

Nivel 1

- Ejercicio 1

A partir de los documentos adjuntos (estructura_datos y datos_introducir), importa las dos tablas. Muestra las principales características del esquema creado y explica las diferentes tablas y variables que existen. Asegúrate de incluir un diagrama que ilustre la relación entre las distintas tablas y variables.

La base de datos “Transactions” contiene dos tablas:

- “transaction” (hechos)
Clave primaria: “id”
Clave foránea: “company_id”

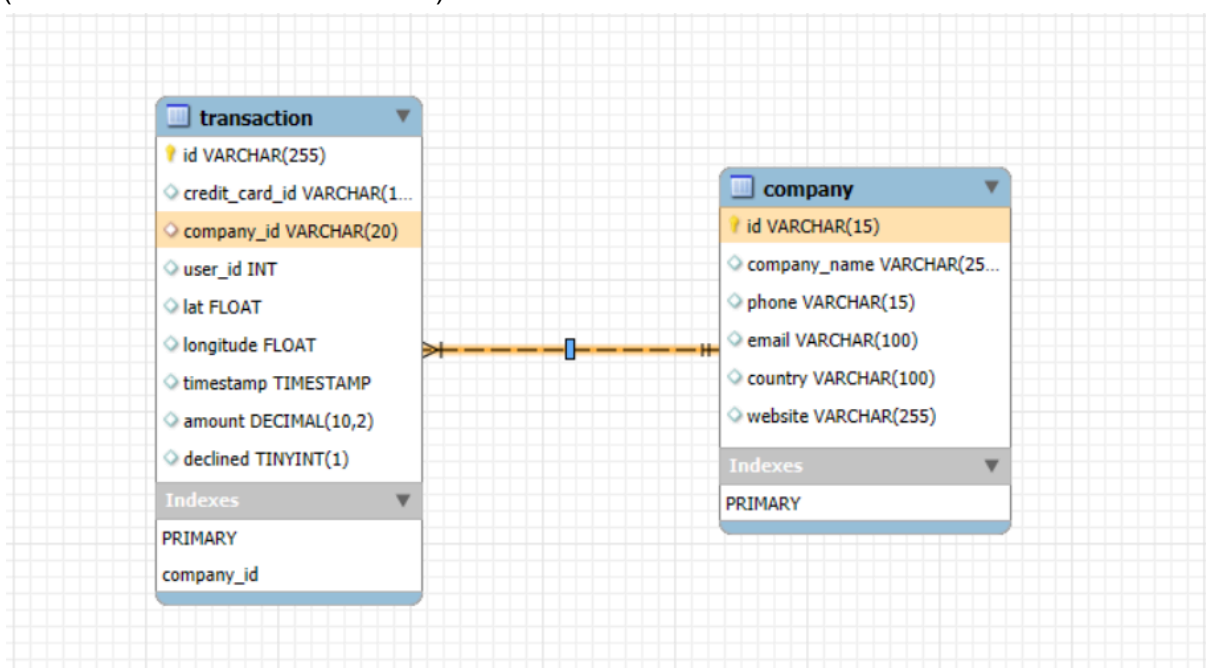
La “transaction” es la tabla principal (hechos) que contiene la información sobre las transacciones realizadas: id de la tarjeta de crédito, id de la empresa, id del usuario, latitud, longitud, la fecha, el monto y si fue declinada o no.

- “company” (dimensión)
Clave primaria: “id”

La “company” tiene la información de las empresas: id, nombre, teléfono, email, país y la página web.

La tabla “company” se relaciona con la tabla de hechos “transaction” a través de la clave foránea “company_id”. Es una relación de 1 uno a muchos ($1 \rightarrow \infty$): una empresa puede tener muchas transacciones, mientras que una transacción pertenece a una sola empresa.

La estructura se asemeja a un modelo estrella simple, donde “transaction” funciona como la tabla de hechos y “company” como una tabla de dimensión, aunque es una versión mínima (1 tabla de hechos + 1 dimensión).



- Ejercicio 2

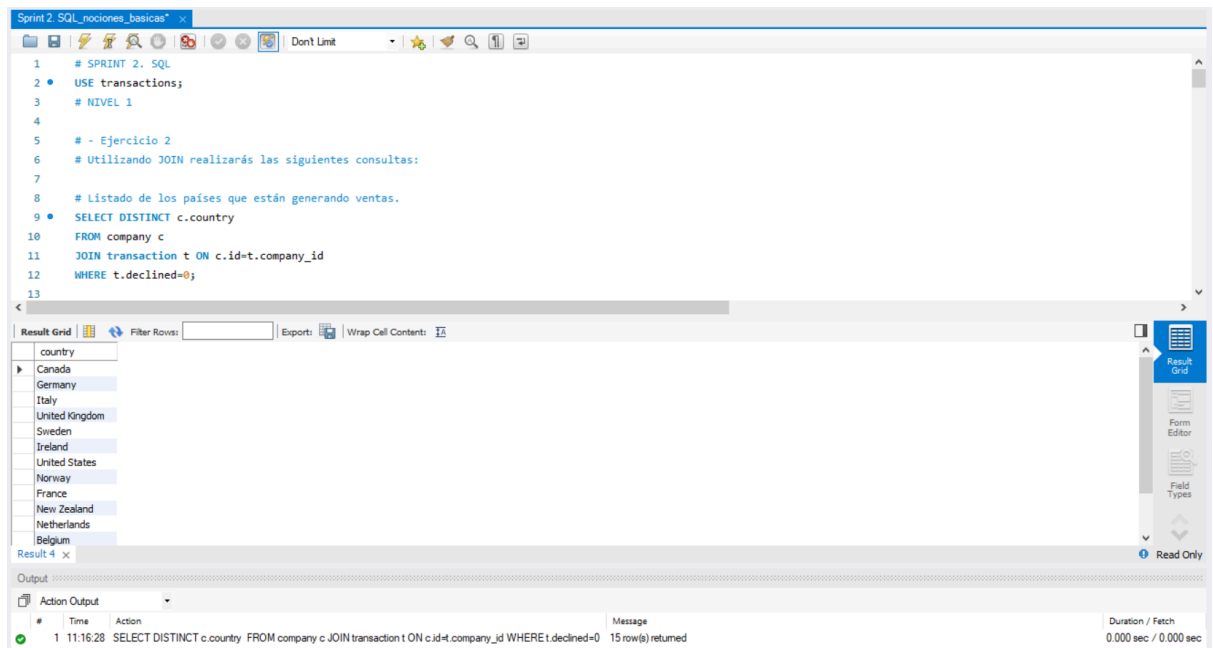
Nota general para el resto de los ejercicios: Transacciones vs Ventas.

Las transacciones son todas las operaciones registradas en el sistema, sin importar el resultado - rechazada/aprobada (declined=1/ declined=0). Cuando un ejercicio pide “transacciones” uso todas las filas de la tabla de transacciones, porque se refiere a la actividad general, no al dinero generado.

Las ventas representan sólo las transacciones aprobadas, es decir, las que generaron dinero real, aunque no lo diga explícitamente. La razón es que una transacción rechazada no genera ingresos, así que incluirla daría resultados incorrectos. Cuando se habla de ventas filtro por “WHERE declined=0”.

Utilizando JOIN realizarás las siguientes consultas:

- Listado de los países que están generando ventas.



The screenshot shows a SQL IDE window titled "Sprint 2. SQL_nociones_basicas". The query editor contains the following SQL code:

```
1 # SPRINT 2. SQL
2 • USE transactions;
3 # NIVEL 1
4
5 # - Ejercicio 2
6 # Utilizando JOIN realizarás las siguientes consultas:
7
8 # Listado de los países que están generando ventas.
9 • SELECT DISTINCT c.country
10 FROM company c
11 JOIN transaction t ON c.id=t.company_id
12 WHERE t.declined=0;
13
```

The results pane shows a list of countries:

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

The output pane shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	11:16:28	SELECT DISTINCT c.country FROM company c JOIN transaction t ON c.id=t.company_id WHERE t.declined=0	15 row(s) returned	0.000 sec / 0.000 sec

Nota: Necesitamos la información de ambas tablas “company” para extraer los países y “transaction” para saber cuales realizan transacciones no rechazadas. Utilizamos DISTINCT para obtener la lista de los países únicos y evitar las repeticiones.

- Desde cuántos países se generan las ventas.

The screenshot shows a SQL IDE window titled "Sprint 2: SQL_nociones_basicas". The query editor contains the following SQL code:

```
1 # SPRINT 2. SQL
2 • USE transactions;
3 # NIVEL 1
4
5 # - Ejercicio 2
6 # Utilizando JOIN realizarás las siguientes consultas:
7
8 # Desde cuántos países se generan las ventas.
9 • SELECT COUNT(DISTINCT country) AS Países_que_compran
10 FROM company c
11 JOIN transaction t ON c.id=t.company_id
12 WHERE t.declined=0;
```

The "Result Grid" shows a single row with the value 15 under the column "Países_que_compran". The "Output" pane shows the execution details: "1 11:37:42 SELECT COUNT(DISTINCT country) AS Países_que_compran FROM company c JOIN transaction t ON c.id=t.... 1 row(s) returned" with a duration of 0.000 sec / 0.000 sec.

Nota: Utilizamos COUNT DISTINCT para calcular la cantidad de los países únicos que generan ventas y evitar las repeticiones, tomamos en cuenta solo los que tienen al menos una transacción aprobada.

- Identifica a la compañía con la mayor media de ventas.

The screenshot shows the same SQL IDE window with a new query:

```
5 # - Ejercicio 2
6 # Utilizando JOIN realizarás las siguientes consultas:
7
8 # Identifica a la compañía con la mayor media de ventas.
9 • SELECT c.id, c.company_name, ROUND(AVG(t.amount),2) AS media_de_ventas
10 FROM company c
11 JOIN transaction t ON c.id=t.company_id
12 WHERE t.declined=0
13 GROUP BY c.id, c.company_name
14 ORDER BY media_de_ventas DESC
15 LIMIT 1;
```

The "Result Grid" shows a single row with the following data:

id	company_name	media_de_ventas
b-2398	Eget Ipsum Ltd	481.85

The "Output" pane shows the execution details: "1 11:40:06 SELECT c.id, c.company_name, ROUND(AVG(t.amount),2) AS media_de_ventas FROM company c JOIN trans... 1 row(s) returned" with a duration of 0.000 sec / 0.000 sec.

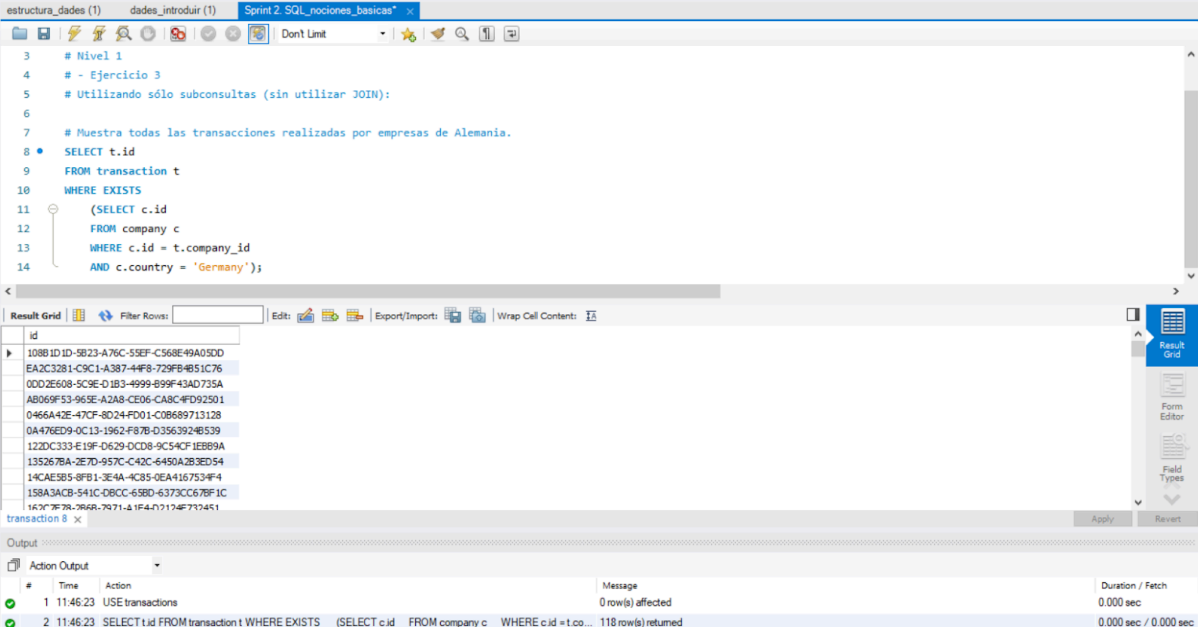
Nota: Necesitamos la información de ambas tablas de "company" el nombre de la compañía y de "transaction" el monte medio (AVG) de las ventas solo de las transacciones no rechazadas. Buscamos la media de ventas de cada una de las empresas, las ordenamos de mayor a menor (DESC) y LIMITamos el resultado a un registro - la compañía con la mayor media. Siguiendo buenas prácticas utilizo JOIN frente a la subconsulta y aplico ROUND para limitar la cantidad de decimales a 2.

- Ejercicio 3

Nota general sobre IN y EXISTS: por buenas prácticas he utilizado EXISTS en vez de IN ya que: IN busca un valor en una lista, pero puede dar problemas si hay NULL o la lista es muy grande, mientras que EXISTS solo comprueba si existe al menos un registro que cumpla la condición; es más seguro y eficiente.

Utilizando sólo subconsultas (sin utilizar JOIN):

- Muestra todas las transacciones realizadas por empresas de Alemania.



The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
3 # Nivel 1
4 # - Ejercicio 3
5 # Utilizando sólo subconsultas (sin utilizar JOIN):
6
7 # Muestra todas las transacciones realizadas por empresas de Alemania.
8 SELECT t.id
9 FROM transaction t
10 WHERE EXISTS
11     (SELECT c.id
12      FROM company c
13      WHERE c.id = t.company_id
14      AND c.country = 'Germany');
```

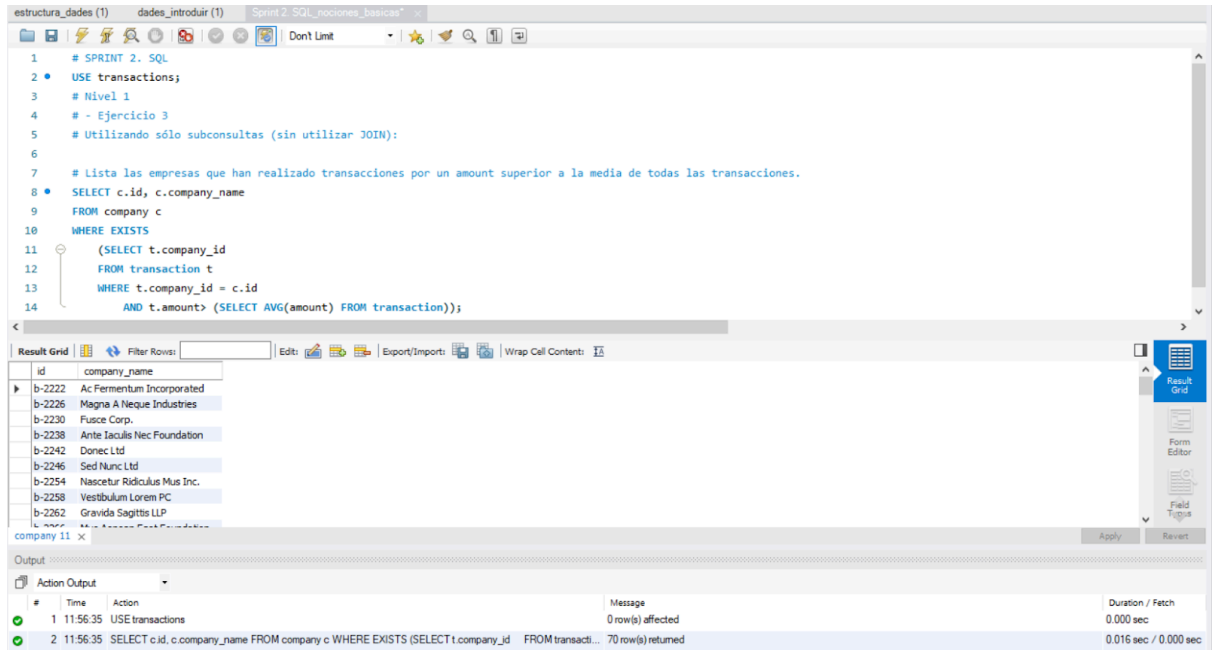
The results pane displays a list of transaction IDs. The first few IDs are: 10881D1D-5823-A76C-55EF-C568E49A05D0, EA2C3281-C9C1-A387-4F8-729FB4B51C76, 00D2E608-5C9E-01B3-4999-B99F43AD735A, AB069F53-969E-A2A8-CE06-CABC4FD92501, 0466A42E-47CF-8024-FD01-C0B689713128, 0A476ED9-0C13-1962-F87B-D3563924B539, 122DC333-E19F-0629-DCD8-9C54CF1EB89A, 135267BA-2E7D-957C-C42C-6450A2B3ED54, 14C4E5B3-8FB1-3E4A-4C35-0EA4167534F4, 158A3ACB-541C-08CC-658D-6373CC676F1C, and 163C7E2A-966A-7A71-41F4471734F7724E1.

The output pane shows the execution of the query. It indicates that 118 rows were returned in 0.000 seconds.

#	Time	Action	Message	Duration / Fetch
1	11:46:23	USE transactions	0 row(s) affected	0.000 sec
2	11:46:23	SELECT t.id FROM transaction t WHERE EXISTS (SELECT c.id FROM company c WHERE c.id = t.co...	118 row(s) returned	0.000 sec / 0.000 sec

Nota: Necesitamos únicamente las transacciones de la tabla “transaction” que cumplen el criterio de pertenecer a una empresa de Alemania (esta información se encuentra en la tabla “company”). Utilizamos la subconsulta con WHERE para filtrar las transacciones de las compañías alemanas. Se puede utilizar WHERE ... IN, pero para seguir las buenas prácticas opto por el uso de EXISTS.

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



The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
1 # SPRINT 2. SQL
2 • USE transactions;
3 # Nivel 1
4 # - Ejercicio 3
5 # Utilizando sólo subconsultas (sin utilizar JOIN):
6
7 # Lista las empresas que han realizado transacciones por un amount superior a la media de todas las transacciones.
8 • SELECT c.id, c.company_name
9 FROM company c
10 WHERE EXISTS
11     (SELECT t.company_id
12      FROM transaction t
13      WHERE t.company_id = c.id
14      AND t.amount > (SELECT AVG(amount) FROM transaction));
```

The results pane displays a table with two columns: 'id' and 'company_name'. The table contains 11 rows of data:

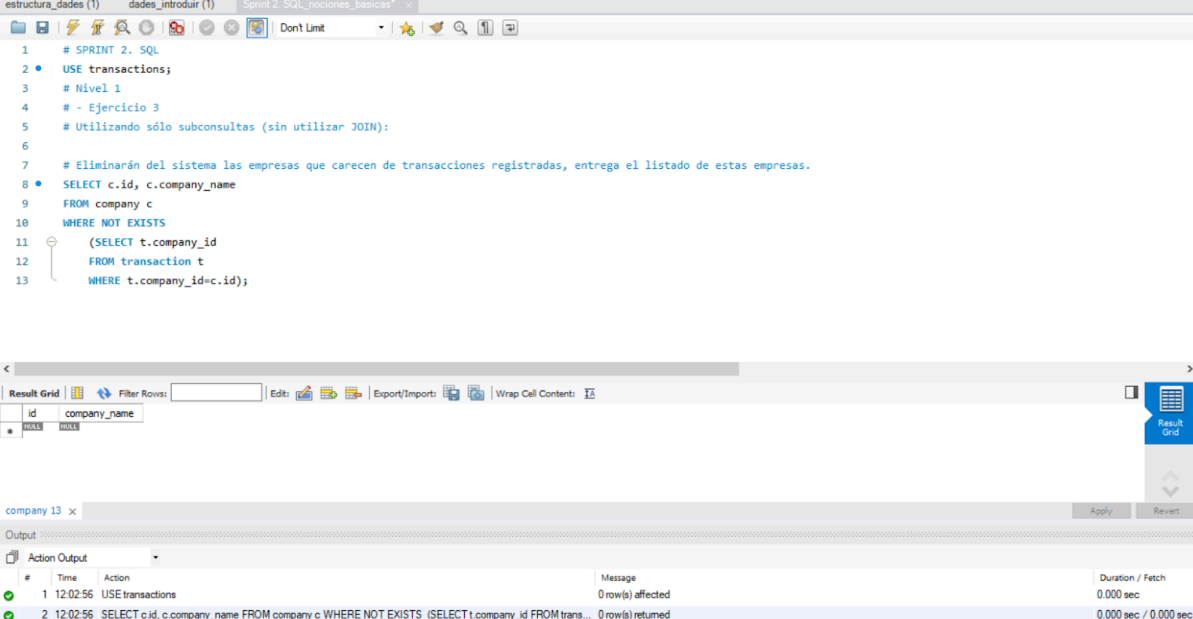
id	company_name
b-2222	Ac Fermentum Incorporated
b-2226	Magna A Nique Industries
b-2230	Fusce Corp.
b-2238	Ante Iaculis Nec Foundation
b-2242	Donec Ltd
b-2246	Sed Nunc Ltd
b-2254	Nascetur Ridiculus Mus Inc.
b-2258	Vestibulum Lorem PC
b-2262	Gravida Sagittis LLP
b-2266	Maecenas Port Foundation
b-2270	Maecenas Port Foundation

The output pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	11:56:35	USE transactions	0 row(s) affected	0.000 sec
2	11:56:35	SELECT c.id, c.company_name FROM company c WHERE EXISTS (SELECT t.company_id FROM transacti...	70 row(s) returned	0.016 sec / 0.000 sec

Nota: Utilizamos la subconsulta para encontrar la media de todas las transacciones y después con la consulta principal obtenemos la lista de los países cuyas transacciones superan la media.

- Eliminarán del sistema las empresas que carecen de transacciones registradas, entrega el listado de estas empresas.



The screenshot shows a SQL IDE interface with a query editor at the top and a results/output pane at the bottom. The query editor contains the following SQL code:

```
1 # SPRINT 2. SQL
2 • USE transactions;
3 # Nivel 1
4 # - Ejercicio 3
5 # Utilizando sólo subconsultas (sin utilizar JOIN):
6
7 # Eliminarán del sistema las empresas que carecen de transacciones registradas, entrega el listado de estas empresas.
8 • SELECT c.id, c.company_name
9 FROM company c
10 WHERE NOT EXISTS
11     (SELECT t.company_id
12      FROM transaction t
13      WHERE t.company_id=c.id);
```

The results pane shows a table with two columns: 'id' and 'company_name'. The table is empty, with 'NULL' values in both columns. Below the table, the 'Action Output' pane shows the execution log:

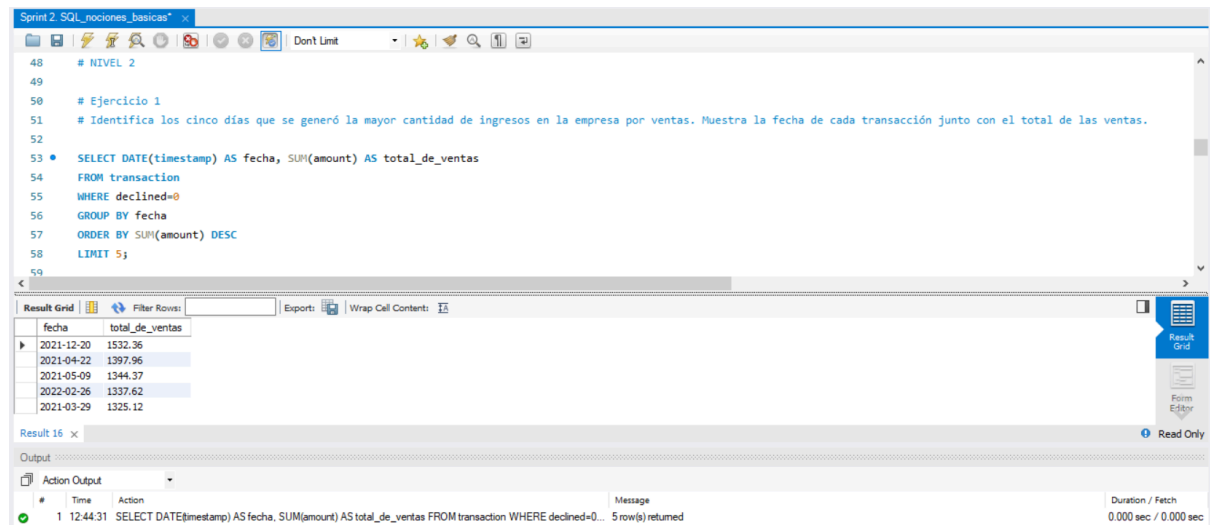
#	Time	Action	Message	Duration / Fetch
1	12.02:56	USE transactions	0 row(s) affected	0.000 sec
2	12.02:56	SELECT c.id, c.company_name FROM company c WHERE NOT EXISTS (SELECT t.company_id FROM trans...	0 row(s) returned	0.000 sec / 0.000 sec

Nota: Aplicamos la subconsulta correlacionada y seleccionamos las transacciones de la tabla “transaction”, después con la consulta principal filtramos las compañías que no tiene las transacciones. El resultado “NULL” nos demuestra que todas las compañías han realizado al menos una transacción.

Nivel 2

- Ejercicio 1

Identifica los cinco días que se generó la mayor cantidad de ingresos en la empresa por ventas. Muestra la fecha de cada transacción junto con el total de las ventas.



The screenshot shows a SQL IDE window titled "Sprint 2 SQL nociones basicas". The query editor contains the following SQL code:

```
48 # NIVEL 2
49
50 # Ejercicio 1
51 # Identifica los cinco días que se generó la mayor cantidad de ingresos en la empresa por ventas. Muestra la fecha de cada transacción junto con el total de las ventas.
52
53 • SELECT DATE(timestamp) AS fecha, SUM(amount) AS total_de_ventas
54 FROM transaction
55 WHERE declined=0
56 GROUP BY fecha
57 ORDER BY SUM(amount) DESC
58 LIMIT 5;
59
```

The "Result Grid" shows the following data:

fecha	total_de_ventas
2021-12-20	1532.36
2021-04-22	1397.96
2021-05-09	1344.37
2022-02-26	1337.62
2021-03-29	1325.12

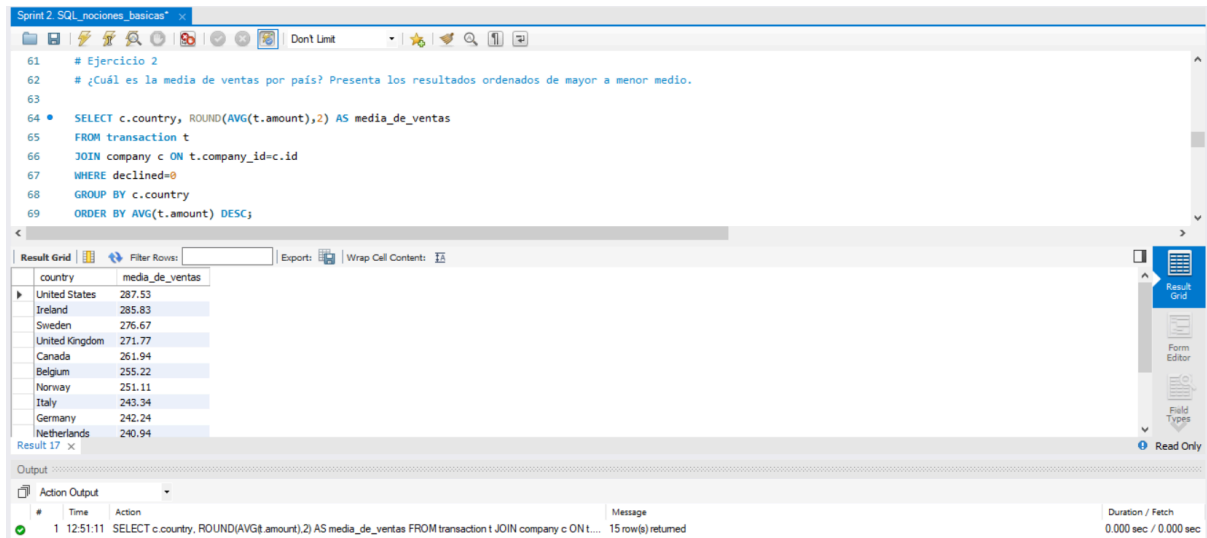
The "Output" section shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	12:44:31	SELECT DATE(timestamp) AS fecha, SUM(amount) AS total_de_ventas FROM transaction WHERE declined=0...	5 row(s) returned	0.000 sec / 0.000 sec

Nota: Para obtener la fecha a partir de timestamp utilizo DATE, agrupo todas las transacciones aprobadas por la fecha y calculo el total de transacciones realizadas en el mismo día. Como necesitamos el TOP-5, ordenamos de mayor a menor (DESC) y LIMITamos el output a 5 registros.

- Ejercicio 2

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



The screenshot shows a SQL IDE window titled "Sprint 2: SQL_nociones_basicas". The query editor contains the following SQL code:

```
61 # Ejercicio 2
62 # ¿Cuál es la media de ventas por país? Presenta los resultados ordenados de mayor a menor medio.
63
64 • SELECT c.country, ROUND(AVG(t.amount),2) AS media_de_ventas
65 FROM transaction t
66 JOIN company c ON t.company_id=c.id
67 WHERE declined=0
68 GROUP BY c.country
69 ORDER BY AVG(t.amount) DESC;
```

Below the query editor, the "Result Grid" displays the results of the query. The grid has two columns: "country" and "media_de_ventas". The results are ordered from highest to lowest average sales.

country	media_de_ventas
United States	287.53
Ireland	285.83
Sweden	276.67
United Kingdom	271.77
Canada	261.94
Belgium	255.22
Norway	251.11
Italy	243.34
Germany	242.24
Netherlands	240.94

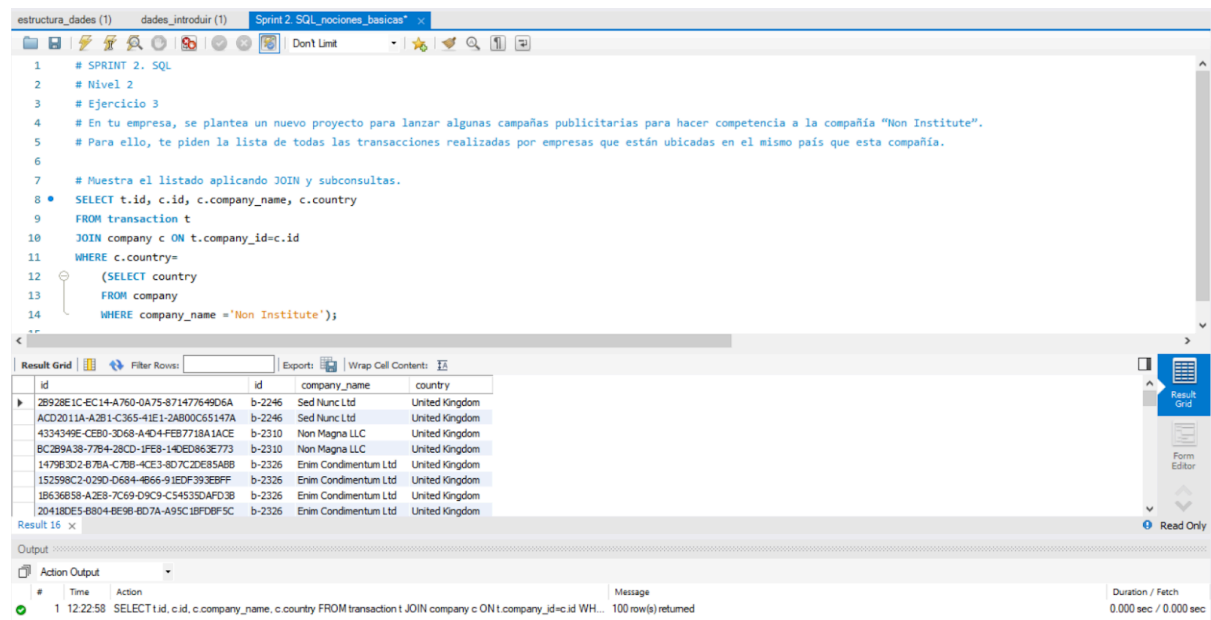
At the bottom of the IDE, the "Output" pane shows the "Action Output" for the query execution. It indicates that the query was executed at 12:51:11 and returned 15 rows.

#	Time	Action	Message	Duration / Fetch
1	12:51:11	SELECT c.country, ROUND(AVG(t.amount),2) AS media_de_ventas FROM transaction t JOIN company c ON t...	15 row(s) returned	0.000 sec / 0.000 sec

- Ejercicio 3

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

- Muestra el listado aplicando JOIN y subconsultas.



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 # SPRINT 2. SQL
2 # Nivel 2
3 # Ejercicio 3
4 # En tu empresa, se plantea un nuevo proyecto para lanzar algunas campañas publicitarias para hacer competencia a la compañía "Non Institute".
5 # Para ello, te piden la lista de todas las transacciones realizadas por empresas que están ubicadas en el mismo país que esta compañía.
6
7 # Muestra el listado aplicando JOIN y subconsultas.
8 SELECT t.id, c.id, c.company_name, c.country
9 FROM transaction t
10 JOIN company c ON t.company_id=c.id
11 WHERE c.country=
12     (SELECT country
13      FROM company
14      WHERE company_name = 'Non Institute');
```

The results grid displays the following data:

id	id	company_name	country
28928E1C-EC14-A760-0A75-871477649D6A	b-2246	Sed Nunc Ltd	United Kingdom
ACD2011A-A2B1-C365-41E1-2A800C65147A	b-2246	Sed Nunc Ltd	United Kingdom
4334349E-CEB0-3D68-A4D4-FEB7718A1ACE	b-2310	Non Magna LLC	United Kingdom
BC2B9A38-778A-28CD-1FE8-14DED863E773	b-2310	Non Magna LLC	United Kingdom
1479B3D2-678A-C78B-4CE3-8D7C2DE85ABB	b-2326	Enim Condimentum Ltd	United Kingdom
152598C2-029D-D684-4B66-91EDF393EBFF	b-2326	Enim Condimentum Ltd	United Kingdom
18636B58-A2E8-7C69-D9C9-C5453DAFD3B	b-2326	Enim Condimentum Ltd	United Kingdom
20418DE5-B804-8E9B-BD7A-A95C1BFD8F5C	b-2326	Enim Condimentum Ltd	United Kingdom

The output pane at the bottom shows the execution message: "SELECT t.id, c.id, c.company_name, c.country FROM transaction t JOIN company c ON t.company_id=c.id WH... 100 row(s) returned". The duration is 0.000 sec / 0.000 sec.

Nota: Necesitamos JOIN para obtener la información de ambas tablas id, nombre y país (solo para ver cuál es) de “company” y id de “transactions”. Utilizamos la subconsulta para obtener la información sobre de qué país es la empresa ‘Non-Institute’ y la colocamos en la condición WHERE porque nos interesan solo las compañías del mismo país.

- Muestra el listado aplicando solo subconsultas.

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

```
2 # Nivel 2
3 # Ejercicio 3
4 # En tu empresa, se plantea un nuevo proyecto para lanzar algunas campañas publicitarias para hacer competencia a la compañía "Non Institute".
5 # Para ello, te piden la lista de todas las transacciones realizadas por empresas que están ubicadas en el mismo país que esta compañía.
6
7 SELECT t.id
8 FROM transaction t
9 WHERE EXISTS
10     (SELECT c.id
11      FROM company c
12      WHERE t.company_id=c.id
13      AND c.country=
14          (SELECT country
15           FROM company
16           WHERE company_name = 'Non Institute'));
```

The results pane shows a list of transaction IDs:

id
2B928E1C-EC14-A760-0A75-871477649D6A
ACD2011A-A2B1-C365-41E1-2AB00C65147A
4334349E-CEB0-3D68-A4D4-FEB7718A1ACE
BC2B9A38-77B4-28CD-1FEB-14DED863E773
1479B3D2-87B4-C7B8-4CE3-8D7C2DE85A8B
152598C2-029D-D684-4B66-91EDF393EBFF
1B636B58-A2E8-7C69-D9C9-C54535DAFD3B
20418DE5-B804-8E9B-8D7A-A95C1BDFBF5C

The output pane shows the following message:

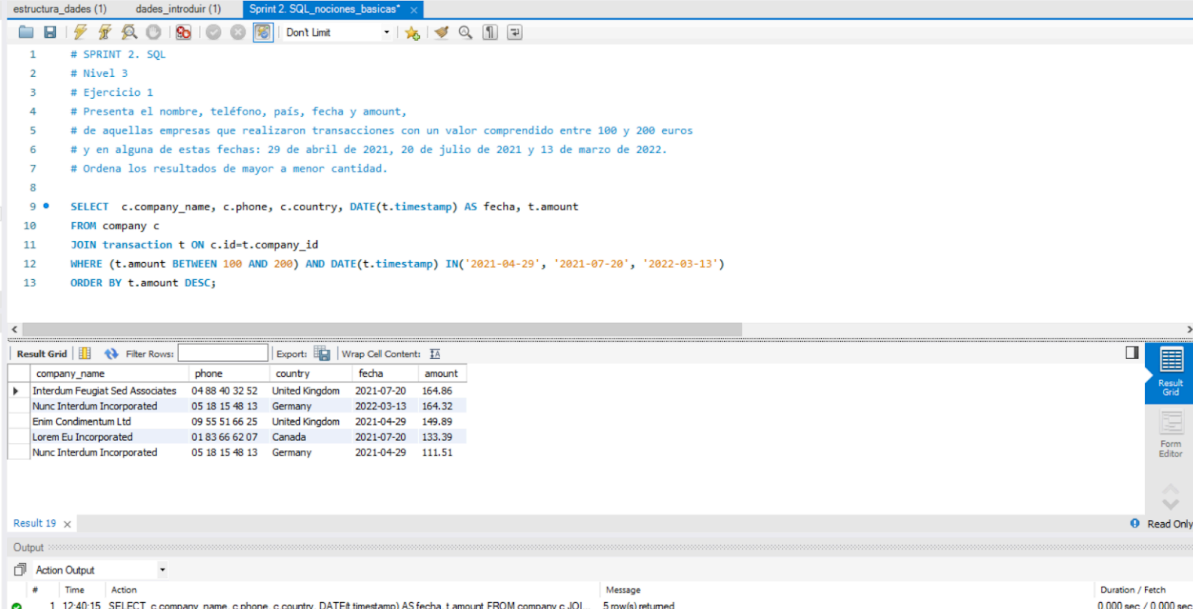
```
1 12:28:34 SELECT t.id FROM transaction t WHERE EXISTS (SELECT c.id FROM company c WHERE t.company_id=c.i... 100 row(s) returned
0.000 sec / 0.000 sec
```

Nota: Como no está permitido utilizar JOIN, hacen falta dos subconsultas (anidadas). Primero ver de qué país es la compañía 'Non-Institute', luego elegir solo las compañías que son de aquel país para poder con la consulta principal visualizar las transacciones de las empresas que son del mismo país que 'Non-Institute'.

Nivel 3

- Ejercicio 1

Presenta el nombre, teléfono, país, fecha y amount, de aquellas empresas que realizaron transacciones con un valor comprendido entre 100 y 200 euros y en alguna de estas fechas: 29 de abril de 2021, 20 de julio de 2021 y 13 de marzo de 2022. Ordena los resultados de mayor a menor cantidad.



The screenshot shows a SQL IDE interface with a query editor at the top and a results grid at the bottom. The query editor contains the following SQL code:

```
1 # SPRINT 2. SQL
2 # Nivel 3
3 # Ejercicio 1
4 # Presenta el nombre, teléfono, país, fecha y amount,
5 # de aquellas empresas que realizaron transacciones con un valor comprendido entre 100 y 200 euros
6 # y en alguna de estas fechas: 29 de abril de 2021, 20 de julio de 2021 y 13 de marzo de 2022.
7 # Ordena los resultados de mayor a menor cantidad.
8
9 • SELECT c.company_name, c.phone, c.country, DATE(t.timestamp) AS fecha, t.amount
10 FROM company c
11 JOIN transaction t ON c.id=t.company_id
12 WHERE (t.amount BETWEEN 100 AND 200) AND DATE(t.timestamp) IN('2021-04-29', '2021-07-20', '2022-03-13')
13 ORDER BY t.amount DESC;
```

The results grid displays the following data:

company_name	phone	country	fecha	amount
Interdum Feugiat Sed Associates	04 88 40 32 52	United Kingdom	2021-07-20	164.86
Nunc Interdum Incorporated	05 18 15 48 13	Germany	2022-03-13	164.32
Enim Condimentum Ltd	09 55 51 66 25	United Kingdom	2021-04-29	149.89
Lorem Eu Incorporated	01 83 66 62 07	Canada	2021-07-20	133.39
Nunc Interdum Incorporated	05 18 15 48 13	Germany	2021-04-29	111.51

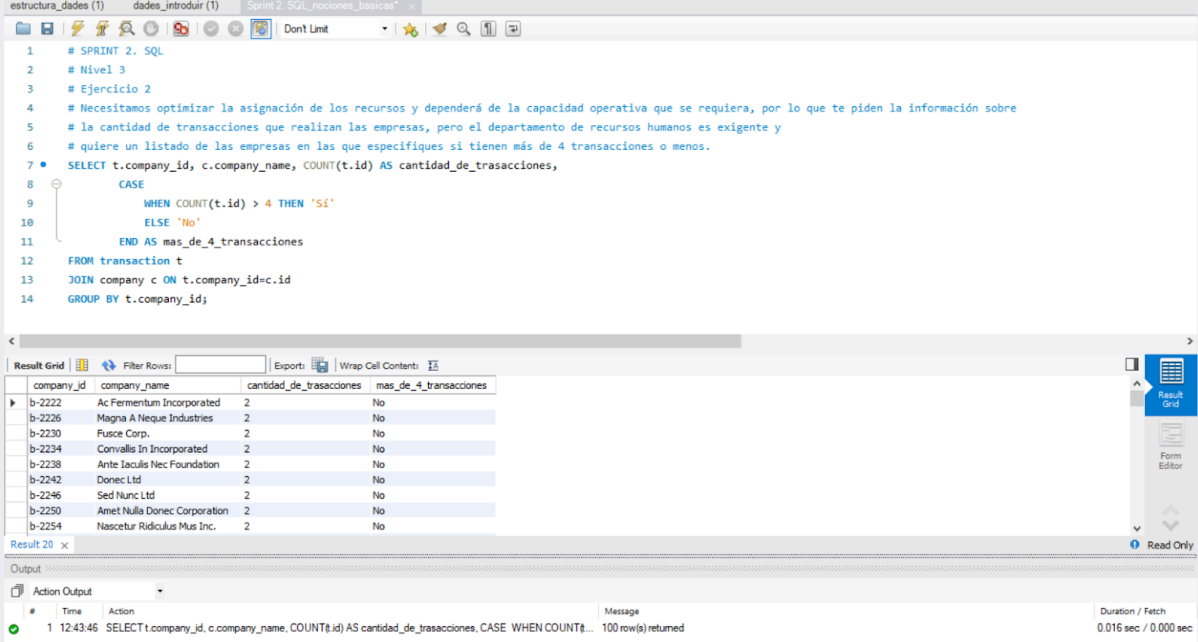
The bottom of the interface shows the 'Action Output' tab with the following message:

```
# Time Action Message Duration / Fetch
1 12:40:15 SELECT c.company_name, c.phone, c.country, DATE(t.timestamp) AS fecha, t.amount FROM company c JOIN... 5 row(s) returned 0.000 sec / 0.000 sec
```

Nota: Tenemos dos condiciones que tienen que cumplirse al mismo tiempo: el valor y la fecha, las colocamos en WHERE y las unimos con AND.

- Ejercicio 2

Necesitamos optimizar la asignación de los recursos y dependerá de la capacidad operativa que se requiera, por lo que te piden la 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 en las que especifiques si tienen más de 4 transacciones o menos.



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

```
1 # SPRINT 2. SQL
2 # Nivel 3
3 # Ejercicio 2
4 # Necesitamos optimizar la asignación de los recursos y dependerá de la capacidad operativa que se requiera, por lo que te piden la información sobre
5 # la cantidad de transacciones que realizan las empresas, pero el departamento de recursos humanos es exigente y
6 # quiere un listado de las empresas en las que especifiques si tienen más de 4 transacciones o menos.
7 • SELECT t.company_id, c.company_name, COUNT(t.id) AS cantidad_de_transacciones,
8     CASE
9         WHEN COUNT(t.id) > 4 THEN 'Si'
10        ELSE 'No'
11    END AS mas_de_4_transacciones
12 FROM transaction t
13 JOIN company c ON t.company_id=c.id
14 GROUP BY t.company_id;
```

The results grid shows the following data:

company_id	company_name	cantidad_de_transacciones	mas_de_4_transacciones
b-2222	Ac Fermentum Incorporated	2	No
b-2226	Magna A Neque Industries	2	No
b-2230	Fusce Corp.	2	No
b-2234	Connullis In Incorporated	2	No
b-2238	Ante Iaculis Nec Foundation	2	No
b-2242	Donec Ltd	2	No
b-2246	Sed Nunc Ltd	2	No
b-2250	Amet Nulla Donec Corporation	2	No
b-2254	Nascetur Ridiculus Mus Inc.	2	No

The output panel shows the execution of the query, indicating that 100 rows were returned in 0.016 seconds.

Nota: Necesitamos agrupar las transacciones por la empresa (GROUP BY) y calcular su cantidad respectiva (COUNT). Se nos solicita indicar explícitamente si hay más de 4 transacciones para lograrlo creamos nueva columna utilizando CASE y detallando las condiciones.