSACTIONS.PRODUCT_IDS) -LENGTH(REPLACE(TRANSACTIONS.PRODUCT_IDS, ',')) +
1 >= Numero.num;
```

## Consulta para ver los resultado de la tabal Transaction\_product

```
SELECT * FROM Transaction_product
```

```
ORDER BY 1;
```

The screenshot displays a database query interface. At the top, the SQL query is entered in two lines: `SELECT * FROM Transaction_product` and `ORDER BY 1;`. Below the query editor, a toolbar contains various icons for editing and exporting. The main area shows the 'Result Grid' with a table of data. The table has two columns: 'Transaction\_id' and 'product\_id'. The data is sorted by 'Transaction\_id'. Below the table, there is a tab labeled 'Transaction\_product 4'. At the bottom, the 'Output' section shows the 'Action Output' with a single entry: a green checkmark, the number '1', the time '23:31:32', the query text, and a message stating '1457 row(s) returned'.

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

#	Time	Action	Message
1	23:31:32	SELECT * FROM Transaction_product ORDER BY 1 LIMIT 0, 50000	1457 row(s) returned

Ahora tenemos que enlazar las tablas Transaction\_product con las tablas transactions y products

### **1 - Enlace Transaction\_product con la tabla transactions.**

```
ALTER TABLE Transaction_product
```

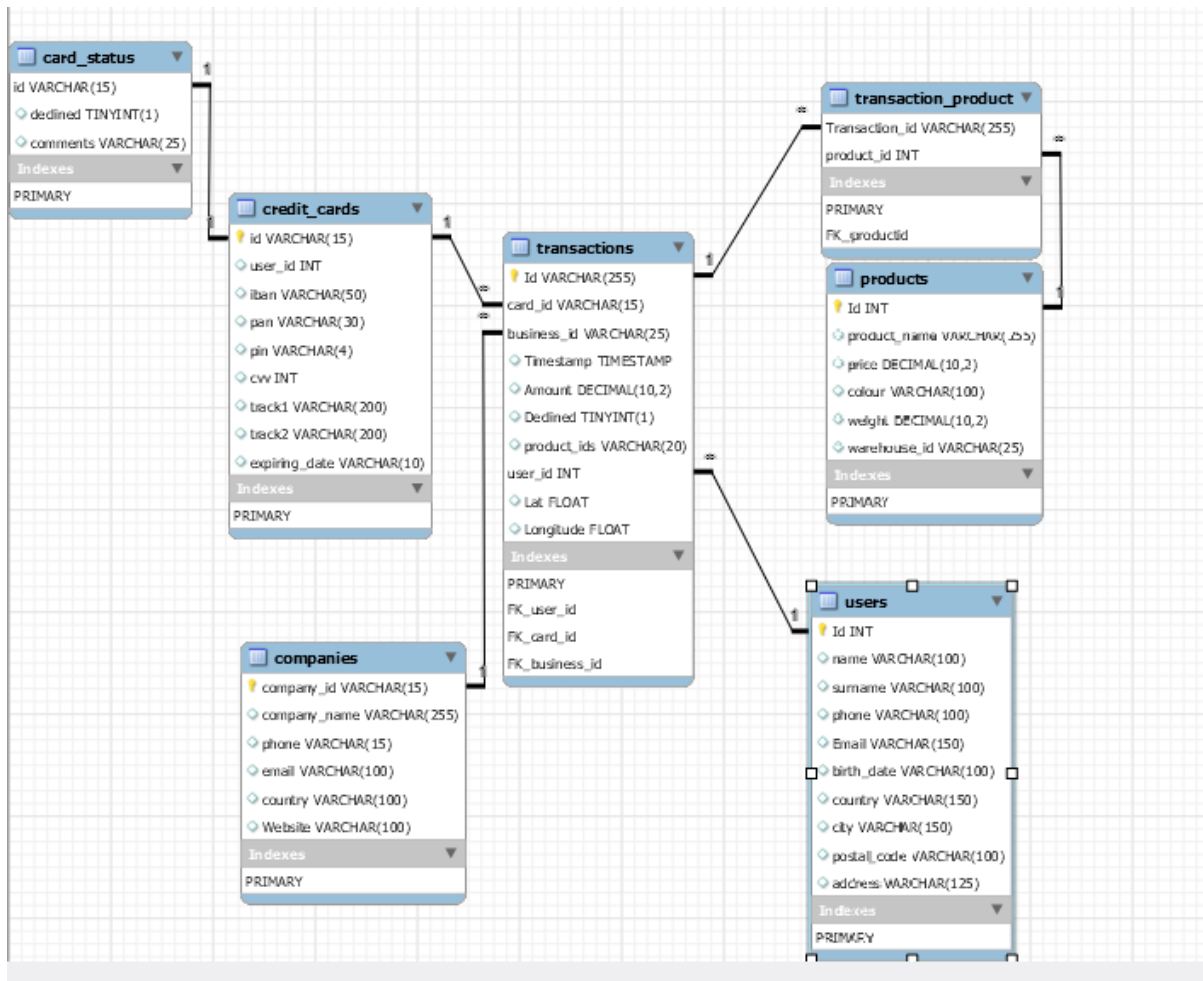
```
ADD CONSTRAINT FK_transactionID FOREIGN KEY (Transaction_ID) REFERENCES transactions(id);
```

### **2 Enlace Transaction\_product con la tabla products.**

```
ALTER TABLE Transaction_product
```

```
ADD CONSTRAINT FK_productid FOREIGN KEY (product_id) REFERENCES products(id);
```

El nuevo diagrama Entidad-Relación, quedará como sigue;



## Exercici 1

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

Para esta consulta usaremos la tabla recientemente creada Transaction\_product  
Agruparemos por producto y contaremos el número de transacciones

```

select product_id,product_name, count(transaction_id)
from Transaction_product
JOIN products
  
```

ON PRODUCTS.ID = Transaction\_product.product\_id

group by product\_id

order by 1 ;

```
468 • select product_id,product_name, count(transaction_id)
469 from Transaction_product
470 JOIN products
471 ON PRODUCTS.ID = Transaction_product.product_id
472 group by product_id
473 order by 1 ;
474
```

Result Grid

	product_id	product_name	count(transaction_id)
▶	1	Direwolf Stannis	61
	2	Tarly Stark	65
	3	duel tourney Lannister	51
	5	skywalker ewok	49
	7	north of Casterly	54
	11	Karstark Dorne	48

Result 6 x

Output

Action Output

#	Time	Action	Message
✓ 1	23:36:51	select product_id,product_name, count(transaction_id)from Transaction_product JOIN pr...	26 row(s) returned

2