

Sprint 2: Bases de dades relacionals i introducció a SQL

Curso de especialización Data Analytics - IT Academy - Barcelona Activa

Nicolás Aros Marzá

12/12/2025

Nivel 1

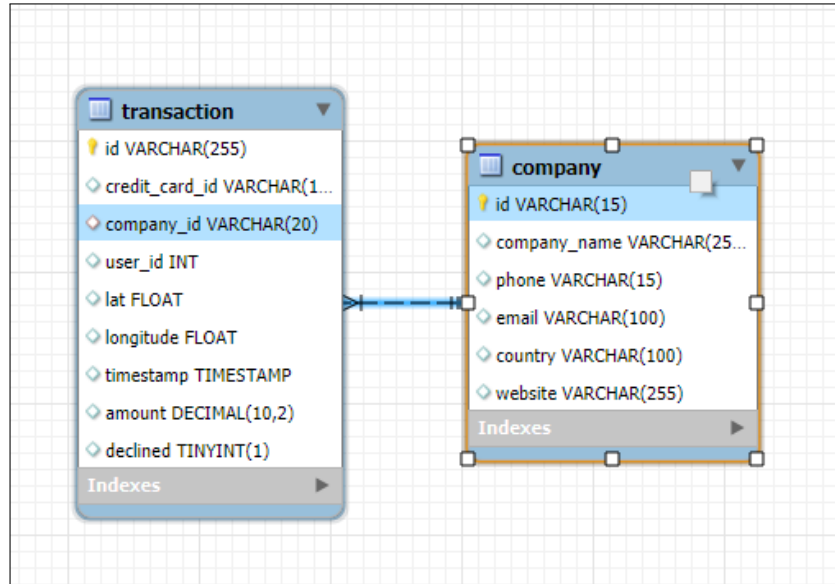
Ejercicio 1

A partir de los documentos adjuntos (estructura_dades y dades_introduir), 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.

El esquema “*transactions*” está compuesto por dos tablas: “*company*” y “*transaction*”, las cuales están relacionadas mediante una variable que identifica a cada empresa. En la tabla “*company*” corresponde a la variable `id` que es la *primary key*, puesto que identifica de forma única a cada fila, es decir a cada compañía. Por su parte, en la tabla “*transaction*” esta variable se llama `company_id`. La relación entre estas tablas es del tipo 1:N, puesto que una fila de la tabla “*company*” puede estar relacionado con muchas filas de la tabla “*transaction*”, pero cada transacción solo se relaciona con una fila de la tabla de empresas.

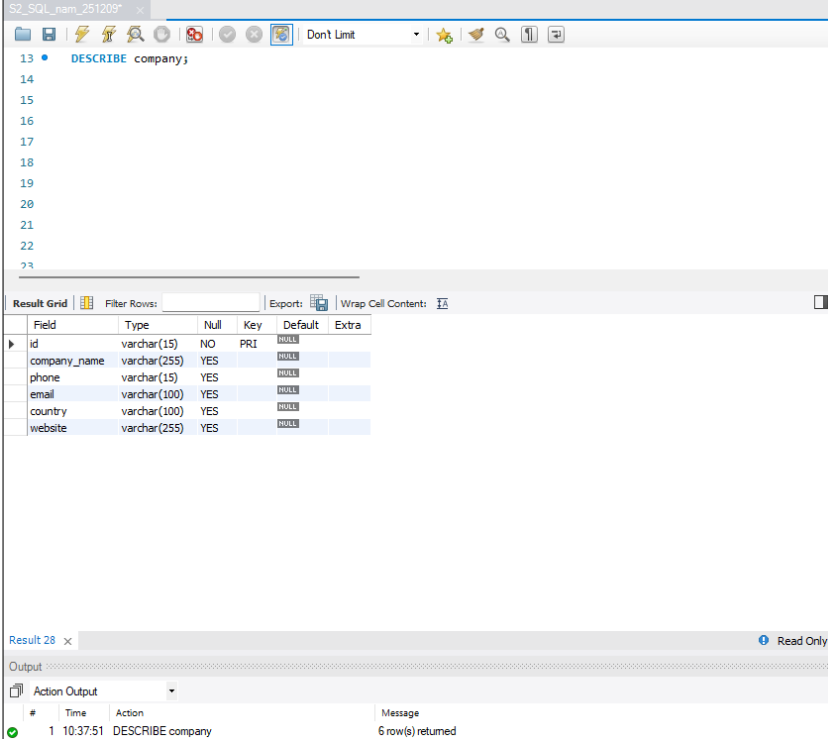
El diagrama de la relación entre las tablas de este esquema es el siguiente:

Diagrama de relación entre las tablas



Las consultas para conocer las características de cada una de las tablas son las siguientes:

Características de la tabla "company"



Result Grid

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
company_name	varchar(255)	YES		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(100)	YES		NULL	
country	varchar(100)	YES		NULL	
website	varchar(255)	YES		NULL	

Result 28 x Read Only

Output

Action Output

#	Time	Action	Message
1	10:37:51	DESCRIBE company	6 row(s) returned

En la tabla "company" se encuentran información básica sobre cada empresa. La variable `id` contiene un código único de identificación; la variable `company_name` indica el nombre de la empresa; las variables `phone`, `email` y `website` corresponden a la información de contacto; mientras que `country` indica el país en que se encuentra la compañía. Todas estas variables son de tipo carácter.

Exploración de la tabla "company"

15 • `SELECT * FROM company;`

16

17

18

19

20

21

22

23

24

25

Result Grid

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.co.uk	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.co.uk	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
b-2262	Gravida Sagittis LLP	03 81 28 33 97	turpis.vitae@google.ca	Sweden	https://naver.com/site
b-2266	Mus Aenean Eget Foundation	06 25 15 52 43	mi.duis@hotmail.net	Sweden	https://instagram.com/group/9
b-2270	Dis Parturient Institute	05 36 29 78 74	purus@protonmail.org	Ireland	https://google.com/one
b-2274	Sed LLC	01 63 16 26 52	at@outlook.com	Belgium	https://reddit.com/fr
b-2278	Arcu LLP	06 46 04 41 45	dui@aol.ca	Norway	https://yahoo.com/sub
b-2282	Pretium Neque Corp.	07 77 48 55 28	eleifend.nec.malesuada@proton...	Australia	https://netflix.com/sub
b-2286	Fringilla LLC	08 29 15 93 57	gravida@protonmail.co.uk	New Zealand	https://reddit.com/user/110
b-2290	Quisque Libero LLC	01 45 48 71 11	cras.molestie.nisi@hotmail.co.uk	China	https://facebook.com/ironman10

company 29 x

Output

Action Output

#	Time	Action	Message
1	10:37:51	DESCRIBE company	6 row(s) returned
2	10:38:47	SELECT * FROM company	100 row(s) returned

La tabla "company" está compuesta por 100 observaciones y ninguna columna tiene valores perdidos.

Características de la tabla "transaction"

16 • DESCRIBE transaction;
17
18
19
20
21
22
23
24
25
26

Field	Type	Null	Key	Default	Extra
id	varchar(255)	NO	PRI		
credit_card_id	varchar(15)	YES			
company_id	varchar(20)	YES	MUL		
user_id	int	YES			
lat	float	YES			
longitude	float	YES			
timestamp	timestamp	YES			
amount	decimal(10,2)	YES			
declined	tinyint(1)	YES			

Result 30 × Read Only

Output

#	Time	Action	Message
1	10:37:51	DESCRIBE company	6 row(s) returned
2	10:38:47	SELECT * FROM company	100 row(s) returned
3	10:39:59	DESCRIBE transaction	9 row(s) returned

En la tabla “*transaction*” cada fila corresponde a una transacción. Cada transacción tiene un valor único en la variable `id`. La variable `credit_card_id` registra la tarjeta con la cual se realizó el pago, en `company_id` se indica la compañía que realizó la transacción y `user_id` es una variable numérica que corresponde a un identificador del usuario. La variable `amount` muestra el monto de la transacción y corresponde a una variable decimal, mientras que la variable `declined` es una variable lógica dicotómica (*tinyint*) que indica si la transacción ha sido rechazada. Por último, se encuentran dos variables que muestran las coordenadas del lugar en donde se realizó la transacción (`lat` y `longitude`) y una variable que informa sobre el momento en el que se realizó `timestamp`.

Exploración de la tabla "transaction"

17 • `SELECT * FROM transaction;`

18

19

20

21

22

23

24

25

26

27

Result Grid

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
00043A49-2949-494B-A5DD-A58AE38B19D0	CcS-9294	b-2458	4713	46.1999	1.43554	2024-08-28 07:16:46	395.43	0
000447FE-B650-4DCF-85DE-C7ED0EE1CAAD	CcS-5019	b-2370	438	41.5972	12.2218	2016-12-21 20:07:18	155.63	0
00045D68-ED2E-4F2F-8186-CEE074D875D0	CcS-6699	b-2390	2118	29.7573	-95.3796	2020-07-14 15:37:45	326.01	0
000481C3-1C26-4FEF-83A0-4CD0EB0048BD	CcS-6696	b-2230	2115	53.5489	-113.503	2017-09-04 19:44:53	161.60	0
00051AA4-9CBE-4268-B070-C38062A183E2	CcS-7606	b-2266	3025	52.2084	5.69081	2017-01-05 18:19:25	148.91	0
0008A312-EDFE-4A4F-BC99-E9C92EC3CA4D	CdJ-3358	b-2598	215	53.5535	-113.499	2023-09-23 04:51:43	294.59	0
0009A151-9BCE-4E31-9053-A468FF77FAAB	CcS-7509	b-2546	2928	51.9362	5.34265	2023-12-31 00:06:36	383.63	0
0009D494-6245-4DF9-955D-2C084191CFFB	CcS-8483	b-2526	3902	45.492	-73.5706	2017-07-18 07:52:02	197.80	0
000A1DEC-CD66-4AB2-A619-71DAB8D4A262	CcS-6467	b-2558	1886	55.7425	-3.30009	2018-09-08 05:29:58	339.94	0
000A1E64-1414-4080-9D92-5678A4D958E2	CcS-5966	b-2550	1385	52.0821	5.28424	2022-09-17 04:02:19	369.71	0
000A5879-3472-41D9-AF60-42D35038543C	CdJ-4569	b-2590	42	39.949	-75.1719	2020-02-07 23:03:45	162.43	0
000AE0D4-F06E-4146-804A-5A2FC73110D7	CcS-8134	b-2426	3553	46.8055	-71.2149	2017-07-10 07:11:41	188.94	0
000BCBF8-A4AB-4E8E-A148-2E7A04E1B385	CcS-6020	b-2598	1439	33.4472	-112.072	2019-05-28 04:23:02	96.22	0
000BD852-E25B-4785-93D9-27051425D458	CcS-5445	b-2330	864	51.4317	5.47489	2024-07-31 09:45:39	92.13	0
000DAABF-ABEA-4F32-A090-89CC07E5D9E9	CcS-6404	b-2454	1823	41.7392	12.6293	2023-07-26 22:39:50	298.18	0
000E44B7-43B4-443D-88B2-ABAD4803D081	CcS-7604	b-2458	3023	54.9751	-3.40465	2018-08-01 11:03:20	171.13	0
000E64A2-2C13-4306-8B25-B292ED79413D	CcS-6737	b-2386	2156	45.4992	-73.5719	2018-05-30 16:09:08	105.03	0
000E668B-4747-47C7-4B79-17A9F1E0737E	CcS-6606	b-2507	1515	30.4074	-8.4635	2010-11-28 07:13:00	730.04	0

transaction 31 x

Output

Action Output

#	Time	Action	Message
1	10:37:51	DESCRIBE company	6 row(s) returned
2	10:38:47	SELECT * FROM company	100 row(s) returned
3	10:39:59	DESCRIBE transaction	9 row(s) returned
4	10:40:37	SELECT * FROM transaction	100000 row(s) returned

La tabla "transaction" está compuesta por 100.000 observaciones y ninguna columna tiene valores perdidos.

Ejercicio 2

Utilizando JOIN realizarás las siguientes consultas:

Listado de los países que están generando ventas.

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

```
22 • SELECT DISTINCT country AS lista_paises
23 FROM transaction
24 INNER JOIN company ON company.id = transaction.company_id
25 WHERE declined = 0;
26
27
28
29
30
31
32
```

The results pane displays a list of countries under the heading 'lista_paises':

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

Below the results, the 'Output' pane shows the execution log:

#	Time	Action	Message
1	10:37:51	DESCRIBE company	6 row(s) returned
2	10:38:47	SELECT * FROM company	100 row(s) returned
3	10:39:59	DESCRIBE transaction	9 row(s) returned
4	10:40:37	SELECT * FROM transaction	100000 row(s) returned
5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned

En esta consulta se utiliza el filtro `declined = 0` para distinguir las ventas de las transacciones. Una venta corresponde a una transacción que no ha sido declinada.

Desde cuántos países se están generando ventas.

Número de países que generan ventas.

```
28 • SELECT COUNT(DISTINCT country) AS num_paises
29 FROM transaction
30 INNER JOIN company ON company.id = transaction.company_id
31 WHERE declined = 0;
32
33
34
35
36
37
38
```

Result Grid

num_paises
15

Result 33 x Read Only

Output

#	Time	Action	Message
✓ 1	10:37:51	DESCRIBE company	6 row(s) returned
✓ 2	10:38:47	SELECT * FROM company	100 row(s) returned
✓ 3	10:39:59	DESCRIBE transaction	9 row(s) returned
✓ 4	10:40:37	SELECT * FROM transaction	100000 row(s) returned
✓ 5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned
✓ 6	10:43:54	SELECT COUNT(DISTINCT country) AS num_paise...	1 row(s) returned

En esta consulta también se filtran solamente las ventas realizadas mediante la condición `declined = 0`.

Identifica la compañía con la media más grande de ventas.

Compañía con la media más grande de ventas

```
34 • SELECT company.id, company_name, ROUND(AVG(amount), 2) AS promedio_ventas
35 FROM transaction
36 INNER JOIN company ON company.id = transaction.company_id
37 WHERE declined = 0
38 GROUP BY company.id, company_name
39 ORDER BY promedio_ventas DESC
40 LIMIT 1;
41
42
43
44
```

Result Grid | Filter Rows: | Exports: | Wrap Cell Content: [IA](#)

id	company_name	promedio_ventas
b-2222	Ac Fermentum Incorporated	284.91

Result 34 x Read Only

Output

#	Time	Action	Message
✓ 1	10:37:51	DESCRIBE company	6 row(s) returned
✓ 2	10:38:47	SELECT * FROM company	100 row(s) returned
✓ 3	10:39:59	DESCRIBE transaction	9 row(s) returned
✓ 4	10:40:37	SELECT * FROM transaction	100000 row(s) returned
✓ 5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned
✓ 6	10:43:54	SELECT COUNT(DISTINCT country) AS num_paise...	1 row(s) returned
✓ 7	10:44:50	SELECT company.id, company_name, ROUND(AVG...	1 row(s) returned

Siguiendo las recomendaciones del documento de buenas prácticas, el cálculo de los promedios se realiza en conjunto de la función `ROUND()` para delimitar la cantidad de decimales. En este caso se solicitó aproximar a 2 dígitos.

Ejercicio 3

Utilizando solamente subconsultas (sin utilizar JOIN):

Muestra todas las transacciones realizadas por empresas alemanas.

Transacciones realizadas por empresas alemanas.

The screenshot shows a SQL IDE interface. At the top, a query is written in a text editor:

```
45 SELECT
46 FROM transaction
47 WHERE EXISTS (
48     SELECT id
49     FROM company
50     WHERE country = 'Germany'
51     AND company.id = transaction.company_id);
52
53
54
55
```

Below the editor is a 'Result Grid' showing a table of transaction data. The table has 10 columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. It contains 35 rows of data, each representing a transaction by a German company.

At the bottom, an 'Output' pane shows the execution log. It lists 8 actions performed by the IDE, including SELECT, DESCRIBE, and the final query execution, along with the number of rows returned for each step.

#	Time	Action	Message
2	10:38:47	SELECT * FROM company	100 row(s) returned
3	10:39:59	DESCRIBE transaction	9 row(s) returned
4	10:40:37	SELECT * FROM transaction	100000 row(s) returned
5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned
6	10:43:54	SELECT COUNT(DISTINCT country) AS num_paise...	1 row(s) returned
7	10:44:50	SELECT company.id, company_name, ROUND(AVG...	1 row(s) returned
8	10:45:45	SELECT * FROM transaction WHERE EXISTS (...	13291 row(s) returned

La subconsulta de esta consulta se realiza mediante la combinación de las funciones **WHERE** y **EXISTS**. Si bien, esta operación también se podría realizar con la función **IN**, se decide por **EXISTS** porque es un método más eficiente.

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

Lista de empresas

```

54 • SELECT id, company_name
55 FROM company
56 WHERE EXISTS (
57     SELECT company_id
58     FROM transaction
59     WHERE amount > (
60         SELECT AVG(amount)
61         FROM transaction
62     ) AND company.id = transaction.company_id;
63
64

```

id	company_name
b-2222	Ac Fermentum Incorporated
b-2226	Magna A Neque Industries
b-2230	Fusce Corp.
b-2234	Convalles In Incorporated
b-2238	Ante Jaculis Nec Foundation
b-2242	Donec Ltd
b-2246	Sed Nunc Ltd
b-2250	Amet Nulla Donec Corporation
b-2254	Nascetur Ridiculus Mus Inc.
b-2258	Vestibulum Lorem PC
b-2262	Gravida Sagittis LLP
b-2266	Mus Aenean Eget Foundation
b-2270	Dis Parturient Institute
b-2274	Sed LLC
b-2278	Arcu LLP
b-2282	Pretium Neque Corp.
b-2286	Fringilla LLC
b-2290	Chorisis Libero LLC

#	Time	Action	Message
3	10:39:59	DESCRIBE transaction	9 row(s) returned
4	10:40:37	SELECT * FROM transaction	100000 row(s) returned
5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned
6	10:43:54	SELECT COUNT(DISTINCT country) AS num_paise...	1 row(s) returned
7	10:44:50	SELECT company_id, company_name, ROUND(AVG...	1 row(s) returned
8	10:45:45	SELECT * FROM transaction WHERE EXISTS (...	13291 row(s) returned
9	10:47:30	SELECT id, company_name FROM company WHER...	100 row(s) returned

Esta consulta se compone de tres subconsultas distintas. En la primera, la que se encuentra al final del código, se compara el monto de cada transacción con el promedio de todas las transacciones. Se seleccionan solo aquellas transacciones que se encuentran sobre el promedio. Mientras que en la segunda se seleccionan todas las compañías que realizaron alguna transacción que cumple este criterio.

El listado muestra 100 empresas, es decir, durante el período observado, todas las compañías realizaron al menos una transacción por un monto sobre el promedio.

Eliminarán del sistema las empresas que no tienen transacciones registradas. Entrega el listado de estas empresas.

Listado de empresas eliminadas.

The screenshot shows a database management interface. At the top, a SQL query is entered in a text area:

```

65 SELECT id, company_name
66 FROM company
67 WHERE NOT EXISTS (
68     SELECT company_id
69     FROM transaction);
70
71
72
73
74
75

```

Below the query editor, there is a "Result Grid" section. It contains a table with two columns: "id" and "company_name". The table is currently empty, with only the column headers visible.

At the bottom of the interface, there is an "Output" section. It displays a log of database actions and their results:

#	Time	Action	Message
4	10:40:37	SELECT * FROM transaction	100000 row(s) returned
5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned
6	10:43:54	SELECT COUNT(DISTINCT country) AS num_paise...	1 row(s) returned
7	10:44:50	SELECT company_id, company_name, ROUND(AVG...	1 row(s) returned
8	10:45:45	SELECT * FROM transaction WHERE EXISTS (...	13291 row(s) returned
9	10:47:30	SELECT id, company_name FROM company WHER...	100 row(s) returned
10	10:48:28	SELECT id, company_name FROM company WHER...	0 row(s) returned

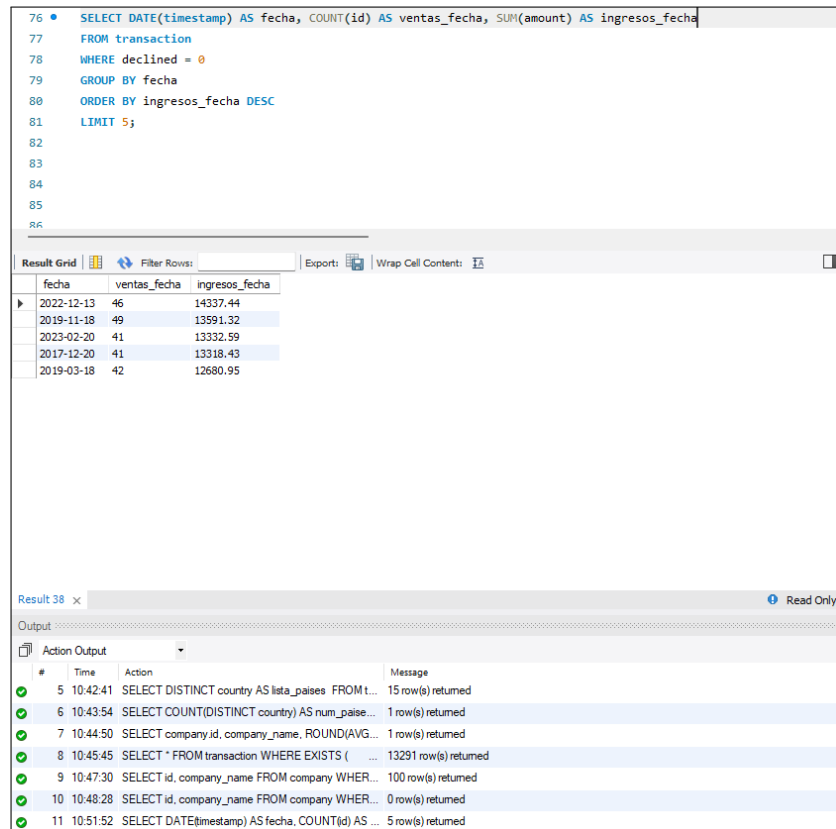
En esta consulta se utiliza la combinación de las funciones WHERE y EXISTS junto con el auxiliar NOT. Con esto, se seleccionan los casos “huérfanos” de la tabla “company”, es decir, aquellas compañías que no tienen ningún registro en la tabla “transactions”. El resultado es una tabla vacía, puesto que no existe ninguna empresa que no haya realizado al menos una transacción en el período observado.

Nivel 2

Ejercicio 1

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

Países que generan más ventas



```
76 • SELECT DATE(timestamp) AS fecha, COUNT(id) AS ventas_fecha, SUM(amount) AS ingresos_fecha
77 FROM transaction
78 WHERE declined = 0
79 GROUP BY fecha
80 ORDER BY ingresos_fecha DESC
81 LIMIT 5;
82
83
84
85
86
```

fecha	ventas_fecha	ingresos_fecha
2022-12-13	46	14337.44
2019-11-18	49	13591.32
2023-02-20	41	13332.59
2017-12-20	41	13318.43
2019-03-18	42	12680.95

Result 38 x Read Only

#	Time	Action	Message
✓ 5	10:42:41	SELECT DISTINCT country AS lista_paises FROM t...	15 row(s) returned
✓ 6	10:43:54	SELECT COUNT(DISTINCT country) AS num_paise...	1 row(s) returned
✓ 7	10:44:50	SELECT company.id, company_name, ROUND(AVG...	1 row(s) returned
✓ 8	10:45:45	SELECT * FROM transaction WHERE EXISTS (...	13291 row(s) returned
✓ 9	10:47:30	SELECT id, company_name FROM company WHER...	100 row(s) returned
✓ 10	10:48:28	SELECT id, company_name FROM company WHER...	0 row(s) returned
✓ 11	10:51:52	SELECT DATE(timestamp) AS fecha, COUNT(id) AS ...	5 row(s) returned

En esta consulta se utilizan las herramientas que ya fueron utilizadas anteriormente. La única novedad es la incorporación de la función `DATE()` que permite transformar una variable de tipo `timestamp` que indica la fecha y el momento del día en que se realizó la transacción, en una variable que solo indique la fecha. Con esta transformación es posible agrupar las ventas realizadas en un mismo día.

Ejercicio 2

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

Mediana de ventas por país.

```
86 • SELECT country, ROUND(AVG(amount), 2) AS promedio_ventas
87 FROM transaction
88 INNER JOIN company ON company.id = transaction.company_id
89 WHERE declined = 0
90 GROUP BY country
91 ORDER BY promedio_ventas DESC;
92
93
94
95
96
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [↗](#)

country	promedio_ventas
Australia	265.54
United States	264.42
Belgium	260.97
Germany	260.83
Ireland	260.39
Spain	260.28
France	259.91
New Zealand	259.59
Norway	259.14
Netherlands	258.34
Italy	258.12
Canada	257.41
Sweden	257.39
United Kingdom	256.68
China	252.60

Result 40 × [Read Only](#)

Output

#	Time	Action	Message
7	10:44:50	SELECT company.id, company_name, ROUND(AVG...	1 row(s) returned
8	10:45:45	SELECT * FROM transaction WHERE EXISTS (...	13291 row(s) returned
9	10:47:30	SELECT id, company_name FROM company WHER...	100 row(s) returned
10	10:48:28	SELECT id, company_name FROM company WHER...	0 row(s) returned
11	10:51:52	SELECT DATE(timestamp) AS fecha, COUNT(id) AS ...	5 row(s) returned
12	10:55:38	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
13	10:55:40	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned

Esta consulta no presenta novedades respecto a las realizadas anteriormente.

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 situadas en el mismo país que esta compañía.

Muestra el listado aplicando JOIN y subconsultas.

```
98 SELECT *
99 FROM transaction
100 INNER JOIN company ON company.id = transaction.company_id
101 WHERE country = (
102     SELECT country
103     FROM company
104     WHERE company_name = "Non Institute");
105
106
107
108
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
008629B4-C9A9-406C-A3D2-71FDA47BC546	CcS-7063	b-2246	2482	45.7666	4.83048	2015-07-30 12:12:42	486.44	0
00872BA4-54A3-4B8E-813F-2D57535AA17A	CcS-8475	b-2246	3894	55.6212	-3.7546	2017-10-26 22:08:26	414.06	0
01F075B1-D7AE-4D02-AAD9-5FFD72A43F3C	CcS-8700	b-2246	4119	55.856	-3.15783	2018-01-27 13:44:36	103.73	0
023FFC8B-E618-4938-8F56-C8DF80540ADD	CcS-7816	b-2246	3235	46.3568	1.82755	2016-12-19 11:53:45	219.28	0
026838EB-EF91-4564-9578-D6F1662AB7C5	CcS-9471	b-2246	4890	42.1332	12.396	2017-01-10 21:09:29	326.87	0
02C2F29E-CEFC-4C1E-A594-F476E8F279C0	CcS-9082	b-2246	4501	39.4662	-0.373246	2020-05-24 01:17:29	155.72	0
02F468DC-426C-47C2-880A-D8B25B7A81AF	CcS-6913	b-2246	2332	52.175	19.3508	2023-03-17 16:36:27	305.35	0
03068E3B-817B-4A49-934E-0E439291A104	CcS-5302	b-2246	721	51.9233	18.926	2021-12-02 23:06:02	339.58	0
0347BFE6-8EB5-4387-8187-0E78E8F2B8FB	CcS-7674	b-2246	3093	45.768	4.84271	2021-12-30 08:40:24	172.93	0
03AEB00E-DC97-4BD3-9C57-6A6DB78026FD	CcS-6121	b-2246	1540	50.8113	10.3145	2018-11-11 11:28:49	114.77	0
03CA36D3-88FF-4DBF-8FD4-4CC7DA4EED2B	CcS-8036	b-2246	3455	52.5178	13.4131	2017-02-25 15:38:21	440.27	0
04494182-96D0-42EB-82FE-5F92C5210537	CcS-6791	b-2246	2210	41.9542	12.4607	2018-05-17 17:53:53	241.59	0
045AACF6-FF85-49FB-90E4-E6730655366A	CcS-5363	b-2246	782	39.2464	-7.90454	2018-08-09 22:12:54	188.58	0
0489FDAC-86A4-4929-954E-9A2272189CAC	CcS-7296	b-2246	2715	51.1647	10.7348	2021-03-30 04:29:22	205.45	0
0490C36A-4B02-4CED-833B-80B3AC86C6C8	CcS-7539	b-2246	2958	46.4281	1.64603	2023-05-20 19:28:00	424.64	0
04A0ABDC-FB25-4E59-A901-84BB79380A33	CcS-4971	b-2246	290	52.0589	5.55327	2022-11-04 05:35:28	461.34	0
0517163F-B670-4BFF-B1F7-46C6B8A80F0F	CcS-4661	b-2246	2070	38.4384	-8.57107	2012-11-28 07:51:38	87.15	0

Result 41 x

Output

Action Output

#	Time	Action	Message
8	10:45:45	SELECT * FROM transaction WHERE EXISTS (...	13291 row(s) returned
9	10:47:30	SELECT id, company_name FROM company WHERE ...	100 row(s) returned
10	10:48:28	SELECT id, company_name FROM company WHERE ...	0 row(s) returned
11	10:51:52	SELECT DATE(timestamp) AS fecha, COUNT(id) AS ...	5 row(s) returned
12	10:55:38	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
13	10:55:40	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
14	10:56:15	SELECT * FROM transaction INNER JOIN company ...	13776 row(s) returned

En la subconsulta WHERE se decide usar como criterio `country =`. Si bien se podría usar `EXISTS` o `IN`, la modalidad utilizada es más sencilla y eficiente.

Muestra el listado aplicando solamente subconsultas.

```

98 SELECT *
99 FROM transaction
100 INNER JOIN company ON company.id = transaction.company_id
101 WHERE country = (
102     SELECT country
103     FROM company
104     WHERE company_name = "Non Institute");
105
106
107
108

```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
008629B4-C9A9-406C-A3D2-71FDA47BC546	CcS-7063	b-2246	2482	45.7666	4.83048	2015-07-30 12:12:42	486.44	0
00B72BA4-54A3-4B8E-813F-2D57535AA17A	CcS-8475	b-2246	3894	55.6212	-3.7546	2017-10-26 22:08:26	414.06	0
01F075B1-D7AE-4D02-AAD9-5FFD72A43F3C	CcS-8700	b-2246	4119	55.856	-3.15783	2018-01-27 13:44:36	103.73	0
023FFC8B-E618-4938-8F56-C8DF80540ADD	CcS-7816	b-2246	3235	46.3568	1.82755	2016-12-19 11:53:45	219.28	0
026838EB-EF91-4564-9578-D6F1662AB7C5	CcS-9471	b-2246	4890	42.1332	12.396	2017-01-10 21:09:29	326.87	0
02C2F29E-CEF2-4C1E-A594-F476E8F279C0	CcS-9082	b-2246	4501	39.4662	-0.373246	2020-05-24 01:17:29	155.72	0
02F468DC-426C-47C2-880A-D8B25B7A81AF	CcS-6913	b-2246	2332	52.175	19.3508	2023-03-17 16:36:27	305.35	0
0306BE3B-817B-4A49-934E-0E439291A104	CcS-5302	b-2246	721	51.9233	18.926	2021-12-02 23:06:02	339.58	0
0347BF6E-8EB5-4387-B187-0E78E8F2B8FB	CcS-7674	b-2246	3093	45.768	4.84271	2021-12-30 08:40:24	172.93	0
03AEBD0E-DC97-4BD3-9C57-6A6DB78026FD	CcS-6121	b-2246	1540	50.8113	10.3145	2018-11-11 11:28:49	114.77	0
03CA36D3-88FF-4DBF-8FD4-4CC7DA4EED2B	CcS-8036	b-2246	3455	52.5178	13.4131	2017-02-25 15:38:21	440.27	0
04494182-96DD-42EB-82FE-5F92C5210537	CcS-6791	b-2246	2210	41.9542	12.4607	2018-05-17 17:53:53	241.59	0
045AACF6-FF85-49FB-90E4-E6730655366A	CcS-5363	b-2246	782	39.2464	-7.90454	2018-08-09 22:12:54	188.58	0
0489FDAC-85A4-4929-954E-9A2272189CAC	CcS-7296	b-2246	2715	51.1647	10.7348	2021-03-30 04:29:22	205.45	0
0490C36A-EB02-4CED-633B-80B3AC86C6C8	CcS-7539	b-2246	2958	46.4281	1.64603	2023-05-20 19:28:00	424.64	0
04A0ABDE-FB25-4E59-A901-B4BB79380A33	CcS-4871	b-2246	290	52.0589	5.55327	2022-11-04 05:35:28	461.34	0
0511103E-B670-8DFF-B1E7-45660B8080F0	CcS-4651	b-2246	7070	50.684	-8.57107	2024-07-08 07:51:38	87.15	0

Result 41 x Read On

Output

#	Time	Action	Message
8	10:45:45	SELECT * FROM transaction WHERE EXISTS (13291 row(s) returned
9	10:47:30	SELECT id, company_name FROM company WHER...	100 row(s) returned
10	10:48:28	SELECT id, company_name FROM company WHER...	0 row(s) returned
11	10:51:52	SELECT DATE(timestamp) AS fecha, COUNT(id) AS ...	5 row(s) returned
12	10:55:38	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
13	10:55:40	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
14	10:56:15	SELECT * FROM transaction INNER JOIN company ...	13776 row(s) returned

En este ejercicio se encuentra una novedad respecto a los anteriores. Se requiere renombrar las tablas mediante la función AS, puesto que como se hace referencia en más de una ocasión a una variable de una tabla. Sin este paso, se retornaría un error, puesto que sería una solicitud ambigua.

Nivel 3

Ejercicio 1

Presenta el nombre, teléfono, país, fecha e importe (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 de 2015, 20 de julio de 2018 y 13 de marzo de 2024. Ordena los resultados de mayor a menor cantidad.

Listado compañías seleccionadas.

```
122 • SELECT company.id, company_name, phone, country, amount, DATE(timestamp) AS fecha
123 FROM transaction
124 INNER JOIN company ON company.id = transaction.company_id
125 WHERE amount BETWEEN 350 AND 400
126 AND (DATE(timestamp) = "2015-04-29"
127 OR DATE(timestamp) = "2018-07-20"
128 OR DATE(timestamp) = "2024-03-13") -- podría usarse IN
129 ORDER BY amount DESC;
130
131
132
```

	id	company_name	phone	country	amount	fecha
▶	b-2566	Aliquam PC	01 45 73 52 16	Germany	399.84	2024-03-13
	b-2294	Auctor Mauris Vel LLP	08 09 28 74 14	United States	399.51	2018-07-20
	b-2402	At Pede Corp.	06 14 48 33 15	Italy	390.69	2015-04-29
	b-2566	Aliquam PC	01 45 73 52 16	Germany	388.29	2024-03-13
	b-2574	Orci Adipiscing Limited	03 18 00 77 81	United Kingdom	373.71	2018-07-20
	b-2286	Fringilla LLC	08 29 15 93 57	New Zealand	367.62	2015-04-29
	b-2426	Pede Cum Ltd	07 62 26 48 38	Norway	356.87	2018-07-20
	b-2294	Auctor Mauris Vel LLP	08 09 28 74 14	United States	353.75	2024-03-13

Result 44 × Read Only

Output

#	Time	Action	Message
✓ 11	10:51:52	SELECT DATE(timestamp) AS fecha, COUNT(id) AS ...	5 row(s) returned
✓ 12	10:55:38	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
✓ 13	10:55:40	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
✓ 14	10:56:15	SELECT * FROM transaction INNER JOIN company ...	13776 row(s) returned
✓ 15	10:57:30	SELECT * FROM transaction AS t WHERE EXISTS (...)	13776 row(s) returned
✓ 16	10:58:26	SELECT company.id, company_name, phone, countr...	8 row(s) returned
✓ 17	10:58:37	SELECT company.id, company_name, phone, countr...	8 row(s) returned

En esta consulta se incorpora la función **BETWEEN** que permite seleccionar un rango de valores en el que los límites están incluidos. Se decide por incluir cada fecha indicada mediante el operador **OR**, sin embargo, podría hacerse más simple con **IN**, pero se han seguido las recomendaciones de las buenas prácticas de codificación en SQL.

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 donde especifiques si tienen más de 400 transacciones o menos.

Listado compañías seleccionadas.

```

135 • SELECT company.id, company_name, COUNT(*) AS num_transacciones,
136     CASE
137     WHEN COUNT(*) < 400 THEN 'Menos de 400'
138     ELSE '400 o más'
139     END AS identificador_num_ventas
140 FROM transaction
141 INNER JOIN company ON company.id = transaction.company_id
142 GROUP BY company_id
143 ORDER BY num_transacciones;
144

```

id	company_name	num_transacciones	identificador_num_ventas
b-2362	Lorem Eu Incorporated	380	Menos de 400
b-2286	Fringilla LLC	397	Menos de 400
b-2470	Nec Luctus LLC	399	Menos de 400
b-2578	Dui Quis Institute	402	400 o más
b-2446	Risus Associates	407	400 o más
b-2226	Magna A Neque Industries	410	400 o más
b-2526	Amet Luctus Vulputate Foundation	411	400 o más
b-2298	Elit Etiam Laoreet Associates	418	400 o más
b-2534	Mattis Foundation	418	400 o más
b-2590	Euismod Mauris Institute	418	400 o más
b-2390	Neque Tellus Imperdiet Corp.	424	400 o más
b-2254	Nascetur Ridiculus Mus Inc.	424	400 o más
b-2490	Nunc Sit Incorporated	424	400 o más
b-2498	Metus Vitae Associates	425	400 o más
b-2606	Sed Est Corp.	426	400 o más
b-2486	Tincidunt Orci Limited	426	400 o más
b-2598	Aliquam Iaculis Lacus Corp.	426	400 o más
b-2387	Magna Tellus Incorporated	430	400 o más

Result 45 x

Output

#	Time	Action	Message
12	10:55:38	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
13	10:55:40	SELECT country, ROUND(AVG(amount), 2) AS prom...	15 row(s) returned
14	10:56:15	SELECT * FROM transaction INNER JOIN company ...	13776 row(s) returned
15	10:57:30	SELECT * FROM transaction AS t WHERE EXISTS (...)	13776 row(s) returned
16	10:58:26	SELECT company.id, company_name, phone, countr...	8 row(s) returned
17	10:58:37	SELECT company.id, company_name, phone, countr...	8 row(s) returned
18	10:59:24	SELECT company.id, company_name, COUNT(*) AS ...	100 row(s) returned

Este último ejercicio requirió la creación de una variable dicotómica que identifique las transacciones según los criterios declarados. Para esto se utiliza la combinación de las funciones CASE y WHEN.