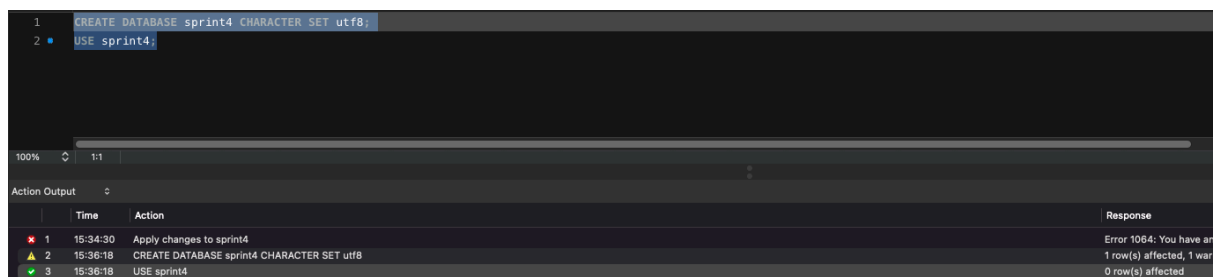


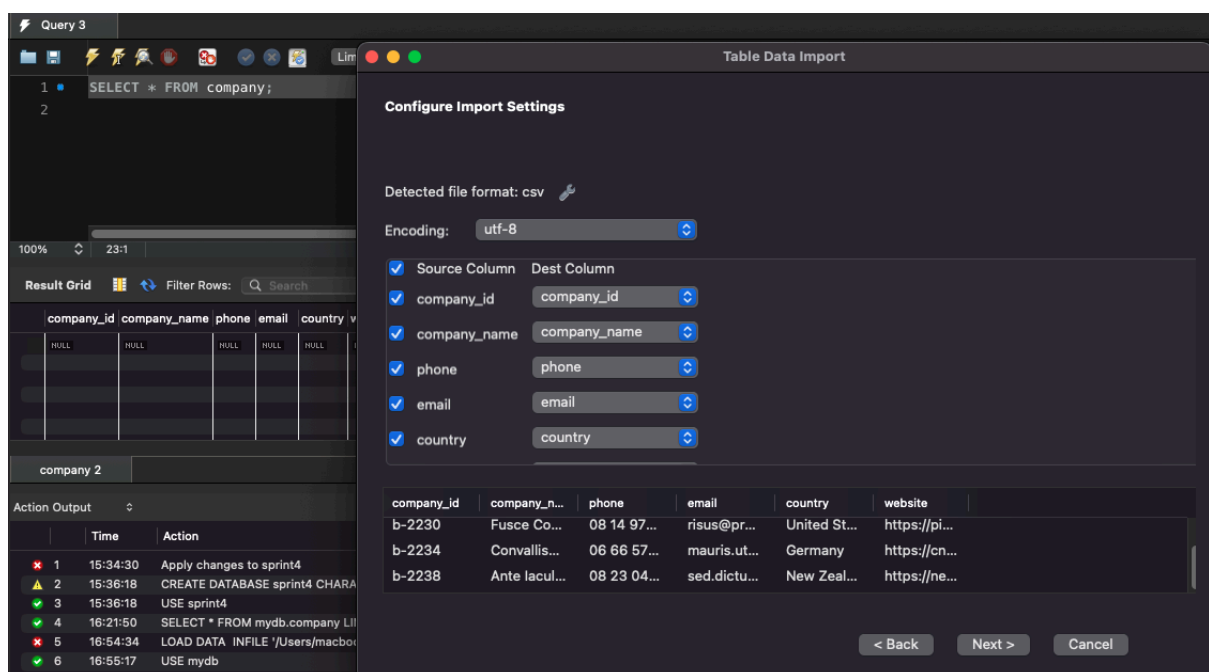
Tasca S4.01. Creació de Base de Dades

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:

Aquí creamos la base de Datos y le indicamos que es la base de datos que queremos usar.



Importamos los datos para la tabla company a través del asistente gráfico, con un select * mostramos los datos anteriormente importados



Tasca S4.01. Creació de Base de Dades

Nivell 1 - Exercici 1

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

The screenshot shows a SQL IDE interface. The top panel displays a SQL query with line numbers 1 through 9. The query is a subquery that selects the name of users from a 'user' table where their user_id is in a list of user_ids from a 'transaction' table, grouped by user_id and having a count of more than 30 transactions.

```
1 SELECT name
2 FROM user
3 WHERE id IN (
4     SELECT user_id
5     FROM transaction
6     GROUP BY user_id
7     HAVING COUNT(id) > 30
8 );
9
```

Below the query editor, the 'Result Grid' is visible, showing a table with one column 'name' and one row with the value 'Lynn'. The 'Action Output' panel at the bottom shows the execution details of the query, including the time (18:18:21), the action (SELECT name FROM user WHERE id IN (SELECT user_id FRO...), the response (1 row(s) returned), and the duration (0.0085 sec / 0).

name
Lynn

Time	Action	Response	Duration / Fetc
18:18:21	SELECT name FROM user WHERE id IN (SELECT user_id FRO...	1 row(s) returned	0.0085 sec / 0

Tasca S4.01. Creació de Base de Dades

Nivell 1 - 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, AVG(t.amount)
2 FROM company c
3 JOIN transaction t
4   ON company_id = business_id
5 JOIN credit_card cc
6   ON t.card_id = cc.id
7 WHERE company_name = 'Donec Ltd'
8 GROUP BY cc.iban;
```

100% 12:8

Result Grid Filter Rows: Search Export:

company_name	iban	AVG(t.amount)
Donec Ltd	PT87806228135092429456346	203.715000

Result 7

Action Output

	Time	Action	Response	Duration / Fetch
✓ 1	15:01:23	SELECT c.company_name, cc.iban, AVG(t.amount) FROM comp...	2 row(s) returned	0.0061 sec / 0.0
✗ 2	15:03:51	SELECT c.company_name, cc.iban, AVG(t.amount) FROM comp...	Error Code: 1054. Unknown column 't.iban' in 'group...	0.0064 sec
✓ 3	15:03:58	SELECT c.company_name, cc.iban, AVG(t.amount) FROM comp...	1 row(s) returned	0.0038 sec / 0.0

Tasca S4.01. Creació de Base de Dades

Nivell 2 - Exercici 1

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:

Quantes targetes estan actives?

Ninguna de las tarjetas tiene mas de una transaccion con lo que solo se declino la tarjeta un maximo de 1 vez

```
1  SELECT cc.id, COUNT(cc.id),t.timestamp
2  FROM credit_card cc
3  JOIN transaction t
4  ON cc.id = t.card_id
5  WHERE t.declined != 0
6  GROUP BY cc.id, t.timestamp;
7
```

100% 1:7

Result Grid Filter Rows: Search Export:

id	COUNT(cc.id)	timestamp
CcU-2938	1	2021-05-09 10:25:08
CcU-2945	1	2021-06-15 00:26:29
CcU-2952	1	2021-05-06 05:33:39
CcU-2959	1	2021-04-17 05:30:17
CcU-2966	1	2021-06-02 06:19:00
CcU-2973	1	2021-07-31 23:03:21
CcU-2980	1	2022-03-05 20:41:20
CcU-2987	1	2021-05-18 12:03:25
CcU-2994	1	2021-08-13 13:44:05
CcU-3001	1	2021-10-13 11:30:20
CcU-3008	1	2021-03-29 16:15:13
CcU-3015	1	2022-01-02 16:59:28
CcU-3022	1	2022-01-31 11:33:27
CcU-3029	1	2022-02-13 16:33:50
CcU-3036	1	2021-08-17 10:19:03
CcU-3043	1	2022-02-12 11:00:08
CcU-3050	1	2021-07-19 07:13:52
CcU-3057	1	2021-05-29 12:44:03

Result 5

Action Output

	Time	Action	Response	Duration / F
✓ 1	15:57:23	SELECT cc.id, COUNT(cc.id),t.timestamp FROM credit_card cc...	87 row(s) returned	0.013 sec / C

Tasca S4.01. Creació de Base de Dades

Creo la tabla credit_card status

```
1 CREATE TABLE credit_card_status(  
2     id INT,  
3     active TINYINT,  
4     credit_card_id VARCHAR(15) UNIQUE,  
5  
6     CONSTRAINT pk_id PRIMARY KEY(id),  
7     CONSTRAINT fk_credit_card FOREIGN KEY(credit_card_id)  
8         REFERENCES credit_card(id)  
9 );  
10  
11
```

100% 28:4

Apply Revert

Action Output

	Time	Action	Response	Duration / Fetch Time
✓ 1	15:57:23	SELECT cc.id, COUNT(cc.id), t.timestamp FROM credit_card cc...	87 row(s) returned	0.013 sec / 0.00008
✗ 2	16:21:52	CREATE TABLE credit_card_status(id INT, active TINYINT, cre...	Error Code: 3780. Referencing column 'credit_card_id...	0.0099 sec
✓ 3	16:23:18	CREATE TABLE credit_card_status(id INT, active TINYINT, cre...	0 row(s) affected	0.041 sec

Con el resultado de la consulta anterior puedo saber que todas las tarjetas estan activas con lo que añado esos datos en un nuevo csv para cargarlos a la tabla credit_card status

script.sql U credit_cards_status.csv credit_cards_status - credit_cards.csv x

Users > macbook > Downloads > credit_cards_status - credit_cards.csv > data




```
1 id,active,credit_card_id  
2 275,1,CcU-2938  
3 274,1,CcU-2945  
4 273,1,CcU-2952  
5 272,1,CcU-2959  
6 271,1,CcU-2966  
7 270,1,CcU-2973  
8 269,1,CcU-2980  
9 268,1,CcU-2987  
10 267,1,CcU-2994  
11 266,1,CcU-3001  
12 265,1,CcU-3008
```

Tasca S4.01. Creació de Base de Dades

Y con los datos en la tabla ya se puede proceder a la consulta


```
1 SELECT COUNT(ccs.id) AS targetas_activas
2 FROM credit_card cc
3 JOIN credit_card_status ccs
4 ON cc.id = ccs.credit_card_id
5 WHERE ccs.active !=0;
```

100% 28:4

Result Grid   Filter Rows: Export: 

targetas_activ...
275

Result 16

Action Output 

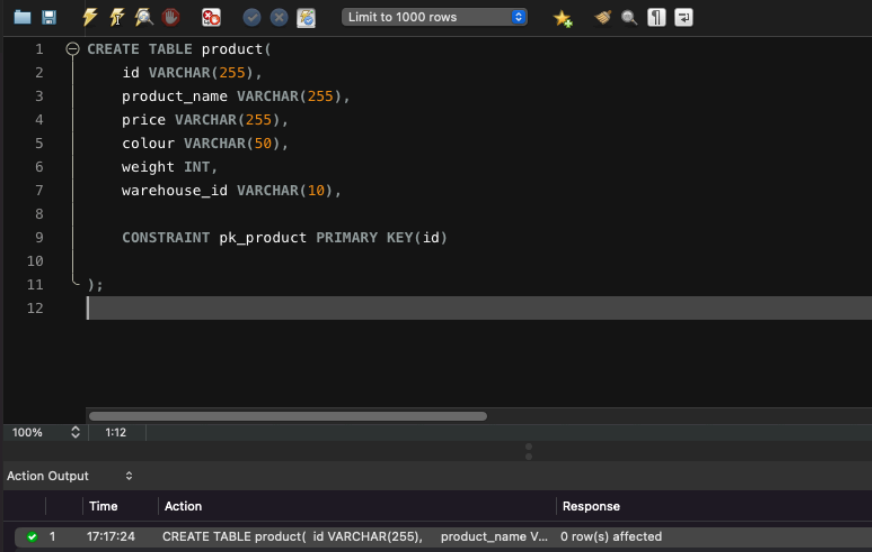
	Time	Action	Response
✓ 1	16:42:38	SELECT COUNT(ccs.id) AS targetas_activas FROM credit_card...	1 row(s) returned

Tasca S4.01. Creació de Base de Dades

Nivell 3 - Exercici 1

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:

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



```
1 CREATE TABLE product(  
2   id VARCHAR(255),  
3   product_name VARCHAR(255),  
4   price VARCHAR(255),  
5   colour VARCHAR(50),  
6   weight INT,  
7   warehouse_id VARCHAR(10),  
8  
9   CONSTRAINT pk_product PRIMARY KEY(id)  
10 );  
11  
12
```

100% 1:12

Action Output

	Time	Action	Response
✓ 1	17:17:24	CREATE TABLE product(id VARCHAR(255), product_name V...	0 row(s) affected

Tasca S4.01. Creació de Base de Dades

Esta es la solución que se me ocurre para hacer la consulta con el diseño actual

```
1 SELECT SUM(CASE WHEN '1' IN (product_ids) THEN 1 ELSE 0 END) AS product_1_sales,
2 SUM(CASE WHEN '2' IN (product_ids) THEN 1 ELSE 0 END) AS product_2_sales,
3 SUM(CASE WHEN '3' IN (product_ids) THEN 1 ELSE 0 END) AS product_3_sales,
4 SUM(CASE WHEN '4' IN (product_ids) THEN 1 ELSE 0 END) AS product_4_sales,
5 SUM(CASE WHEN '5' IN (product_ids) THEN 1 ELSE 0 END) AS product_5_sales,
6 SUM(CASE WHEN '6' IN (product_ids) THEN 1 ELSE 0 END) AS product_6_sales,
7 SUM(CASE WHEN '7' IN (product_ids) THEN 1 ELSE 0 END) AS product_7_sales,
8 SUM(CASE WHEN '8' IN (product_ids) THEN 1 ELSE 0 END) AS product_8_sales,
9 SUM(CASE WHEN '9' IN (product_ids) THEN 1 ELSE 0 END) AS product_9_sales
10 FROM transaction;
11
```

100% 18:10

Result Grid Filter Rows: Search Export:

product_1_sales	product_2_sal...	product_3_sal...	product_4_sal...	product_5_sal...	product_6_sal...	product_7_sales	product_8_sal...	product_9_sal...
4	3	4	0	7	0	4	0	0

Result 4

Action Output

	Time	Action	Response	Duration /
✓ 1	17:56:03	SELECT SUM(CASE WHEN '1' IN (product_ids) THEN 1 ELSE 0 E...	1 row(s) returned	0.0035 se

Sin embargo, optimizaría este modelo añadiendo una nueva tabla en la base de datos, que podría llamarse **product_transaction**, con una relación muchos a muchos (N-M). Un producto puede estar en muchas transacciones, y una transacción puede contener muchos productos. En esta nueva tabla se almacenarían dos claves foráneas que referencian a las tablas **product** y **transaction**. De esta manera, se podría optimizar la consulta para saber cuántas veces se ha vendido un producto del mismo tipo.

Esta sería la consulta.

```
SELECT tp.product_id, COUNT(p.id) AS sales_count
FROM transaction_product
GROUP BY tp.product_id;
```