

DATA ANALYTICS

IT ACADEMY | BCN ACTIVA

SPRINT 02

Bases de datos relacionales e introducción a SQL
29 de abril de 2025



Letiane Benincá
benincalf@gmail.com

IT ACADEMY



DESCRIPCIÓN

Revisar los conceptos básicos para el uso de bases de datos relacionales. En este sprint, comenzarás tu experiencia práctica con una base de datos que contiene información de una empresa dedicada a la venta de productos en línea. En esta actividad, te enfocarás en datos relacionados con las transacciones realizadas y la información corporativa de las empresas que participaron.

NIVEL 1

EJERCICIO 01

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

La primera tabla es **Transaction**, que actúa como la tabla de hechos y contiene información sobre transacciones realizadas por usuarios. La segunda tabla es **Company**, que actúa como una tabla de dimensiones y almacena datos descriptivos de las empresas. La creación y estructura de las tablas es la siguiente:

```

estructura_datos x
4
5      -- Creamos la tabla company
6      CREATE TABLE IF NOT EXISTS company (
7          id VARCHAR(15) PRIMARY KEY,
8          company_name VARCHAR(255),
9          phone VARCHAR(15),
10         email VARCHAR(100),
11         country VARCHAR(100),
12         website VARCHAR(255)
13     );
14
15
16     -- Creamos la tabla transaction
17     CREATE TABLE IF NOT EXISTS transaction (
18         id VARCHAR(255) PRIMARY KEY,
19         credit_card_id VARCHAR(15) REFERENCES credit_card(id),
20         company_id VARCHAR(20),
21         user_id INT REFERENCES user(id),
22         lat FLOAT,
23         longitude FLOAT,
24         timestamp TIMESTAMP,
25         amount DECIMAL(10, 2),
26         declined BOOLEAN,
27         FOREIGN KEY (company_id) REFERENCES company(id)
28     );
  
```

TIPOLOGIA DE LOS DATOS:

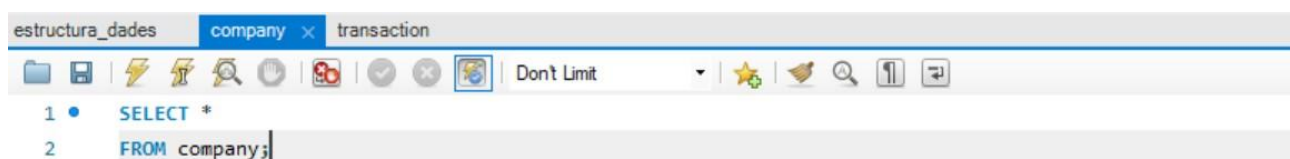
VARCHAR(n)	Almacenar texto de longitud variable, donde <i>n</i> representa la cantidad máxima de caracteres que puede contener.
INT	Almacena valores enteros, ya sean positivos o negativos, sin incluir decimales.
FLOAT	Dato numérico que admite cifras decimales.
TINYINT(1)	Utilizado para representar valores booleanos, donde 1 equivale a verdadero (TRUE) y 0 a falso (FALSE).
TIMESTAMP	Almacena fechas y horas en un formato: 2025-03-10 14:30:00.

DESCRIPCIÓN DE LAS TABLAS:

TABLA COMPANY

Esta tabla está diseñada para almacenar información descriptiva y estática sobre las empresas. Los tipos de datos predominantes son del tipo **VARCHAR(n)**, que permite almacenar cadenas de texto con una longitud máxima definida, las columnas son:

- **id (VARCHAR (15))**: La clave primaria que identifica de manera única a cada empresa.
- **company_name (VARCHAR (255))**: El nombre de la empresa.
- **phone (VARCHAR (15))**: teléfono de contacto de la empresa.
- **email (VARCHAR (100))**: email de la empresa.
- **country (VARCHAR (100))**: El país donde opera la empresa.
- **website (VARCHAR (255))**: La dirección URL del sitio web corporativo.



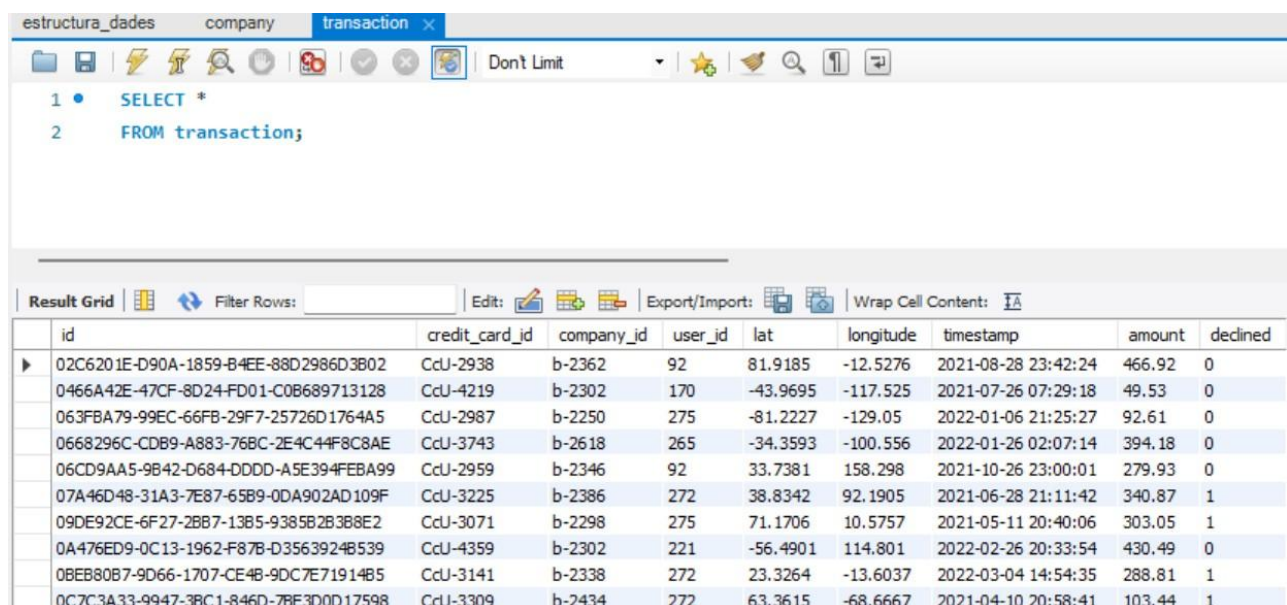
id	company_name	phone	email	country	website
b-2222	Ac Fermentum Incorporated	06 85 56 52 33	donec.porttitor.tellus@yahoo.net	Germany	https://instagram.com/site
b-2226	Magna A Neque Industries	04 14 44 64 62	risus.donec.nibh@icloud.org	Australia	https://whatsapp.com/group/9
b-2230	Fusce Corp.	08 14 97 58 85	risus@protonmail.edu	United States	https://pinterest.com/sub/cars
b-2234	Convallis In Incorporated	06 66 57 29 50	mauris.ut@aol.couk	Germany	https://cnn.com/user/110
b-2238	Ante Iaculis Nec Foundation	08 23 04 99 53	sed.dictum.proin@outlook.ca	New Zealand	https://netflix.com/settings
b-2242	Donec Ltd	01 25 51 37 37	at.iaculis@hotmail.couk	Norway	https://nytimes.com/user/110
b-2246	Sed Nunc Ltd	02 62 64 73 48	nibh@yahoo.org	United Kingdom	https://cnn.com/one
b-2250	Amet Nulla Donec Corporation	07 15 25 14 74	mattis.integer.eu@protonmail.net	Italy	https://netflix.com/sub/cars
b-2254	Nascetur Ridiculus Mus Inc.	06 26 87 61 84	suspendisse.dui@icloud.net	United States	https://ebay.com/sub
b-2258	Vestibulum Lorem PC	02 02 87 33 40	aenean.massa.integer@aol.net	Belgium	https://pinterest.com/sub/cars

El número después de VARCHAR, como VARCHAR (15) o VARCHAR (255), indica la longitud máxima de caracteres que puede almacenar ese campo en la base de datos.

TABLA TRANSACTION

Esta tabla contiene información relacionada con las transacciones realizadas. Aquí encontramos una mayor variedad de tipos de datos, a seguir vemos la definición de las columnas:

- **id (VARCHAR (255))**: Clave primaria que identifica de manera única cada transacción.
- **credit_card_id (VARCHAR (15))**: Un identificador asociado a la tarjeta de crédito utilizada, almacenado como texto sensible.
- **company_id (VARCHAR (20))**: Una clave foránea que establece una relación con la tabla **company**, permitiendo conectar cada transacción con la empresa correspondiente.
- **user_id (INT)**: Un identificador numérico que referencia al usuario que realizó la transacción.
- **lat (FLOAT)**: Latitud de la ubicación de la transacción.
- **longitude (FLOAT)**: Longitud de la ubicación de la transacción.
- **timestamp (TIMESTAMP)**: Registro de fecha y hora que indica cuándo ocurrió la transacción.
- **amount (DECIMAL (10,2))**: El monto de la transacción, almacenado como un número decimal con dos posiciones después del punto.
- **declined (TINYINT)**: Un valor booleano (0 o 1) que indica si la transacción fue rechazada (1) o aprobada (0).



The screenshot shows a database management interface with a tab labeled 'transaction'. Below the tab, there is a query editor with the following SQL code:

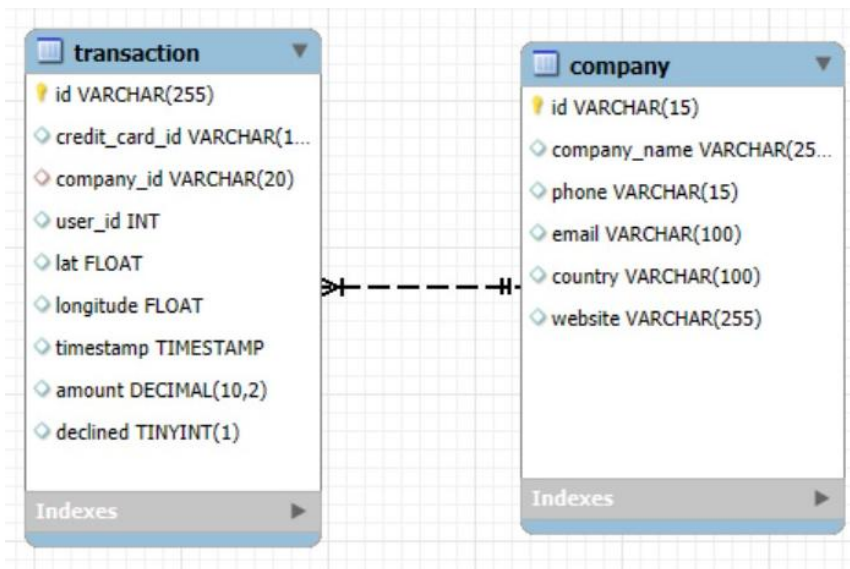
```
1 SELECT *
2 FROM transaction;
```

Below the query editor, there is a 'Result Grid' showing the data from the 'transaction' table. The grid has 10 columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The data is as follows:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
02C6201E-D90A-1859-B4EE-88D2986D3B02	CcU-2938	b-2362	92	81.9185	-12.5276	2021-08-28 23:42:24	466.92	0
0466A42E-47CF-8D24-FD01-C0B689713128	CcU-4219	b-2302	170	-43.9695	-117.525	2021-07-26 07:29:18	49.53	0
063FBA79-99EC-66FB-29F7-25726D1764A5	CcU-2987	b-2250	275	-81.2227	-129.05	2022-01-06 21:25:27	92.61	0
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	CcU-3743	b-2618	265	-34.3593	-100.556	2022-01-26 02:07:14	394.18	0
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	CcU-2959	b-2346	92	33.7381	158.298	2021-10-26 23:00:01	279.93	0
07A46D48-31A3-7E87-65B9-0DA902AD109F	CcU-3225	b-2386	272	38.8342	92.1905	2021-06-28 21:11:42	340.87	1
09DE92CE-6F27-2B87-13B5-9385B2B388E2	CcU-3071	b-2298	275	71.1706	10.5757	2021-05-11 20:40:06	303.05	1
0A476ED9-0C13-1962-F87B-D3563924B539	CcU-4359	b-2302	221	-56.4901	114.801	2022-02-26 20:33:54	430.49	0
08EB80B7-9D66-1707-CE4B-9DC7E71914B5	CcU-3141	b-2338	272	23.3264	-13.6037	2022-03-04 14:54:35	288.81	1
0C7C3A33-9947-3BC1-846D-7BE3D0D17598	CcU-3309	b-2434	272	63.3615	-68.6667	2021-04-10 20:58:41	103.44	1

La relación entre las tablas es de uno a muchos (1: N), donde una empresa puede estar asociada con múltiples transacciones, pero cada transacción está vinculada a una única empresa. Esta

relación se establece mediante la columna **company_id** en la tabla **Transaction**, que es una clave foránea que referencia la columna **id** en la tabla **Company**.



EJERCICIO 02

Utilizando JOIN realizarás las siguientes consultas:

- Listado de los países que están realizando compras.

Utilizando el **SELECT DISTINCT** para evitar repeticiones, hacemos **JOIN** (inner) entre las dos tablas para encontrar el nombre de los países que están realizando compras, totalizando en 15 países y retorna 16 filas, por el valor NULL.

Consulta:

```
SELECT DISTINCT country AS Países
FROM company co
JOIN transaction tr
ON co.id = tr.company_id;
```

The screenshot shows a SQL IDE window titled '29042025_LB_Data_Analytics_S...'. The query editor contains the following SQL code:

```

1  -- Nivel 01. Ejercicio 02:
2  -- Listado de los países que están realizando compras:
3  • SELECT DISTINCT country AS Países
4    FROM company co
5    JOIN transaction tr
6    ON co.id = tr.company_id;

```

Below the query editor, the 'Result Grid' tab is active, displaying a list of countries:

Países
Germany
Australia
United States
New Zealand
Norway
United Kingdom
Italy
Belgium
Sweden
Ireland
China
Canada

At the bottom, the 'Output' tab shows the execution results:

#	Time	Action	Message	Duration / Fetch
12	11:02:02	SELECT COUNT(DISTINCT country) AS Total_Paises FROM company co J...	1 row(s) returned	0.016 sec / 0.000 sec

b) Desde cuántos países se realizan las compras.

Con **COUNT** contamos el número de países y utilizamos **DISTINCT** para evitar repetirlos. Unimos las tablas con **JOIN** (inner) entre las claves primarias (*transaction.company_id* y *company.id*). Nos resultan 15 países, como ya verificado, anteriormente, en la consulta inicial.

Consulta:

```

SELECT COUNT (DISTINCT country) AS Total_Paises
FROM company co
JOIN transaction tr
ON co.id = tr.company_id;

```


The screenshot shows a SQL IDE window titled '29042025_LB_Data Analytics_S...'. The query editor contains the following SQL code:

```

8  -- Desde cuántos países se realizan las compras:
9  • SELECT COUNT(DISTINCT country) AS Total_Paises
10 FROM company co
11 JOIN transaction tr
12 ON co.id = tr.company_id;
13

```

The 'Result Grid' shows a single row with the value 15 for the column 'Total_Paises'.

The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
14	11:04:29	SELECT COUNT(DISTINCT country) AS Total_Paises FROM company co J...	1 row(s) returned	0.000 sec / 0.000 sec
15	11:04:59	SELECT COUNT(DISTINCT country) AS Total_Paises FROM company co J...	1 row(s) returned	0.016 sec / 0.000 sec

c) Identifica la compañía con el promedio más grande de ventas.

Para esta consulta, empezamos seleccionando las columnas que queremos mostrar: el identificador de la empresa (`co.company_id`), para evitar repetidos o empates, el nombre de la empresa (`co.company_name`) y para calcular el promedio, usamos ***ROUND(AVG(tr.amount), 2)*** (para redondear a dos decimales) y le ponemos un alias como ***AS Promedio_ventas***.

Filtramos solo las transacciones aceptadas con ***WHERE declined = 0*** y agrupamos los datos por empresa usando ***GROUP BY 1,2*** (`company_id` y `company_name`, por su posición en la consulta). Luego, ordenamos el resultado de mayor a menor con ***ORDER BY 3 DESC*** (por la posición 3 de la consulta) y limitamos la salida a una sola fila con ***LIMIT 1***, para obtener la empresa con el promedio más alto.

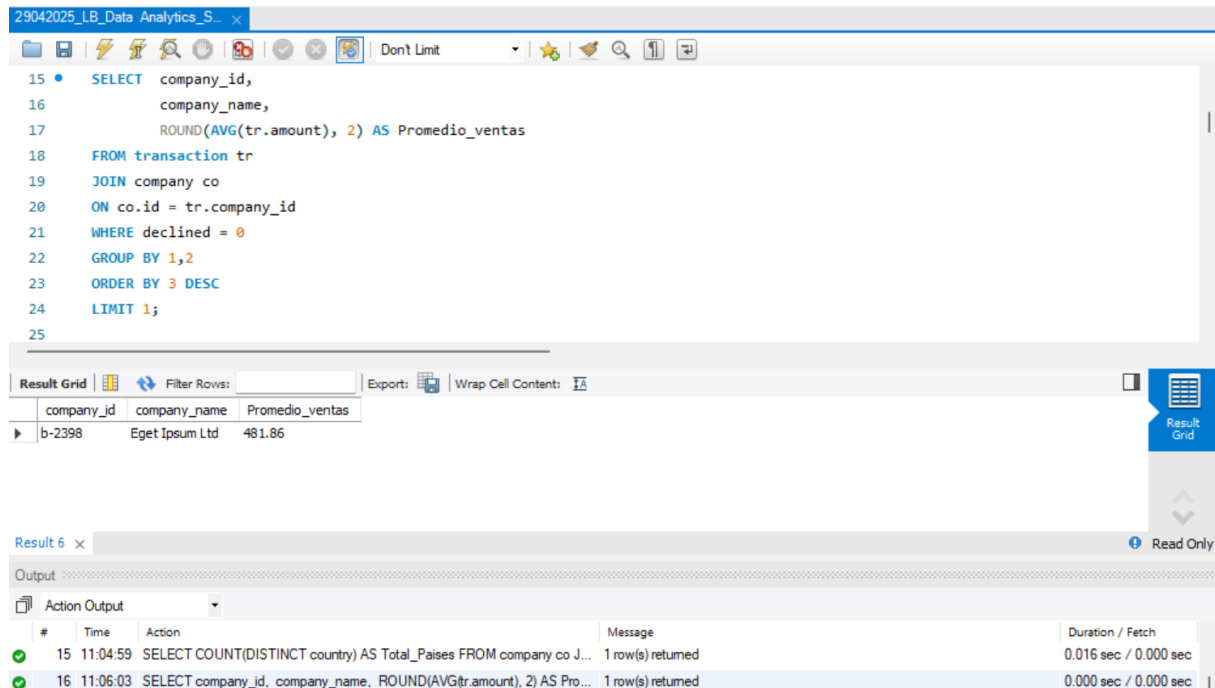
Consulta:

```

SELECT      company_id,
            company_name,
            ROUND(AVG(tr.amount), 2) AS Promedio_ventas
FROM transaction tr
JOIN company co
ON co.id = tr.company_id
WHERE declined = 0
GROUP BY 1,2

```

ORDER BY 3 DESC
LIMIT 1;



The screenshot shows a SQL query editor window titled '29042025_LB_Data Analytics_S...'. The query is as follows:

```

15 SELECT company_id,
16        company_name,
17        ROUND(AVG(tr.amount), 2) AS Promedio_ventas
18 FROM transaction tr
19 JOIN company co
20 ON co.id = tr.company_id
21 WHERE declined = 0
22 GROUP BY 1,2
23 ORDER BY 3 DESC
24 LIMIT 1;
25

```

Below the query, the 'Result Grid' shows the following data:

company_id	company_name	Promedio_ventas
b-2398	Eget Ipsum Ltd	481.86

At the bottom, the 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
15	11:04:59	SELECT COUNT(DISTINCT country) AS Total_Paises FROM company co J...	1 row(s) returned	0.016 sec / 0.000 sec
16	11:06:03	SELECT company_id, company_name, ROUND(AVG(tr.amount), 2) AS Pro...	1 row(s) returned	0.000 sec / 0.000 sec

Como resultado de la consulta la empresa Eget Ipsum Ltd (b-2398) tiene el mayor promedio de ventas de \$ 481.86.

EJERCICIO 03

Utilizando solo subconsultas (sin utilizar JOIN):

- a) Muestra todas las transacciones realizadas por empresas de Alemania.

En esta consulta se utilizó el * para seleccionar todas las columnas de la tabla transaction. Se utilizó la subconsulta para localizar solo las empresas ubicadas en Alemania, para esto se utilizó el filtro **WHERE company_id IN country (campo país) = 'Germany'**.

Consulta:

```

SELECT *
FROM transaction
WHERE company_id IN (
    SELECT id
    FROM company
    WHERE country = 'Germany');

```


The screenshot shows a SQL IDE window titled "2025_LB_Sprint02". The query editor contains the following SQL code:

```

25
26 -- Nivel 01. Ejercicio 03:
27 -- Muestra todas las transacciones realizadas por empresas de Alemania.
28 • SELECT *
29 FROM transaction
30 WHERE company_id IN (
31     SELECT id
32     FROM company
33     WHERE country = 'Germany');
34

```

Below the query editor is the "Result Grid" showing the results of the query. The table has 10 columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The results are as follows:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B1D1D-5B23-A76C-55EF-C568E49A05DD	CcU-2938	b-2222	275	83.7839	-178.86	2021-07-07 17:43:16	293.57	0
EA2C3281-C9C1-A387-44F8-729FB4B51C76	CcU-2938	b-2222	275	20.2004	-116.84	2021-05-09 10:25:08	119.36	1
0DD2E608-5C9E-D1B3-4999-B99F43AD735A	CcU-2959	b-2234	275	9.68811	130.282	2021-04-17 05:30:17	252.47	1
AB069F53-965E-A2A8-CE06-CA8C4FD92501	CcU-2959	b-2234	275	1.64819	-158.007	2021-04-15 13:37:18	60.99	0
0466A42E-47CF-8D24-FD01-C0B689713128	CcU-4219	b-2302	170	-43.9695	-117.525	2021-07-26 07:29:18	49.53	0

Below the result grid is the "Output" section, which shows the execution of the query. It includes a table with columns: #, Time, Action, and Message. The output shows two actions:

#	Time	Action	Message
12	12:54:23	SELECT company_id, company_name, ROUND(AVG(tr.amount), 2) AS Pro...	1 row(s) returned
13	13:02:40	SELECT * FROM transaction WHERE company_id IN (SELECT id FROM c...	118 row(s) returned

Como resultado de la consulta, han retornado 118 transacciones realizadas por empresas ubicadas en Alemania.

- b) Lista las empresas que han realizado transacciones por un monto superior al promedio de todas las transacciones.

Para esta consulta, si ha seleccionado los nombres de las empresas (company_name) de la tabla company. Para filtrar solo las empresas que nos interesan, usamos un **WHERE** que busca las empresas cuyo id coincide con los company_id (sin usar join) de transacciones con importes (amount) mayores al promedio global de todas las transacciones. Ese promedio lo calculamos en una subconsulta dentro de otra subconsulta. Finalmente, ordenamos el resultado alfabéticamente con **ORDER BY 1 ASC** para que los nombres de las empresas salgan en orden.

Consulta:

```

SELECT company_name
FROM company
WHERE id IN (
    SELECT company_id
    FROM transaction
    WHERE amount > (
        SELECT AVG(amount)
        FROM transaction))
ORDER BY 1 ASC;

```

The screenshot shows a SQL IDE window titled "2025_LB_Sprint02". The query editor contains the following SQL code:

```

35 -- Lista las empresas que han realizado transacciones por un monto superior al promedio de todas las transacciones.
36 SELECT company_name
37 FROM company
38 WHERE id IN (
39     SELECT company_id
40     FROM transaction
41     WHERE amount > (
42         SELECT AVG(amount)
43         FROM transaction))
44 ORDER BY 1 ASC;

```

The "Result Grid" shows the following data:

company_name
A Institute
Ac Fermentum Incorporated
Ac Industries
Aliquam PC
Aliquet Diam Limited

The "Output" pane shows the execution results:

#	Time	Action	Message	Duration / Fetch
16	13:12:16	SELECT company_name FROM company WHERE id IN (SELECT compan...	Error Code: 1064. You have an error in your SQL syntax; check the manual t...	0.000 sec
17	13:12:19	SELECT company_name FROM company WHERE id IN (SELECT compan...	70 row(s) returned	0.000 sec / 0.000 sec

Como resultado tenemos a 70 empresas que tienen un monto mayor que el promedio de todas las transacciones.

- c) Se eliminarán del sistema las empresas que no tienen transacciones registradas; entrega el listado de estas empresas.

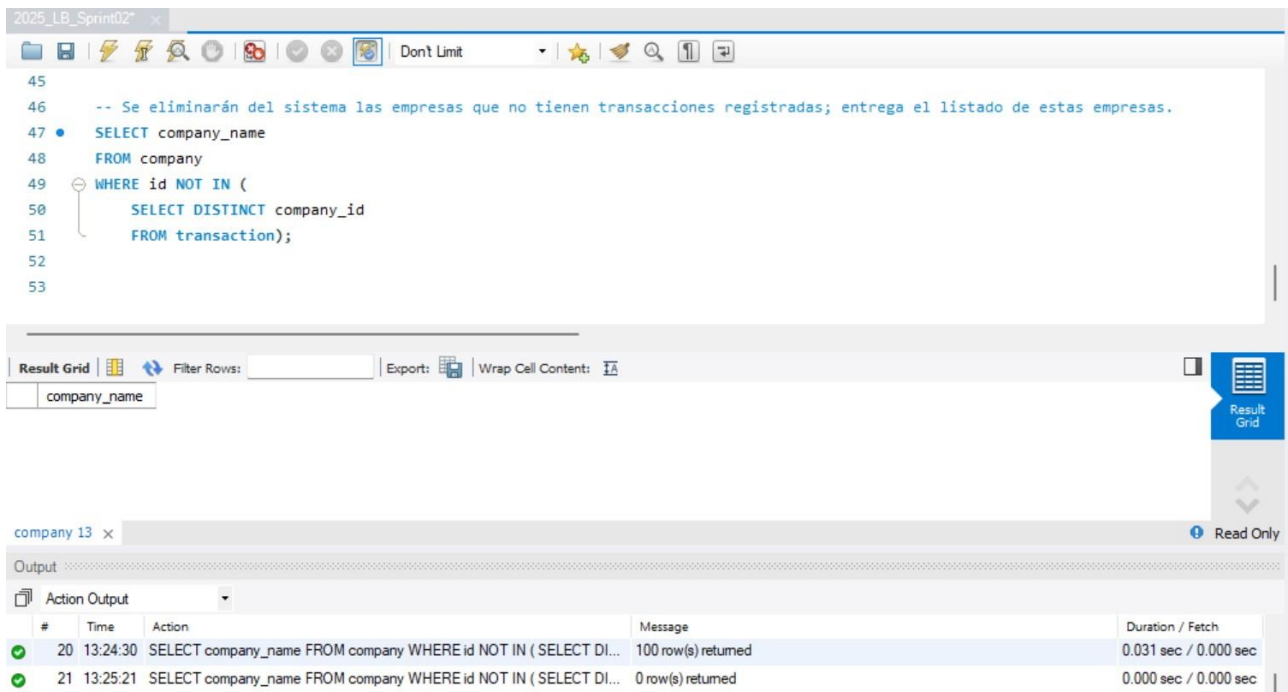
Seleccionamos el nombre de la empresa desde la tabla company y usamos **WHERE id NOT IN** para encontrar las que no tienen transacciones registradas, o sea, las que no aparecen en la tabla transaction. En la subconsulta, usamos **DISTINCT** para evitar que busque el mismo company_id más de una vez si hay duplicados.

Consulta:

```

SELECT company_name
FROM company
WHERE id NOT IN (
    SELECT DISTINCT company_id
    FROM transaction);

```



The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows a SQL query being executed:

```
45
46 -- Se eliminarán del sistema las empresas que no tienen transacciones registradas; entrega el listado de estas empresas.
47 • SELECT company_name
48 FROM company
49 WHERE id NOT IN (
50     SELECT DISTINCT company_id
51     FROM transaction);
52
53
```

The bottom pane shows the 'Result Grid' with a single column header 'company_name'. Below it, the 'Output' pane shows the 'Action Output' table:

#	Time	Action	Message	Duration / Fetch
20	13:24:30	SELECT company_name FROM company WHERE id NOT IN (SELECT DI...	100 row(s) returned	0.031 sec / 0.000 sec
21	13:25:21	SELECT company_name FROM company WHERE id NOT IN (SELECT DI...	0 row(s) returned	0.000 sec / 0.000 sec

No ha retornado ninguna empresa, en este caso, si puede entender que todas las empresas tienen como mínimo un registro en la base de datos.

NIVEL 2

EJERCICIO 01

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

En esta consulta, seleccionamos la variable de **DATE(timestamp)**, que contiene las fechas de las transacciones, para saber el total de ventas se utilizó el **SUM(amount)** y para asegurarse que todas las ventas fueron transacciones hechas o finalizadas, aplicamos el filtro **WHERE decline = 0**. Al final, fue agrupado por la fecha (posición en la consulta 1) y **ORDER BY 2**, ordenando por el Total_Ventas (posición en la consulta 2). Por fin, para presentar solo los 5 días con mayores ingresos, se ha delimitado utilizando el **LIMIT 5**.

```
Consulta:      SELECT      DATE(timestamp) AS Fecha,
                      SUM(amount) AS Total_Ventas
FROM transaction
WHERE declined = 0
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```

The screenshot shows a SQL IDE window titled "2025_LB_Sprint02". The query editor contains the following SQL code:

```
-- Identifica los cinco días en los que se generó la mayor cantidad de ingresos para la empresa por ventas.
-- Muestra la fecha de cada transacción junto con el total de las ventas.
SELECT DATE(timestamp) AS Fecha,
       SUM(amount) AS Total_Ventas
FROM transaction
WHERE declined = 0
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```

Below the query editor, the "Result Grid" shows the following data:

Fecha	Total_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

At the bottom, the "Output" pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
26	13:27:55	SELECT DISTINCT company_name FROM transaction tr LEFT JOIN compa...	100 row(s) returned	0.000 sec / 0.000 sec
27	13:35:01	SELECT DATE(timestamp) AS Fecha, SUM(amount) AS Total_Ventas FRO...	5 row(s) returned	0.016 sec / 0.000 sec

EJERCICIO 02

¿Cuál es el promedio de ventas por país? Presente los resultados ordenados de mayor a menor promedio.

Para esta consulta, seleccionamos el country de la tabla company, para presentar el promedio por nombre de país, usamos el **AVG** para calcular el promedio de ventas de tabla transaction, usando el **ROUND 2**, para redondear el resultado a dos casas decimales. Para realizar esta consulta fue necesario hacer el JOIN entre las tablas. Por último, agrupamos por país (posición 1 de la consulta) y ordenamos **ORDER BY 2 DESC**, por el **Total_Ventas** (posición 2 de la consulta) de forma descendente para presentar del mayor al menor valor.

Consulta:

```
SELECT      country AS Países,
            ROUND(AVG(amount), 2) AS Total_Ventas

FROM transaction tr
JOIN company co
ON co.id = tr.company_id
WHERE declined = 0
GROUP BY 1
ORDER BY 2 DESC;
```

2025_LB_Sprint02

```
-- ¿Cuál es el promedio de ventas por país? Presente los resultados ordenados de mayor a menor promedio.
65 SELECT country AS Países,
66      ROUND(AVG(amount), 2) AS Total_Ventas
67
68 FROM transaction tr
69 JOIN company co
70 ON co.id = tr.company_id
71 WHERE declined = 0
72 GROUP BY 1
73 ORDER BY 2 DESC;
```

Result Grid

Países	Total_Ventas
United States	287.53
Ireland	285.83
Sweden	276.67
United Kingdom	271.77
Canada	261.94
Belgium	255.22

Result 19

Output

#	Time	Action	Message	Duration / Fetch
28	14:49:26	SELECT country AS Países, ROUND(AVG(amount), 2) AS Total_Ventas FR...	Error Code: 1054. Unknown column 'country' in field list	0.000 sec
29	14:49:57	SELECT country AS Países, ROUND(AVG(amount), 2) AS Total_Ventas FR...	15 row(s) returned	0.000 sec / 0.000 sec

Como resultado si presentan el total de ventas de los 15 países.

EJERCICIO 03

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.

a) Mostrar el listado aplicando JOIN y subconsultas:

Primero, seleccionamos todas las columnas de las tablas transaction y company, juntándolas con un **JOIN** para conectar los datos a través del company_id. Luego, filtramos las transacciones de empresas que están en el mismo país que la empresa llamada 'Non Institute', usando una subconsulta para obtener ese país. Finalmente, excluimos a la propia 'Non Institute' del resultado con **AND NOT company_name = 'Non Institute'**. Así nos aseguramos de traer solo las transacciones de otras empresas del mismo país.

Consulta:

```
SELECT *
FROM transaction tr
JOIN company co
ON co.id = tr.company_id
WHERE co.country = (
    SELECT country
    FROM company
    WHERE company_name = 'Non Institute')
AND NOT company_name = 'Non Institute';
```

The screenshot shows a SQL IDE window titled '2025_LB_Sprint02'. The query editor contains the SQL query from the previous block. Below the editor, the 'Result Grid' shows the results of the query. The results are displayed in a table with columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined, id, and company. The first few rows are visible, showing transactions for companies like 'Sed Nunc' and 'Non Magr'. Below the result grid, the 'Output' pane shows the execution log, indicating that the query returned 100 rows in 0.000 seconds.

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined	id	company
2B928E1C-EC14-A760-0A75-871477649D6A	CcU-2980	b-2246	275	-41.0496	161.685	2021-08-10 08:14:49	383.73	0	b-2246	Sed Nunc
ACD2011A-A2B1-C365-41E1-2AB00C65147A	CcU-2980	b-2246	275	-54.4792	-82.7974	2022-03-05 20:41:20	60.07	1	b-2246	Sed Nunc
4334349E-CEB0-3D68-A4D4-FEB7718A1ACE	CcU-3092	b-2310	275	-20.4859	150.87	2021-05-03 22:37:23	458.74	0	b-2310	Non Magr
BC2B9A38-77B4-28CD-1FE8-14DED863E773	CcU-3092	b-2310	275	-78.0295	18.5295	2021-10-18 07:27:35	477.95	1	b-2310	Non Magr
1479B3D2-B76A-C7BB-4CE3-8D7C2DE85ABB	CcU-2994	b-2326	133	66.2672	172.399	2021-08-09 00:58:07	309.45	0	b-2326	Enim Con

Llegamos a un resultado de 70 empresas en el mismo país de la empresa Non Institute.

b) Muestra el listado utilizando solo subconsultas:

En este caso es el mismo que la consulta anterior, pero sin utilizar el **JOIN**, utilizando dos subconsultas en el **WHERE** utilizando el operador **IN**.

Consulta:

```
SELECT *
FROM transaction
WHERE company_id IN (
    SELECT id
    FROM company
    WHERE country = (
        SELECT country
        FROM company
        WHERE company_name = "Non Institute")
    AND NOT company_name = "Non Institute");
```

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

```
SELECT *
FROM transaction
WHERE company_id IN (
    SELECT id
    FROM company
    WHERE country = (
        SELECT country
        FROM company
        WHERE company_name = "Non Institute")
    AND NOT company_name = "Non Institute");
```

The result grid displays the following data:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
2B928E1C-EC14-A760-0A75-871477649D6A	CcU-2980	b-2246	275	-41.0496	161.685	2021-08-10 08:14:49	383.73	0
ACD2011A-A2B1-C365-41E1-2AB00C65147A	CcU-2980	b-2246	275	-54.4792	-82.7974	2022-03-05 20:41:20	60.07	1
4334349E-CEB0-3D68-A4D4-FEB7718A1ACE	CcU-3092	b-2310	275	-20.4859	150.87	2021-05-03 22:37:23	458.74	0
BC2B9A38-77B4-28CD-1FE8-14DED863E773	CcU-3092	b-2310	275	-78.0295	18.5295	2021-10-18 07:27:35	477.95	1
147983D2-87BA-C7BB-4CE3-8D7C2DE85ABB	CcU-2994	b-2326	133	66.2672	172.399	2021-08-09 00:58:07	309.45	0

The output section shows the following action output:

#	Time	Action	Message	Duration / Fetch
38	15:21:38	SELECT * FROM transaction WHERE company_id IN (SELECT id FRO...	70 row(s) returned	0.000 sec / 0.000 sec
39	15:22:10	SELECT * FROM transaction WHERE company_id IN (SELECT id FRO...	70 row(s) returned	0.000 sec / 0.000 sec

Como en el ejercicio anterior, llegamos a las 70 empresas en el mismo país de la empresa Non Institute.

NIVEL 3

EJERCICIO 01

Presenta el nombre, teléfono, país, fecha y *amount* (cantidad) 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.

Para esta consulta, seleccionamos las variables pedidas: nombre, teléfono, país, fecha y *amount*, inicialmente hacemos el **JOIN** entre las dos tablas y mediante el filtro **WHERE** buscamos los montos entre 100 y 200 (**BETWEEN 100 AND 200**) y las fechas delimitadas para la consulta (**DATE(tr.timestamp) IN ('2021-04-29', '2021-07-20', '2022-03-13')**). Por último, ordenamos el resultado conforme el valor más alto.

```
Consulta:  SELECT co.company_name,
            co.phone,
            co.country,
            DATE(tr.timestamp) AS date,
            tr.amount
FROM transaction tr
JOIN company co
ON tr.company_id = co.id
WHERE tr.amount BETWEEN 100 AND 200
AND DATE(tr.timestamp) IN ('2021-04-29', '2021-07-20', '2022-03-13')
ORDER BY 4 DESC;
```

The screenshot shows a SQL IDE window titled '2025_LB_Sprint02'. The query editor contains the following SQL code:

```

103 -- 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.
104 -- Ordena los resultados de mayor a menor cantidad.
105 • SELECT co.company_name, co.phone, co.country, DATE(tr.timestamp) AS date, tr.amount
106 FROM transaction tr
107 JOIN company co
108 ON tr.company_id = co.id
109 WHERE tr.amount BETWEEN 100 AND 200
110 AND DATE(tr.timestamp) IN ('2021-04-29', '2021-07-20', '2022-03-13')
111 ORDER BY 4 DESC;

```

Below the query editor, the 'Result Grid' shows 5 rows of data:

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

The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
41	15:28:38	SELECT co.company_name, co.phone, co.country, DATE(tr.timestamp) AS date, tr.amount	5 row(s) returned	0.000 sec / 0.000 sec
42	15:29:38	SELECT co.company_name, co.phone, co.country, DATE(tr.timestamp) AS date, tr.amount	5 row(s) returned	0.000 sec / 0.000 sec

Como resultado tenemos las 5 filas con los datos de cada empresa, de acuerdo con las fechas determinadas.

EJERCICIO 02

Necesitamos optimizar la asignación de los recursos, lo cual dependerá de la capacidad operativa que se requiera. Por ello, te piden información sobre la cantidad de transacciones que realizan las empresas. Sin embargo, el departamento de recursos humanos es exigente y quiere un listado de las empresas donde especifiques si tienen más de 4 transacciones o menos.

Para esta consulta, utilizamos el **CASE WHEN**, para crear una nueva columna que muestre el resultado del cálculo.

Consulta:

```

SELECT co.company_name,
       CASE WHEN COUNT(tr.id) > 4 THEN 'More than 4'
            WHEN COUNT(tr.id) <= 4 THEN 'less than 4'
            END AS 'Número de transacciones'
FROM transaction tr
JOIN company co
ON tr.company_id = co.id
GROUP BY 1;

```

2025_LB_Sprint02

```

116 -- Sin embargo, el departamento de recursos humanos es exigente y quiere un listado de las empresas donde especifiques
117 -- si tienen más de 4 transacciones o menos.
118 • SELECT co.company_name,
119     CASE WHEN COUNT(tr.id) > 4 THEN 'More than 4'
120          WHEN COUNT(tr.id) <= 4 THEN 'less than 4'
121          END AS 'Número de transacciones'
122 FROM transaction tr
123 JOIN company co
124 ON tr.company_id = co.id
125 GROUP BY 1;

```

Result Grid

company_name	Número de transacciones
Ac Fermentum Incorporated	less than 4
Magna A Neque Industries	less than 4
Fusce Corp.	less than 4
Convallis In Incorporated	less than 4
Ante Iaculis Nec Foundation	less than 4

Result 31

Output

Action Output

#	Time	Action	Message	Duration / Fetch
43	15:38:06	SELECT co.company_name, CASE WHEN COUNT(tr.id) > 4 THEN 'More th...	100 row(s) returned	0.000 sec / 0.000 sec

En este caso, como resultado tenemos una nueva columna, que podemos ordenar el número de transacciones hechas por más o menos de 4.