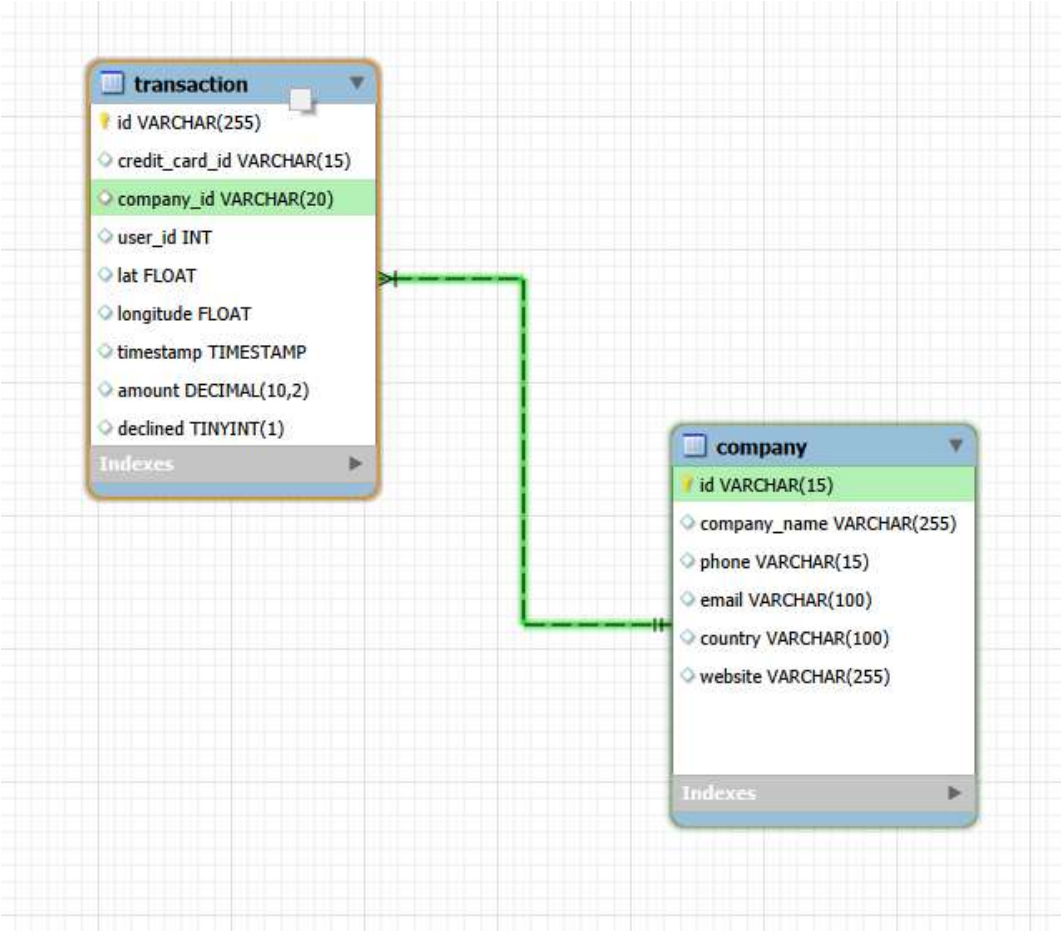
NIVEL 1

Ejercicio 1



La BBDD está compuesta por dos tablas principales: **company** y **transaction**, relacionadas entre sí mediante una clave foránea. El tipo de relación es de 1:N (uno a muchos). Se incluyen distintos tipos de datos: VARCHAR, INT, FLOAT, TIMESTAMP, DECIMAL, TINYINT.

La tabla **company** tiene como PRIMARY KEY el identificador **id VARCHAR(15)** que es único para cada empresa.

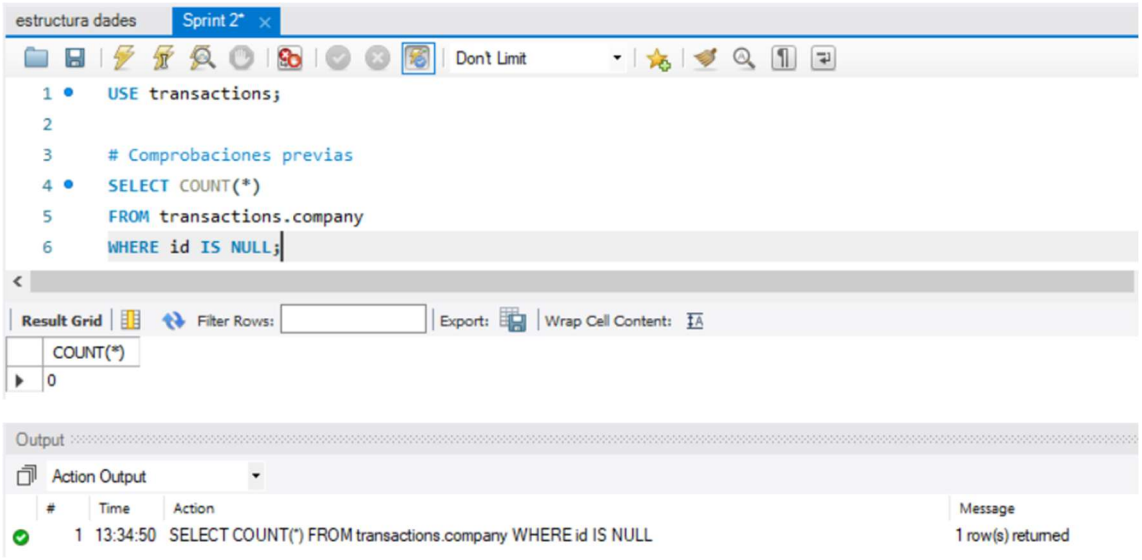
Por otro lado, la tabla **transaction** tiene como PRIMARY KEY **id VARCHAR(255)** que corresponde al identificador único de la transacción y como FOREIGN KEY **company_id VARCHAR(20)** que hace referencia a **company.id**.

Ejercicio 2 – Utilizando sólo JOIN

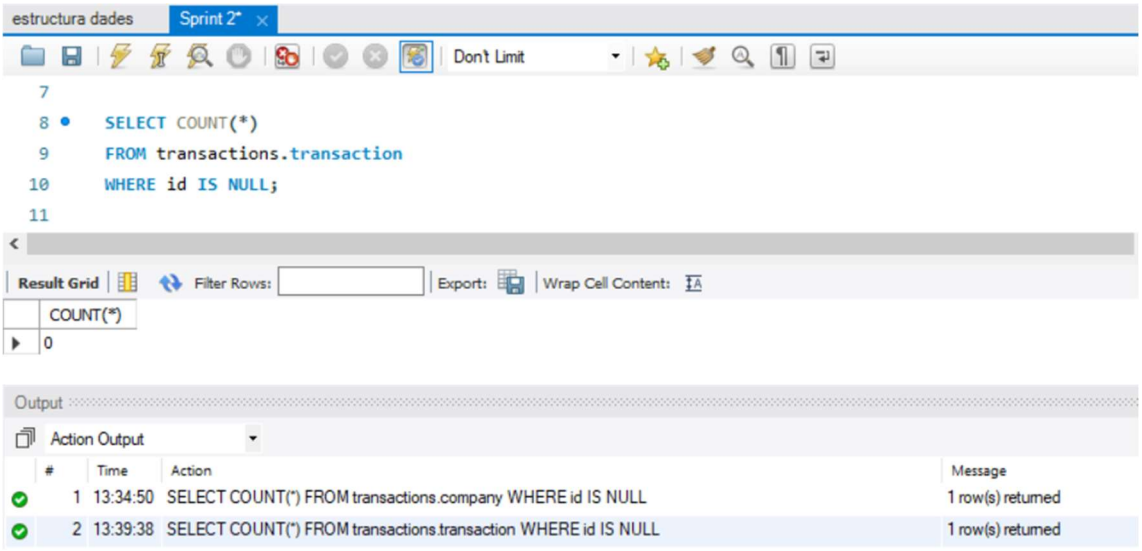
Para comprobar que la base de datos funciona correctamente, se realizan las siguientes acciones:

Comprobaciones previas

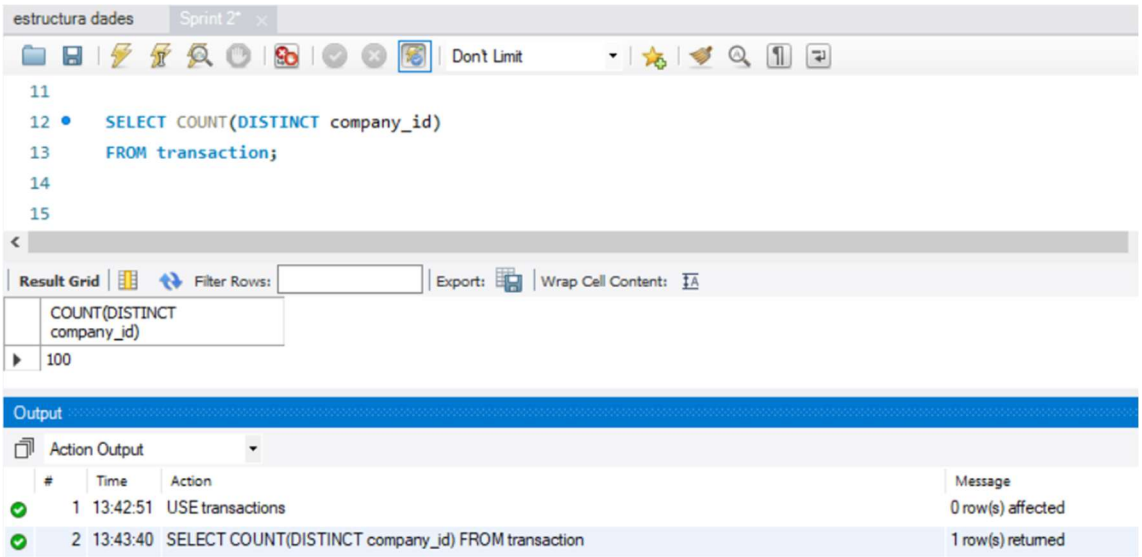
- Comprobación de valores NULL de la tabla company:

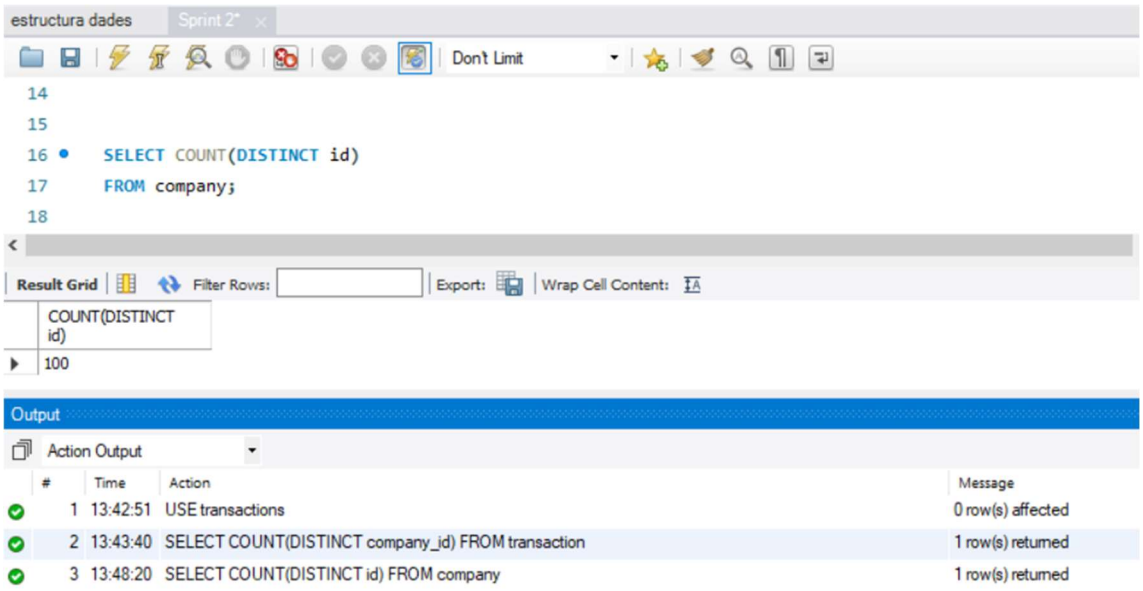


- Comprobación de valores NULL de la tabla transaction



- Comprobación de valores diferentes, misma cantidad de empresas en ambas tablas

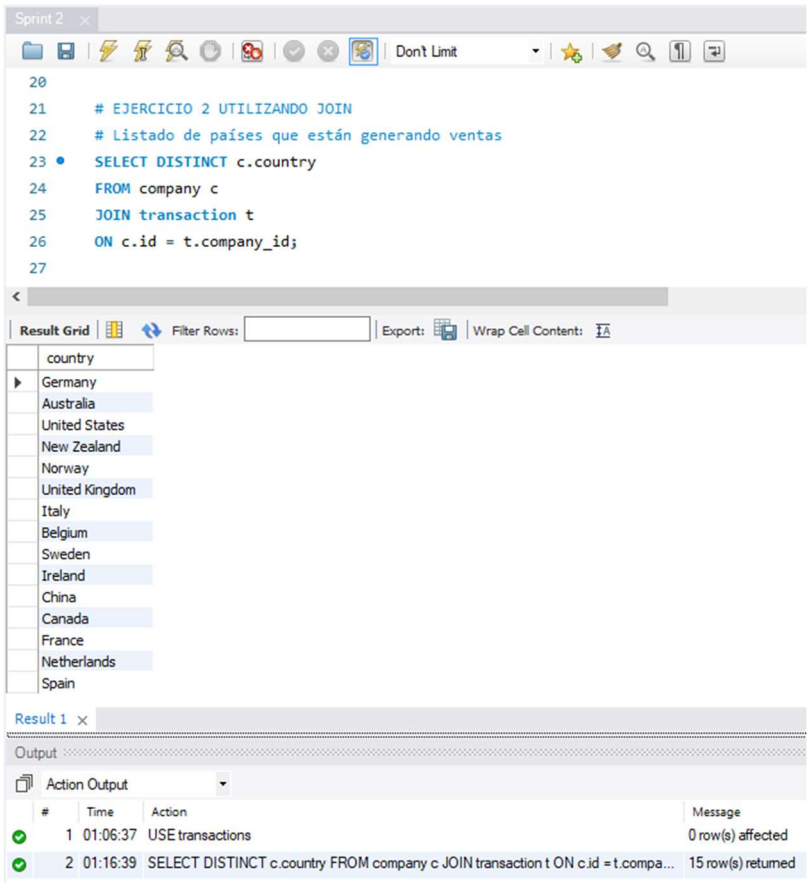




Como se puede observar, hay la misma cantidad de empresas (valores diferentes) tanto en la tabla company como en la tabla transaction.

Ejercicio 2.1.

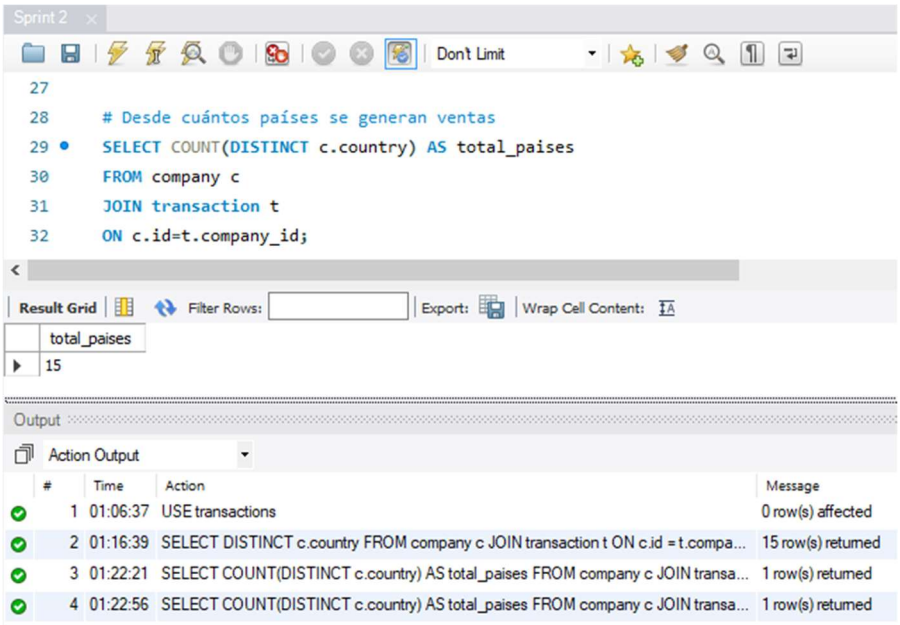
- Listado de los países que están generando ventas:



Aquí se usa el SELECT DISTINCT para obtener los países únicos que tienen al menos una transacción registrada. Para cruzar los datos y obtener sólo los que tienen coincidencias en ambas tablas usamos JOIN (INNER JOIN).

Ejercicio 2.2.

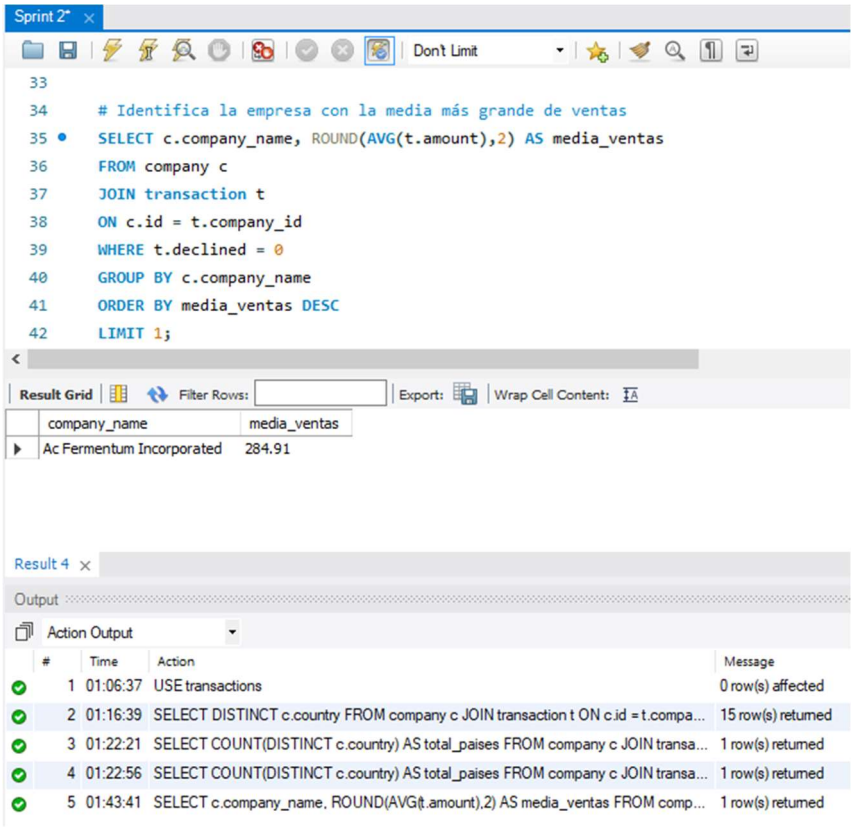
- Desde cuántos países se generan las ventas



En esta consulta se pide un número entero por este motivo usamos COUNT DISTINCT de la columna **country** de la tabla company y se le otorga un alias con AS total_paises.

Ejercicio 2.3.

- Identifica la empresa con la media más grande de ventas

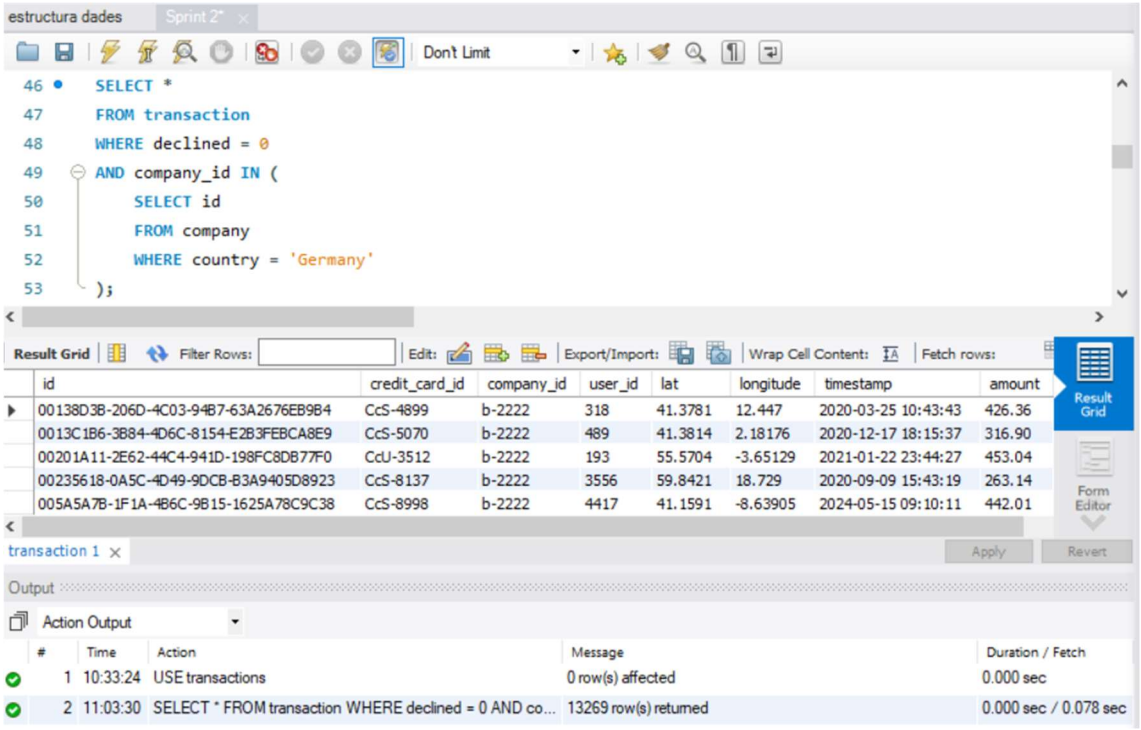


En esta consulta se introduce la función AVG para obtener la media o promedio de la columna **amount**, se redondea a dos decimales con **ROUND**, se agrupa por nombre de la empresa, se ordena de forma descendente y se limita al primer registro mediante **LIMIT 1**. Además, se usa la expresión **declined = 0** (0 representa FALSE mientras que 1 representa TRUE) para eliminar las transacciones que hayan sido rechazadas.

Ejercicio 3 – Utilizando sólo subconsultas

Ejercicio 3.1.

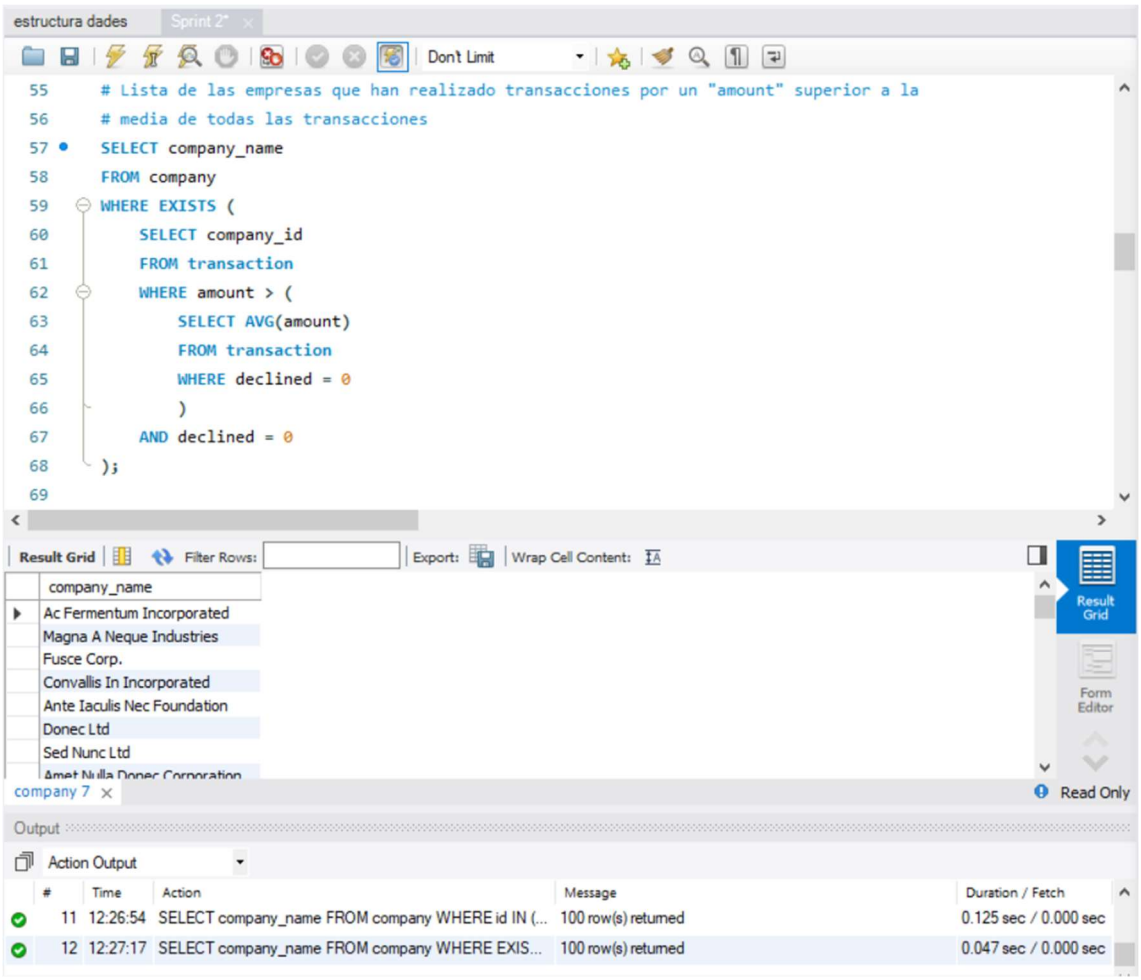
- Muestra todas las transacciones realizadas por empresas de Alemania



En esta consulta primero realizamos una subconsulta para delimitar el país 'Germany'. Una vez acotado el país, la introducimos en la OUTER QUERY mediante la FOREIGN KEY **company_id** en la tabla **transaction** que la vincula a la PRIMARY KEY **id** en la tabla **company**.

Ejercicio 3.2.

- Lista de las empresas que han realizado transacciones por un amount superior a la media de todas las transacciones



En este ejercicio tenemos 3 capas o niveles de consultas. QUERY-SUBQUERY NIVEL 2-SUBQUERY 1, es decir, hay una SUBQUERY dentro de otra SUBQUERY que a su vez está dentro de una MAIN QUERY (OUTER QUERY).

En la primera SUBQUERY se busca el promedio **(AVG)** del amount de todas las transacciones excluyendo las transacciones rechazadas (**declined = 0**).

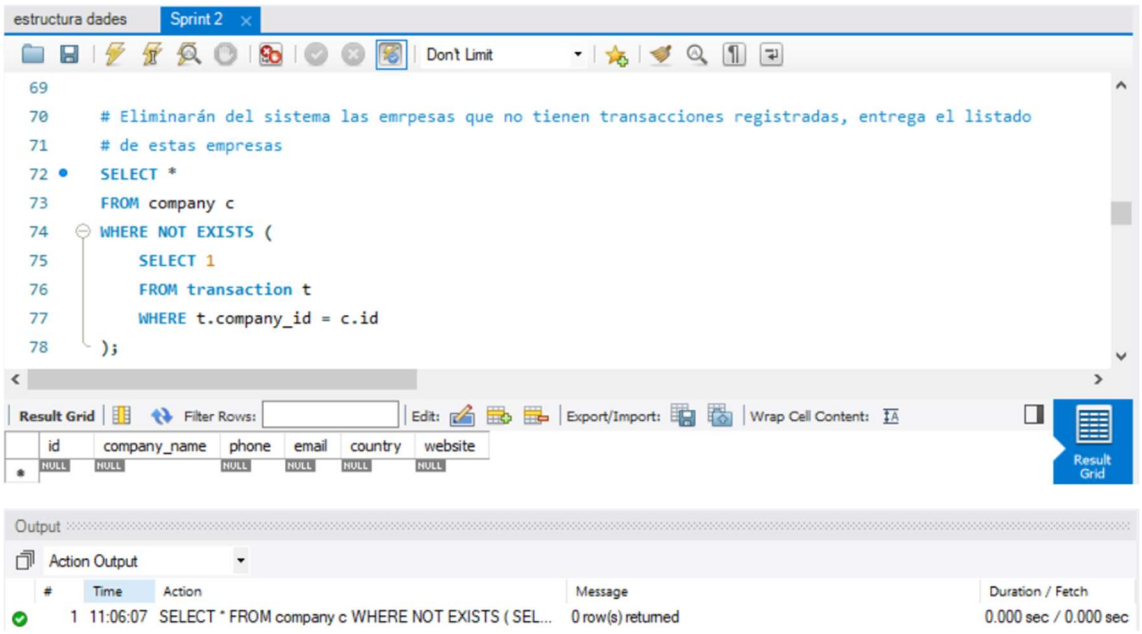
A continuación, introducimos esta información a la SUBQUERY superior mediante la cláusula **WHERE** para encontrar las empresas que cumplan dicha condición y también excluimos las transacciones rechazadas (**declined = 0**), este paso se repite porque ahora sí que sirve para descartar las transacciones rechazadas mientras que en la SUBQUERY que busca el **AVG** se usa para no desvirtuar el resultado final del promedio.

Por último, buscamos las empresas que cumple las condiciones establecidas en las SUBQUERIES.

Ejercicio 3.3.

- Eliminarán del sistema las empresas que no tienen transacciones registradas, entrega el listado de estas empresas.

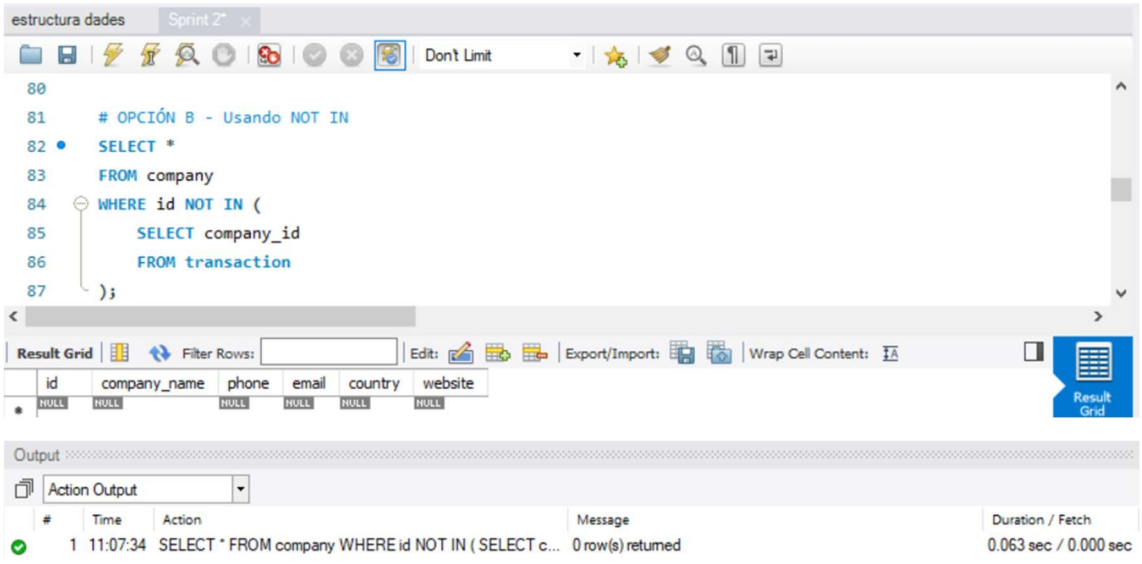
Opción A – Subconsulta correlacionada



En este ejercicio buscaremos, mediante una SUBQUERY CORRELACIONADA, las empresas de la tabla **company** que no figuren (NOT EXISTS) en la tabla **transaction**. Con SELECT usamos 1 porque no nos interesa recuperar todas las columnas y filas de la tabla especificada, sólo saber si existen registros devolviendo 1 para cada fila que coincida.

En esta última consulta no incluimos declined = 0 porque el objetivo es encontrar empresas que no tiene ningún registro en la tabla **transaction**, independientemente de si fueron rechazadas o no.

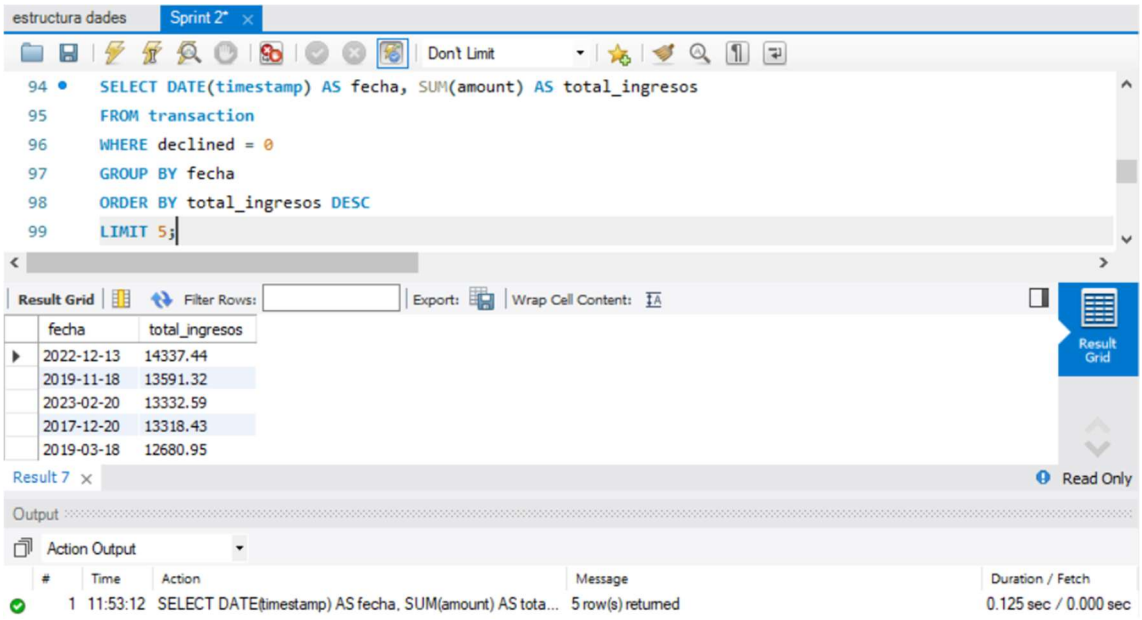
Opción B – Usando NOT IN



NIVEL 2

Ejercicio 1

Identifica los cinco días que se generaron la cantidad más grande de ingresos a la empresa por ventas. Muestra la fecha de cada transacción junto con el total de las ventas.

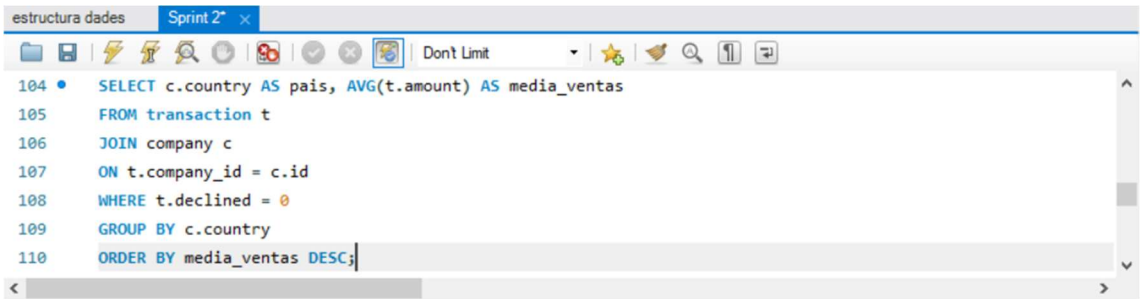


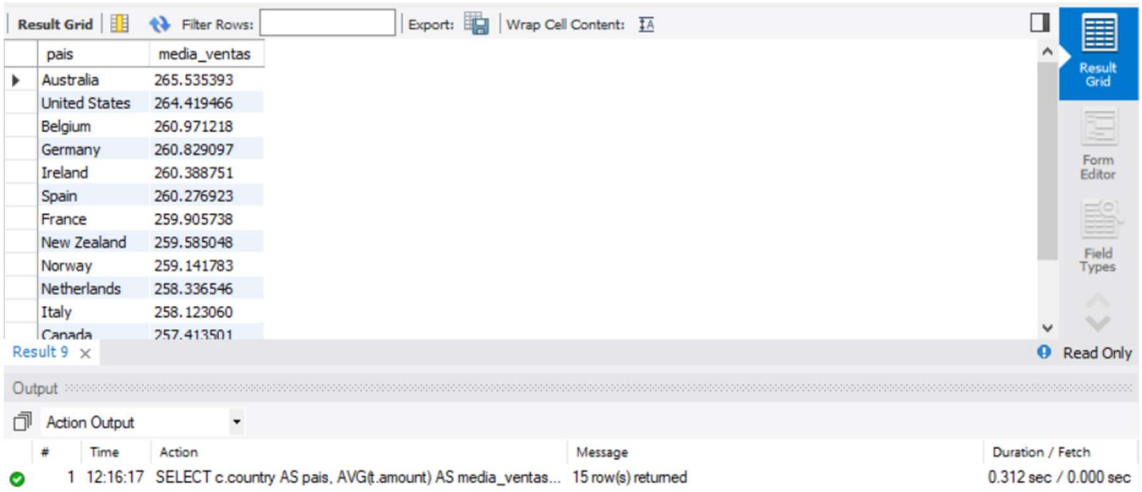
Para esta consulta, debemos extraer la fecha de la columna **timestamp**, sumar el **amount** de todas las transacciones que no fueron rechazadas (declined = 0) y agrupar por día.

Se usa **DATE()** para ignorar la hora del **timestamp**. Filtramos con **WHERE** declined = 0 para asegurar que sólo contamos ventas exitosas. Finalmente, ordenamos de mayor a menor y limitamos a los 5 primeros resultados.

Ejercicio 2

¿Cuál es la media de ventas por país? Presenta los resultados ordenados de mayor a menor media.





pais	media_ventas
Australia	265.535393
United States	264.419466
Belgium	260.971218
Germany	260.829097
Ireland	260.388751
Spain	260.276923
France	259.905738
New Zealand	259.585048
Norway	259.141783
Netherlands	258.336546
Italy	258.123060
Canada	257.413501

Result 9

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:16:17	SELECT c.country AS pais, AVG(t.amount) AS media_ventas...	15 row(s) returned	0.312 sec / 0.000 sec

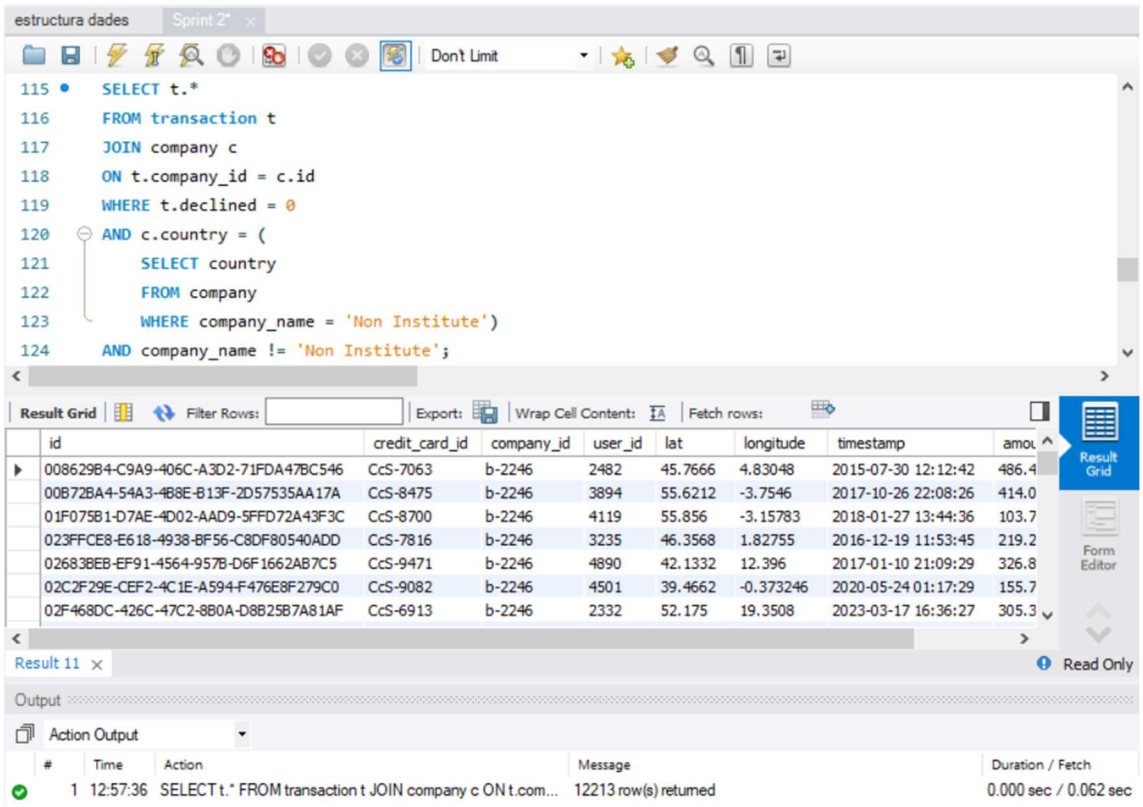
Para responder a esta consulta, es necesario realizar un **JOIN** entre ambas tablas, ya que la información del país está en la tabla **company** y el monto de la venta en **transaction**.

Se unen las tablas a través de **company_id** de la tabla **transaction** e id de la tabla **company**. La función **AVG()** calcula el promedio de los montos y el **ORDER BY ... DESC** asegura que los países con mayor media aparezcan primero.

Ejercicio 3

En tu empresa, se plantea un nuevo proyecto para lanzar algunas campañas publicitarias para hacer competencia a la empresa “Non Institute”. Para esto, te piden una lista de todas las transacciones realizadas por empresas que están situadas en el mismo país que esta empresa.

- Muestra el listado aplicando JOIN y subconsultas



estructura dades — Sprint 2*

```
115 • SELECT t.*
116   FROM transaction t
117   JOIN company c
118   ON t.company_id = c.id
119   WHERE t.declined = 0
120   AND c.country = (
121       SELECT country
122       FROM company
123       WHERE company_name = 'Non Institute')
124   AND company_name != 'Non Institute';
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amou
008629B4-C9A9-406C-A3D2-71FDA47BC546	CcS-7063	b-2246	2482	45.7666	4.83048	2015-07-30 12:12:42	486.4
00B72BA4-54A3-4B8E-B13F-2D57535AA17A	CcS-8475	b-2246	3894	55.6212	-3.7546	2017-10-26 22:08:26	414.0
01F075B1-D7AE-4D02-AAD9-5FFD72A43F3C	CcS-8700	b-2246	4119	55.856	-3.15783	2018-01-27 13:44:36	103.7
023FFCE8-E618-4938-BF56-C8DF80540ADD	CcS-7816	b-2246	3235	46.3568	1.82755	2016-12-19 11:53:45	219.2
02683BEB-EF91-4564-957B-D6F1662AB7C5	CcS-9471	b-2246	4890	42.1332	12.396	2017-01-10 21:09:29	326.8
02C2F29E-CEF2-4C1E-A594-F476E8F279C0	CcS-9082	b-2246	4501	39.4662	-0.373246	2020-05-24 01:17:29	155.7
02F468DC-426C-47C2-8B0A-D8B25B7A81AF	CcS-6913	b-2246	2332	52.175	19.3508	2023-03-17 16:36:27	305.3

Result 11

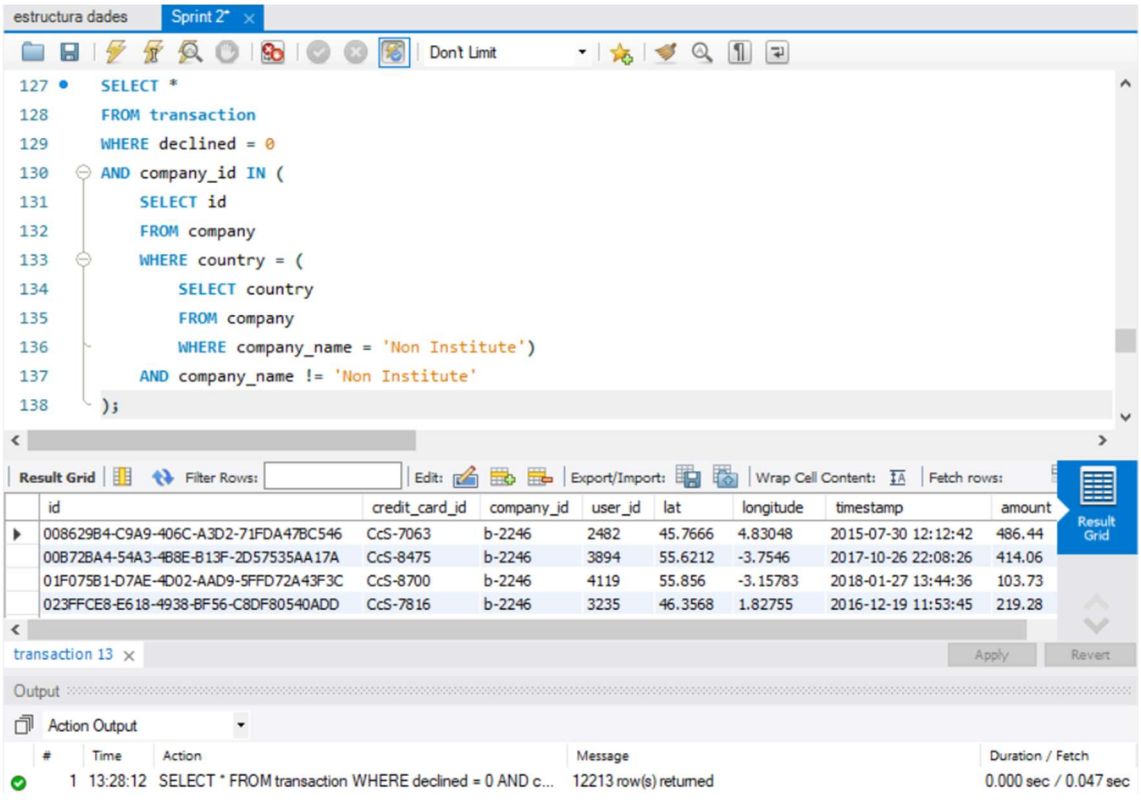
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:57:36	SELECT t.* FROM transaction t JOIN company c ON t.com...	12213 row(s) returned	0.000 sec / 0.062 sec

En este caso, la subconsulta obtiene el país de “Non Institute”, y el **JOIN** permite filtrar tanto por el país de la empresa como por el estado de la transacción. Como interesa analizar las otras empresas del país, es decir, la competencia local, se añade la condición **c.company_name != ‘Non Institute’**.

- Muestra el listado aplicando sólo subconsultas



Aquí se filtra directamente en la tabla **transaction**. La subconsulta anidada identifica primero el país objetivo y luego recupera todos los IDs de empresas de dicho país para filtrar la tabla principal

NIVEL 3

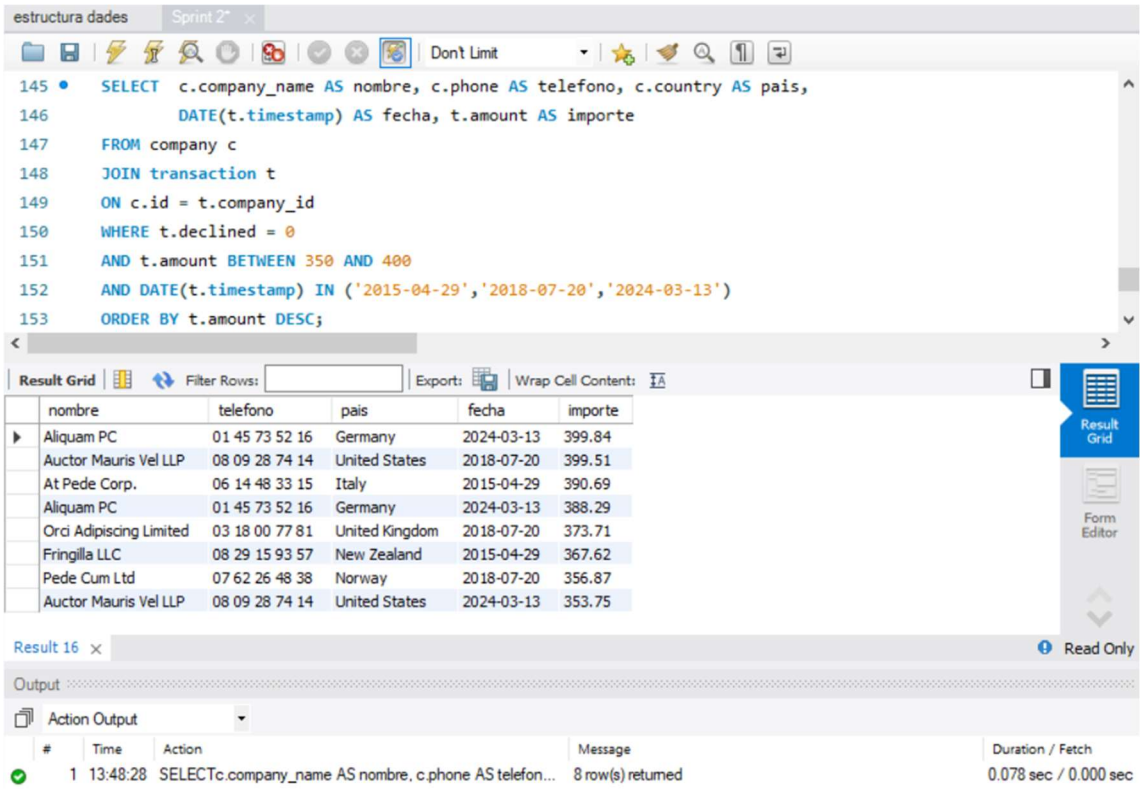
Ejercicio 1

Presenta el nombre, teléfono, país, fecha y amount, de aquellas empresas que realizaron transacciones con un valor comprendido entre 350 y 400 euros y en alguna de estas fechas: 29 de abril del 2015, 20 de julio del 2018 y 13 de marzo del 2024. Ordena los resultados de mayor a menor cantidad.

Se ha utilizado la función **DATE(timestamp)** para asegurar que la comparación se haga sólo por el día, ignorando la hora, y el operador **IN** para agrupar las tres fechas específicas.

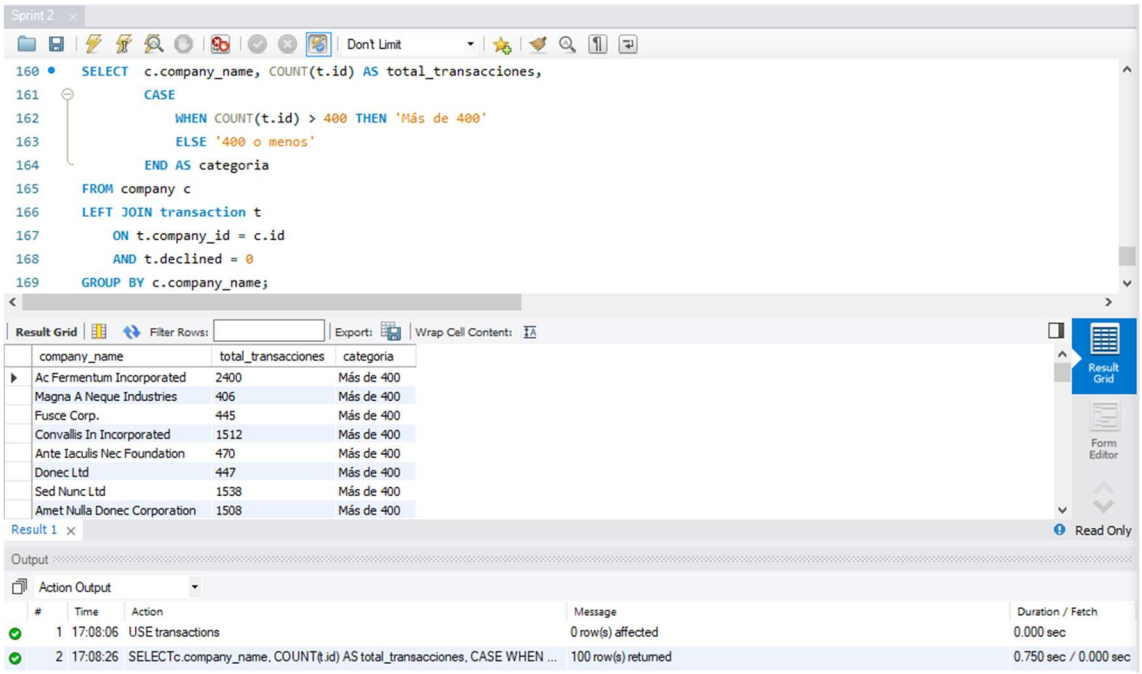
El rango de importes se ha acotado usando el operador **BETWEEN 350 AND 400** incluyendo ambos límites. Como se ha visto anteriormente, se añade la condición **declined = 0** para mostrar sólo las transacciones exitosas.

Por último, se utiliza **ORDER BY t.amount DESC** para cumplir con el requisito de mayor a menor cantidad.



Ejercicio 2

Necesitamos optimizar la asignación de los recursos y dependerá de la capacidad operativa que se requiera, por lo cual te piden información sobre la cantidad de transacciones que realizan las empresas, pero el departamento de recursos humanos es exigente y quiere un listado de las empresas donde especifiques si tienen más de 400 transacciones o menos.



Para esta consulta se usa **LEFT JOIN** porque queremos que aparezcan todas las empresas, incluso las que tienen 0 transacciones. Si se usara **INNER JOIN**, las empresas sin transacciones desaparecerían.

Con **COUNT(t.id)** cuenta las transacciones por empresa. A continuación, se aplica **CASE** que clasifica según la cantidad. Se asemeja a una función de Python. Dentro del **LEFT JOIN** se incluye la condición **declined = 0** para filtrar las transacciones rechazadas.

Por último, se agrupa por **c.company_name**.