

SPRINT 1 – NIVEL 1

SPRINT 1 - EJERCICIO – 1 - Estructura de las Tablas

Database: transactions

#La base de datos transactions se compone de dos tablas: company y transaction

DESCRIBE company;

clave primaria de company es el campo id, que no puede ser en blanco (not null)

la tabla tiene un total de 6 campos, todos ellos de tipo varchar;

SELECT COUNT(*) FROM company;

#la tabla tiene un total de 100 filas

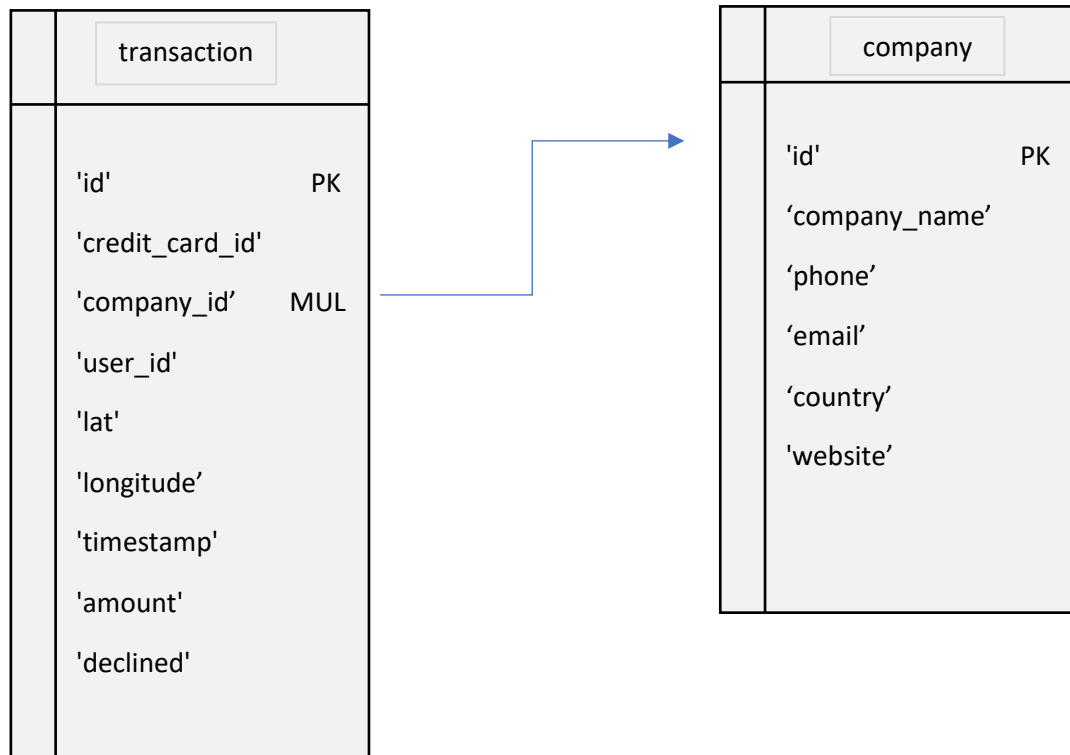
DESCRIBE transaction;

clave primaria de company es el campo id, que no puede ser en blanco (not null)

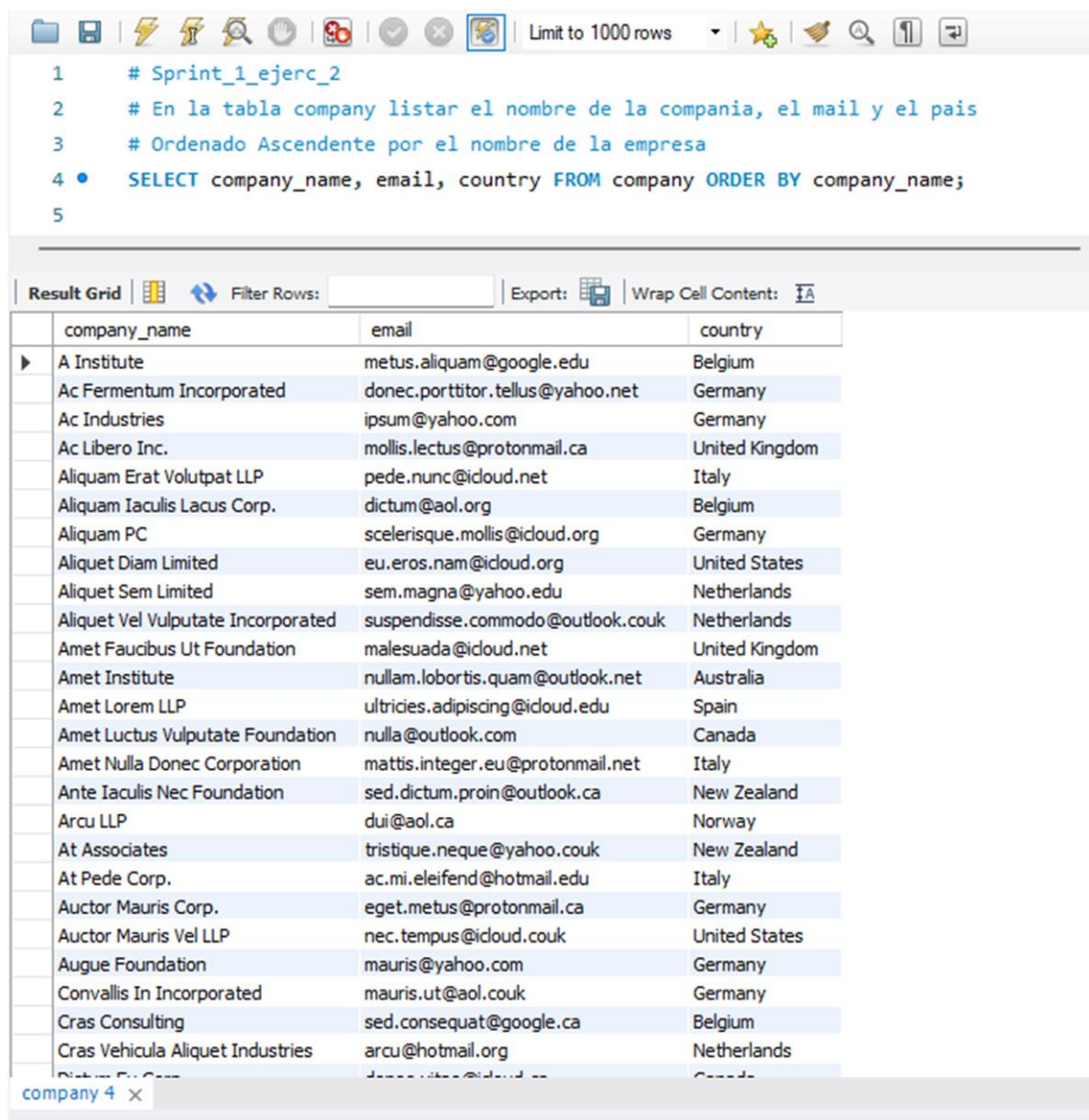
la tabla tiene un total de 9 campos, la tipologia es varchar, int, float, timestamp, decimal, y tinyint

SELECT COUNT(*) FROM transaction;

#la tabla tiene un total de 587 filas



SPRINT_1 - EJERCICIO – 2 – Tabla Compañías ordenada por nombre



Limit to 1000 rows

```
1 # Sprint_1_ejerc_2
2 # En la tabla company listar el nombre de la compania, el mail y el pais
3 # Ordenado Ascendente por el nombre de la empresa
4 • SELECT company_name, email, country FROM company ORDER BY company_name;
5
```

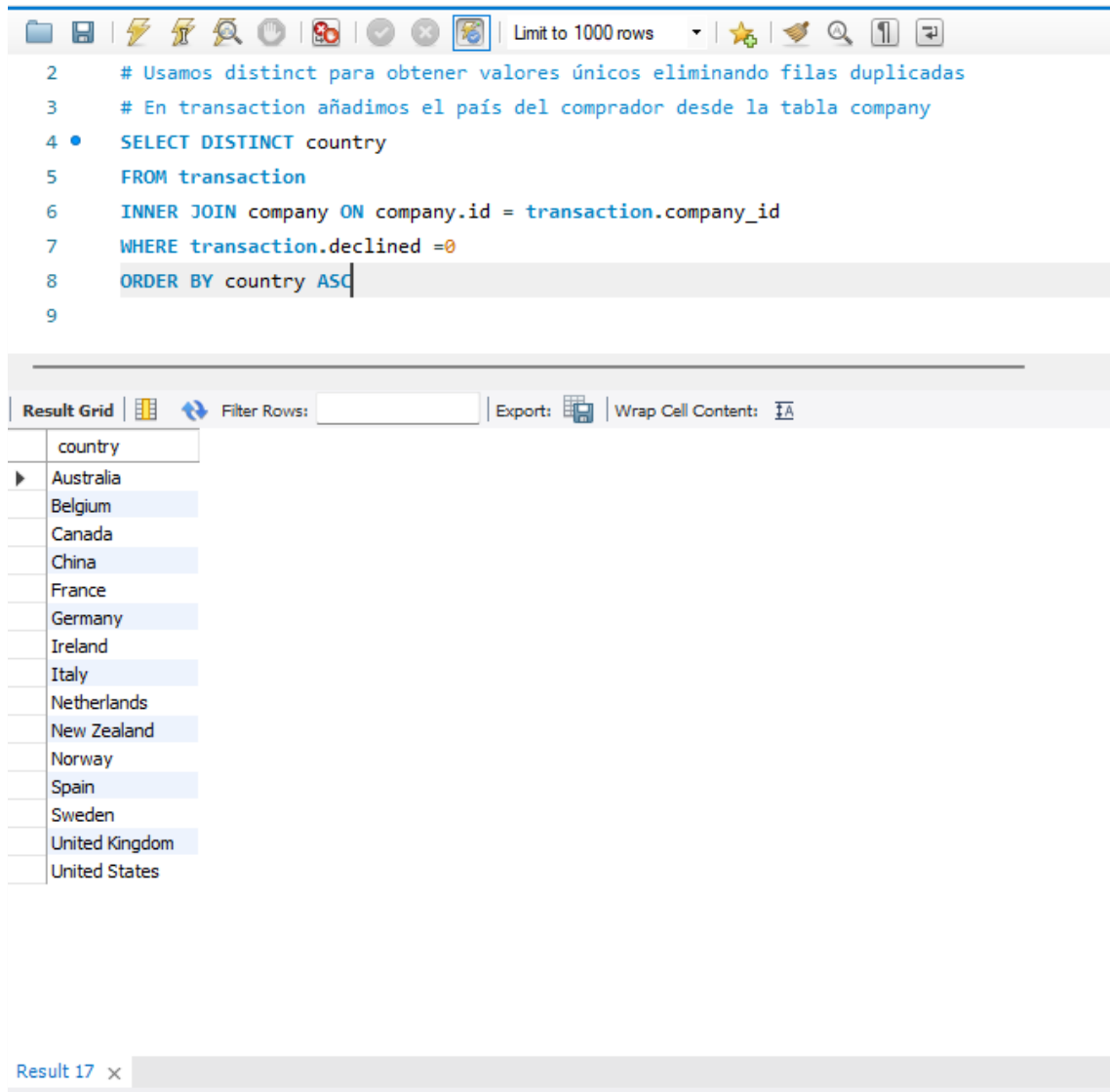
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	company_name	email	country
▶	A Institute	metus.aliquam@google.edu	Belgium
	Ac Fermentum Incorporated	donec.porttitor.tellus@yahoo.net	Germany
	Ac Industries	ipsum@yahoo.com	Germany
	Ac Libero Inc.	mollis.lectus@protonmail.ca	United Kingdom
	Aliquam Erat Volutpat LLP	pede.nunc@icloud.net	Italy
	Aliquam Iaculis Lacus Corp.	dictum@aol.org	Belgium
	Aliquam PC	scelerisque.mollis@icloud.org	Germany
	Aliquet Diam Limited	eu.eros.nam@icloud.org	United States
	Aliquet Sem Limited	sem.magna@yahoo.edu	Netherlands
	Aliquet Vel Vulputate Incorporated	suspendisse.commodo@outlook.couk	Netherlands
	Amet Faucibus Ut Foundation	malesuada@icloud.net	United Kingdom
	Amet Institute	nullam.lobortis.quam@outlook.net	Australia
	Amet Lorem LLP	ultrices.adipiscing@icloud.edu	Spain
	Amet Luctus Vulputate Foundation	nulla@outlook.com	Canada
	Amet Nulla Donec Corporation	mattis.integer.eu@protonmail.net	Italy
	Ante Iaculis Nec Foundation	sed.dictum.proin@outlook.ca	New Zealand
	Arcu LLP	dui@aol.ca	Norway
	At Associates	tristique.neque@yahoo.couk	New Zealand
	At Pedo Corp.	ac.mi.eleifend@hotmail.edu	Italy
	Auctor Mauris Corp.	eget.metus@protonmail.ca	Germany
	Auctor Mauris Vel LLP	nec.tempus@icloud.couk	United States
	Augue Foundation	mauris@yahoo.com	Germany
	Convallis In Incorporated	mauris.ut@aol.couk	Germany
	Cras Consulting	sed.consequat@google.ca	Belgium
	Cras Vehicula Aliquet Industries	arcu@hotmail.org	Netherlands
	Dictum Erat Corp.	donec.ante@icloud.net	Canada

company 4 x

SPRINT 1 EJERCICIO 3 - Países de las empresas que han comprado

*La compra tiene que haber llegado a buen fin, es decir, no haber sido declinada.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
2  # Usamos distinct para obtener valores únicos eliminando filas duplicadas
3  # En transaction añadimos el país del comprador desde la tabla company
4  • SELECT DISTINCT country
5  FROM transaction
6  INNER JOIN company ON company.id = transaction.company_id
7  WHERE transaction.declined =0
8  ORDER BY country ASC
9
```

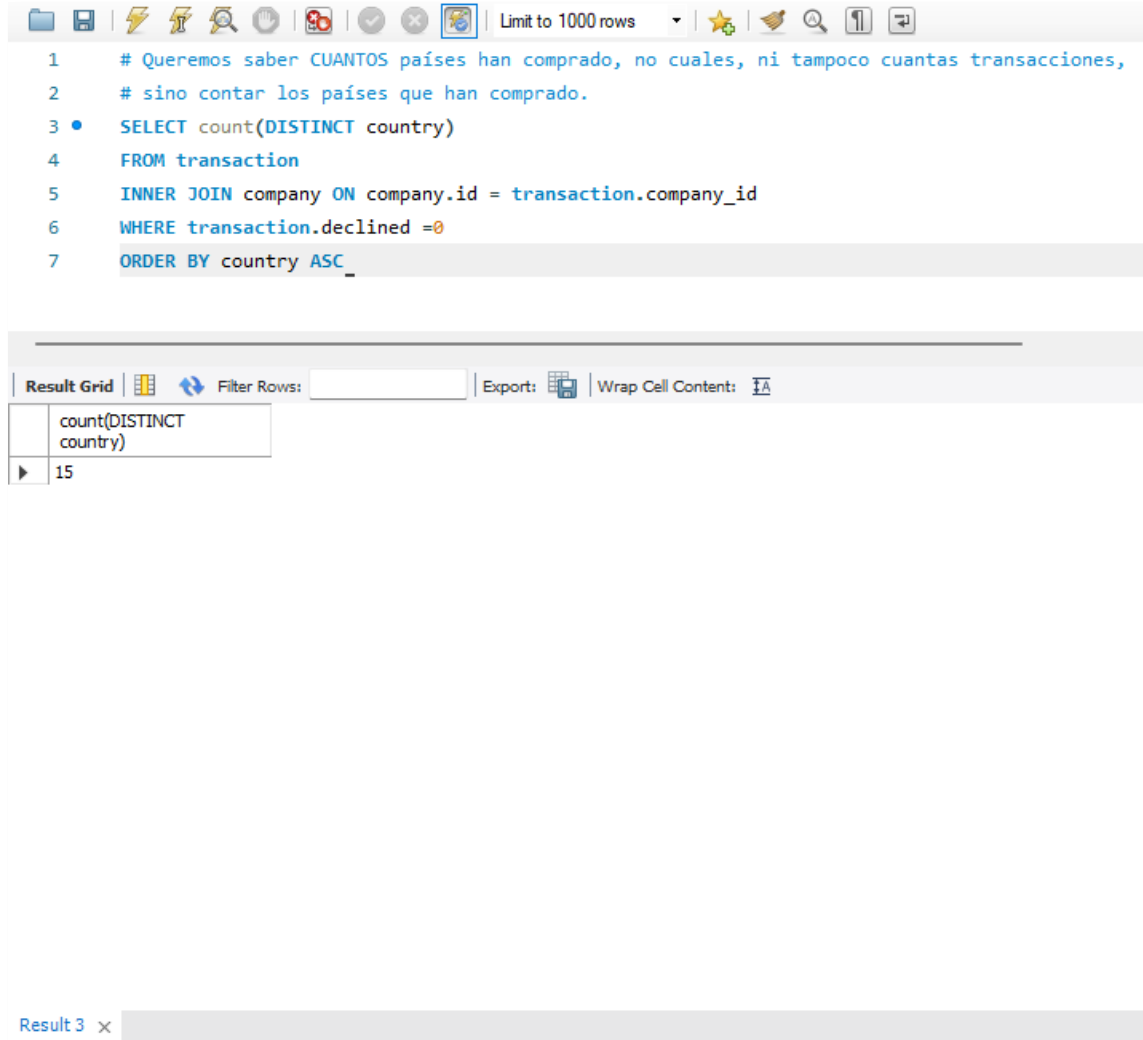
Below the editor is the 'Result Grid' section. It has a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are displayed in a table with one column, 'country', listing 15 countries in ascending order:

country
Australia
Belgium
Canada
China
France
Germany
Ireland
Italy
Netherlands
New Zealand
Norway
Spain
Sweden
United Kingdom
United States

At the bottom, a tab labeled 'Result 17' is visible.

SPRINT 1 EJERCICIO 4 - Quieren saber el Número de países compradores

*Entendemos que comprador es aquella transacción que no ha sido declinada, es decir, que ha llegado a buen fin.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
1 # Queremos saber CUANTOS países han comprado, no cuales, ni tampoco cuantas transacciones,
2 # sino contar los países que han comprado.
3 • SELECT count(DISTINCT country)
4 FROM transaction
5 INNER JOIN company ON company.id = transaction.company_id
6 WHERE transaction.declined =0
7 ORDER BY country ASC
```

Below the editor is the 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are displayed in a table:

	count(DISTINCT country)
▶	15

At the bottom left, there is a tab labeled 'Result 3' with a close button (x).

SPRINT_1_EJERCICIO_5 - Detalles (nombre y país)sobre la compañía con el número b-2354

Sabiendo el id de la empresa, queremos más detalles de su País así como su nombre

```
1 # Queremos saber el País y el Nombre con la compañía identificada con el número b-2354
2 SELECT id,company_name,country FROM company WHERE id = 'b-2354';
```

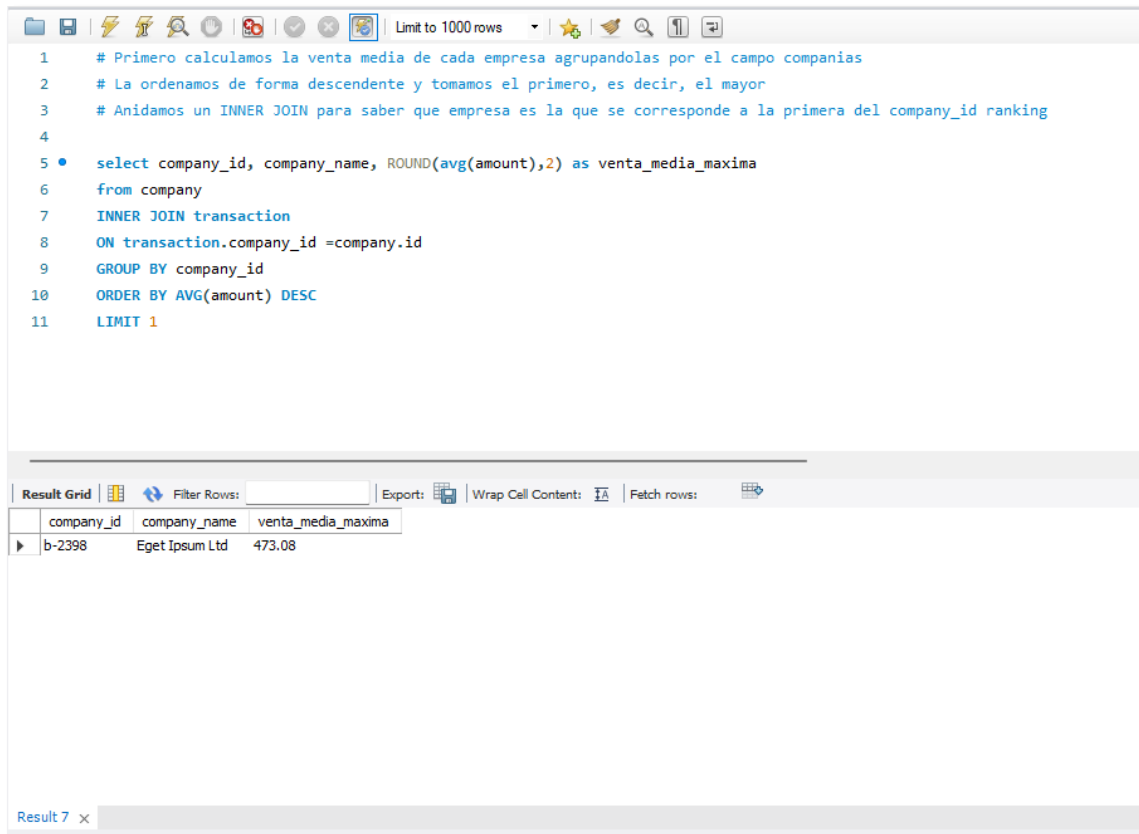
Result Grid

	id	company_name	country
▶	b-2354	Ac Libero Inc.	United Kingdom
*	NULL	NULL	NULL

SPRINT 1 EJERCICIO 6 - Queremos saber la empresa con la mayor venta media

Inicialmente identificamos la primera empresa de este ranking mediante los métodos AVG, GROUP BY, ORDER, y LIMIT.

Adicionalmente, anidamos un INNER JOIN para etiquetar el nombre de la empresa según el código id de la empresa identificada.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, a 'Limit to 1000 rows' dropdown, and search/execution tools. The SQL editor contains the following code:

```
1 # Primero calculamos la venta media de cada empresa agrupandolas por el campo companias
2 # La ordenamos de forma descendente y tomamos el primero, es decir, el mayor
3 # Anidamos un INNER JOIN para saber que empresa es la que se corresponde a la primera del company_id ranking
4
5 • select company_id, company_name, ROUND(avg(amount),2) as venta_media_maxima
6 from company
7 INNER JOIN transaction
8 ON transaction.company_id =company.id
9 GROUP BY company_id
10 ORDER BY AVG(amount) DESC
11 LIMIT 1
```

Below the editor is a 'Result Grid' section with a 'Filter Rows' input, an 'Export' button, 'Wrap Cell Content' and 'Fetch rows' options. The results are displayed in a table:

company_id	company_name	venta_media_maxima
b-2398	Eget Ipsum Ltd	473.08

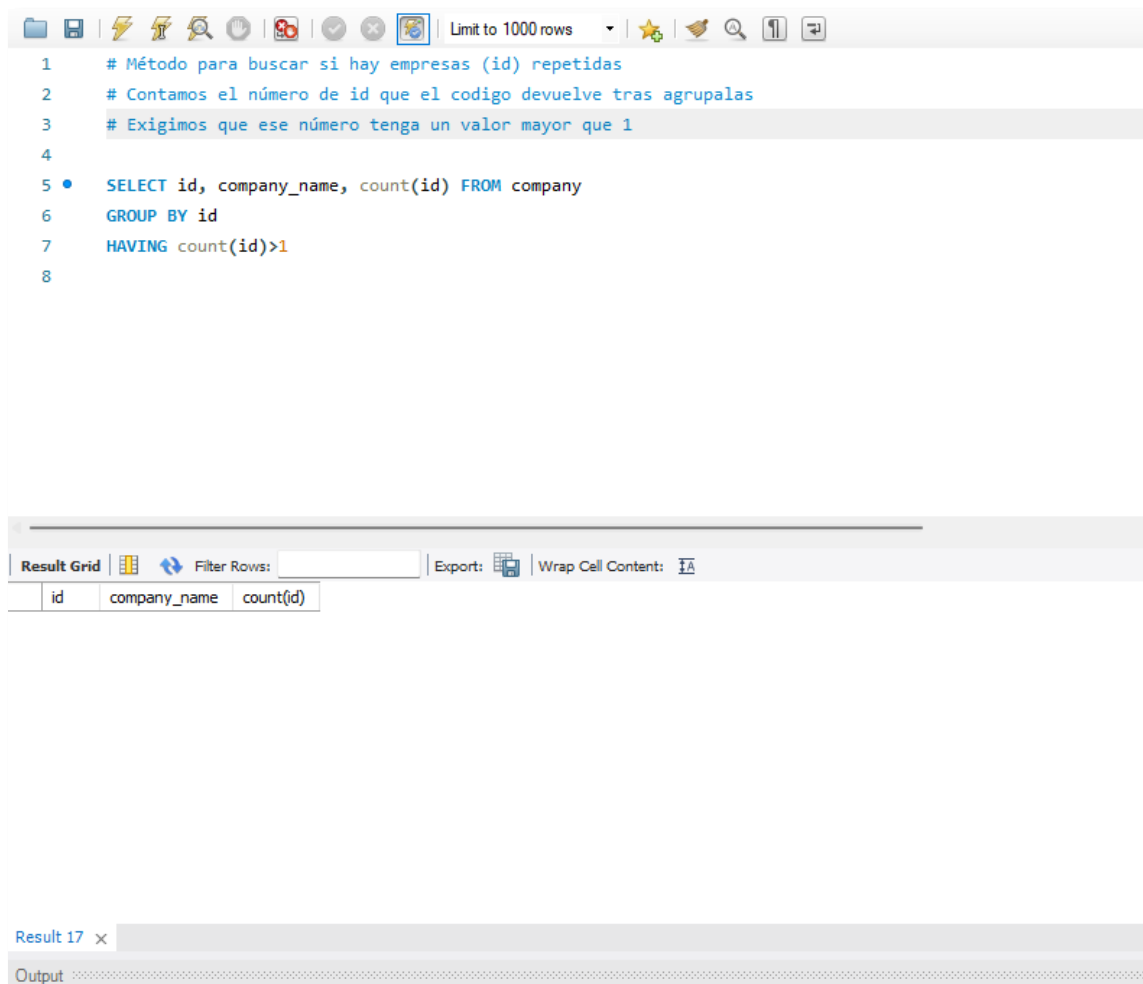
At the bottom, a tab labeled 'Result 7' is visible.

SPRINT 1 – NIVEL 2

SPRINT 1- NIVEL 2 - EJERCICIO 1 - Buscamos valores repetidos entre las compañías según id

El método usado es el count para contar las empresas y el having para exigir que tenga un valor superior a 1.

4



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search. The query editor contains the following SQL code:

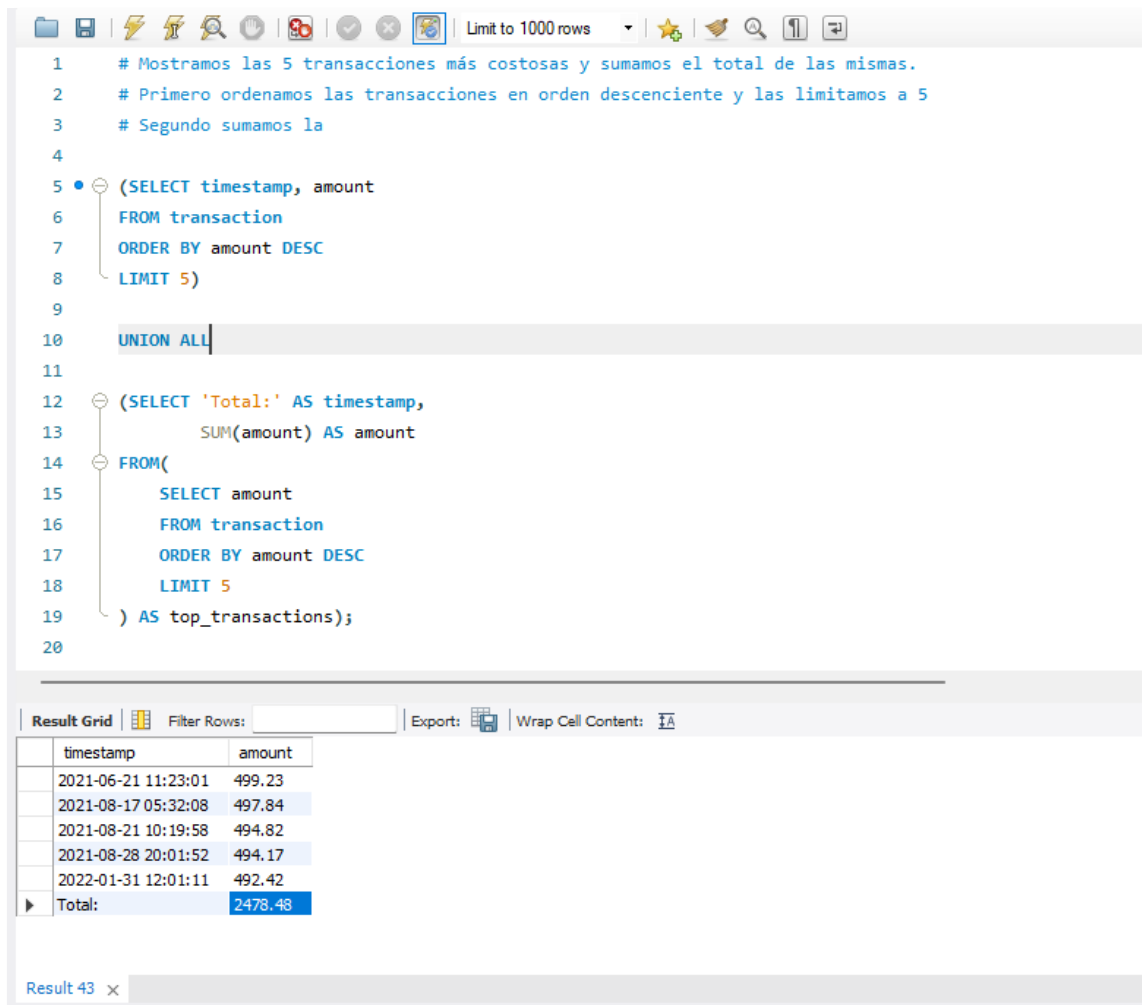
```
1 # Método para buscar si hay empresas (id) repetidas
2 # Contamos el número de id que el código devuelve tras agruparlas
3 # Exigimos que ese número tenga un valor mayor que 1
4
5 • SELECT id, company_name, count(id) FROM company
6   GROUP BY id
7   HAVING count(id)>1
8
```

Below the query editor is the 'Result Grid' section. It has a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The grid header shows the columns: id, company_name, and count(id). The grid body is currently empty.

At the bottom, there is a 'Result 17' tab and an 'Output' pane.

SPRINT 1- NIVEL 2 - EJERCICIO 2 - Buscamos 5 transacciones más costosas y las sumamos

Buscamos el timestamp las 5 transacciones de mayor valor, adicionalmente las sumamos para obtener el monto todos de esta agregación.



The screenshot shows a SQL IDE interface with a query editor and a result grid. The query is as follows:

```
1  # Mostramos las 5 transacciones más costosas y sumamos el total de las mismas.
2  # Primero ordenamos las transacciones en orden descendiente y las limitamos a 5
3  # Segundo sumamos la
4
5  (SELECT timestamp, amount
6   FROM transaction
7   ORDER BY amount DESC
8   LIMIT 5)
9
10 UNION ALL
11
12 (SELECT 'Total:' AS timestamp,
13      SUM(amount) AS amount
14 FROM(
15     SELECT amount
16     FROM transaction
17     ORDER BY amount DESC
18     LIMIT 5
19 ) AS top_transactions);
20
```

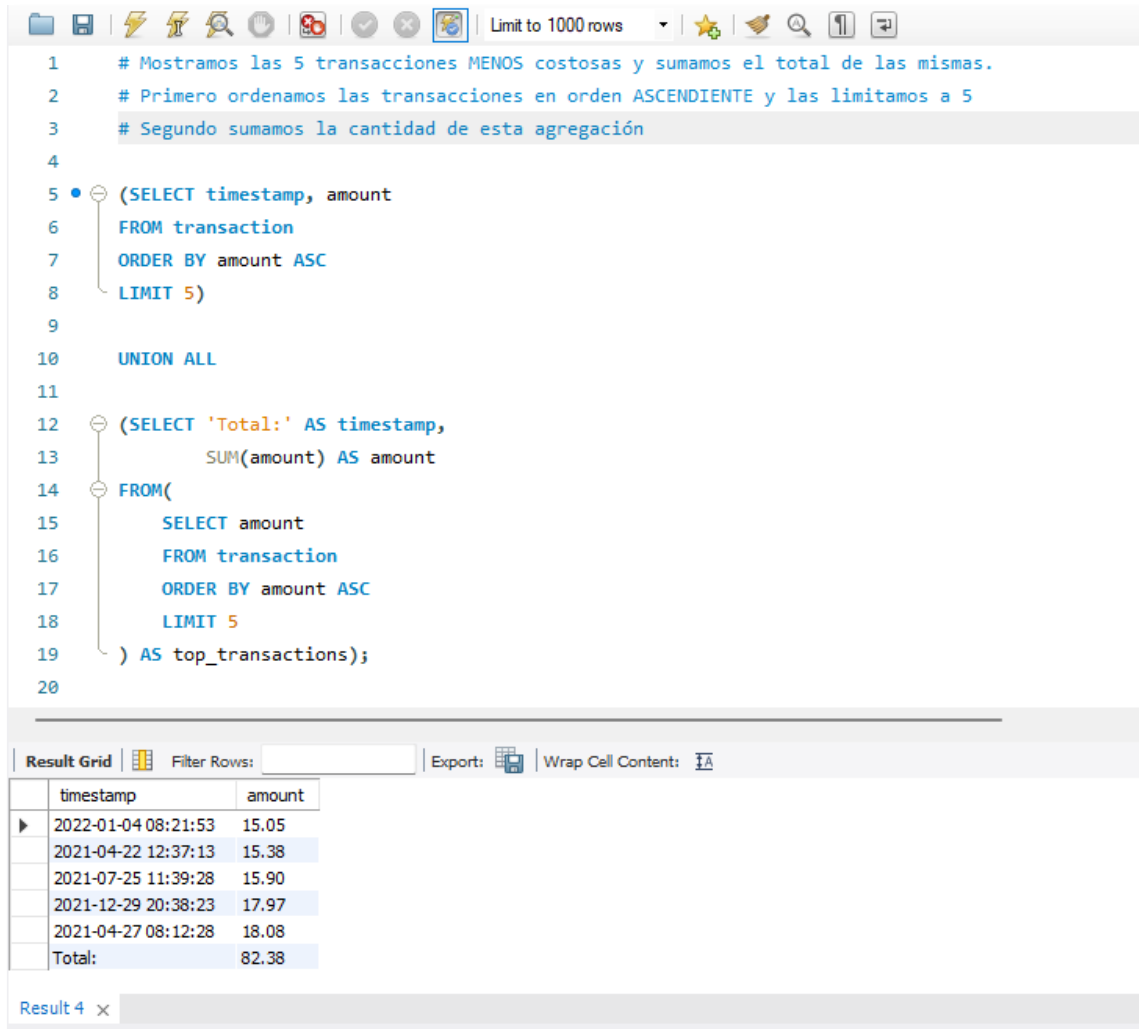
The result grid shows the following data:

timestamp	amount
2021-06-21 11:23:01	499.23
2021-08-17 05:32:08	497.84
2021-08-21 10:19:58	494.82
2021-08-28 20:01:52	494.17
2022-01-31 12:01:11	492.42
Total:	2478.48

Result 43 x

SPRINT_1- NIVEL 2 - EJERCICIO_3 - Buscamos 5 transacciones menores y las sumamos

Buscamos el timestamp de las 5 transacciones de menor valor, adicionalmente las sumamos para obtener el monto todos de esta agregación.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a dropdown menu set to "Limit to 1000 rows". The SQL editor contains the following query:

```
1  # Mostramos las 5 transacciones MENOS costosas y sumamos el total de las mismas.
2  # Primero ordenamos las transacciones en orden ASCENDIENTE y las limitamos a 5
3  # Segundo sumamos la cantidad de esta agregación
4
5  (SELECT timestamp, amount
6     FROM transaction
7     ORDER BY amount ASC
8     LIMIT 5)
9
10 UNION ALL
11
12 (SELECT 'Total:' AS timestamp,
13        SUM(amount) AS amount
14     FROM(
15         SELECT amount
16         FROM transaction
17         ORDER BY amount ASC
18         LIMIT 5
19     ) AS top_transactions);
20
```

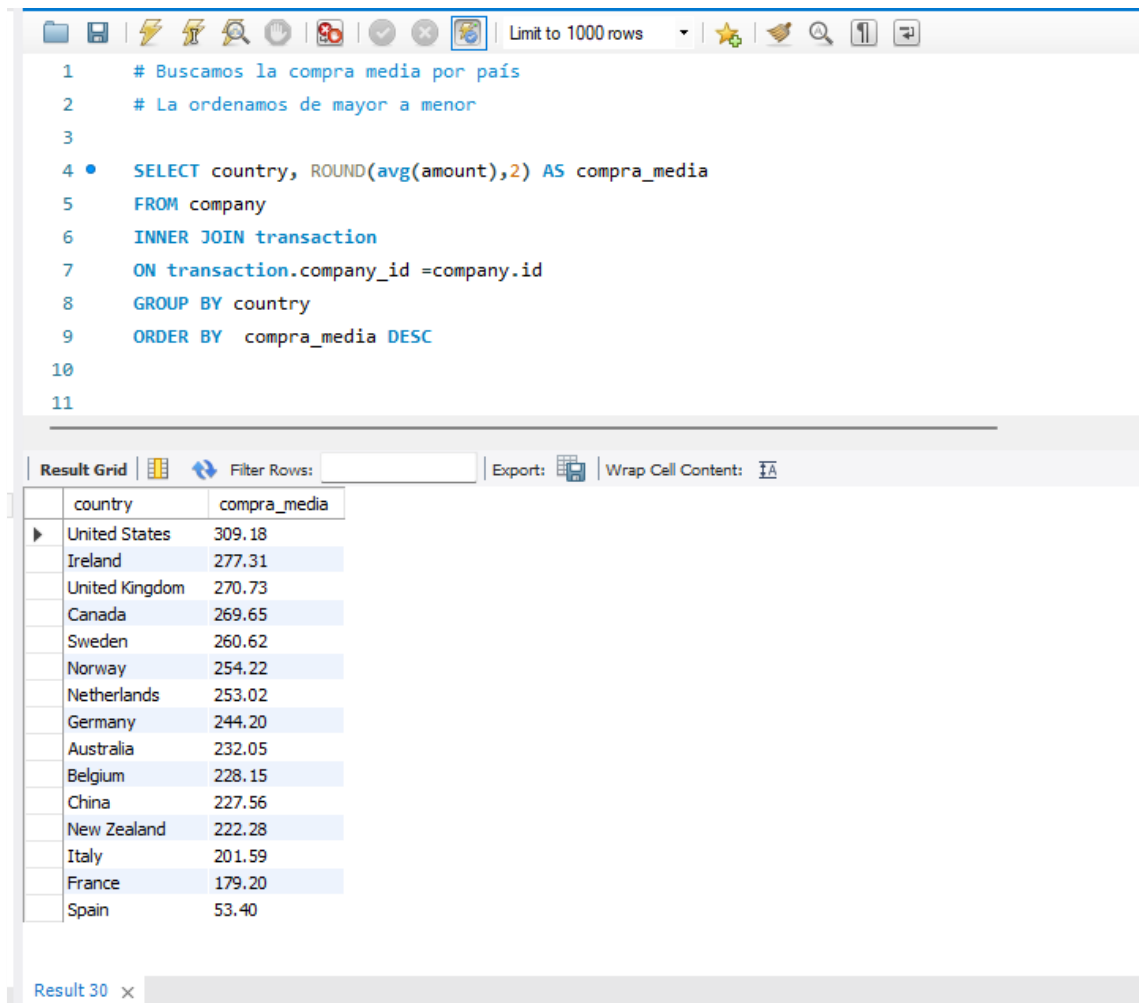
Below the editor, the "Result Grid" tab is active, displaying the query results. The results are shown in a table with two columns: "timestamp" and "amount". The first five rows represent the transactions with the lowest amounts, and the final row is a summary row labeled "Total:".

timestamp	amount
2022-01-04 08:21:53	15.05
2021-04-22 12:37:13	15.38
2021-07-25 11:39:28	15.90
2021-12-29 20:38:23	17.97
2021-04-27 08:12:28	18.08
Total:	82.38

At the bottom of the IDE, a tab labeled "Result 4" is visible.

SPRINT 1- NIVEL 2 - EJERCICIO 4 - Compra media por País ordenada de forma descendiente

Tras el inner join agregamos por país y buscamos la compra media, finalmente ordenamos la tabla en orden descendiente



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 # Buscamos la compra media por país
2 # La ordenamos de mayor a menor
3
4 • SELECT country, ROUND(avg(amount),2) AS compra_media
5 FROM company
6 INNER JOIN transaction
7 ON transaction.company_id =company.id
8 GROUP BY country
9 ORDER BY compra_media DESC
10
11
```

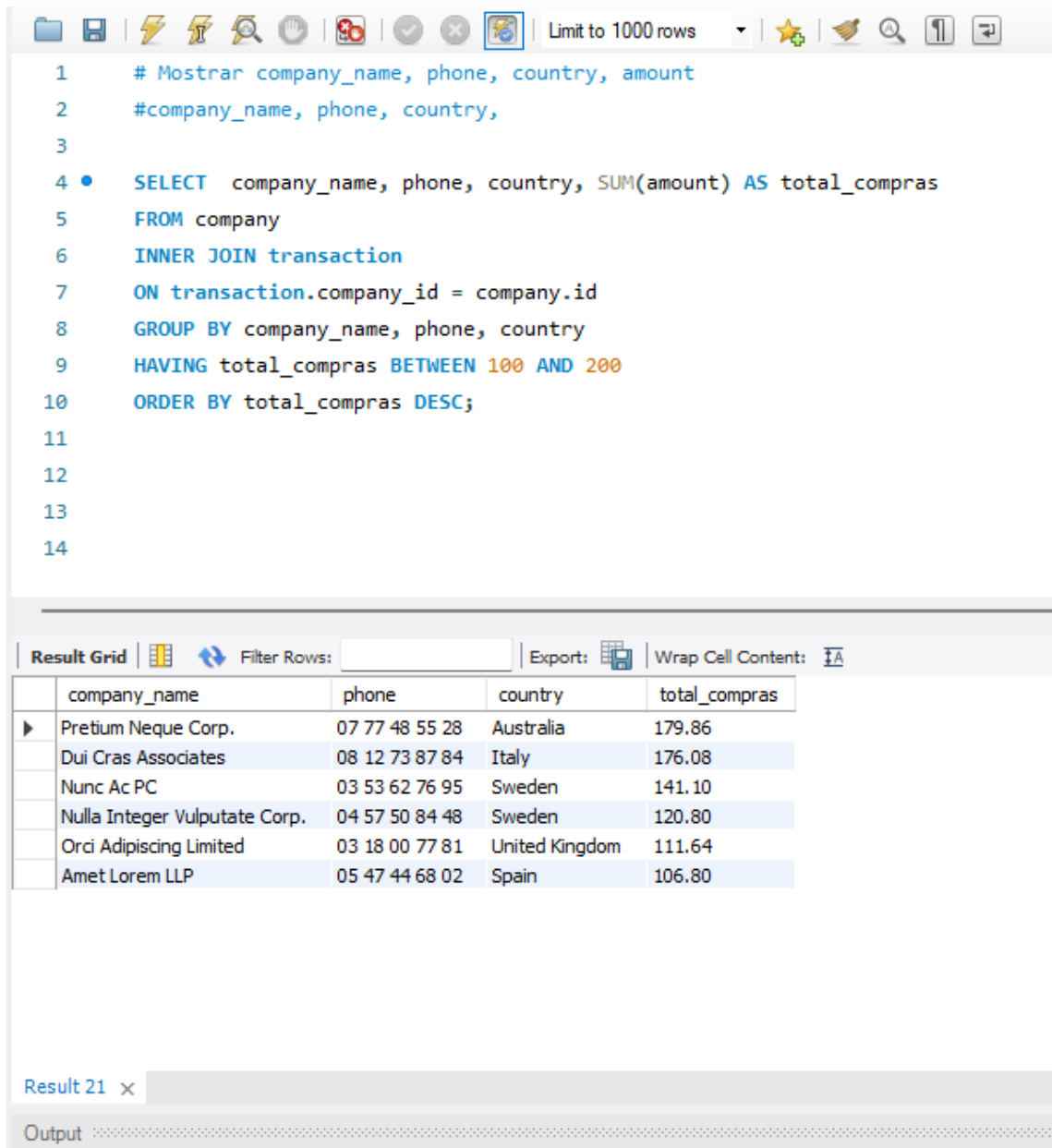
Below the query editor, there is a 'Result Grid' section with a 'Filter Rows' input and an 'Export' button. The grid displays the following data:

	country	compra_media
▶	United States	309.18
	Ireland	277.31
	United Kingdom	270.73
	Canada	269.65
	Sweden	260.62
	Norway	254.22
	Netherlands	253.02
	Germany	244.20
	Australia	232.05
	Belgium	228.15
	China	227.56
	New Zealand	222.28
	Italy	201.59
	France	179.20
	Spain	53.40

At the bottom left, it says 'Result 30' with a close button.

SPRINT 1- NIVEL 3 - EJERCICIO 1 - Nombre, Phone, Country con Total de compra por compañía en order descendiente siempre que las compras estén entre 100 y 200 € ordenado de forma descendiente

Calculo inicialmente la suma de las compras agregada por compañía, y uso el having (el where no es posible con operadores) para delimitar el rango de la compra, finalmente lo ordeno de forma descendiente.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
1  # Mostrar company_name, phone, country, amount
2  #company_name, phone, country,
3
4  • SELECT company_name, phone, country, SUM(amount) AS total_compras
5  FROM company
6  INNER JOIN transaction
7  ON transaction.company_id = company.id
8  GROUP BY company_name, phone, country
9  HAVING total_compras BETWEEN 100 AND 200
10 ORDER BY total_compras DESC;
11
12
13
14
```

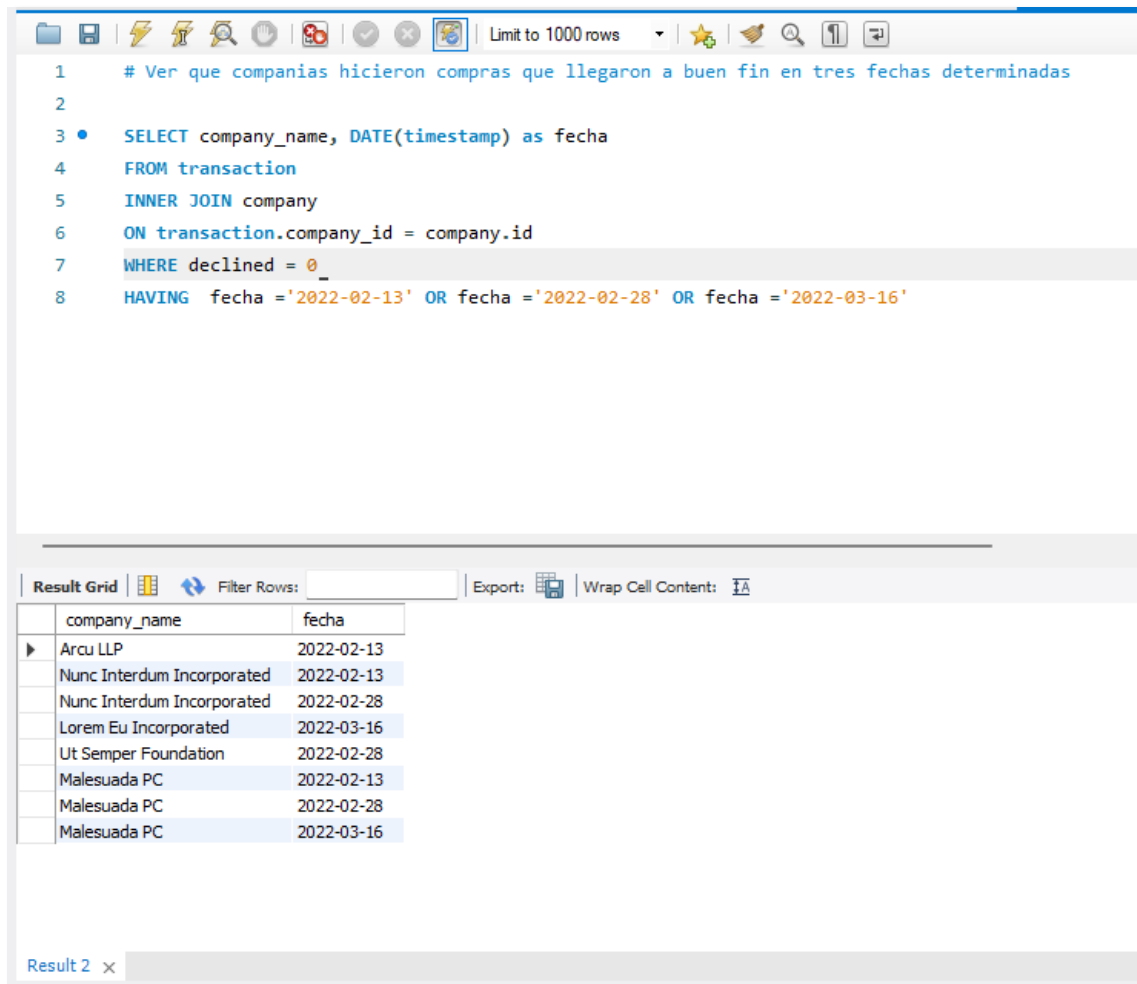
Below the editor is the 'Result Grid' section, which includes a 'Filter Rows' input, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are displayed in a table with 5 columns: company_name, phone, country, and total_compras. The table contains 7 rows of data, sorted by total_compras in descending order.

	company_name	phone	country	total_compras
▶	Pretium Neque Corp.	07 77 48 55 28	Australia	179.86
	Dui Cras Associates	08 12 73 87 84	Italy	176.08
	Nunc Ac PC	03 53 62 76 95	Sweden	141.10
	Nulla Integer Vulputate Corp.	04 57 50 84 48	Sweden	120.80
	Orci Adipiscing Limited	03 18 00 77 81	United Kingdom	111.64
	Amet Lorem LLP	05 47 44 68 02	Spain	106.80

At the bottom, there is a 'Result 21' tab and an 'Output' section.

SPRINT 1- NIVEL 3 - EJERCICIO 2 - Nombre de las compañías que hicieron compras que llegaron a buen fin en 3 días

Tras hacer el Inner Join para ver el nombre de la compañía, buscamos registros con operaciones no declinadas en tres fechas determinadas.



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1  # Ver que companias hicieron compras que llegaron a buen fin en tres fechas determinadas
2
3  • SELECT company_name, DATE(timestamp) as fecha
4  FROM transaction
5  INNER JOIN company
6  ON transaction.company_id = company.id
7  WHERE declined = 0
8  HAVING fecha = '2022-02-13' OR fecha = '2022-02-28' OR fecha = '2022-03-16'
```

Below the query editor, the results are displayed in a table with columns 'company_name' and 'fecha'.

company_name	fecha
Arcu LLP	2022-02-13
Nunc Interdum Incorporated	2022-02-13
Nunc Interdum Incorporated	2022-02-28
Lorem Eu Incorporated	2022-03-16
Ut Semper Foundation	2022-02-28
Malesuada PC	2022-02-13
Malesuada PC	2022-02-28
Malesuada PC	2022-03-16

At the bottom of the interface, there is a tab labeled 'Result 2' with a close button (x).