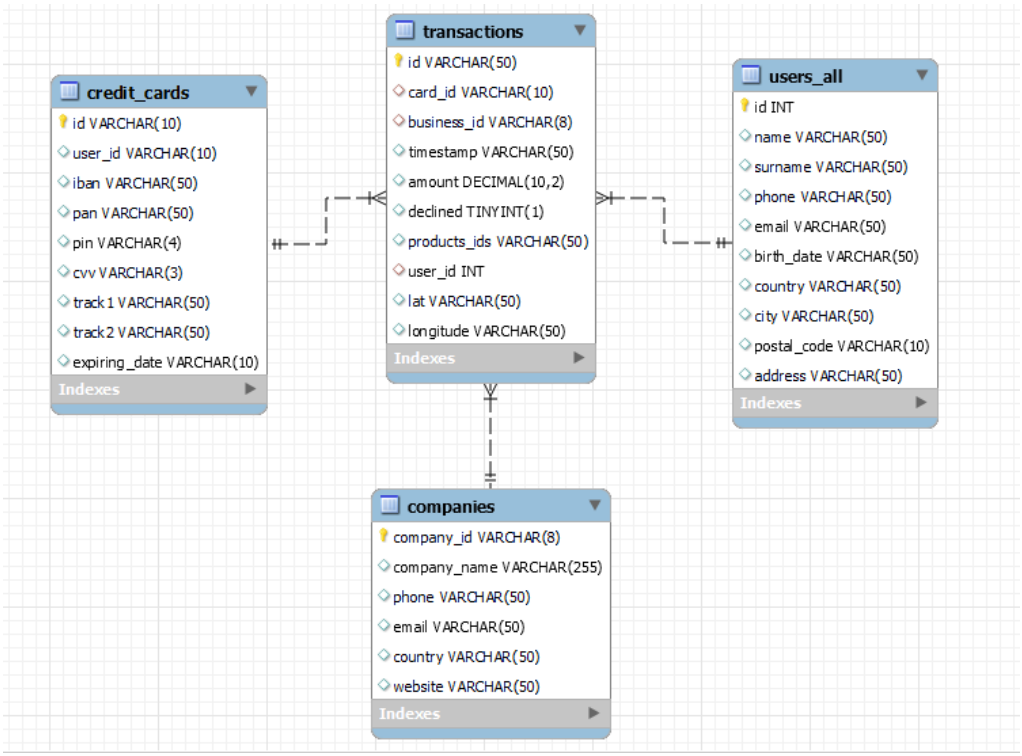
Nivel 1

La **base de datos** creada se llama **"TransactionsT4".** Inicialmente tendrá 4 tablas:

- Companies
- Credit_cards
- Users_all
- Transactions

Las tablas siguen un esquema de tipo estrella, con la tabla **Transactions como tabla de hechos** y las otras como tabla de dimensiones. Todas estas tienen una **relación de 1 a N con la tabla central**, ya que cada compañía, usuario y tarjeta de crédito puede realizar muchas transacciones.

DB TransactionsT4

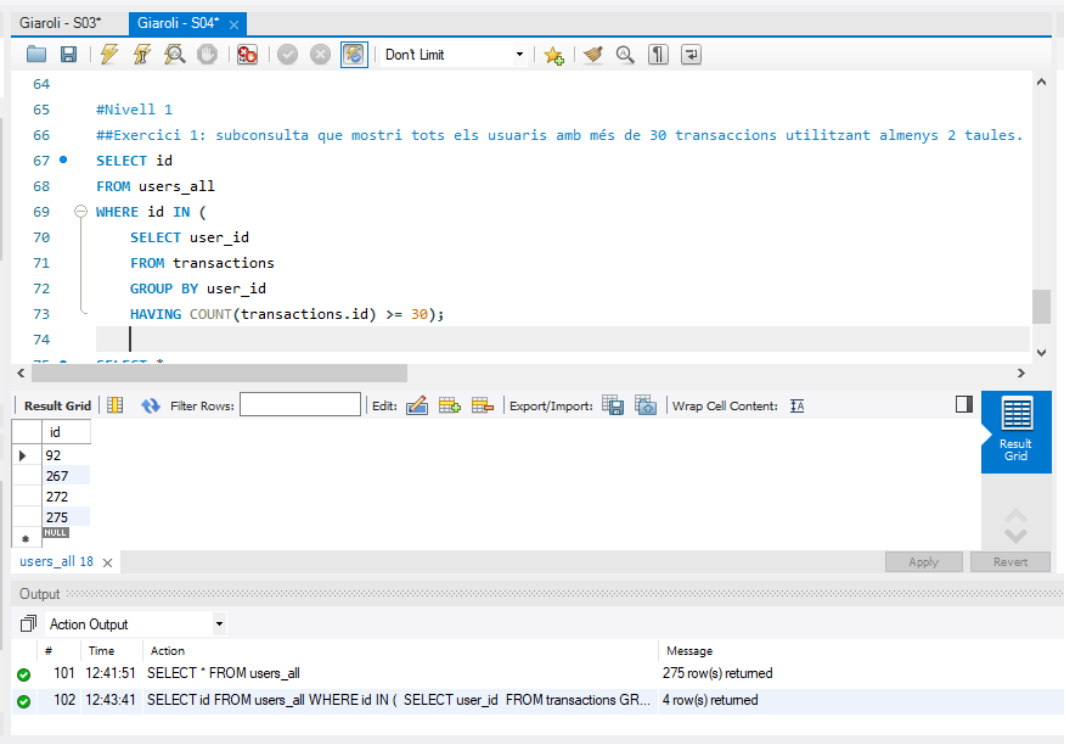


Para una explicación de la creación de la base de datos y de las tablas ver la sección ANEXO al final del documento.

Ejercicio 1

Los usuarios con más de 30 transacciones son los que tienen los siguientes ID:

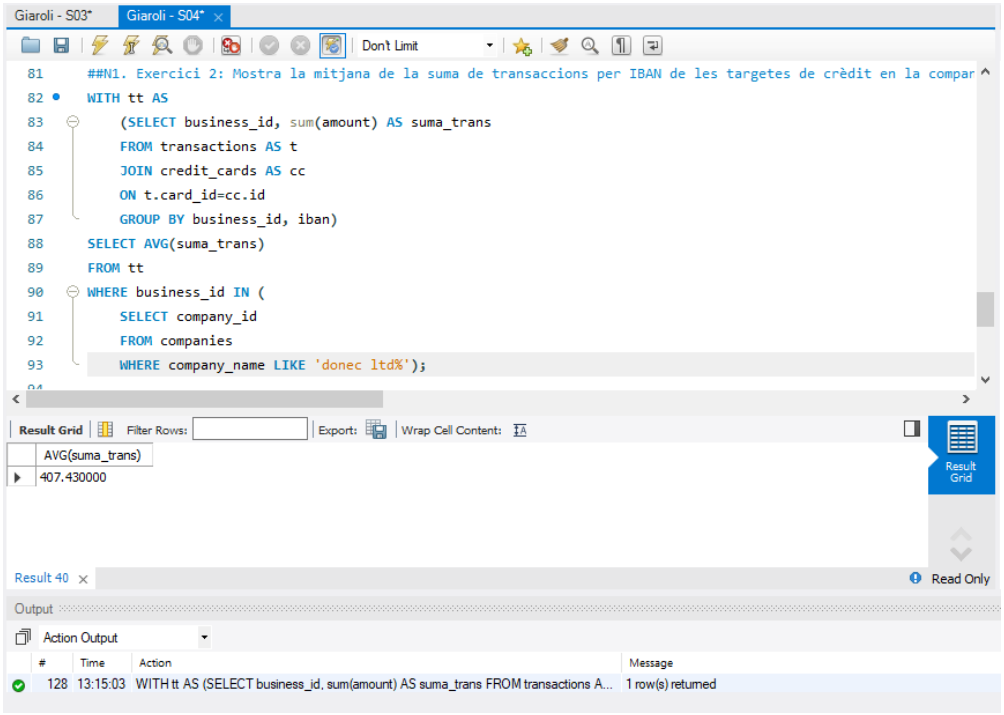
- 92
- 267
- 272
- 275



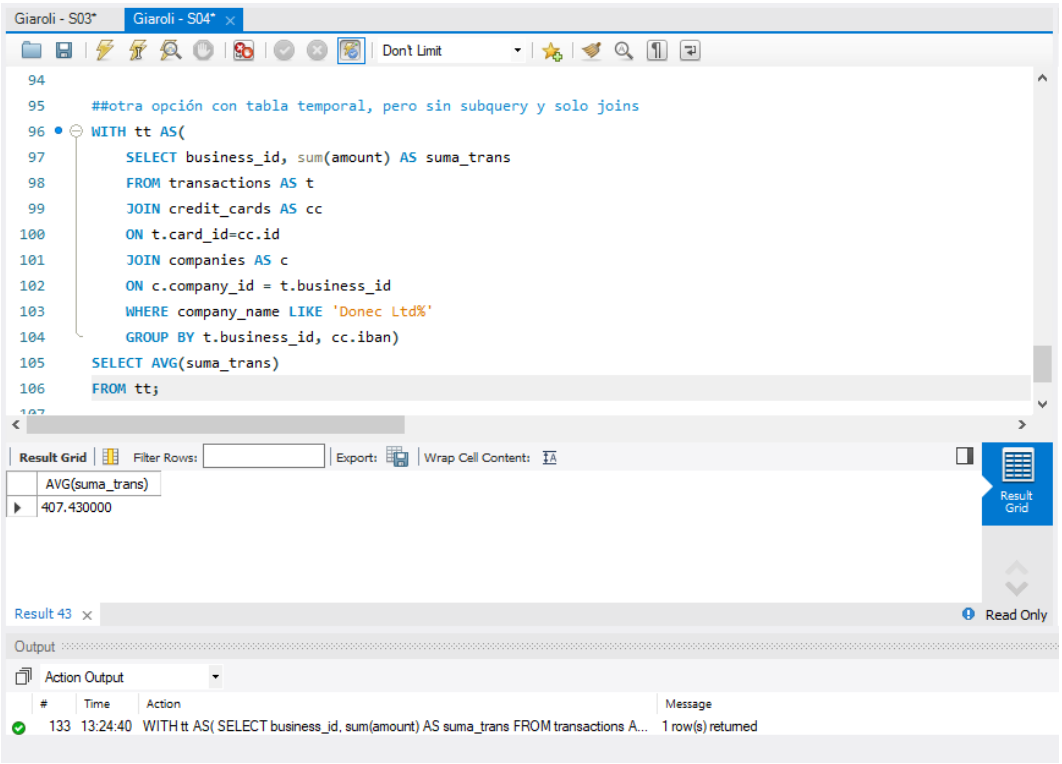
Ejercicio 2

La media de la suma de transacciones por IBAN de las tarjetas de crédito de la compañía Donec Ltd. es de 407.43.

Para obtener este resultado, primero obtenemos la suma de las transacciones agrupadas por Id de compañía e IBAN en una tabla temporal y, luego, calculamos el promedio de esas sumas para la compañía Donec.

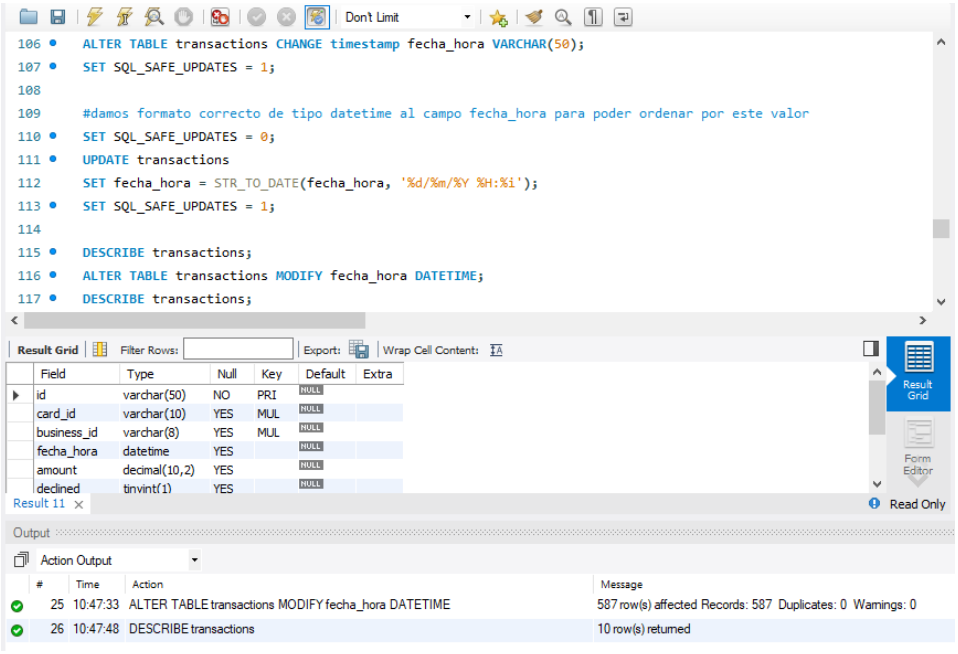


Otra opción de hacerlo, también con tabla temporal, pero sin subquery y sólo joins sería la siguiente:

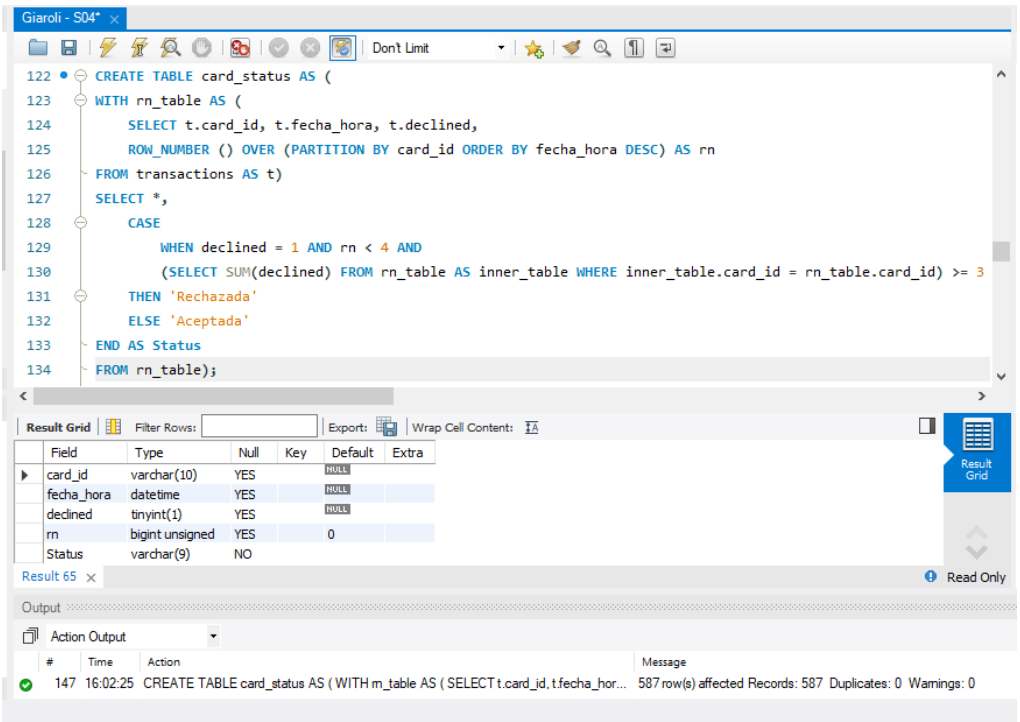


Nivel 2

Antes de crear la tabla, cambiaremos el nombre del campo timestamp a fecha_hora para evitar errores y luego le daremos formato de tipo datetime para poder ordenar las transacciones por fecha.

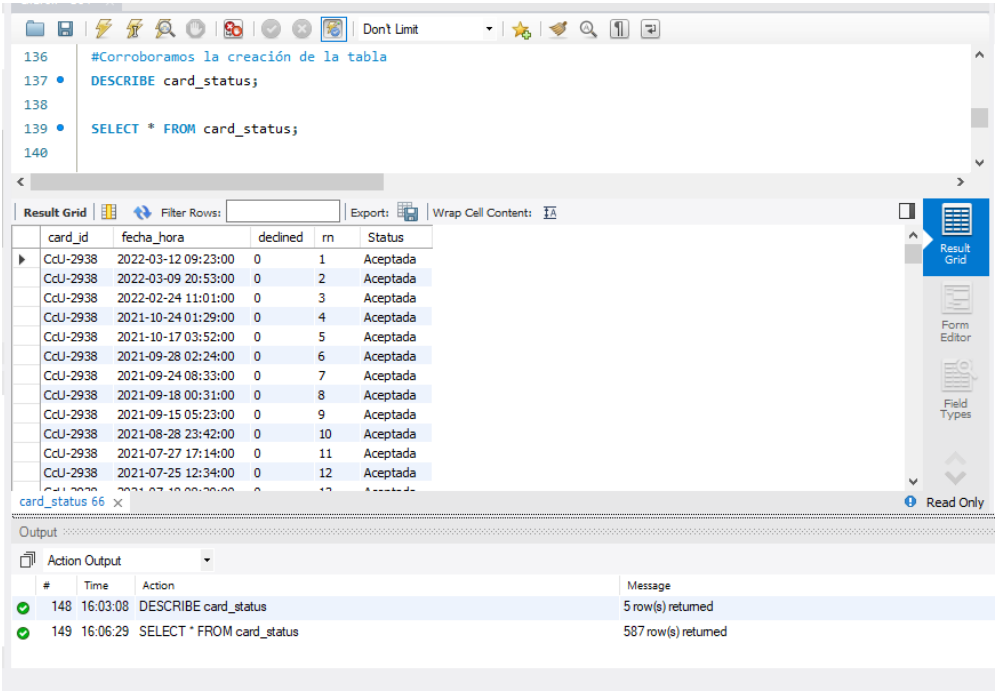


Creamos la tabla para poder hacer la consulta. La función row_number nos permite asignar un número a cada transacción, partition by nos permite hacer una partición del conteo y recomenzarlo por cada tarjeta y con order by podemos ordenar las transacciones por fecha realizada.



Por último, añadimos una columna “Status” que analiza cada caso (CASE WHEN) y asigna el status “Rechazado” sólo si la suma de las últimas tres transacciones es igual o mayor que 3, es decir si las últimas 3 transacciones han sido declinadas para el mismo número de card_id.

Luego corroboramos la creación de la tabla:

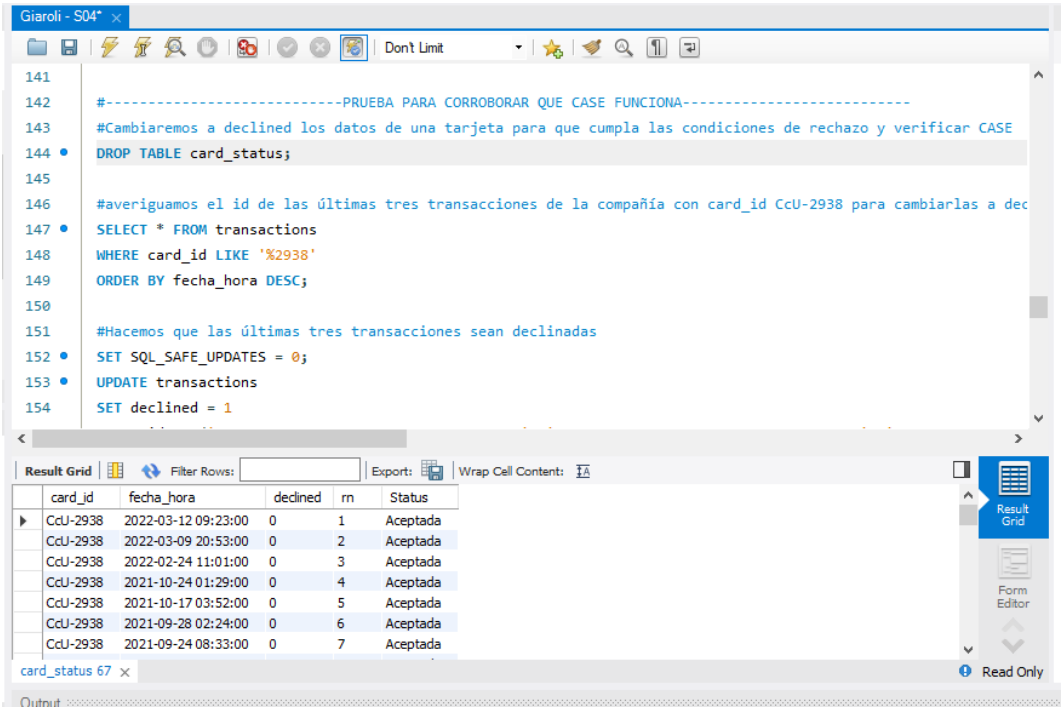


```
#Corroboramos la creación de la tabla
DESCRIBE card_status;

SELECT * FROM card_status;
```

card_id	fecha_hora	declined	rn	Status
CcU-2938	2022-03-12 09:23:00	0	1	Aceptada
CcU-2938	2022-03-09 20:53:00	0	2	Aceptada
CcU-2938	2022-02-24 11:01:00	0	3	Aceptada
CcU-2938	2021-10-24 01:29:00	0	4	Aceptada
CcU-2938	2021-10-17 03:52:00	0	5	Aceptada
CcU-2938	2021-09-28 02:24:00	0	6	Aceptada
CcU-2938	2021-09-24 08:33:00	0	7	Aceptada
CcU-2938	2021-09-18 00:31:00	0	8	Aceptada
CcU-2938	2021-09-15 05:23:00	0	9	Aceptada
CcU-2938	2021-08-28 23:42:00	0	10	Aceptada
CcU-2938	2021-07-27 17:14:00	0	11	Aceptada
CcU-2938	2021-07-25 12:34:00	0	12	Aceptada

```
Output
Action Output
#  Time      Action                          Message
148 16:03:08 DESCRIBE card_status            5 row(s) returned
149 16:06:29 SELECT * FROM card_status       587 row(s) returned
```

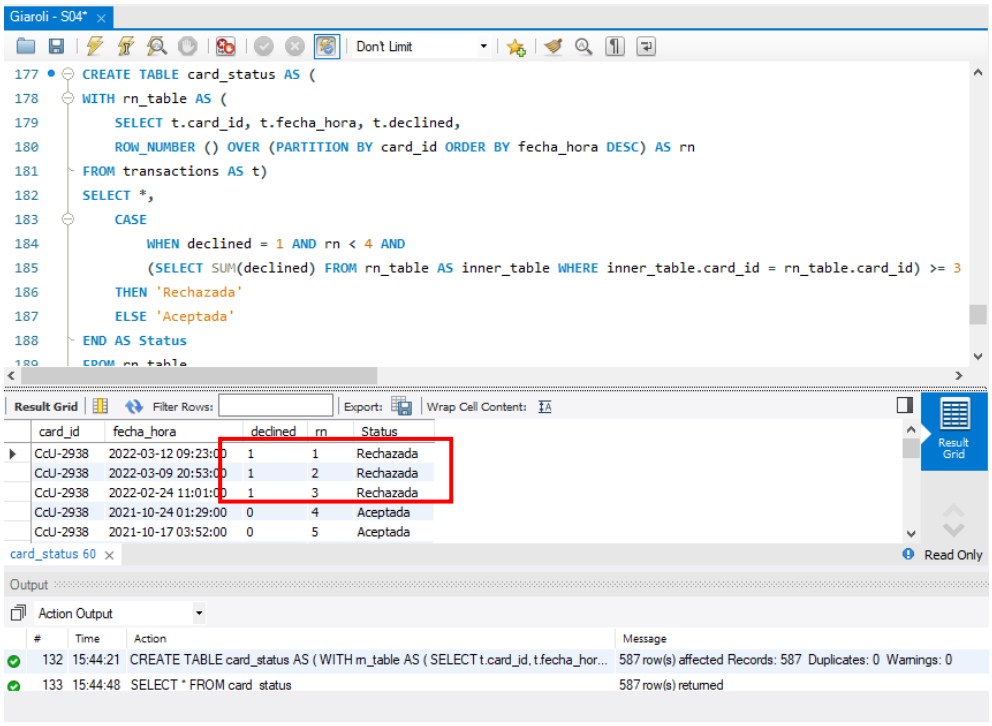
Como todas las transacciones por tarjeta salen aceptadas, editaremos dos transacciones para que se cumpla el caso de rechazo, para ver si las analiza correctamente.



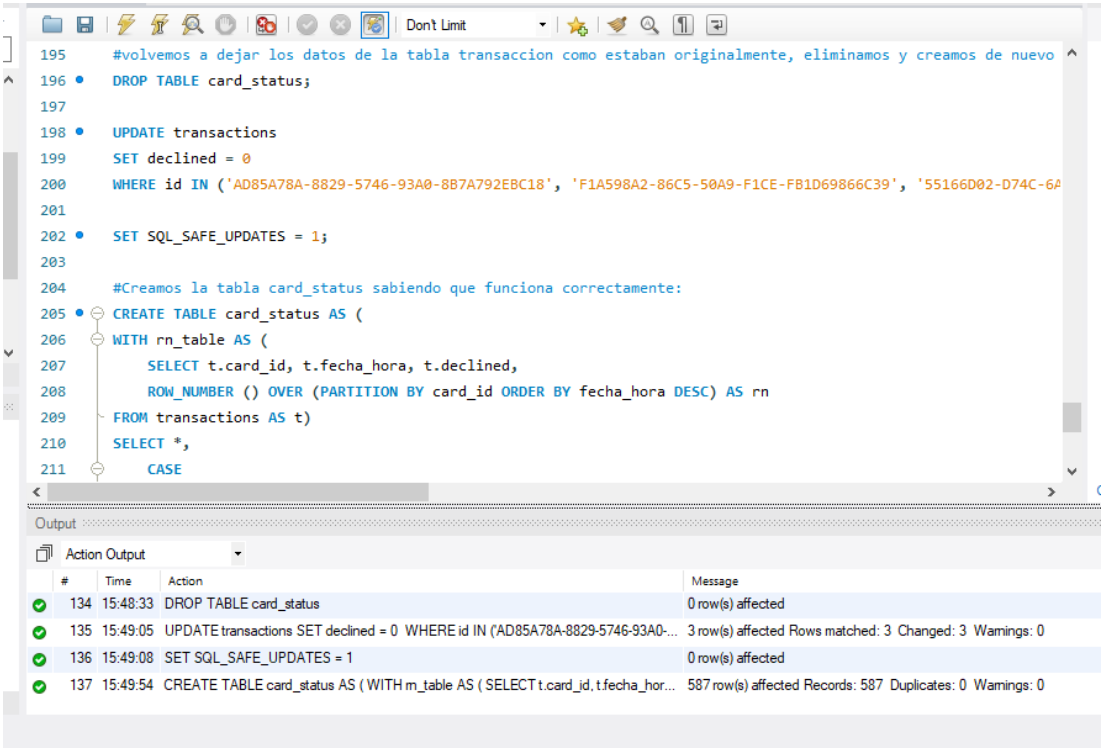
```
#----------------------------PRUEBA PARA CORROBORAR QUE CASE FUNCIONA-------------------------
#Cambiaremos a declined los datos de una tarjeta para que cumpla las condiciones de rechazo y verificar CASE
DROP TABLE card_status;

#averiguamos el id de las últimas tres transacciones de la compañía con card_id CcU-2938 para cambiarlas a dec
SELECT * FROM transactions
WHERE card_id LIKE '%2938'
ORDER BY fecha_hora DESC;

#Hacemos que las últimas tres transacciones sean declinadas
SET SQL_SAFE_UPDATES = 0;
UPDATE transactions
SET declined = 1
```

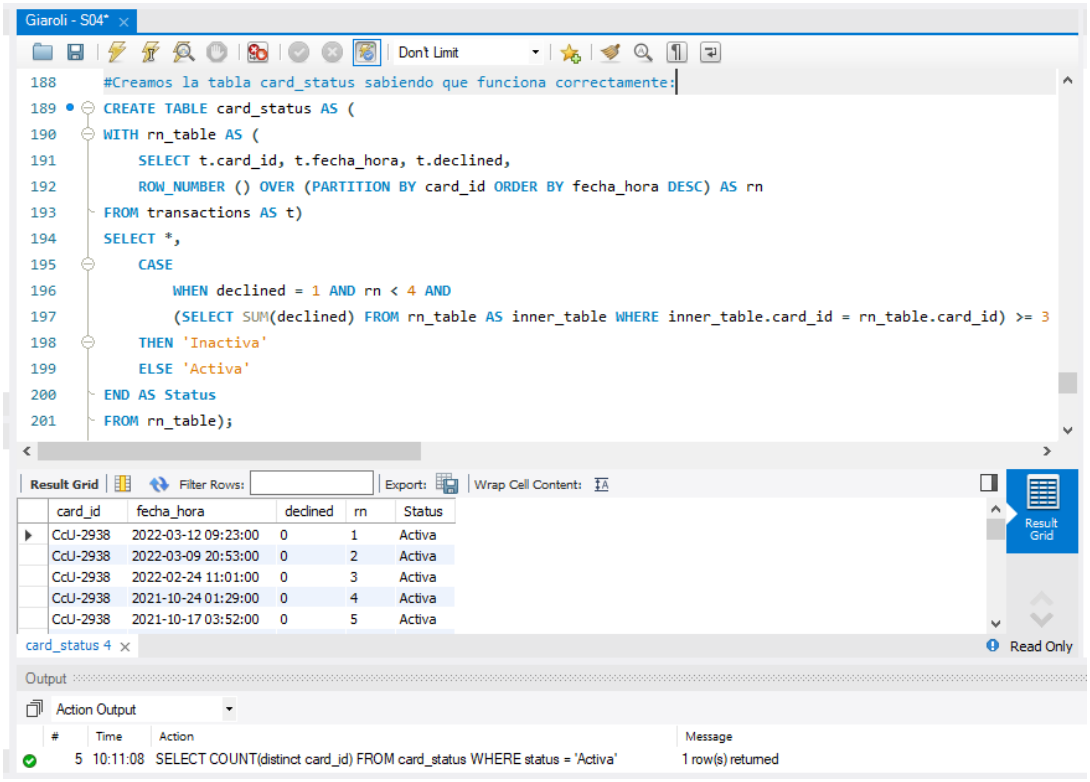
card_id	fecha_hora	declined	rn	Status
CcU-2938	2022-03-12 09:23:00	0	1	Aceptada
CcU-2938	2022-03-09 20:53:00	0	2	Aceptada
CcU-2938	2022-02-24 11:01:00	0	3	Aceptada
CcU-2938	2021-10-24 01:29:00	0	4	Aceptada
CcU-2938	2021-10-17 03:52:00	0	5	Aceptada
CcU-2938	2021-09-28 02:24:00	0	6	Aceptada
CcU-2938	2021-09-24 08:33:00	0	7	Aceptada

Comprobamos con las transacciones editadas que el código funciona:



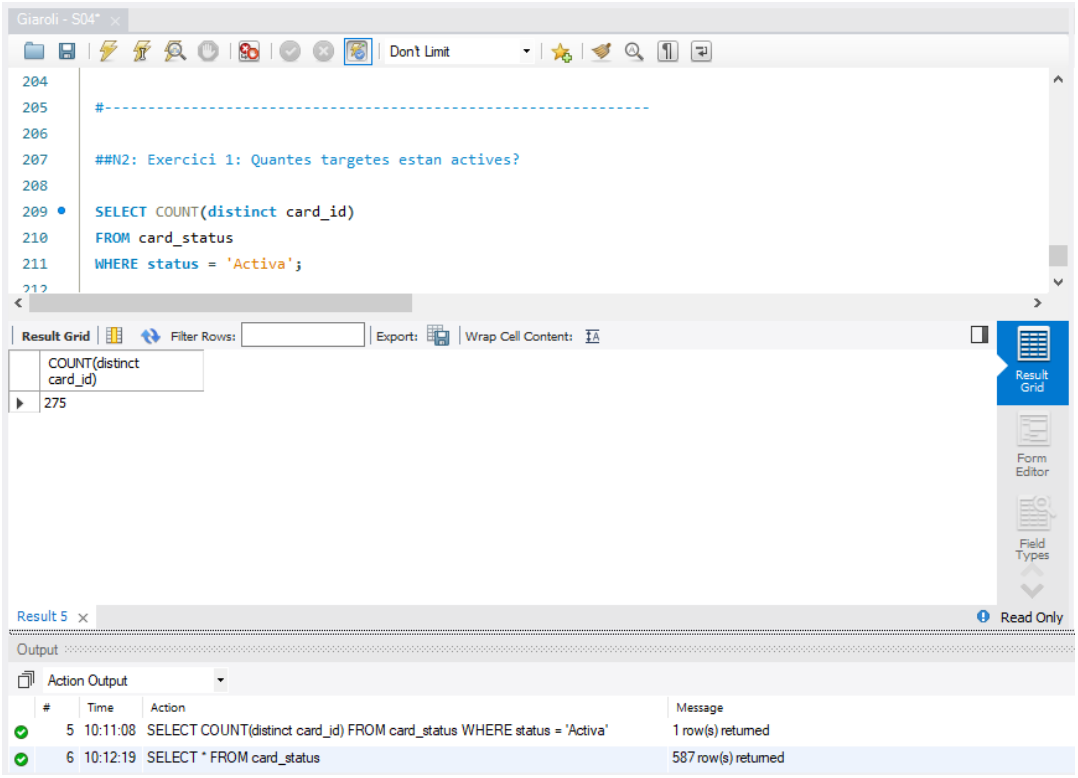
Volvemos a colocar la tabla Transactions con sus datos originales y creamos la tabla Card_status sabiendo que funciona correctamente, cambiamos la nomenclatura de case a Activa/Inactiva en lugar de Aceptada/ Rechazada:





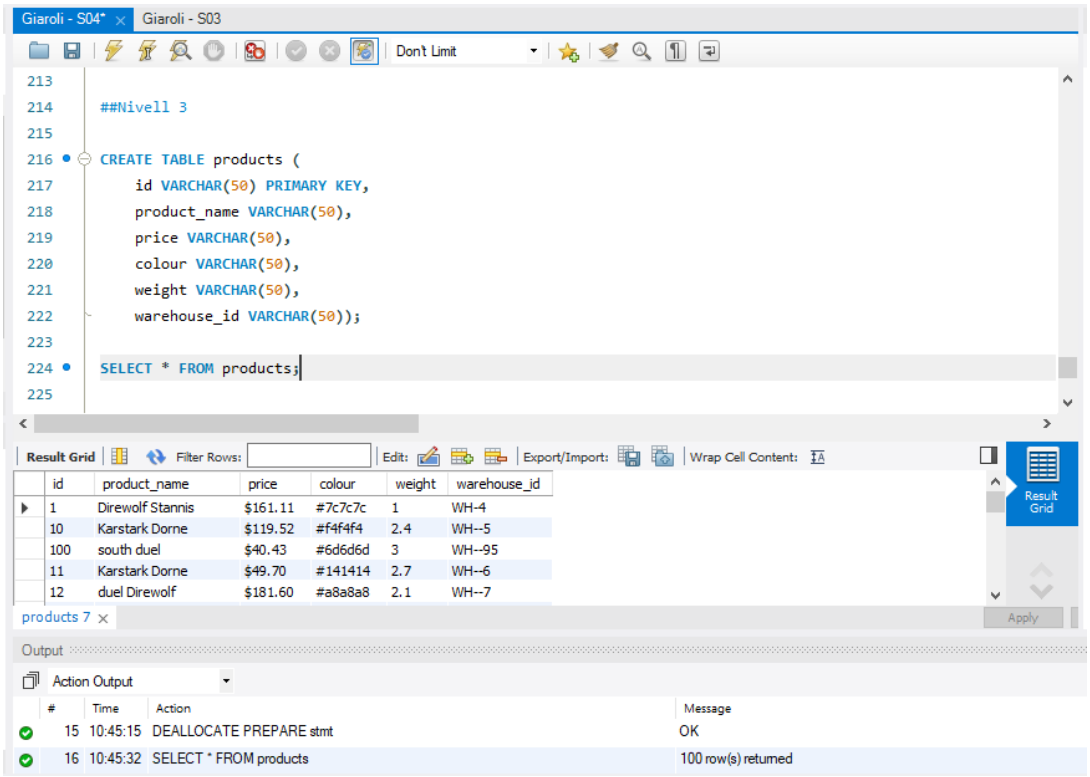
Ejercicio 1

Hay 275 tarjetas activas, es decir todas las tarjetas. Ya que de las 587 transacciones realizadas, 87 fueron declinadas, pero cada operación declinada ha sido con diferentes tarjetas. Con lo que ninguna tarjeta ha tenido las últimas tres transacciones declinadas como para considerarse rechazada.

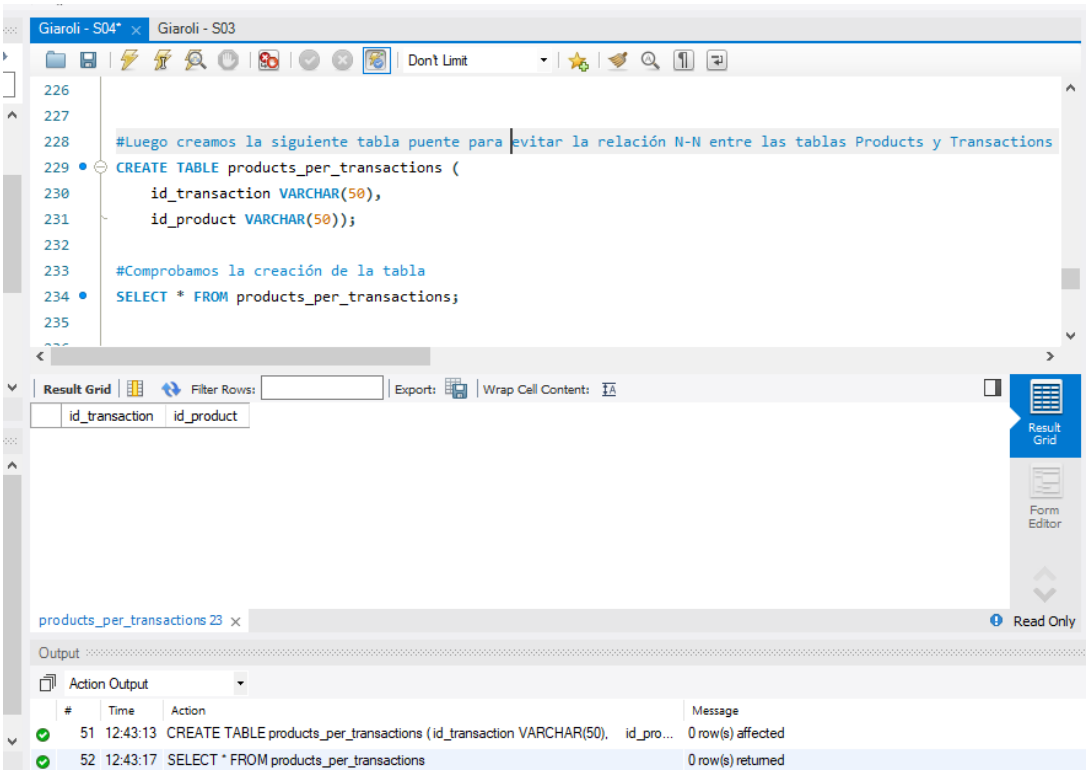


Nivel 3

Primero creamos la tabla Products e importamos los datos:



Luego creamos la tabla puente “Products_per_transactions” entre Products y Transactions para romper la relación N-N entre ambas.

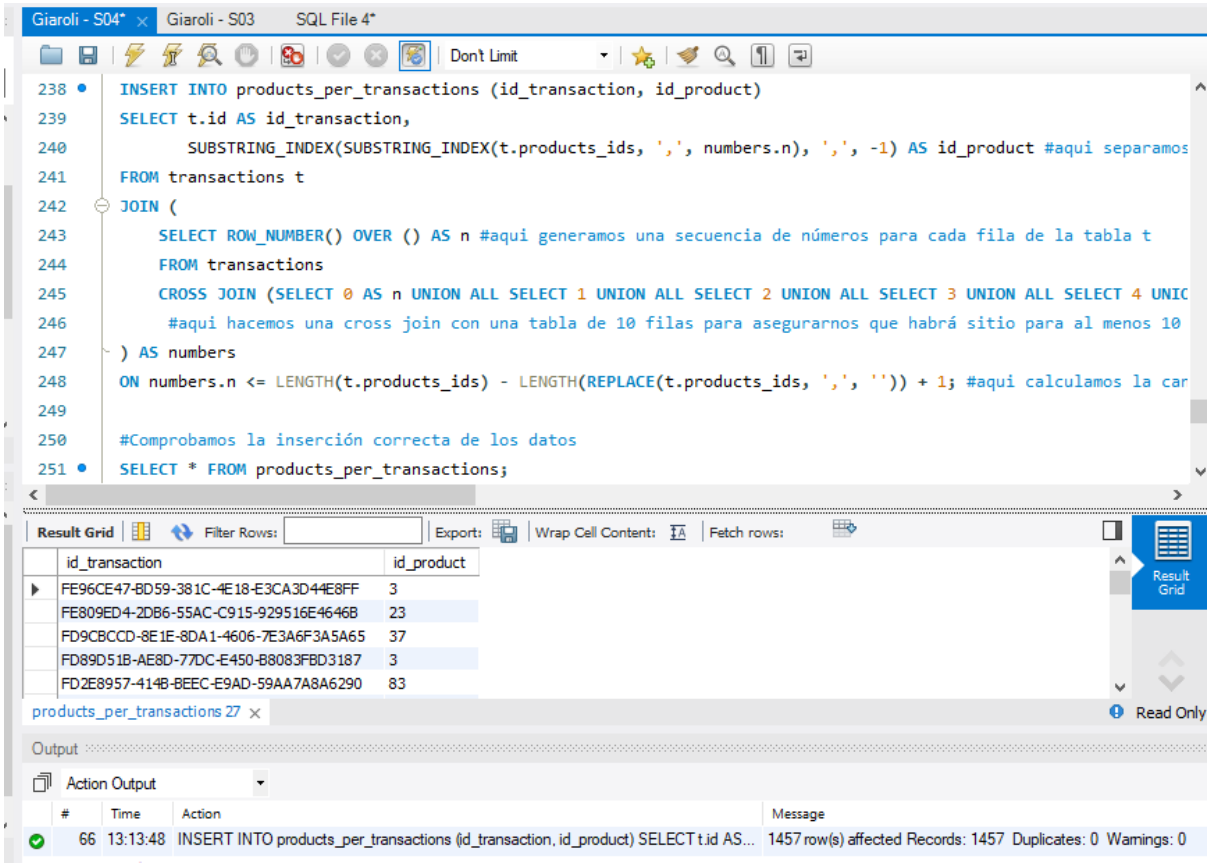


Esta tabla puente se compone de dos campos:

- uno es id_transaccion que contiene los id de cada transacción realizada y actúa como FK con la tabla Transaccions, con la que tiene una relación de N-1.
- Otro es id_product que contiene los id de cada producto y actúa como FK con la tabla Products, con la que tiene una relación de N-1.

Añadiremos las FK luego de insertar los datos en la tabla para facilitar la inserción.

A continuación, insertamos los datos en la tabla puente a partir de las columnas que ya existen en la tabla Transacions.

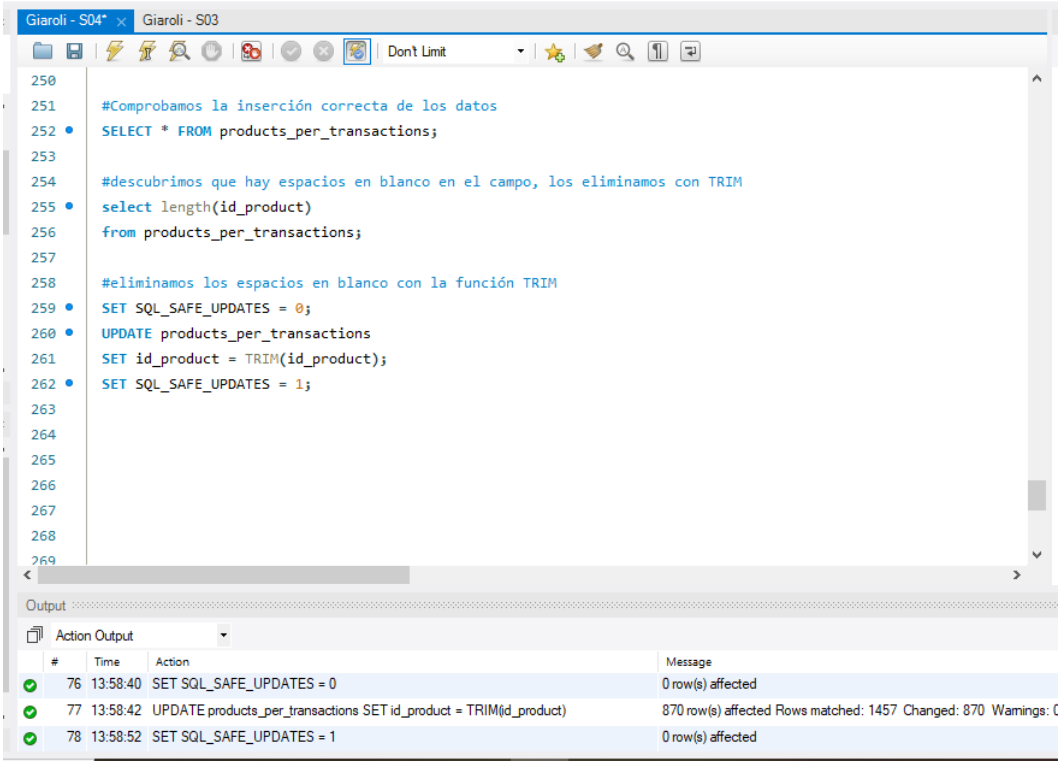


Para ello, seguiremos los siguientes pasos:

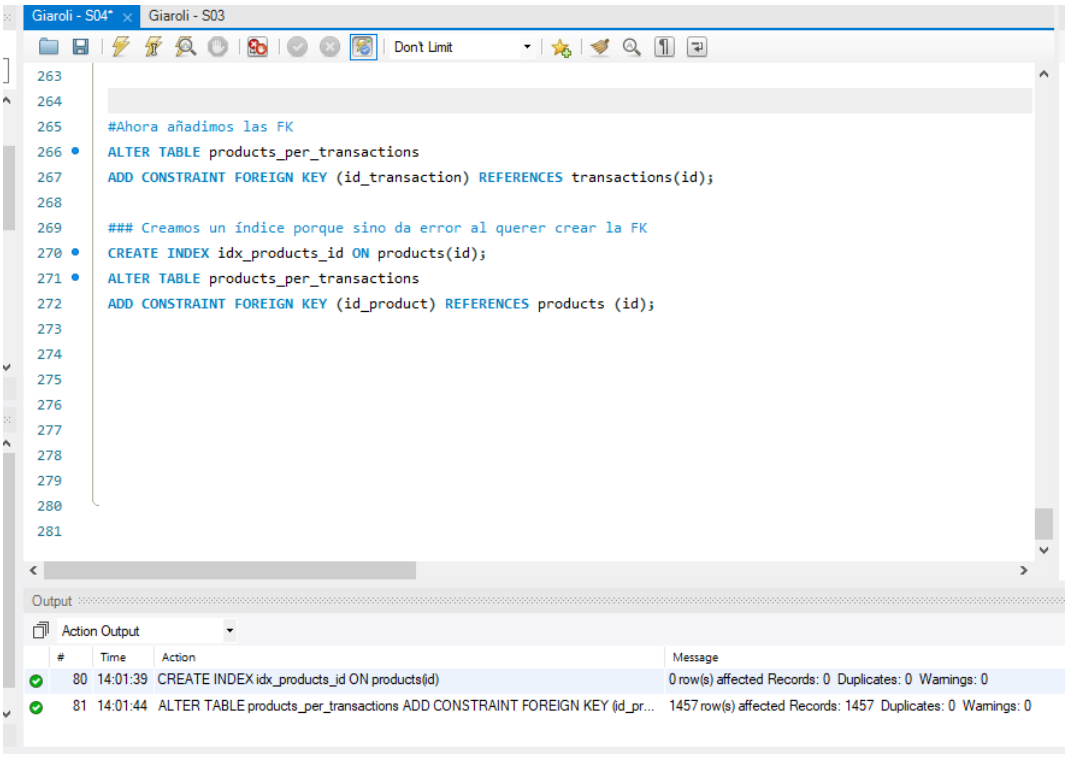
1) Separamos cada producto del campo products_ids con la función **SUBSTRING_INDEX**, la aplicamos dos veces porque sino nos quedaría el primer producto sin separar
2) Hacemos una **JOIN** con una secuencia de números generada con la función **ROW_NUMBER** para cada fila de la tabla transactions
3) Hacemos una **CROSS JOIN** (es decir un producto cartesiano) con una tabla ficticia de 10 números, para asegurar que al menos habrá 10 filas disponibles para productos por cada transacción
4) La JOIN la hacemos teniendo en cuenta el cálculo de la cantidad de productos que hay en cada campo, para ello restamos la función **LENGTH**

la longitud de la cadena con comas menos la sin comas, para saber la cantidad de comas que hay y le sumamos uno.

Al comprobar la creación de la tabla encontramos que el campo id_product tiene espacios en blanco, los quitamos con la función TRIM:



Ahora creamos las FK, añadimos un índice en la tabla products porque sino no nos deja crearla:

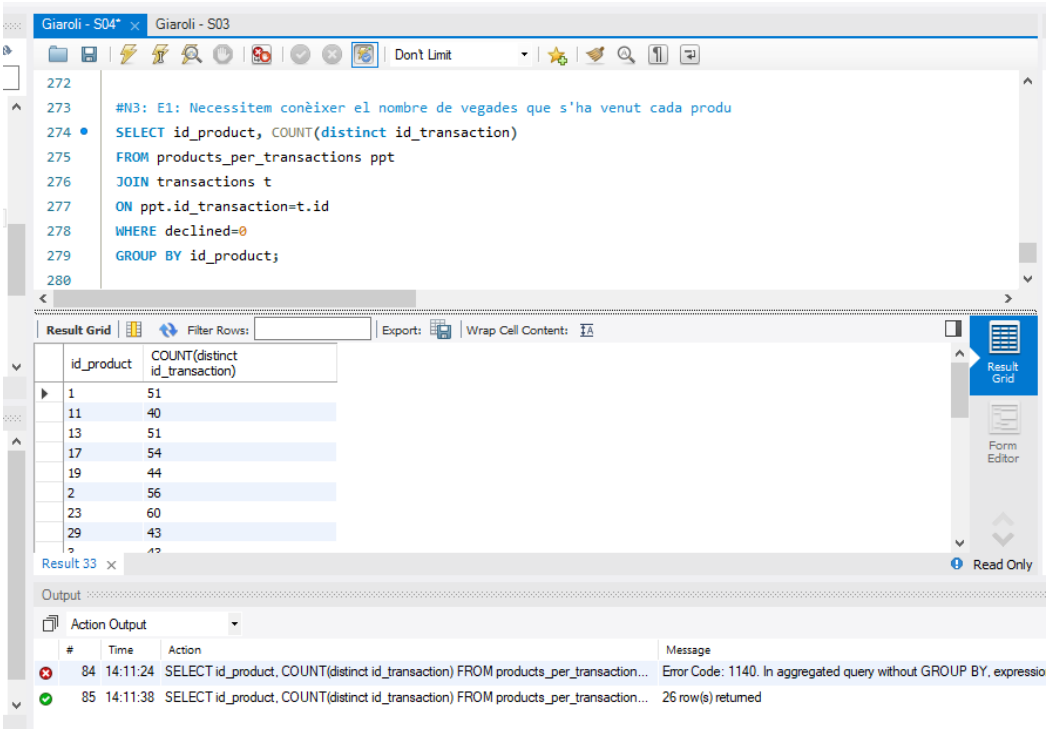


Ejercicio 1

Para obtener un listado de la cantidad de veces que se ha vendido cada producto usamos la siguiente consulta:

Como el enunciado especifica productos “vendidos” tendremos en cuenta que la transaccion no haya sido declinada (declined = 0).

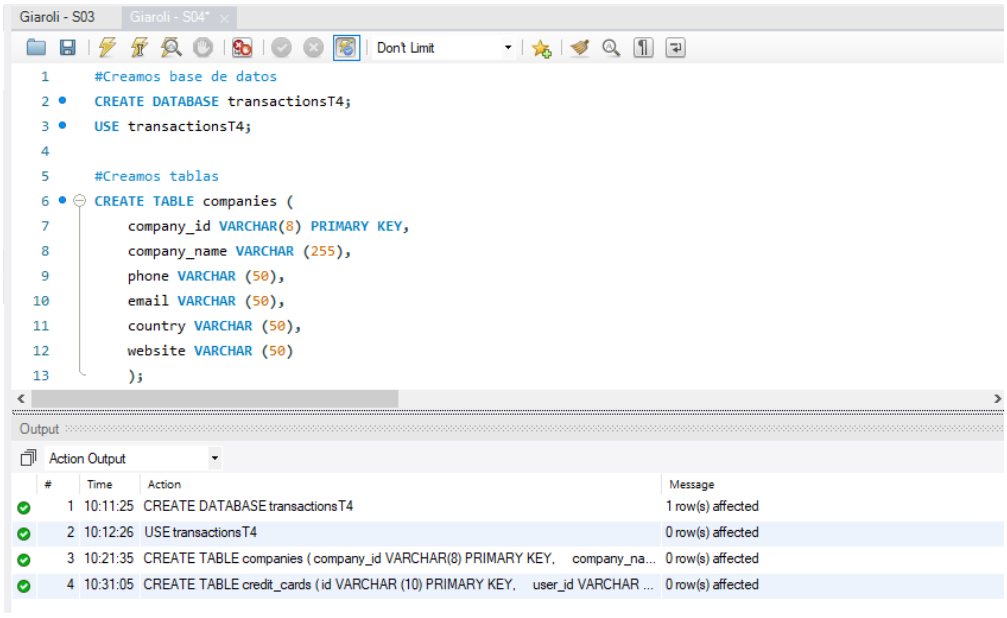
Por ejemplo en el primer renglón podemos ver que el producto con id = 1 se ha vendido 51 veces.



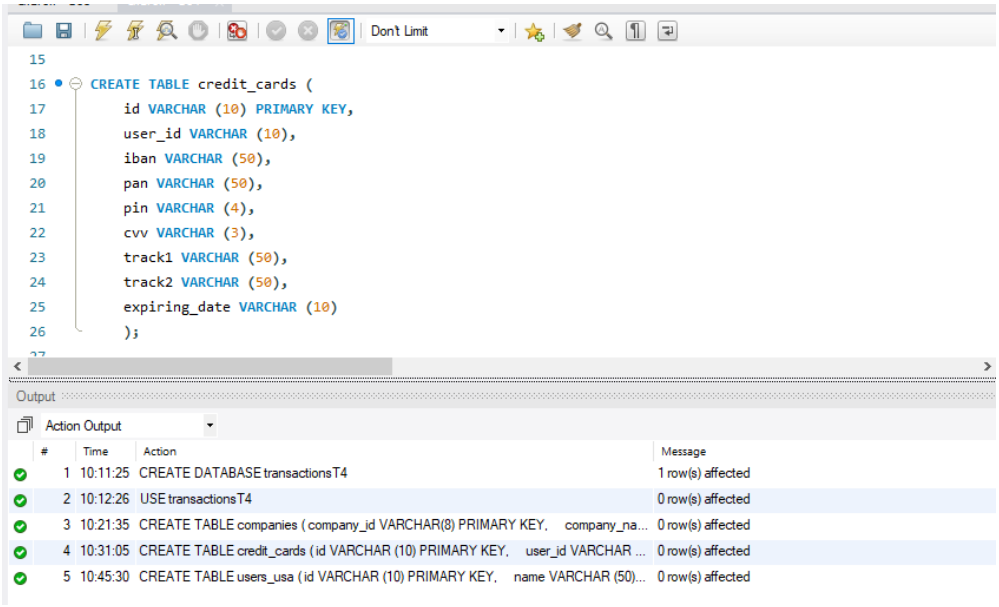
ANEXO

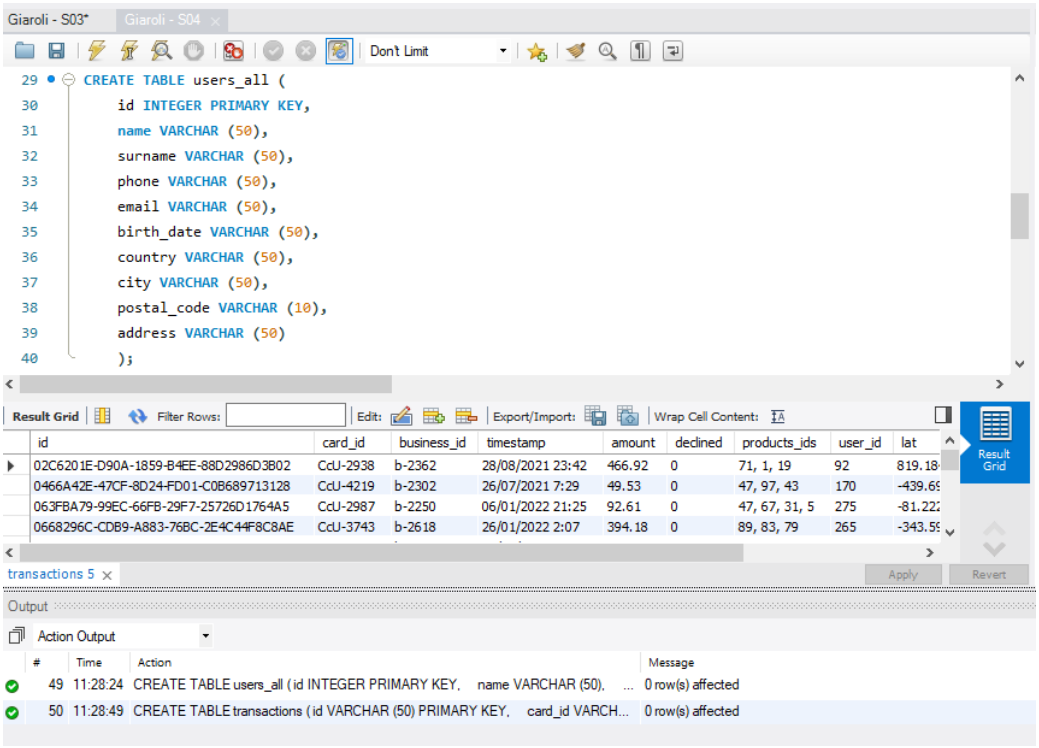
Para crear la base de datos y el modelo realizamos los siguientes comandos:

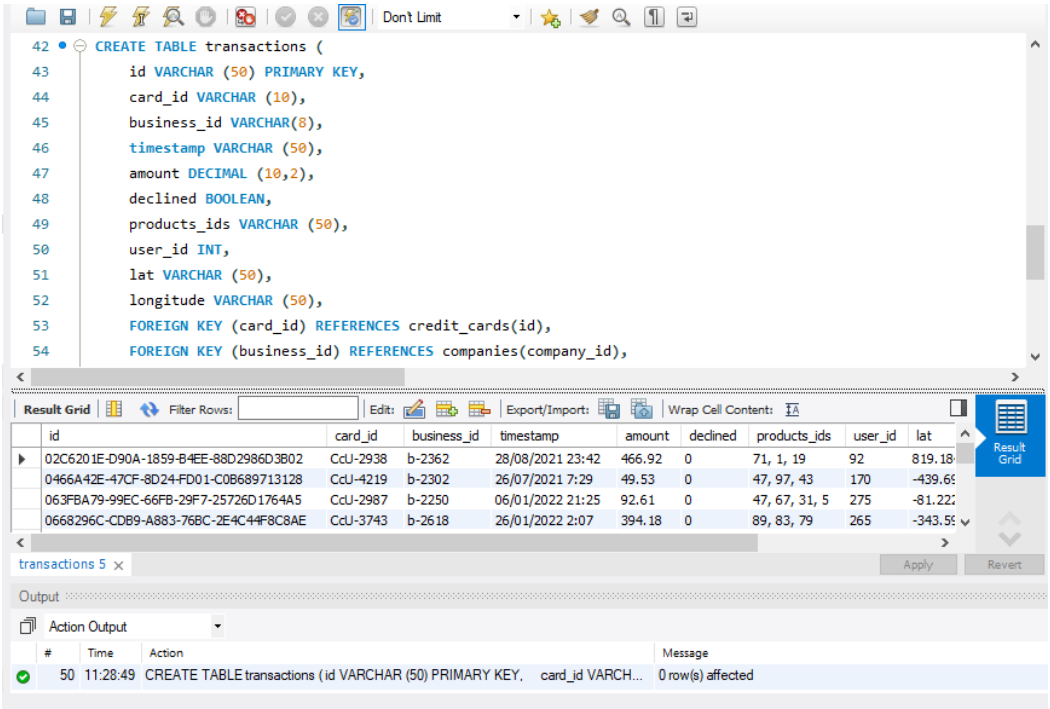
1) Creamos la Base de datos y la tabla companies



2) Creamos las tablas credit_Cards, users_all y transactions







3) **Insertamos los datos en cada tabla** usando la función "Table Date Import Wizard".

4) Y **verificamos la inserción** de los datos con el comando SELECT *