

## Sprint 4 - Tasca S4.01. Creació de Base de Dades

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

### Nivell 1

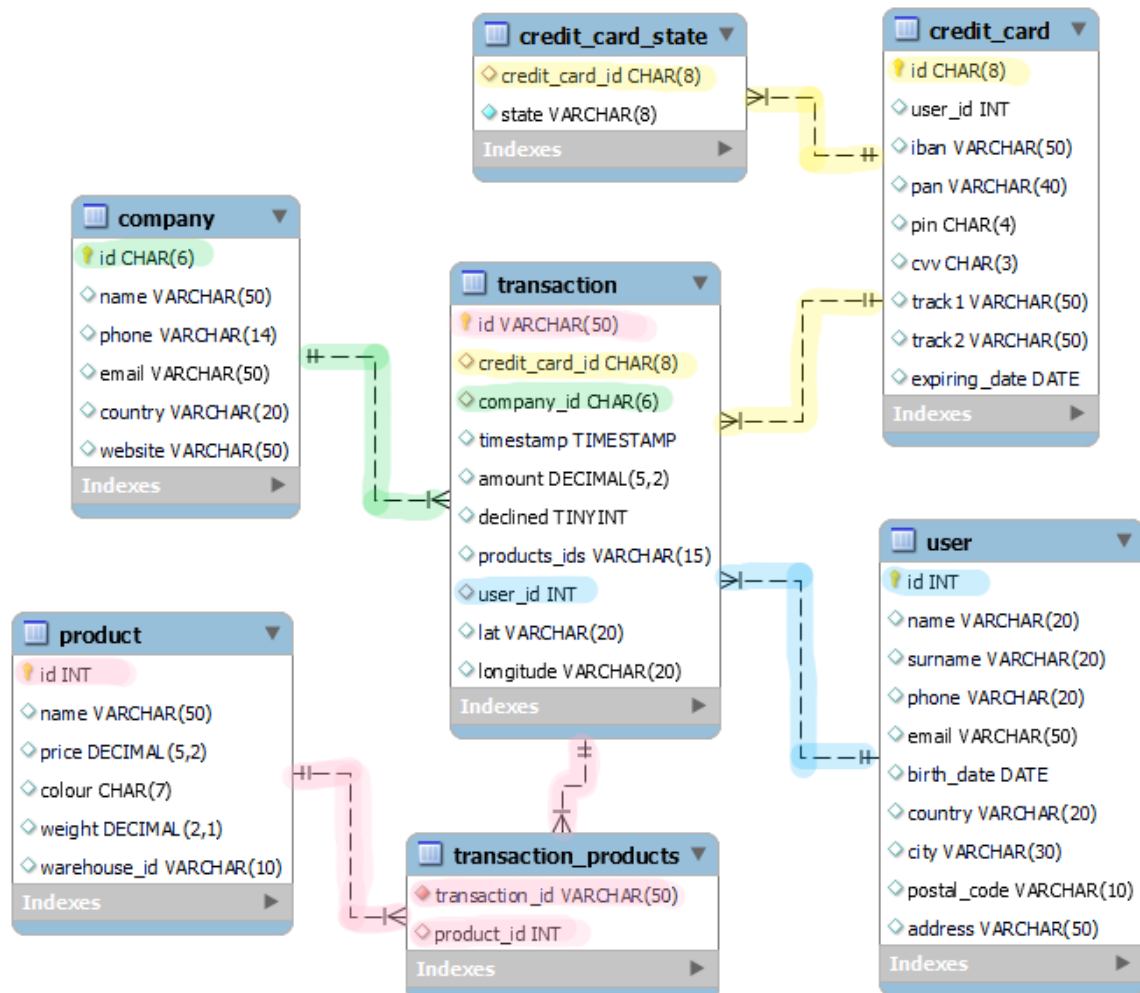
Descàrrega els arxius CSV, estudia'ls i dissenya una base de dades amb un esquema d'estrella que contingui, almenys 4 taules de les quals puguis realitzar les següents consultes:

Antes que nada, descargué los archivos proporcionados para poder ver los datos que contienen y así poder modelar el esquema que utilizaría para mi base de datos: cantidad de tablas, columnas dentro de las tablas, formato de las columnas, y el tipo de modelo que utilizaría.

Una vez tenía la idea del modelo de estrella listo empecé con la carga de datos.

Con los datos cargados empecé la transformación, estandarizando los datos en sus respectivas columnas y las columnas a su respectivos formatos (VARCHAR a: CHAR, INT, DEC, DATES).

Con los datos ya cargados lo último que faltaba era agregar las FOREIGN KEYS para empezar a trabajar con la base de datos.



## - Exercici 1

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

Lo primero que hice fue un conteo de las transacciones (id), donde lo agrupé por los user\_id y le pedí que también me mostrara dichos user\_id.

Lo siguiente fue solicitar todos los datos de la tabla user utilizando los ids de la Query anterior como un filtro, para así obtener solamente los datos de los usuarios que tienen más de 30 transacciones registradas.

```
SELECT *
FROM user AS u
HAVING id IN
    (SELECT user_id
     FROM transaction AS t
     GROUP BY user_id
     HAVING COUNT(id) > 30);
```

The screenshot shows a database IDE window titled "Sprint\_4 - Abelardo Lopez". The SQL editor contains two queries. The first query counts transactions per user. The second query uses a subquery to filter users with more than 30 transactions. The "Result Grid" shows the results of the second query, displaying user details for four users. The "Output" section shows the execution logs for both queries, indicating that 4 rows were returned for each.

```
185 # Nivell 1
186 ## Exercici 1
187 ### Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.
188 #Para hacer un conteo de cuantas transacciones registradas tiene un usuario.
189 • SELECT user_id, COUNT(id) AS tran_count
190 FROM transaction
191 GROUP BY user_id
192 HAVING COUNT(id) > 30;
193
194 #Para saber los datos de los usuarios con mas de 30 transacciones registradas, utilizando el conteo anterior como filtro.
195 • SELECT *
196 FROM user AS u
197 HAVING id IN
198     (SELECT user_id
199      FROM transaction AS t
200      GROUP BY user_id
201      HAVING COUNT(id) > 30);
202
```

id	name	surname	phone	email	birth_date	country	city	postal_code	address
92	Lynn	Riddle	1-387-885-4057	vitae.aliquet@outlook.edu	1984-09-21	United States	Bozeman	61871	P.O. Box 712, 7907 Est St.
267	Ocean	Nelson	079-481-2745	aenean@yahoo.com	1991-12-26	Canada	Charlottetown	85X 3P4	Ap #732-8357 Pede, Rd.
272	Hedwig	Gilbert	064-204-8788	sem.eget@icloud.edu	1991-04-16	Canada	Tuktoyaktuk	Q4C 3G7	P.O. Box 496, 5145 Sapien Road
275	Kenyon	Hartman	082-871-7248	convallis.ante.lectus@yahoo.com	1982-08-03	Canada	Richmond	R8H 2K2	8564 Facilisi. St.
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Output

#	Time	Action	Message
✓ 1	10:45:05	SELECT user_id, COUNT(id) AS tran_count FROM transaction GROUP BY user_id HAVING COUNT(id) > 30 LIMIT 0, 1000	4 row(s) returned
✓ 2	10:45:20	SELECT * FROM user AS u HAVING id IN (SELECT user_id FROM transaction AS t GROUP BY user_id HAVING COUNT(id) > 30) LIMIT 0, 1000	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.**

Para este ejercicio necesito datos de 3 tablas distintas, necesito la columna "iban" de la tabla credit\_card, la columna "amount" de la tabla transaction y la columna "name" de la tabla company.

Una vez aclaré eso lo siguiente es ver el planteamiento de como accederé a esos datos.

En la primera opción usé un JOIN entre las tablas credit\_card y transaction, y por otro lado usé una SubQuery en el WHERE con la intención de filtrar el id de la compañía cuyo nombre sea "Donec Ltd".

En la segunda opción junté las 3 tablas con JOINS y le pedí que me mostrara las 3 columnas que iba a utilizar, este resultado lo utilicé como una tabla temporal en el FROM y en la Query principal realicé el conteo y filtrado necesario para obtener el resultado demandado.

```
SELECT cc.iban, ROUND(AVG(t.amount),2) AS average
FROM transaction AS t
JOIN credit_card AS cc ON t.credit_card_id = cc.id
WHERE t.company_id IN
    (SELECT id
     FROM company
     WHERE name = "Donec Ltd")
GROUP BY cc.iban;
```

The screenshot shows a SQL IDE window titled "Sprint\_4 - Abelardo Lopez". It contains two SQL queries and their results.

**Query 1 (Line 207):**

```
SELECT cc.iban, ROUND(AVG(t.amount),2) AS average
FROM transaction AS t
JOIN credit_card AS cc ON t.credit_card_id = cc.id
WHERE t.company_id IN
    (SELECT id
     FROM company
     WHERE name = "Donec Ltd")
GROUP BY cc.iban;
```

**Query 2 (Line 217):**

```
SELECT iban, ROUND(AVG(amount),2) AS average
FROM
    (SELECT cc.iban, t.amount, c.name
     FROM transaction AS t
     JOIN credit_card AS cc ON t.credit_card_id = cc.id
     JOIN company AS c ON t.company_id = c.id) AS utility_table
WHERE name = "Donec Ltd"
GROUP BY iban;
```

**Result Grid:**

iban	average
PT87806228135092429456346	203.72

**Output:**

#	Time	Action	Message
1	11:09:19	SELECT cc.iban, ROUND(AVG(t.amount),2) AS average FROM transaction AS t JOIN credit_card AS cc ON t.credit_card_id = cc.id WHERE t.compan...	1 row(s) returned
2	11:09:19	SELECT iban, ROUND(AVG(amount),2) AS average FROM (SELECT cc.iban, t.amount, c.name FROM transaction AS t JOIN credit_card AS cc ON t.cr...	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:

Para crear la tabla valoré que para que una tarjeta esté inactiva debería de tener como mínimo 3 transacciones rechazadas y con esa valoración cree la siguiente tabla.

```
CREATE TABLE IF NOT EXISTS credit_card_state AS
SELECT credit_card_id,
       CASE
         WHEN (COUNT(declined) = 1) >= 3 THEN "Inactive"
         ELSE "Active"
       END AS "state"
FROM transaction
GROUP BY credit_card_id;
```

The screenshot shows a SQL IDE window titled "Sprint\_4 - Abelardo Lopez". The editor contains the following SQL code:

```
226 # Nivell 2
227 # 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:
228 #Considerando que para que una tarjeta esté inactiva debería de haber sido declinada las últimas 3 veces,
229 #con esto hago el conteo de las veces que una tarjeta a sido rechazada, y si fue rechazada más de 3 veces le doy el estado de inactiva,
230 #luego uso esa información para crear una nueva tabla.
231 • CREATE TABLE IF NOT EXISTS credit_card_state AS
232 SELECT credit_card_id,
233        CASE
234          WHEN (COUNT(declined) = 1) >= 3 THEN "Inactive"
235          ELSE "Active"
236        END AS "state"
237 FROM transaction
238 GROUP BY credit_card_id;
239
240 #Para visualizar las tabla creada.
241 • SELECT *
242 FROM credit_card_state;
243
244 #Para agregar la FOREIGN KEY.
245 • ALTER TABLE credit_card_state
246 ADD FOREIGN KEY (credit_card_id) REFERENCES credit_card(id);
```

The Output window shows the following results:

#	Time	Action	Message
3	11:52:40	CREATE TABLE IF NOT EXISTS credit_card_state AS SELECT credit_card_id, CASE WHEN (COUNT(declined) = 1) >= 3 THEN "Inactive" ELSE "...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0
4	11:53:00	SELECT * FROM credit_card_state LIMIT 0, 1000	275 row(s) returned
5	12:00:14	ALTER TABLE credit_card_state ADD FOREIGN KEY (credit_card_id) REFERENCES credit_card(id)	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

## - Ejercicio 1

### ¿Cuántas tarjetas están activas?

Todas las tarjetas están activas. En esta parte del ejercicio lo que hice fue ordenar las transacciones de la más reciente a la más antigua, agrupándolas por `credit_card_id`. Una vez ordenado se lleva a cabo un conteo de las rechazadas para ver si llegan hasta 3. Y al final se les da el estado de inactivas a las tarjetas de crédito que llegaron hasta las 3 transacciones rechazadas.

```
WITH date_order AS (  
    SELECT  
        timestamp, credit_card_id, declined,  
        ROW_NUMBER() OVER (PARTITION BY "credit_card_id" ORDER BY  
"timestamp" DESC) row_order  
    FROM transaction),  
last_three AS (  
    SELECT credit_card_id, COUNT(declined) AS declined_count  
    FROM date_order  
    WHERE row_order <= 3 AND declined = 1  
    GROUP BY credit_card_id)  
SELECT credit_card_id,  
    CASE  
        WHEN declined_count = 3 THEN "Inactive"  
        ELSE "Active"  
    END AS "state"  
FROM last_three  
GROUP BY credit_card_id  
ORDER BY credit_card_id;
```

The screenshot shows a SQL IDE window titled "Sprint\_4 - Abelardo Lopez". The query editor contains the same SQL code as in the previous block. The query is executed, and the output pane shows "0 row(s) returned".

credit_card_id	state
----------------	-------

Output: 0 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:

Primero hay que crear una tabla temporal que contiene una lista de números del 1 al 4 (que es el máximo de ids en la columna products\_ids). La idea es utilizar esta lista de números para acceder a las distintas posiciones posibles del string

```
SELECT 1 AS num UNION ALL SELECT 2 UNION ALL SELECT 3 UNION ALL SELECT 4;
```

Después hay que dividir los strings para tener un solo id, ahí es donde uso num para que haga la iteración por las distintas posiciones posibles. Dependiendo del valor de "num" en el SUBSTRING\_INDEX() interno, el SUBSTRING\_INDEX() externo será capaz de darme los valores correspondientes a esa posición, incluyendo la primera.

Y uso la función TRIM() para eliminar los espacios vacíos que hay antes o después de los ids.

```
TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(products_ids, ',', num), ',', -1)) AS product_id
```

El primer CHAR\_LENGTH() me dice el largo del string, el segundo elimina las comas y me dice el largo del string sin las comas, al restar el uno con el otro se la cantidad de comas dentro de cada string. Si la cantidad de comas es igual o mayor que "num - 1" se efectuara la iteración sin problemas.

La condición en el JOIN me permite saber la cantidad de comas "," que hay en el string, y por ende la cantidad de ids en ese string (0 comas = 1 id, 1 coma = 2 ids).

```
CHAR_LENGTH(products_ids) - CHAR_LENGTH(REPLACE(products_ids, ',', '')) >= num - 1.
```

```
CREATE TABLE IF NOT EXISTS transaction_products AS
```

```
SELECT
```

```
    t.id AS transaction_id,
```

```
    TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(t.products_ids, ',', num), ',', -1)) AS
```

```
product_id
```

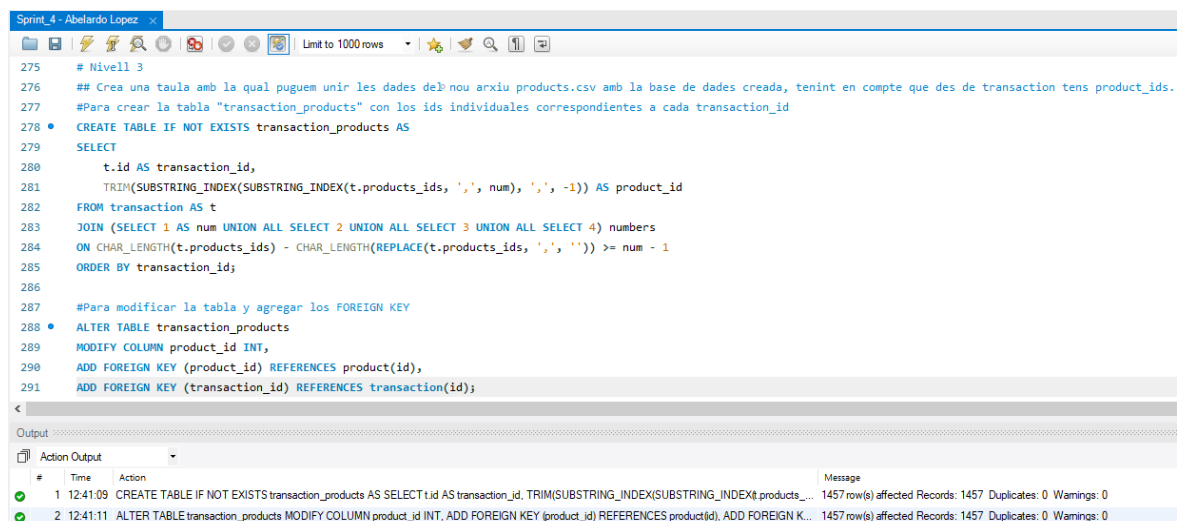
```
FROM transaction AS t
```

```
JOIN (SELECT 1 AS num UNION ALL SELECT 2 UNION ALL SELECT 3 UNION ALL SELECT 4)
```

```
numbers
```

```
ON CHAR_LENGTH(t.products_ids) - CHAR_LENGTH(REPLACE(t.products_ids, ',', '')) >= num - 1
```

```
ORDER BY transaction_id;
```



The screenshot shows a SQL IDE window titled "Sprint\_4 - Abelardo Lopez". The main editor displays a series of SQL queries for creating a table and adding foreign keys. The queries are as follows:

```
275 # Nivell 3
276 ## 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.
277 #Para crear la tabla "transaction_products" con los ids individuales correspondientes a cada transaction_id
278 • CREATE TABLE IF NOT EXISTS transaction_products AS
279 SELECT
280     t.id AS transaction_id,
281     TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(t.products_ids, ',', num), ',', -1)) AS product_id
282 FROM transaction AS t
283 JOIN (SELECT 1 AS num UNION ALL SELECT 2 UNION ALL SELECT 3 UNION ALL SELECT 4) numbers
284 ON CHAR_LENGTH(t.products_ids) - CHAR_LENGTH(REPLACE(t.products_ids, ',', '')) >= num - 1
285 ORDER BY transaction_id;
286
287 #Para modificar la tabla y agregar los FOREIGN KEY
288 • ALTER TABLE transaction_products
289     MODIFY COLUMN product_id INT,
290     ADD FOREIGN KEY (product_id) REFERENCES product(id),
291     ADD FOREIGN KEY (transaction_id) REFERENCES transaction(id);
```

The bottom of the window shows the "Output" panel with the following messages:

#	Time	Action	Message
1	12:41:09	CREATE TABLE IF NOT EXISTS transaction_products AS SELECT t.id AS transaction_id, TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(t.products_ids, ',', num), ',', -1)) AS product_id FROM transaction AS t JOIN (SELECT 1 AS num UNION ALL SELECT 2 UNION ALL SELECT 3 UNION ALL SELECT 4) numbers ON CHAR_LENGTH(t.products_ids) - CHAR_LENGTH(REPLACE(t.products_ids, ',', '')) >= num - 1 ORDER BY transaction_id;	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0
2	12:41:11	ALTER TABLE transaction_products MODIFY COLUMN product_id INT, ADD FOREIGN KEY (product_id) REFERENCES product(id), ADD FOREIGN KEY (transaction_id) REFERENCES transaction(id);	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0

## - Exercici 1

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

Una vez se ha resuelto el grueso del problema realizar esta Query es muy sencillo. Solo tenemos que hacer un conteo de los product\_id de la tabla transaction\_products y agruparlos por el nombre del producto de la tabla product. Al estar estos datos en tablas distintas usamos un JOIN para acceder a la información de ambas tablas fácilmente.

```
SELECT p.name AS product_name, COUNT(tp.product_id) AS sales_count
FROM product AS p
JOIN transaction_products AS tp ON p.id = tp.product_id
GROUP BY name;
```

The screenshot shows a database IDE window titled "Sprint\_4 - Abelardo Lopez". The SQL editor contains the following query:

```
293 ## Exercici 1
294 ### Necessitem conèixer el nombre de vegades que s'ha venut cada producte.
295 #¿Cuántas veces se ha vendido cada producto?
296 • SELECT p.name AS product_name, COUNT(tp.product_id) AS sales_count
297 FROM product AS p
298 JOIN transaction_products AS tp ON p.id = tp.product_id
299 GROUP BY name;
```

Below the editor, the "Result Grid" displays the query results:

product_name	sales_count
Direwolf Stannis	106
Tarly Stark	65
duel tourney Lannister	51
skywalker ewok	100
north of Casterly	54
Karstark Dorne	48
palpatine chewbacca	60
skywalker ewok sith	61
dooku solo	49

The "Output" pane at the bottom shows the execution details:

#	Time	Action	Message
3	12:46:31	SELECT p.name AS product_name, COUNT(tp.product_id) AS sales_count FROM product AS p JOIN transaction_products AS tp ON p.id = tp.product_...	24 row(s) returned