SPRINT 4

Lucía Trinidad Fernández Gagliano

29/01/2025

Corregido por:

NIVELL 1

PARA PODER REALIZAR LOS EJERCICIOS DEL NIVEL 1:

-- > primero creamos la base de datos

```
12 --- CREAMOS BASE DE DATOS
14 --- CREATE database transtrans;
16 • USE transtrans;
```

-- > luego creamos las tablas de dimensiones users, credit_card, company

```
-- CREAMOS LAS TABLAS DE 'DIMENSIONES'
21 • DROP TABLE IF exists users;
22 ullet CREATE TABLE IF NOT EXISTS users (
     id INT NOT NULL PRIMARY KEY,
23
     name VARCHAR (50),
     surname VARCHAR (100),
25
phone VARCHAR (50),
27 email VARCHAR (100),
28 birth_date VARCHAR (30),
29
    country VARCHAR (100),
    city VARCHAR (100),
30
31
     postal_code VARCHAR (20),
      address VARCHAR (100)
33
```

```
35
36 • DROP TABLE IF exists companies;
37 • ○ CREATE TABLE IF NOT EXISTS companies (
38 company_id VARCHAR(10) PRIMARY KEY,
39 company_name VARCHAR (100),
40 phone VARCHAR (30),
41 email VARCHAR (100),
42 country VARCHAR (100),
43 website VARCHAR (100)
44 );
```

```
46 • DROP TABLE IF exists credit_cards;
id VARCHAR (10) PRIMARY KEY,
49
     user_id VARCHAR (10),
50
     iban VARCHAR (50),
51
     pan VARCHAR (50),
    pin VARCHAR(4),
52
53
    CVV VARCHAR(3),
    track1 VARCHAR (100),
54
55 track2 VARCHAR (100),
56 expiring_date VARCHAR (20)
57 );
```

-- > configuramos MySQL Workbench para que permita la carga de archivos CSV desde la carpeta indicada

```
63 • SHOW VARIABLES LIKE 'secure_file_priv'; -- > para ver dónde se alojan los archivos que se cargan
      -- > 'C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\'
      -- > subo las tablas de datos a utilizar a la carpeta Uploads
67
     #EJECUTO:
68 • SET GLOBAL local_infile=on;
69 • SHOW GLOBAL VARIABLES LIKE 'local infile'; #ON
70
71
    # INTEGRO LA SIGUIENTE ORDEN EN EL Manage Conections - Advanced . Test Conection:
72
     # OPT_LOCAL_INFILE=1
      # ME SIGUE DANDO ERROR
     # Error Code: 2068. LOAD DATA LOCAL INFILE file request rejected due to restrictions on access.
76
      _____
     ### | PRUEBO QUITANDO EL 'LOCAL' Y AHORA SÍ ME CARGA LOS DATOS !!! (emoción)
77
```

-- > Cargamos lo datos de cada tabla de dimensiones:

```
79 -- > CARGAMOS DATOS DE LA TABLA USERS
80
81 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca.csv"
82 INTO TABLE users
83 FIELDS TERMINATED BY ','
84 ENCLOSED BY '"'
85 LINES TERMINATED BY '\r\n' # agregamos \r porque birthday tiene una coma que divide la celda
86 IGNORE 1 ROWS;
87
88 • LOAD DATA 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' # agregamos \r porque birthday tiene una coma que divide la celda
93 IGNORE 1 ROWS;
```

```
95 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv"
 96
       INTO TABLE users
 97
       FIELDS TERMINATED BY ','
       ENCLOSED BY '"'
       LINES TERMINATED BY '\r\n' # agregamos \r porque birthday tiene una coma que divide la celda
100
      IGNORE 1 ROWS;
101
102
       -- > CARGAMOS DATOS DE LA TABLA COMPANY
104 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv"
105
     INTO TABLE companies
     FIELDS TERMINATED BY ','
106
     ENCLOSED BY """
     LINES TERMINATED BY '\n'
      IGNORE 1 ROWS;
111
      -- > CARGAMOS DATOS DE LA TABLA credit_cards
112
113 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit cards.csv"
      INTO TABLE credit cards
115 FIELDS TERMINATED BY ','
116 ENCLOSED BY '"'
117 LINES TERMINATED BY '\n'
118
    IGNORE 1 ROWS;
```

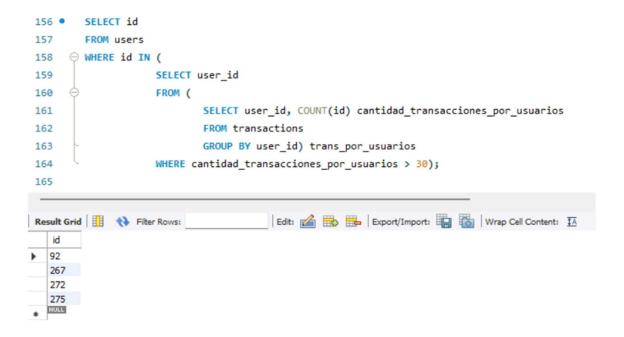
-- > y finalmente creamos la tabla de hechos *transactions* y le cargamos sus datos.

```
DROP TABLE IF exists transactions;
124 • ⊖ CREATE TABLE IF NOT EXISTS transactions (
125
                 id VARCHAR(255) PRIMARY KEY,
                 card_id VARCHAR(15), #es varchar(10)
126
                 business id VARCHAR(10), # ok
127
128
                 timestamp TIMESTAMP,
                 amount DECIMAL(10, 2),
129
                 declined boolean,
130
                 product ids varchar (200),
131
                 user_id INT, #ok
132
                 lat FLOAT,
133
                 longitude FLOAT,
134
         FOREIGN KEY (card_id) REFERENCES credit_cards(id),
135
         FOREIGN KEY (business id) REFERENCES companies(company id),
136
         FOREIGN KEY (user_id) REFERENCES users(id)
137
       );
138
142 •
       LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv"
      INTO TABLE transactions
143
     FIELDS TERMINATED BY ';' # esta tabla está separada por ;
      ENCLOSED BY ""
145
146
     LINES TERMINATED BY '\n'
     IGNORE 1 ROWS;
```

-- > AHORA SÍ, CON LOS DATOS CARGADOS, PASAMOS A LOS EJERCICIOS

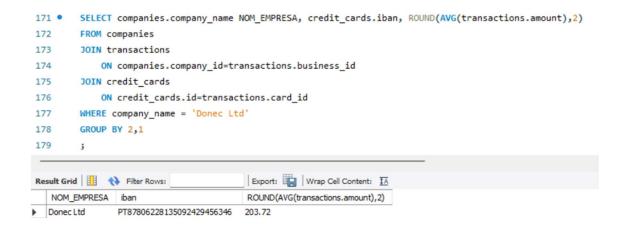
NIVELL 1 - EXERCICI 1

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



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.

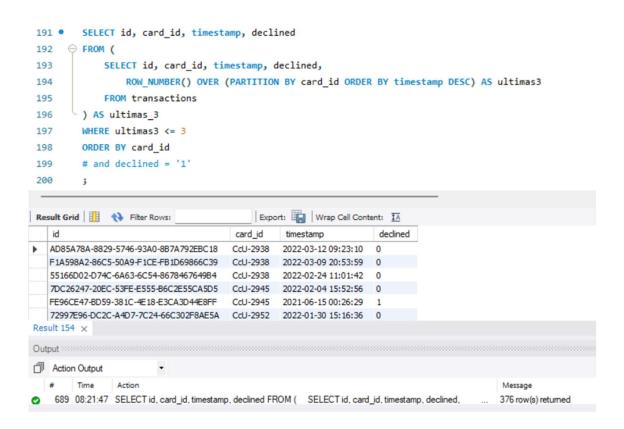


NIVELL 2 - EXERCICI 1

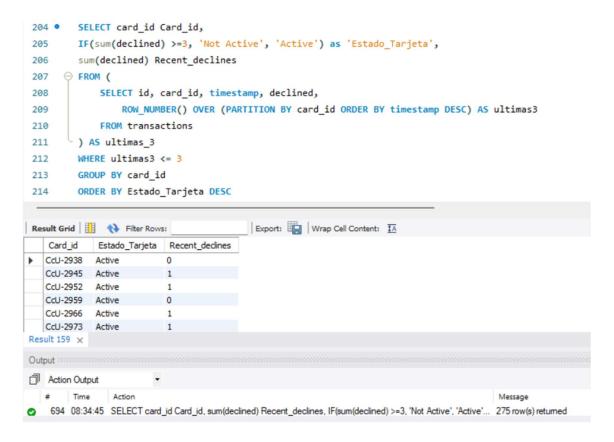
- -- > Crea una nova taula que reflecteixi l'estat de les targetes de crèdit basat en si les últimes tres transaccions van ser declinades i genera la següent consulta:
- -- > Quantes targetes estan actives?

ACLARACIÓN: CREÉ DOS TABLAS PORQUE LA PRIMERA NO ME TERMINABA DE CONVENCER. EN LA SEGUNDA VERSIÓN FINAL ADEMÁS DE VER SI LA TARJETA ESTÁ ACTIVA O NO, SE PUEDE VER EL NÚMERO DE DECLINES RECIENTES (EN LAS ÚLTIMAS TRES TRANSACCIONES).

--> Primero buscamos el modo de seleccionar las última tres transcacciones por usuario de acuerdo al timestamp utilizando la función ROW NUMBER()



--> Luego generamos una columna mediante el IF que me distinga a cada tarjeta como Activa o No Activa si las últimas 3 transacciones fueron declinadas



--> CREAMOS LA TABLA card_status con la columna de Recent_declines

-- > chequeamos cómo queda la tabla

```
235 • CREATE TABLE card_status
      SELECT card_id Card_id,
       IF(sum(declined) >=3, 'Not Active', 'Active') as 'Estado_Tarjeta',
238
      sum(declined) Recent_declines
239 🧇 FROM (
240
           SELECT id, card_id, timestamp, declined,
               ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS ultimas3
241
242
            FROM transactions
      ) AS ultimas_3
243
244 WHERE ultimas3 <= 3
245 GROUP BY card_id
     ORDER BY Estado_Tarjeta DESC
249 • SELECT * from card_status;
250
| Edit: 🚄 🖶 | Export/Import: 🏣 🚡 | Wrap Cell Content: 🖽
  Card_id Estado_Tarjeta Recent_declines
                      1
  CcU-2938 Active
  CcU-2945 Active
  CcU-2952 Active
                        1
  CcU-2959 Active 0
  CcU-2966 Active
card_status 197 ×
Action Output
746 10:59:44 SELECT id, card_id, timestamp, declined FROM ( SELECT id, card_id, timestamp, declined,
                                                                                   376 row(s) returned
747 11:02:13 SELECT * from card_status
                                                                                    275 row(s) returned
```

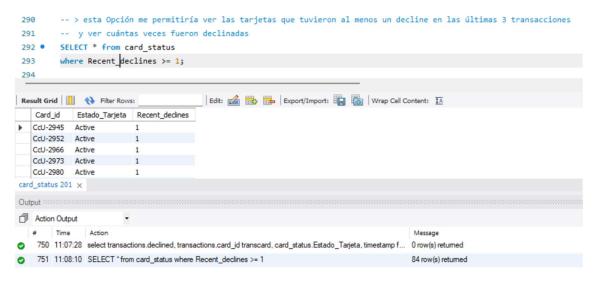
-- > AGREGAMOS LA PRIMERY KEY A LA NUEVA TABLA Y LA FK QUE LA CONECTE A LA TABLA credit cards

```
252
253 • ALTER TABLE card_status ADD PRIMARY KEY(card_id);
254
255 • ALTER TABLE credit_cards
256 ADD FOREIGN KEY (id) REFERENCES card_status(card_id);
```

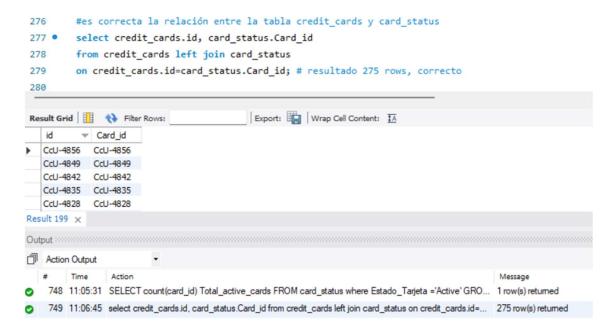
-- > A PARTIR DE AHORA, NUESTRO MODELO SE CONVIERTE EN UN MODELO 'COPO DE NIEVE' < --

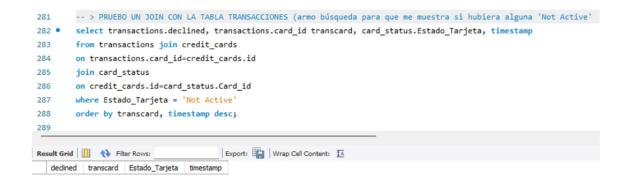
--> VEMOS CUÁNTAS DE LAS TARJETAS ESTÁN ACTIVAS

--> esta Opción me permite ver las tarjetas que tuvieron al menos un decline en las últimas 3 transacciones y ver cuántas veces fueron declinadas



--> HACEMOS CORROBORACIONES:







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.

-- CREAMOS LA TABLA products

```
346 -- CREAMOS LA TABLA products
347 ● DROP TABLE IF EXISTS products;
348 ● ○ CREATE TABLE IF NOT EXISTS products (
349 id INT PRIMARY KEY NOT NULL,
350 product_name VARCHAR(200),
351 price VARCHAR (200),
352 colour VARCHAR(20),
353 weight DECIMAL (10,2),
354 warehouse_id VARCHAR(100)
355 );
```

-- CARGAMOS DATOS

Me genera conflicto con el símbolo \$ en la columna Price. Resuelvo mediante: SET price = REPLACE(@price_raw, '\$', ");

```
-- CON REEMPLAZO DEL $ EN PRICE

370 LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv"

INTO TABLE products

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS

(id, product_name, @price_raw, colour, weight, warehouse_id)

SET price = REPLACE(@price_raw, '$', ''); #QUITAMOS EL $ DE PRICE
```

-- CAMBIAMOS EL CAMPO 'PRICE' DE VARCHAR A DECIMAL y CORROBORAMOS LA CARGA DE DATOS, OK ;)

```
-- CAMBIAMOS EL CAMPO 'PRICE' DE VARCHAR A DECIMAL
380
381 •
         ALTER TABLE products MODIFY price DECIMAL (10,2);
382
        -- CORROBORAMOS LA CARGA DE DATOS, OK ;)
383
         SELECT * from products;
384 •
385
                                        Edit: 🚄 🖶 Export/Import: 📳
warehouse id
        product name
                           price
                                  colour
        Direwolf Stannis
                           161.11
                                  #7c7c7c
                                           1.00
                                                  WH-4
   1
   2
       Tarly Stark
                          9.24
                                  #919191 2.00
                                                 WH-3
   3
        duel tourney Lannister
                           171.13 #d8d8d8
                                           1.50
                                                  WH-2
       warden south duel
                          71.89
                                  #111111 3.00
                                                 WH-1
        skywalker ewok
   5
                           171.22 #dbdbdb
                                           3.20
                                                  WH-0
        dooku solo
   6
                           136.60
                                  #c4c4c4
                                           0.80
                                                  WH--1
products 166 x
Output
Action Output
        Time
               Action
   701 08:51:36 SELECT * from card_status2 where Recent_declines >= 1
```

-- PROBLEMA:

- -- la columna 'produc_ids' de la tabla transactions, al tener varios product ids en cada registro no me matchea con los id de la tabla products
- -- CREAMOS UNA TABLA INTERMEDIA PARA EVITAR LA RELACIÓN MUCHOS A MUCHOS ENTRE LA TABLA PRODUCTS Y LA TABLA TRANSACTIONS

-- utilizamos estrategia mediante tabla JSON para separar los valores de la columna produc_ids de la tabla transactions

```
408 •
          DROP TABLE IF EXISTS products transactions;
          CREATE TABLE IF NOT EXISTS products_transactions
 409 •
          SELECT DISTINCT
 410
 411
              transactions.id AS transaction id,
 412
              product id
 413
         FROM
 414
              transactions
 415
          JOIN
             JSON TABLE(
 416
                  CONCAT('["', REPLACE(transactions.product_ids, ',', '","'), '"]'),
 417
 418
                   '$[*]' COLUMNS (product_id INT PATH '$')
 419
 420
              ON jt.product_id = product_id
         LEFT JOIN
421
              products
 422
 423
              ON jt.product_id = product_id
         ORDER BY
 424
 425
              transactions.id;
       SELECT * FROM products_transactions;
428
                                   Edit: 🕍 📆 📙 Export/Import: 🏣 👸 | Wrap Cell Content: 🔣 | Fetch rows:
transaction_id
                                product_id
▶ 02C6201E-D90A-1859-B4EE-88D2986D3B02
  122DC333-E19F-D629-DCD8-9C54CF1EBB9A 1
  1753A288-9FC1-52E6-5C39-A1FFB97B0D3A
  1A6CECFB-2E3A-65A3-72D9-2FDB58A1E4BA 1
  1EA2B262-D507-AD14-4374-4D532967113F
products_transactions 168 ×
Output :
Action Output
 # Time
            Action
                                                                           Message

    703 08:59:24 SELECT * FROM products_transactions

                                                                          1457 row(s) returned
430
        --> GENERAMOS LA PRIMARY KEY COMPUESTA DE NUESTRA TABLA INTERMEDIA products_transactions
        _____
432 • ALTER TABLE products transactions
      ADD PRIMARY KEY (transaction_id, product_id)
433
434
```

```
436
       --> GENERAMOS LAS FOREING KEYS EN LAS TABLA products_transactions CON LAS TABLAS transactions y products
437
438
       # LA FK CON transactions ME FUNCIONA
439 • ALTER TABLE products_transactions
       ADD CONSTRAINT fk_transactions_id
       FOREIGN KEY (transaction_id)
      REFERENCES transactions(id)
442
444 • ALTER TABLE products_transactions
       ADD CONSTRAINT fk_product_id
       FOREIGN KEY (product_id)
446
       REFERENCES products(id)
448
```

.....

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

```
451
       -- > Necessitem conèixer el nombre de vegades que s'ha venut cada producte.
       ______
454 • SELECT product id, COUNT(transaction id)cantidad vendida
     FROM products_transactions
    GROUP BY product id
457
     ORDER BY product_id;
Export: Wrap Cell Content: IA
  product_id cantidad_vendida
          61
  1
  2
         65
  3
          51
         49
  5
Result 170 ×
Action Output
 705 09:02:29 SELECT product_id, COUNT(transaction_id)cantidad_vendida FROM products_transactions GROU... 26 row(s) returned
```

--> EXTRA: así podemos corroborar la cantidad de ventas de todos los productos presentes en la tabla 'products' viendo también cuál no ha tenido ninguna venta hasta ahora

```
SELECT products.id, cantidad_vendida from products
463
      ⊖ left join (
                 SELECT product_id, COUNT(transaction_id)cantidad_vendida
464
                  FROM products_transactions
465
466
                  GROUP BY product_id
467
                  ORDER BY product_id) cantidades
              ON products.id=cantidades.product_id;
468
469
Export: Wrap Cell Content: IA
         cantidad_vendida
   1
         61
   2
         65
   3
         51
   5
         49
Result 176 ×
Output
Action Output
       Time
               Action
                                                                                            Message
  711 09:05:12 SELECT product_id, COUNT(transaction_id)cantidad_vendida FROM products_transactions GROU... 26 row(s) returned
712 09:05:26 SELECT products.id, cantidad_vendida from products left join (SELECT product_id, COUNT(transac... 100 row(s) returned
```