

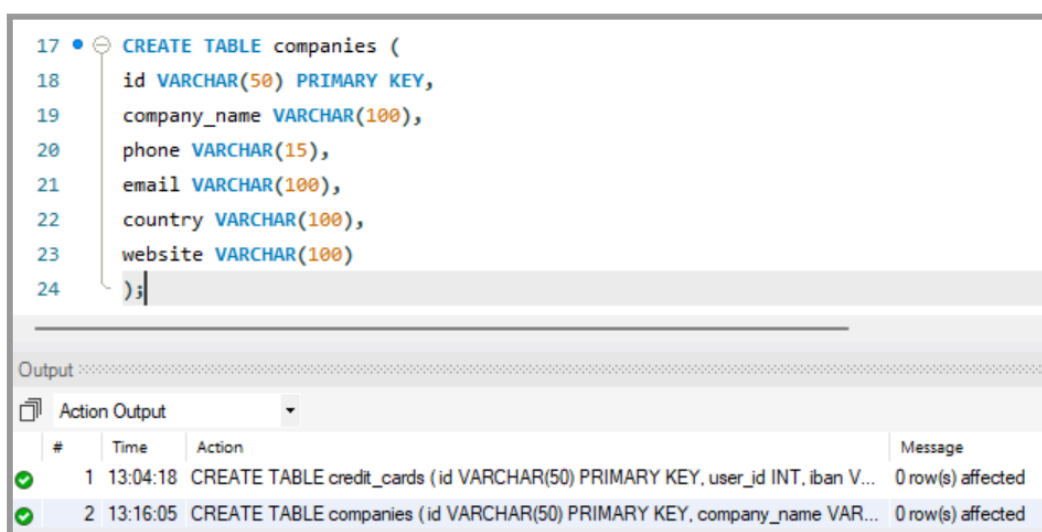
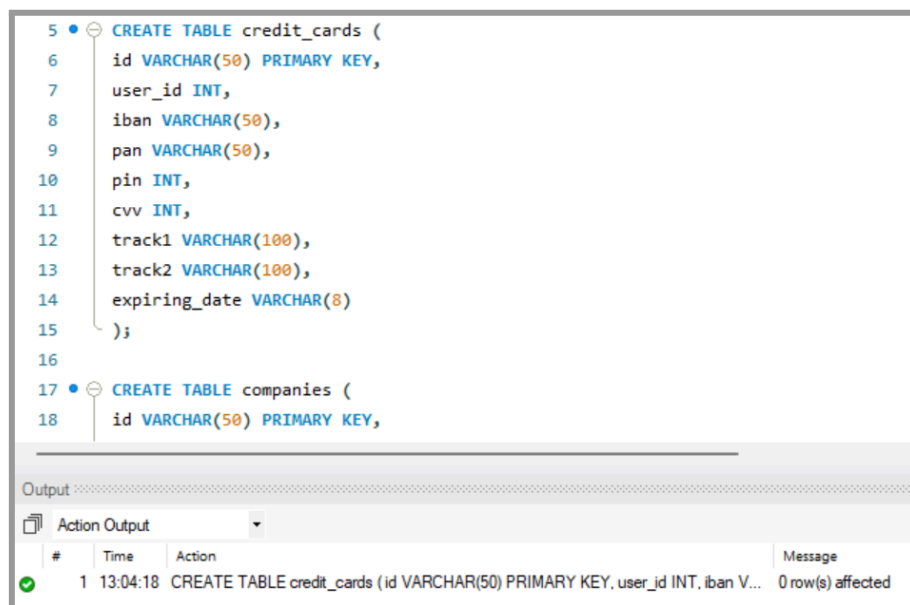
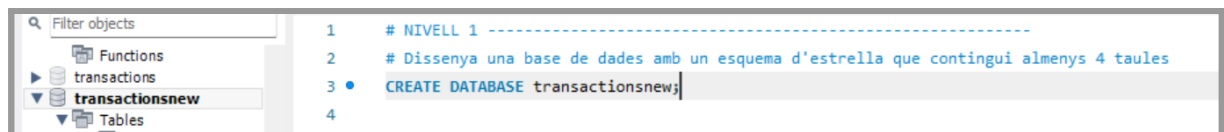
Sprint 4

Tasca S4.01. Creació de Base de Dades

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:

Se crea la base de datos **transactionsnew** y cuatro tablas: *credit_cards*, *companies*, *users*, *transactions*.



```
26 • CREATE TABLE users (  
27     id INT PRIMARY KEY,  
28     name VARCHAR(50),  
29     surname VARCHAR(50),  
30     phone VARCHAR(15),  
31     email VARCHAR(100),  
32     birth_date VARCHAR(12),  
33     country VARCHAR(100),  
34     city VARCHAR(50),  
35     postal_code VARCHAR(10),  
36     address VARCHAR(100)  
37 );
```

Output

Action Output

#	Time	Action	Message
✓ 1	13:17:58	CREATE TABLE users (id INT PRIMARY KEY, name VARCHAR(50), surname VARC...	0 row(s) affected

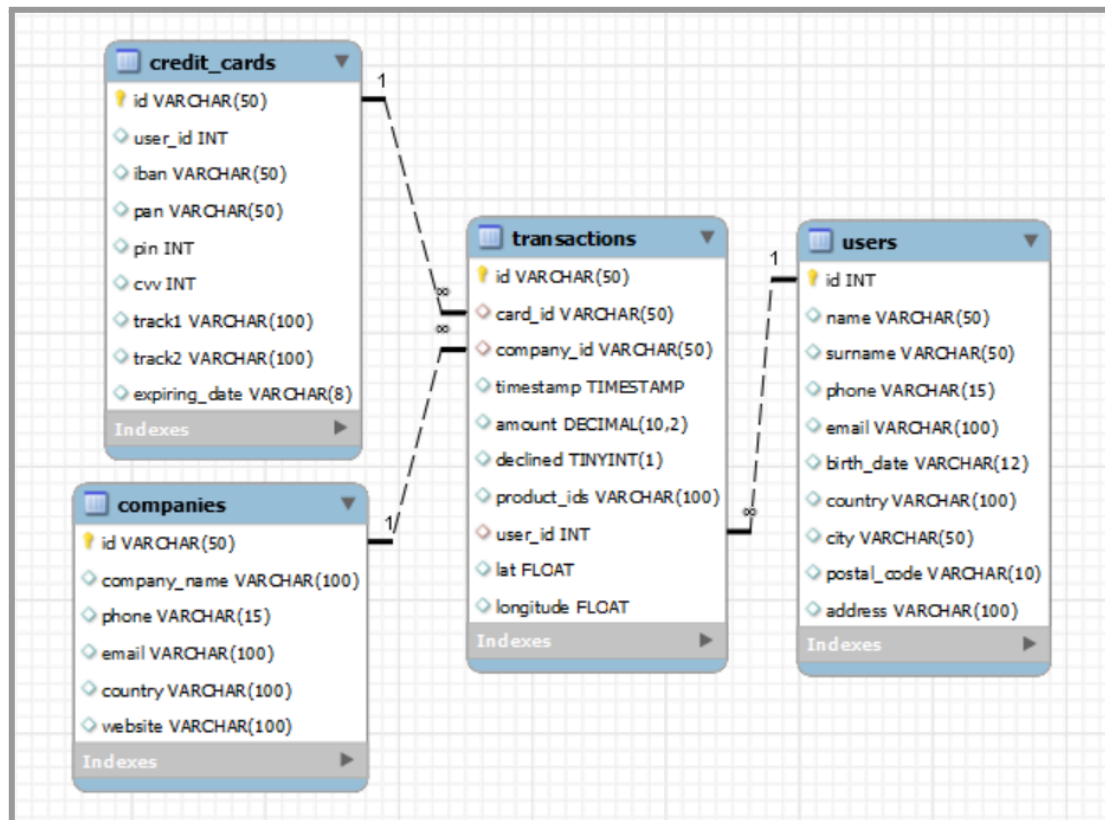
```
39 • CREATE TABLE transactions (  
40     id VARCHAR(50) PRIMARY KEY,  
41     card_id VARCHAR(50),  
42     company_id VARCHAR(50),  
43     timestamp TIMESTAMP,  
44     amount DECIMAL(10,2),  
45     declined TINYINT(1),  
46     product_ids VARCHAR(100),  
47     user_id INT,  
48     lat FLOAT,  
49     longitude FLOAT,  
50     FOREIGN KEY (card_id) REFERENCES credit_cards(id),  
51     FOREIGN KEY (company_id) REFERENCES companies(id),  
52     FOREIGN KEY (user_id) REFERENCES users(id)  
53 );  
54
```

Output

Action Output

#	Time	Action	Message
✓ 1	13:17:58	CREATE TABLE users (id INT PRIMARY KEY, name VARCHAR(50), surname VARC...	0 row(s) affected
⚠ 2	13:23:42	CREATE TABLE transactions (id VARCHAR(50) PRIMARY KEY, card_id VARCHAR(...	0 row(s) affected, 1 warning(s): 1681 Integer display width is deprecated

Queda creado así el siguiente esquema de estrella con la tabla de hechos, *transactions*, como centro, y las tablas restantes como dimensiones. Las relaciones son de muchos a uno y relacionan las foreign keys de la tabla *transactions* con las primary keys de las tablas de dimensiones.



Para cargar los datos desde los archivos CSV con código SQL se colocan los archivos en la carpeta que MySQL habilita como fuente para cargar archivos (intenté modificar los permisos pero sin éxito). El siguiente código muestra la ubicación de la carpeta: SHOW VARIABLES LIKE "secure_file_priv"

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

A la ruta del archivo se le agrega una backslash extra para que MySQL la lea correctamente. Se agregan las condiciones para que el CVS sea leído correctamente: el caracter que separa el campo es una coma o un punto y coma según el caso, en algunos archivos se debe especificar el final de línea con intro (\r) o nueva línea (\n). En todos los casos se ignora la primera fila, que son los nombres de los campos.

Se cargan los datos de los archivos CVS en las tablas respectivas y se confirma a continuación que se hayan cargado correctamente.

credit_cards

```

59 # Se cargan los datos en cada tabla, confirmando a continuación que se hayan cargado correctamente
60 • LOAD DATA
61 INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\credit_cards.csv'
62 INTO TABLE credit_cards
63 FIELDS TERMINATED BY ','
64 IGNORE 1 ROWS
65 ;

```

Output

Action Output

#	Time	Action	Message
1	11:16:30	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\credi...	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0


```

67 • SELECT * FROM credit_cards;
68

```

Result Grid

	id	user_id	iban	pan	pin	cvv	track1	track2	expiring_date
▶	CdU-2938	275	TR301950312213576817638661	5424465566813633	3257	984	%B8383712448554646^...	%B7653863056044187=800...	10/30/22
	CdU-2945	274	DO26854763748537475216568689	5142423821948828	9080	887	%B4621311609958661^...	%B4149568437843501=510...	08/24/23
	CdU-2952	273	BG45IVQL52710525608255	4556 453 55 5287	4598	438	%B2183285104307501^...	%B6778580257827162=690...	06/29/21
	CdU-2959	272	CR7242477244335841535	372461377349375	3583	667	%B7281111956795320^...	%B4246154489281853=280...	02/24/23
	CdU-2966	271	BG72LKTQ15627628377363	448566 886747 7265	4900	130	%B4728932322756223^...	%B2318571115599881=890...	10/29/24
	CdU-2973	270	PT87806228135092429456346	544 58654 54343 384	8760	887	%B4761405253275637^...	%B7816169831446746=131...	01/30/25

credit_cards 4 x

Output

Action Output

#	Time	Action	Message
1	11:21:54	SELECT * FROM credit_cards	275 row(s) returned

companies

```
69 • LOAD DATA
70 INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\companies.csv'
71 INTO TABLE companies
72 FIELDS TERMINATED BY ','
73 IGNORE 1 ROWS
74 ;
```

Output

Action Output

#	Time	Action	Message
✓ 1	11:23:28	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\com...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0

```
76 • SELECT * FROM companies;
77
```

Result Grid

	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

companies 5 ×

Output

Action Output

#	Time	Action	Message
✓ 1	11:32:53	SELECT * FROM companies	100 row(s) returned

users

```

78 • LOAD DATA
79   INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_ca.csv'
80   INTO TABLE users
81   FIELDS TERMINATED BY ','
82   ENCLOSED BY '"'
83   LINES TERMINATED BY '\\r\\n'
84   IGNORE 1 ROWS
85   ;

```

Output

Action Output

#	Time	Action	Message
1	11:36:20	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\user...	75 row(s) affected Records: 75 Deleted: 0 Skipped: 0 Warnings: 0

```

87 • LOAD DATA
88   INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_uk.csv'
89   INTO TABLE users
90   FIELDS TERMINATED BY ','
91   ENCLOSED BY '"'
92   LINES TERMINATED BY '\\r\\n'
93   IGNORE 1 ROWS
94   ;

```

Output

Action Output

#	Time	Action	Message
1	11:36:20	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\user...	75 row(s) affected Records: 75 Deleted: 0 Skipped: 0 Warnings: 0
2	11:37:32	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\user...	50 row(s) affected Records: 50 Deleted: 0 Skipped: 0 Warnings: 0

```

96 • LOAD DATA
97   INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv'
98   INTO TABLE users
99   FIELDS TERMINATED BY ','
100  ENCLOSED BY '"'
101  LINES TERMINATED BY '\\r\\n'
102  IGNORE 1 ROWS
103  ;
104

```

Output

Action Output

#	Time	Action	Message
2	11:37:32	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\us...	50 row(s) affected Records: 50 Deleted: 0 Skipped: 0 Warnings: 0
3	11:38:10	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\us...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0

```

105 • SELECT * FROM users;

```

106

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

	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

users 6 x

Output

Action Output

#	Time	Action	Message
1	11:38:53	SELECT * FROM users	275 row(s) returned

transactions

```
107 • LOAD DATA
108   INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv'
109   INTO TABLE transactions
110   FIELDS TERMINATED BY ';'
111   IGNORE 1 ROWS
112   ;
```

Output

Action Output

#	Time	Action	Message
✓ 1	11:40:03	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\trans...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0

```
114 • SELECT * FROM transactions;
```

Result Grid

	id	card_id	company_id	timestamp	amount	declined	product_ids	user_id	lat	longitude
▶	02C6201E-D90A-1859-B4EE-88D2986D3802	CcU-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	CcU-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	CcU-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-2E4C44F8C8AE	CcU-3743	b-2618	2022-01-26 02:07:14	394.18	0	89, 83, 79	265	-34.3593	-100.556
	06CD9AA5-9B42-D684-DDDD-A5E394FEB999	CcU-2959	b-2346	2021-10-26 23:00:01	279.93	0	43, 31	92	33.7381	158.298
	07A46D48-31A3-7E87-65B9-0DA902AD109F	CcU-3225	b-2386	2021-06-28 21:11:42	340.87	1	47, 23	272	38.8342	92.1905

transactions 7

Output

Action Output

#	Time	Action	Message
✓ 1	11:41:14	SELECT * FROM transactions	587 row(s) returned

Exercici 1

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

Si se intenta hacer esta consulta solo con subqueries, devuelve el listado de los usuarios con más de 30 transacciones pero no la cantidad de transacciones, por lo tanto se hace una JOIN además de la subconsulta, ya que es un dato útil.

```
116 # EXERCICI 1 --- Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.
117 # Si se intenta hacer esta consulta solo con subqueries, devuelve el listado de los usuarios con más de 30 transacciones pero no la
    cantidad de transacciones, por lo tanto se hace una JOIN además de la subconsulta, ya que es un dato que es útil saber.
118 • SELECT users.id, CONCAT(users.name, ' ', users.surname) AS Name, auxt.Transactions
119 FROM users
120 JOIN (SELECT user_id, COUNT(id) AS Transactions
121       FROM transactions
122       GROUP BY user_id
123       HAVING COUNT(id) > 30) auxt
124 ON users.id = auxt.user_id
125 ;
```

id	Name	Transactions
92	Lynn Riddle	39
267	Ocean Nelson	52
272	Hedwig Gilbert	76
275	Kenyon Hartman	48

Result 10 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:30:40	SELECT users.id, CONCAT(users.name, ' ', users.surname) AS Name, auxt.Transactions...	4 row(s) returned	0.000 sec / 0.000 sec

Exercici 2

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

```
128 # EXERCICI 2 --- Mostra la mitjana d'amount per IBAN de les targetes de crèdit a la companyia Donec Ltd, utilitza almenys 2 taules.
129 • SELECT companies.company_name AS Company, credit_cards.iban AS IBAN, ROUND(AVG(transactions.amount), 2) AS 'Avg. amount'
130 FROM transactions
131 JOIN credit_cards
132 ON transactions.card_id = credit_cards.id
133 JOIN companies
134 ON transactions.company_id = companies.id
135 WHERE company_name = 'Donec Ltd'
136 GROUP BY IBAN
137 ;
```

Result Grid

	Company	IBAN	Avg. amount
▶	Donec Ltd	PT87806228135092429456346	203.72

Result 12 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	12:33:12	SELECT companies.company_name AS Company, credit_cards.iban AS IBAN, ROU...	1 row(s) returned	0.015 sec / 0.000 sec

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:

En el script de creació de la taula se crea una CTE en la que se dividen per `card_id` y se ordenan por fecha las filas con la función `DENSE_RANK()` que en el campo `latest3` ordena las fechas de forma descendiente. Este campo más adelante se usa para filtrar las últimas tres transacciones indicando `latest3 <= 3`.

```
143 # Se crea una CTE en la que se dividen por card_id y se ordenan por fecha las filas
144 WITH auxtransactions AS (
145     SELECT
146         card_id,
147         timestamp,
148         declined,
149         DENSE_RANK() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS latest3
150     FROM transactions
```

Result Grid

	card_id	timestamp	declined	latest3
▶	CcU-2938	2022-03-12 09:23:10	0	1
	CcU-2938	2022-03-09 20:53:59	0	2
	CcU-2938	2022-02-24 11:01:42	0	3
	CcU-2938	2021-10-24 01:29:53	0	4
	CcU-2938	2021-10-17 03:52:48	0	5

Result 14 x

Output

Action Output

#	Time	Action	Message
✓ 1	12:50:13	SELECT card_id, timestamp, declined, DENSE_RANK() OVER (PAR...	587 row(s) returned

En la SELECT que usa la CTE auxtransactions se filtran las últimas tres transacciones en el WHERE y con CASE se asigna la etiqueta 'blocked' a las transacciones que tienen tres instancias en que *declined* sea True (es decir, en que la suma de los valores de *declined* es 3).

```
139 # NIVELL 2
140 # Crea una nova taula que reflecteixi l'estat de les targetes de crèdit basat en si les últimes tres transaccions van ser declinades
141 • CREATE TABLE card_status
142 AS
143 # Se crea una CTE en la que se dividen por card_id y se ordenan por fecha las filas
144 WITH auxtransactions AS (
145     SELECT
146         card_id,
147         timestamp,
148         declined,
149         DENSE_RANK() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS latest3
150     FROM transactions
151 )
152 # En la SELECT que usa la CTE auxtransactions en su FROM, se filtran las últimas tres transacciones en el WHERE y con CASE se asigna la
    etiqueta 'active' a las transacciones que no tienen ninguna instancia declined (que la suma de los valores de decline es 0 en las tres
    transacciones)
153 SELECT
154     card_id,
155     CASE
```

Output

#	Time	Action	Message	Duration / Fetch
1	13:06:54	CREATE TABLE card_status AS # Se crea una CTE en la que se dividen por card_id ...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0	0.047 sec

```
153 SELECT
154     card_id,
155     CASE
156         WHEN SUM(declined) = 3 THEN 'blocked'
157         ELSE 'active'
158     END AS status
159 FROM auxtransactions
160 WHERE latest3 <= 3
161 GROUP BY card_id
162 ;
```

Luego se agrega la foreign key para vincular la tabla a transactions.

```
164 # Se agrega la foreign key para vincular la tabla a transactions
165 • ALTER TABLE card_status
166 ADD FOREIGN KEY (card_id) REFERENCES credit_cards(id)
167 ;
```

Output

Action Output

#	Time	Action	Message
✓ 1	13:09:11	ALTER TABLE card_status ADD FOREIGN KEY (card_id) REFERENCES credit_card...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

Se comprueba la tabla.

```
169 # Comprobación de la tabla
170 • SELECT *
171 FROM card_status
172 ;
173
```

Result Grid

card_id	status
CcU-2938	active
CcU-2945	active
CcU-2952	active
CcU-2959	active
CcU-2966	active

card_status 15 x

Output

Action Output



#	Time	Action	Message
✓ 1	13:09:41	SELECT * FROM card_status	275 row(s) returned

Exercici 1

Quantes targetes estan actives?

Todas las tarjetas están activas.

```
174 # EXERCICI 1 --- Quantes targetes estan actives?
175 • SELECT COUNT(status) AS active_cards
176 FROM card_status
177 WHERE status = 'active'
178 ;
179
```

Result Grid  Filter Rows: Export:  Wrap Cell Content: 

	active_cards
▶	275

Result 17 x

Output



Action Output

#	Time	Action	Message
✓ 1	13:10:31	SELECT COUNT(status) AS active_cards FROM card_status WHERE status = 'active'	1 row(s) returned

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:

- Se crea una tabla puente llamada bridge_products con los campos *transaction_id* (vinculado con el campo *id* de la tabla transactions) y *product_id* (vinculado con el campo *id* de la tabla products).
- Se genera una CTE en la que se dividen los números de productos que están separados por coma en el campo *product_ids*. Se utiliza la función SUBSTRING_INDEX para extraer cada valor antes de la coma y los resultados se almacenan en campos llamados *p1*, *p2*, *p3*, *p4*.
- En la siguiente subquery se colocan los valores de los campos *p1*, *p2*, *p3* y *p4* de la tabla aux en un solo campo llamado *product_id* utilizando la función UNION, y no Union All, para no incluir los valores repetidos (que se generaron en el paso anterior, pues SUBSTRING_INDEX devuelve el último valor si no encuentra uno nuevo). Entiendo que generar valores repetidos no es la forma más eficiente de código, pero considero que al ser una CTE que se genera una sola vez para crear una tabla, no es un uso constante de recursos.
- Se ordena por *id* para mayor claridad al visualizar la tabla.
- Se crea la tabla products y se agregan las foreign keys para vincular las tablas transactions y products a través de la tabla puente.
- Se introducen los datos en tabla products.

Detalle de la SELECT con la que se genera la CTE para extraer los valores numéricos de *product_ids*

```
186 # Se genera una CTE en la que se dividen los números de productos que están separados por coma en el campo product_ids. Se utiliza la
187 función SUBSTRING_INDEX para extraer cada valor antes de la coma y los resultados se almacenan en campos llamados p1, p2, p3, p4.
188 WITH aux
189 AS
190 SELECT id,
191 SUBSTRING_INDEX(transactions.product_ids, ',', 1) AS p1,
192 SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', 2), ',', -1) AS p2,
193 SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', 3), ',', -1) AS p3,
194 SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', 4), ',', -1) AS p4
195 FROM transactions)
```

Result Grid

id	p1	p2	p3	p4
02C6201E-D90A-1859-B4EE-88D2986D3B02	71	1	19	19
0466A42E-47CF-8D24-FD01-C0B689713128	47	97	43	43
063FBA79-99EC-66FB-29F7-25726D1764A5	47	67	31	5
0668296C-CDB9-A883-76BC-2E4C4F8C8AE	89	83	79	79
06CD9AA5-9B42-D684-DDDD-A5E394FEB499	43	31	31	31

Result 18 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:38:50	SELECT id, SUBSTRING_INDEX(transactions.product_ids, ',', 1) AS p1, SUBSTRIN...	587 row(s) returned	0.000 sec / 0.000 sec

Query completa de la creación de la tabla *bridge_products*

```

        tabla products)
184 • CREATE TABLE bridge_products (transaction_id VARCHAR(50), product_id INT)
185 AS
186 # Se genera una CTE en la que se dividen los números de productos que están separados por
    coma en el campo product_ids. Se utiliza la función SUBSTRING_INDEX para extraer cada valor
    antes de la coma y los resultados se almacenan en campos llamados p1, p2, p3, p4.
187 WITH aux
188 AS (
189     SELECT id,
190         SUBSTRING_INDEX(transactions.product_ids, ',', 1) AS p1,
191         SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', 2), ',', -1) AS p2,
192         SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', 3), ',', -1) AS p3,
193         SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', 4), ',', -1) AS p4
194     FROM transactions)
195 # En la siguiente subquery se colocan los valores de los campos p1, p2, p3 y p4 de la tabla
    aux en un solo campo llamado product_id utilizando la función UNION, y no UNION ALL, para
    no incluir los valores repetidos (que se generaron en el paso anterior, pues
    SUBSTRING_INDEX devuelve el último valor si no encuentra uno nuevo)
196 SELECT id AS transaction_id, product_id
197 FROM (SELECT id, p1 AS product_id
198     FROM aux
199     UNION
200     SELECT id, p2
201     FROM aux
202     UNION
203     SELECT id, p3
204     FROM aux
205     UNION
206     SELECT id, p4
207     FROM aux
208 ) AS aux2
209 # Se ordena por id para mayor claridad al visualizar la tabla
210 ORDER BY 1
211 ;
212
213 # Se crea la tabla products

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	08:39:51	CREATE TABLE bridge_products (transaction_id VA...	1457 row(s) affected Records: 1457 Duplicates: 0 ...	0.094 sec

Creación de la tabla products

```

214 # Se crea la tabla products
215 • CREATE TABLE products (
216     id INT PRIMARY KEY,
217     product_name VARCHAR(100),
218     price VARCHAR(20),
219     colour VARCHAR(20),
220     weight DECIMAL(8,2),
221     warehouse_id VARCHAR(15)
222 );

```

Output

Action Output

#	Time	Action	Message
✓ 1	13:44:15	CREATE TABLE products (id INT PRIMARY KEY, product_name VARCHAR(100), pri...	0 row(s) affected

Se agregan las FK para vincular las tablas *transactions* y *products* a través de la tabla puente

```
224 # Se agregan las foreign keys para vincular las tablas transactions y products a través de la tabla puente
225 • SET FOREIGN_KEY_CHECKS = 0
226 ;
227 • ALTER TABLE bridge_products
228   ADD FOREIGN KEY (transaction_id) REFERENCES transactions(id),
229   ADD FOREIGN KEY (product_id) REFERENCES products(id)
230 ;
231 • SET FOREIGN_KEY_CHECKS = 1
232 ;
233
```

Output

#	Time	Action	Message
✓ 1	13:45:16	SET FOREIGN_KEY_CHECKS = 0	0 row(s) affected
✓ 2	13:45:19	ALTER TABLE bridge_products ADD FOREIGN KEY (transaction_id) REFERENCES t...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Se introducen los datos en la tabla products

```
234 # Se introducen los datos en la tabla products
235 • LOAD DATA
236   INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\products.csv'
237   INTO TABLE products
238   FIELDS TERMINATED BY ','
239   IGNORE 1 ROWS
240 ;
```

Output

#	Time	Action	Message
✓ 1	13:47:38	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\prod...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0

Se comprueba la tabla *products*

242 # Se comprueba la tabla products

243 • `SELECT * FROM products;`

244

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

	id	product_name	price	colour	weight	warehouse_id
▶	1	Direwolf Stannis	\$161.11	#7c7c7c	1.00	WH-4
	2	Tarly Stark	\$9.24	#919191	2.00	WH-3
	3	duel tourney Lannister	\$171.13	#d8d8d8	1.50	WH-2
	4	warden south duel	\$71.89	#111111	3.00	WH-1
	5	skywalker ewok	\$171.22	#dbdbdb	3.20	WH-0

products 19 x

Output

Action Output

#	Time	Action	Message
✓ 1	13:48:39	SELECT * FROM products	100 row(s) returned

Se comprueba la tabla *bridge_products*

245 # Se comprueba la tabla bridge_products

246 • `SELECT * FROM bridge_products;`

247

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

	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

bridge_products 20 x

Output

Action Output

#	Time	Action	Message
✓ 1	13:49:29	SELECT * FROM bridge_products	1457 row(s) returned

Exercici 1

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

```
248 # Exercici 1 --- Necessitem conèixer el nombre de vegades que s'ha venut cada producte.
249 • SELECT product_id, product_name, COUNT(product_id)
250 FROM products
251 JOIN bridge_products ON products.id = bridge_products.product_id
252 JOIN transactions ON transactions.id = bridge_products.transaction_id
253 WHERE declined = 0
254 GROUP BY 1
255
```

Result Grid

	product_id	product_name	COUNT(product_id)
▶	1	Direwolf Stannis	51
	19	dooku solo	44
	71	Tully Dorne	44
	43	duel	54
	47	Tully	50

Result 22 x

Output

Action Output

#	Time	Action	Message
✓ 1	13:50:47	SELECT product_id, product_name, COUNT(product_id) FROM products JOIN bridg...	26 row(s) returned