

Tasca S2.01. Nocions bàsiques SQL

Descripció

Repasar les nocions bàsiques per a l'ús de base de dades relacionals. En aquest sprint, iniciaràs la teva experiència pràctica amb una base de dades que conté informació d'una empresa dedicada a la venda de productes en línia. En aquesta activitat, t'enfocaràs en dades relacionades amb les transaccions efectuades i la informació corporativa de les empreses que van participar.

Important

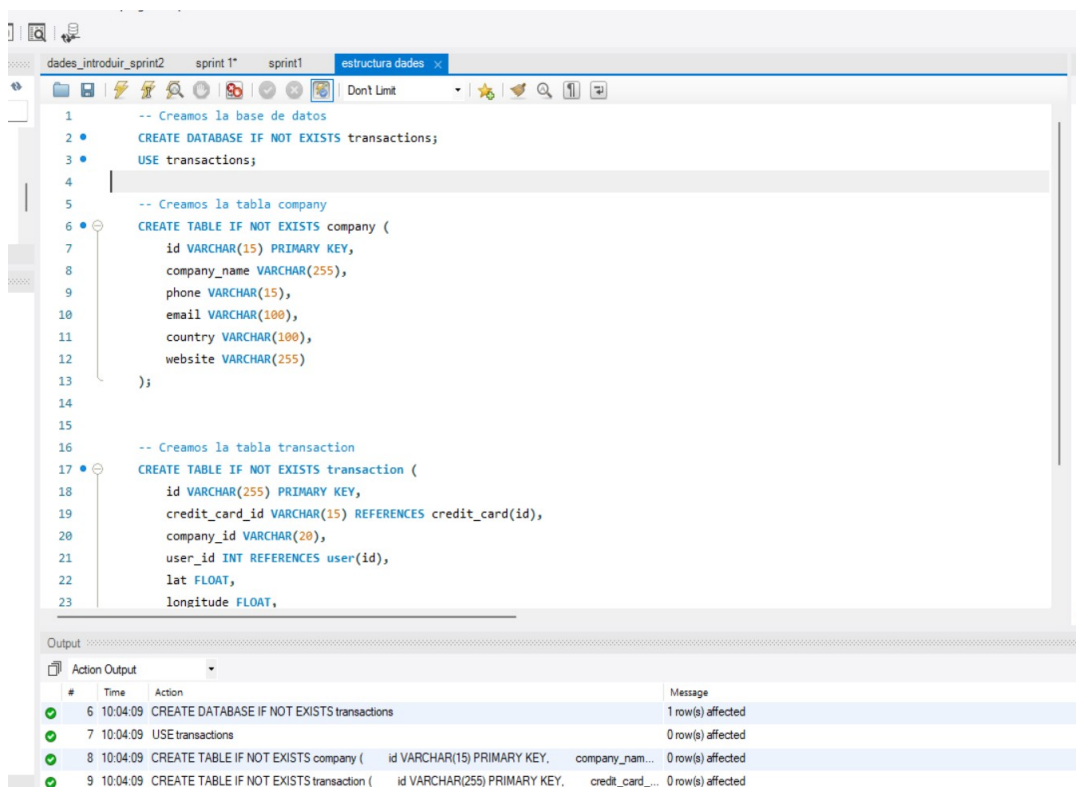
Totes les transformacions i importacions que se't demanen en aquesta tasca s'han de realitzar utilitzant **codi SQL**. **NO ES PERMET fer els canvis utilitzant el wizard.**

Nivell 1

Exercici 1

A partir dels documents adjunts (estructura_dades i dades_introduir), importa les dues taules. Mostra les característiques principals de l'esquema creat i explica les diferents taules i variables que existeixen. Assegura't d'incloure un diagrama que il·lustri la relació entre les diferents taules i variables.

Primero de todo cargamos el archivo adjunto de la estructura de datos donde crearemos la base de datos Transactions, en el mismo archivo crearemos las tablas que en este caso son dos con los nombres Company y Transaction y ejecutamos el archivo para que se creen la base de datos y las tablas de la respectiva base de datos



```
1 -- Creamos la base de datos
2 • CREATE DATABASE IF NOT EXISTS transactions;
3 • USE transactions;
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,
```

#	Time	Action	Message
6	10:04:09	CREATE DATABASE IF NOT EXISTS transactions	1 row(s) affected
7	10:04:09	USE transactions	0 row(s) affected
8	10:04:09	CREATE TABLE IF NOT EXISTS company (id VARCHAR(15) PRIMARY KEY, company_nam...	0 row(s) affected
9	10:04:09	CREATE TABLE IF NOT EXISTS transaction (id VARCHAR(255) PRIMARY KEY, credit_card_...	0 row(s) affected

```

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   );
29
30

```

#	Time	Action	Message
6	10:04:09	CREATE DATABASE IF NOT EXISTS transactions	1 row(s) affected
7	10:04:09	USE transactions	0 row(s) affected
8	10:04:09	CREATE TABLE IF NOT EXISTS company (id VARCHAR(15) PRIMARY KEY, company_name...	0 row(s) affected
9	10:04:09	CREATE TABLE IF NOT EXISTS transaction (id VARCHAR(255) PRIMARY KEY, credit_card_...	0 row(s) affected

Una vez creada la base de datos con sus diferentes tablas realizamos la introducción de los datos de las tablas mediante el archivo adjunto dades a introducir y ejecutamos el archivo para que se cargen todos los datos en la base de datos transactions.

```

1  USE transactions;
2  -- Insertamos datos de company
3
4  INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
5  INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
6  INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
7  INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
8  INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
9  INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
10 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
11 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
12 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
13 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
14 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
15 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
16 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
17 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
18 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
19 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
20 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
21 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
22 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (
23 INSERT INTO company (id, company_name, phone, email, country, website) VALUES (

```

#	Time	Action	Message
100107	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, de...	1 row(s) affected
100108	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, de...	1 row(s) affected
100109	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, de...	1 row(s) affected
100110	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, de...	1 row(s) affected

Realizamos una comprobación para ver si se han cargado los datos realizando una consulta a la tabla Company

Navigation pane on the left shows the database structure:

- sys
- tienda
- transactions
 - company
 - transaction

Table: company

Columns:

- id: varchar(15) PK
- company_name: varchar(255)
- phone: varchar(15)
- email: varchar(100)
- country: varchar(100)
- website: varchar(255)

Query executed:

```
SELECT * FROM transactions.company;
```

Result Grid:

id	company_name	phone	email	country	website
b-2222	Ac Fermentum Incorporated	06 85 56 52 33	donec.porrtitor.tellus@yahoo.net	Germany	https://instagram.com/site
b-2226	Magna A Neque Industries	04 14 44 64 62	rius.donec.nibh@icloud.org	Australia	https://whatsapp.com/group/9
b-2230	Fusce Corp.	08 14 97 58 85	rius@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 laculis Nec Foundation	08 23 04 99 53	sed.dictum.pron@outlook.ca	New Zealand	https://netflix.com/settings

Action Output:

#	Time	Action	Message
100108	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, d...	1 row(s) affected
100109	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, d...	1 row(s) affected
100110	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, d...	1 row(s) affected
100111	10:43:57	SELECT * FROM transactions.company	100 row(s) returned

Realizamos la misma comprobación en la tabla Transaction

Navigation pane on the left shows the database structure:

- sys
- tienda
- transactions
 - company
 - transaction

Table: company

Columns:

- id: varchar(15) PK
- company_name: varchar(255)
- phone: varchar(15)
- email: varchar(100)
- country: varchar(100)
- website: varchar(255)

Query executed:

```
SELECT * FROM transactions.transaction;
```

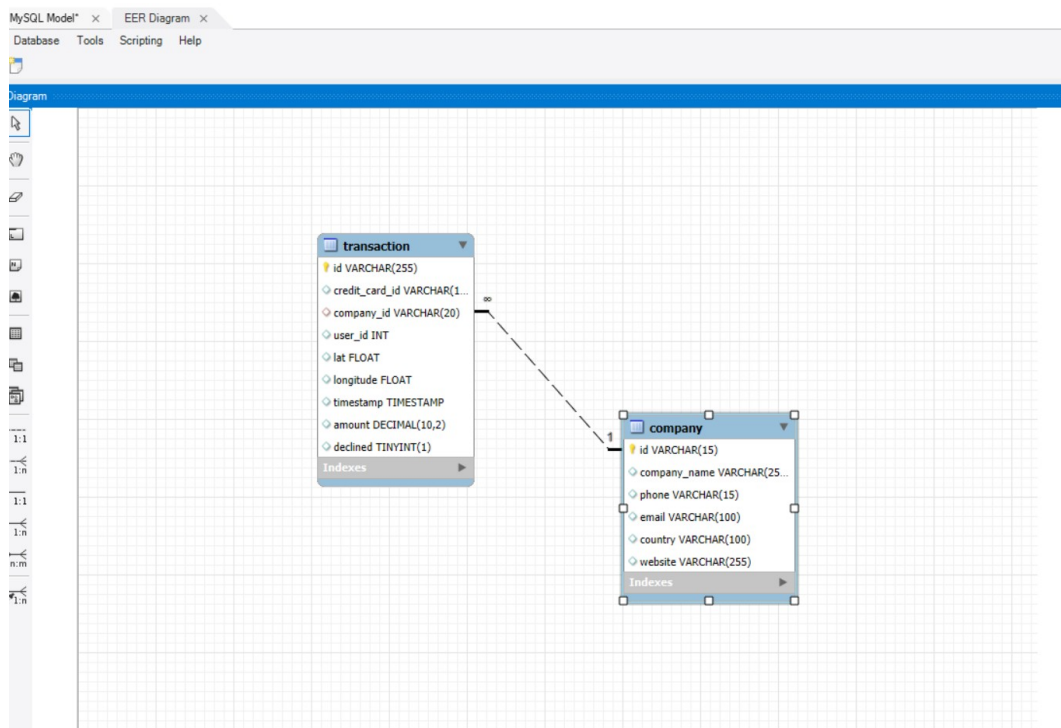
Result Grid:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
00043A49-2949-494E-A5D0-A5BAE38B190D	Cc5-9294	b-2458	4713	46.1999	1.43554	2024-08-28 07:16:46	395.43	0
000447FE-8650-40CF-85DE-C7ED0EE1CAAD	Cc5-5019	b-2370	438	41.5972	12.2218	2016-12-21 20:07:18	155.63	0
00045D68-ED2E-4F3F-8186-CEE074D875D0	Cc5-6699	b-2390	2118	29.7573	-95.3796	2020-07-14 15:37:45	326.01	0
000481C3-1C26-4FEF-83A0-4CD0E9004B8D	Cc5-6696	b-2230	2115	53.5489	-113.503	2017-09-04 19:44:53	161.60	0
00051AA4-9CBE-4268-B070-C38062A1B3E2	Cc5-7606	b-2266	3025	52.2084	5.69081	2017-01-05 18:19:25	148.91	0

Action Output:

#	Time	Action	Message
100109	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, d...	1 row(s) affected
100110	10:23:13	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, d...	1 row(s) affected
100111	10:43:57	SELECT * FROM transactions.transaction	100 row(s) returned
100112	10:50:49	SELECT * FROM transactions.transaction	100000 row(s) returned

Una vez realizada la comprobación para ver que todo esta correcto analizamos las dos tablas de la que consta la base de Datos.



La relación entre las tablas seria desde la tabla Company de 1 a muchos hacia la tabla Transaction, ya que una company puede tener muchas transacciones pero una transacción solo se puede vincular a una compañía.

También observamos que la línea que nos une las 2 tablas es discontinua lo que quiere decir que la relación entre estas dos tablas es débil o en algunos casos puede ser opcional. Esto quiere decir que aunque hay relación la foreign key puede tener valores nulos.

TABLA COMPANY (Tabla de dimensiones) consta las siguientes columnas

- ID Primary Key de la tabla Not Null
- COMPANY_NAME (nombre de la compañía)
- PHONE (telefono asociado a la compañía)
- EMAIL (correo electronico de la compañía)
- COUNTRY (pais donde esta ubicada la compañía)
- WEBSITE (pagina web donde esta registrada la información de la empresa)

TABLE TRANSACTION (Tabla de hechos) consta de las siguientes columnas

- ID primary Key de la tabla Not Null.
- CREDIT_CARD_ID (identificador de la tarjeta de credito)
- COMPANY_ID Foreign Key que se relaciona con la Primary Key ID de la tabla Company
- USER_ID (identificador del usuario/a que realiza la transacción)
- LAT (nos da la latitud desde donde se realiza la transacción)

- LONGITUDE (nos da la longitud desde donde se realiza la transacción)
- TIMESTAMP (nos da la fecha y la hora cuando se realiza la transacción)
- AMOUNT (nos da el importe de cada transacción)
- DECLINED (Indicador de si la transacción ha sido aceptada o rechazada)

Exercici 2

Utilitzant JOIN realitzaràs les següents consultes

A - Llistat dels països que estan generant vendes.

The screenshot shows the MySQL Model* interface with a SQL query editor and a result grid. The query is as follows:

```

1  -- Exercici 2
2  -- Utilitzant JOIN realitzaràs les següents consultes:
3
4  -- a-Llistat dels països que estan generant vendes.
5  • SELECT DISTINCT c.country
6    FROM company c
7   JOIN transaction t ON c.id = t.company_id
8   WHERE t.declined = 0;

```

The result grid displays the following countries:

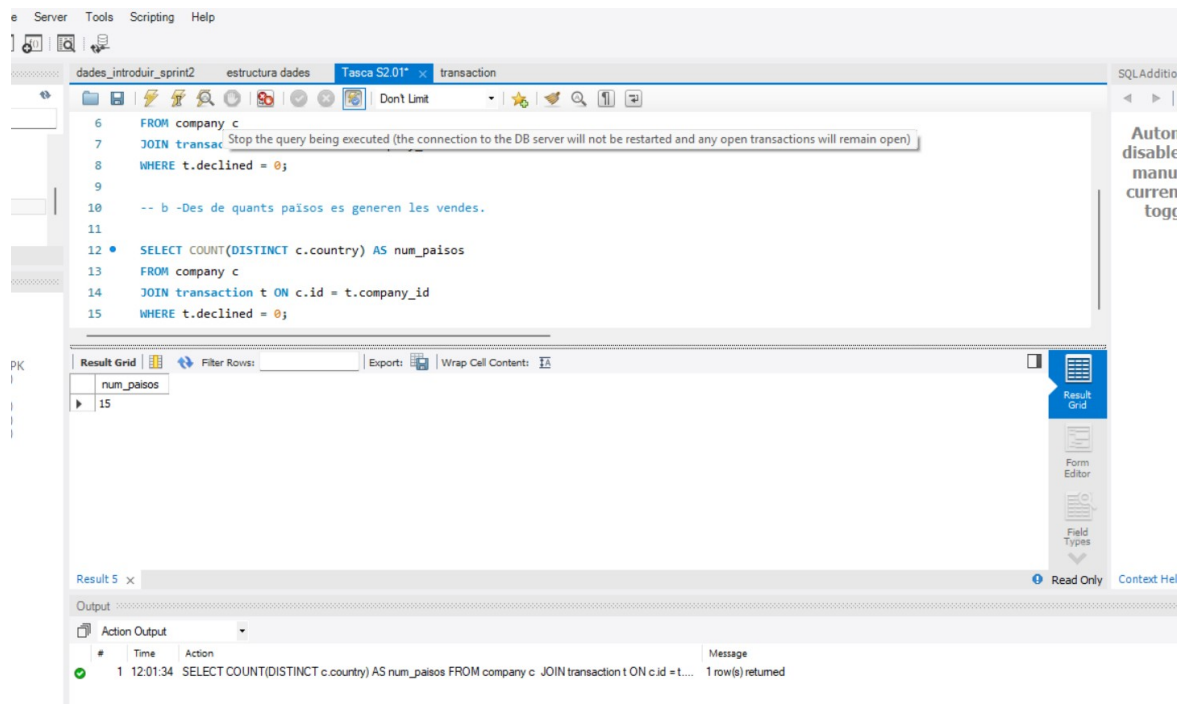
country
Netherlands
Sweden
Ireland
United States
Belgium
Canada
Germany
Norway
France
Italy
United Kingdom
New Zealand

The output section shows the execution details:

#	Time	Action	Message
1	11:48:53	SELECT DISTINCT c.country FROM company c JOIN transaction t ON c.id = t.company_id WHERE t.d...	15 row(s) returned

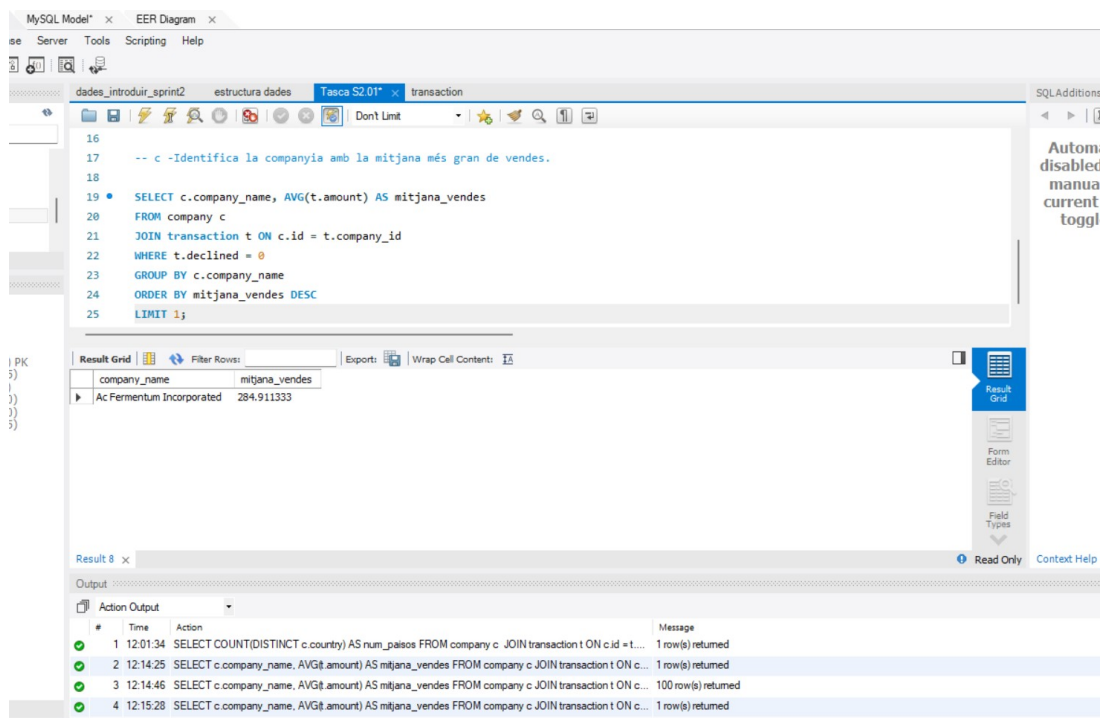
Para obtener la cantidad de países que realizan ventas hacemos un SELECT DISTINCT de Company.country para que nos de el país una sola vez y realizamos un join entre las tablas company y transaction y le damos la condición en el WHERE que la columna Declined = 0 para saber que la transacción se ha realizado correctamente ya que el 1 quiere decir que no se ha realizado la transacción sino que se ha declinado.

B - Des de quants països es generen les vendes.



En este ejercicio realizamos un SELECT COUNT DISTINCT de los países de la tabla company que lo renombramos como num_paises el COUNT no hace un conteo de los países y el distinct no da una unidad sin repetición del país. Realizamos un join con la tabla transaction para saber que países han realizado ventas con el condicional WHERE y que la columna Declined sea igual a 0, para saber que la transacción se ha realizado correctamente.

C- Identifica la companyia amb la mitjana més gran de vendes.



En este ejercicio realizamos un Select de company_name y realizamos un AVG del transaction.amount para que nos de el importe medio de las ventas en la condición Where ponemos que la transaction.declined sea 0 para saber que la transacción se ha realizado seguidamente usaremos un Group BY para agruparlo todo por el company_name de esta manera nos dará la media de ventas por cada compañía seguidamente realizamos un ORDER BY de la media de ventas en orden descendente para que nos salga los primeros con el mayor numero de ventas y finalmente lo limitamos a 1 para que solo nos de la compañía con mayor media de ventas.

Exercici 3

Utilitzant només subconsultes (sense utilitzar JOIN):

A- Mostra totes les transaccions realitzades per empreses d'Alemanya.

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

```
-- Exercici 3
-- Utilitzant només subconsultes (sense utilitzar JOIN):
-- a- Mostra totes les transaccions realitzades per empreses d'Alemanya.

SELECT *
FROM transaction
WHERE declined = 0 AND company_id IN (
    SELECT id
    FROM company
    WHERE country = 'Germany');

-- b-Llista les empreses que han realitzat transaccions per un amount superior a la mitjana de totes les transaccions.
SELECT DISTINCT company_name
FROM company
```

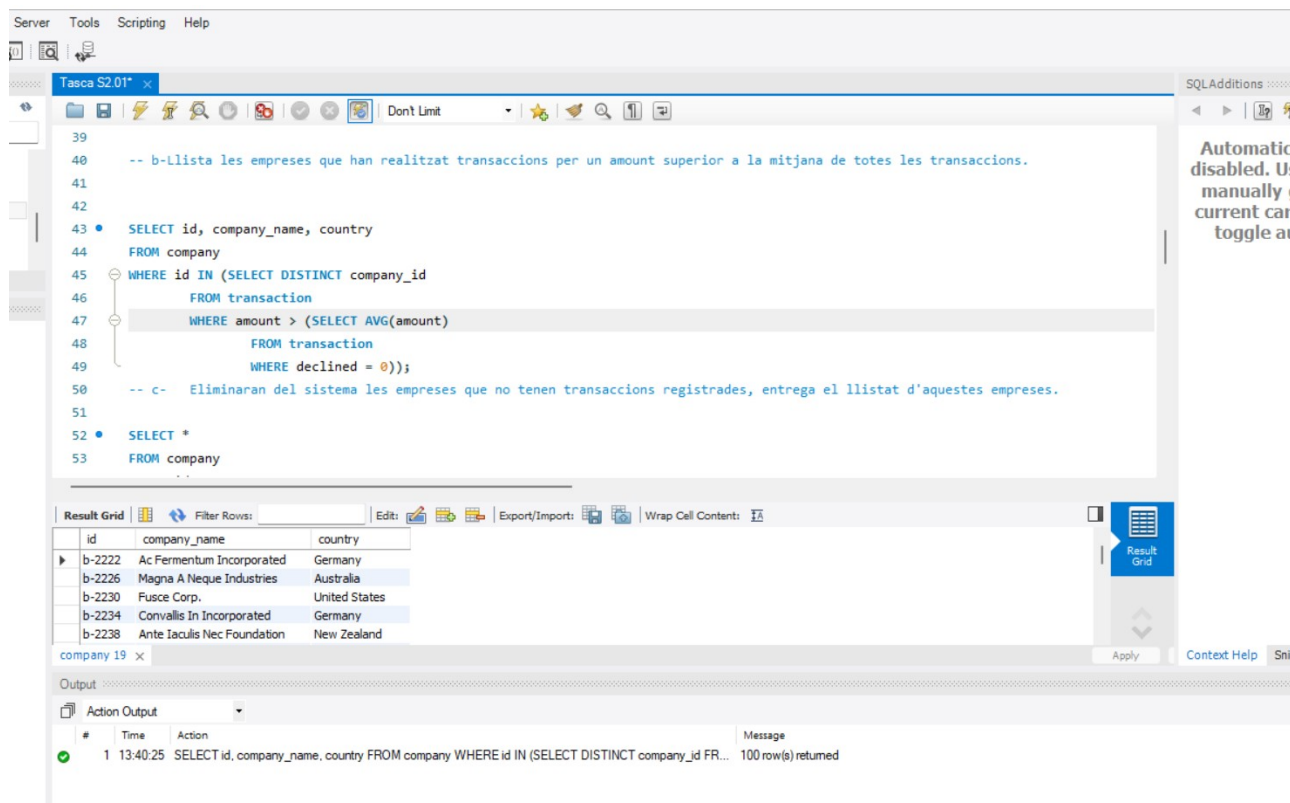
The result grid displays the following data:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
00138D3B-206D-4C03-94B7-63A2676EB9B4	CcS-4899	b-2222	318	41.3781	12.447	2020-03-25 10:43:43	426.36	0
0013C1B6-3B84-4D6C-8154-E2B3FEBCA8E9	CcS-5070	b-2222	489	41.3814	2.18176	2020-12-17 18:15:37	316.90	0
00201A11-2E62-44C4-941D-198FC8D877F0	CcU-3512	b-2222	193	55.5704	-3.65129	2021-01-22 23:44:27	453.04	0
00235618-0A5C-4D49-9DCB-83A9405D8923	CcS-8137	b-2222	3556	59.8421	18.729	2020-09-09 15:43:19	263.14	0
005A5A7B-1F1A-4B6C-9B15-1625A78C9C38	CcS-8998	b-2222	4417	41.1591	-8.63905	2024-05-15 09:10:11	442.01	0

The output section shows the execution of the query, indicating that 13269 rows were returned.

En este ejercicio hemos realizado una subconsulta para obtener solo las empresas que son de Alemania y en la consulta principal hemos pedido todos los datos de las transacciones y el Where hemos puesto el company_id hemos usado el operador IN y la subconsulta para que la respuesta final solo nos de las transacciones que están relacionadas con la subconsulta.

B- Llista les empreses que han realitzat transaccions per un amount superior a la mitjana de totes les transaccions.



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

```
39
40 -- b-Llista les empreses que han realitzat transaccions per un amount superior a la mitjana de totes les transaccions.
41
42
43 • SELECT id, company_name, country
44 FROM company
45 WHERE id IN (SELECT DISTINCT company_id
46 FROM transaction
47 WHERE amount > (SELECT AVG(amount)
48 FROM transaction
49 WHERE declined = 0));
50 -- c- Eliminaran del sistema les empreses que no tenen transaccions registrades, entrega el llistat d'aquestes empreses.
51
52 • SELECT *
53 FROM company
```

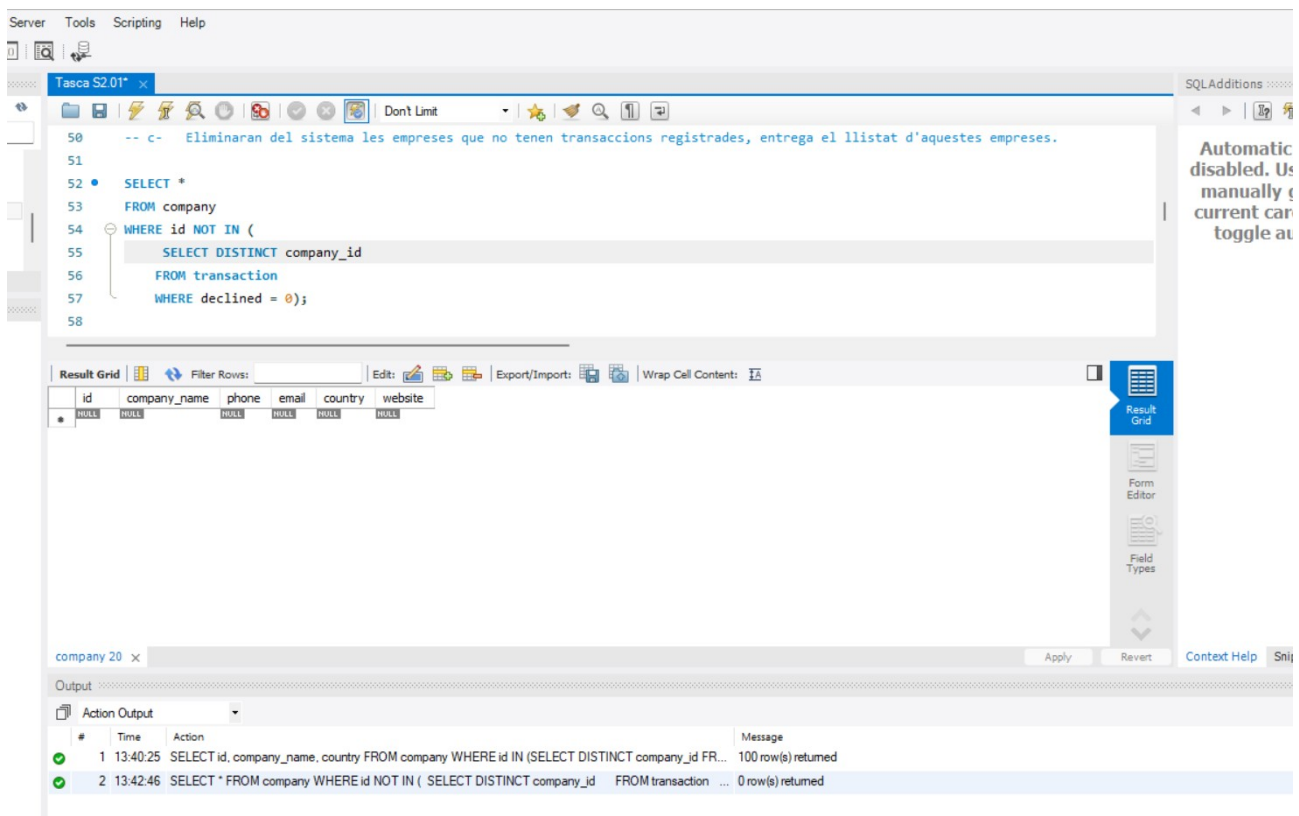
The result grid displays the following data:

id	company_name	country
b-2222	Ac Fermentum Incorporated	Germany
b-2226	Magna A Neque Industries	Australia
b-2230	Fusce Corp.	United States
b-2234	Convallis In Incorporated	Germany
b-2238	Ante Iaculis Nec Foundation	New Zealand

The output pane at the bottom shows the execution message: "1 13:40:25 SELECT id, company_name, country FROM company WHERE id IN (SELECT DISTINCT company_id FR... 100 row(s) returned".

Para este ejercicio realizamos 2 subconsultas anidadas en la primera subconsulta la media total de amount y en la otra subconsulta pedimos que compañías tienen ventas con un amount superior a la media y en la consulta principal pedimos el ID, NOMBRE y PAIS.

C- Eliminaran del sistema les empreses que no tenen transaccions registrades, entrega el llistat d'aquestes empreses.



The screenshot shows a SQL IDE interface with a query editor, a result grid, and an output window.

Query Editor: The query is written in SQL and includes a comment in Catalan. It uses a subquery with the `NOT IN` operator to filter out companies that have transactions.

```
50 -- c- Eliminaran del sistema les empreses que no tenen transaccions registrades, entrega el llistat d'aquestes empreses.
51
52 SELECT *
53 FROM company
54 WHERE id NOT IN (
55     SELECT DISTINCT company_id
56     FROM transaction
57     WHERE declined = 0);
58
```

Result Grid: The grid shows the columns `id`, `company_name`, `phone`, `email`, `country`, and `website`. The first row contains the value `NULL` for all columns.

Output Window: The output shows two actions executed successfully.

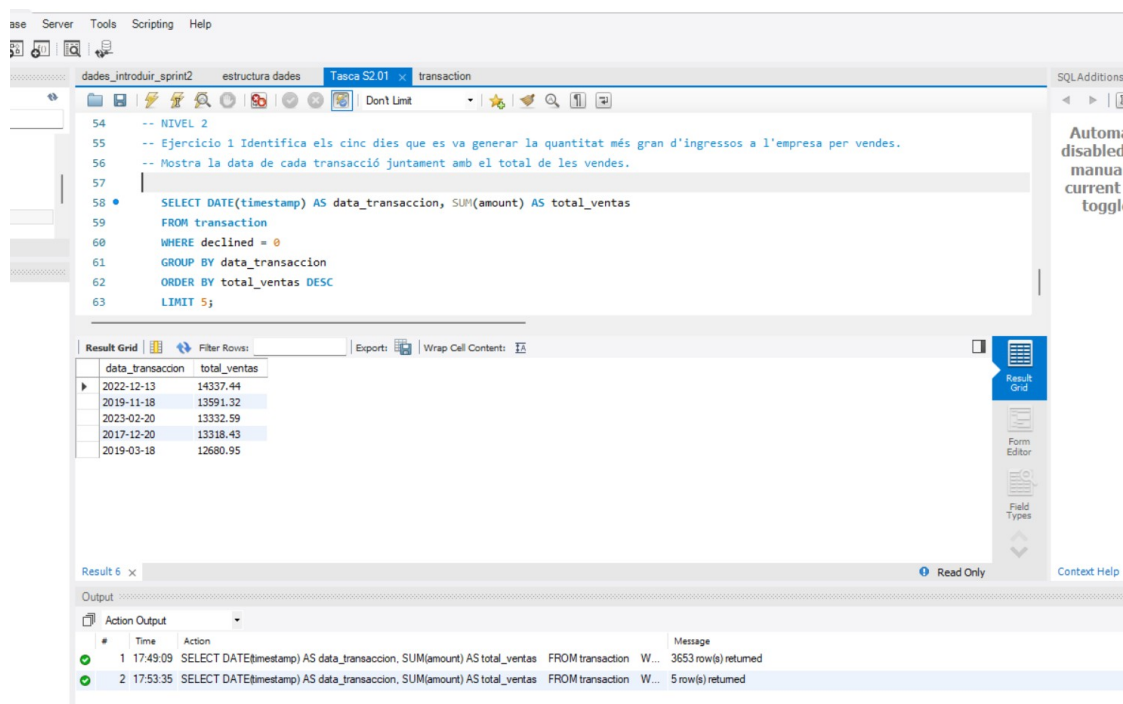
#	Time	Action	Message
1	13:40:25	SELECT id, company_name, country FROM company WHERE id IN (SELECT DISTINCT company_id FR...	100 row(s) returned
2	13:42:46	SELECT * FROM company WHERE id NOT IN (SELECT DISTINCT company_id FROM transaction ...	0 row(s) returned

En este ejercicio realizamos una subconsulta en la que solicitamos todas las compañías que tienen transferencias realizadas pero en la consulta principal le pedimos toda la información de las compañías que no están incluidas en la subconsulta usando el operador `NOT IN`.

Nivell 2

Exercici 1

Identifica els cinc dies que es va generar la quantitat més gran d'ingressos a l'empresa per vendes. Mostra la data de cada transacció juntament amb el total de les vendes.



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

```
-- NIVEL 2
-- Ejercicio 1 Identifica els cinc dies que es va generar la quantitat més gran d'ingressos a l'empresa per vendes.
-- Mostra la data de cada transacció juntament amb el total de les vendes.

SELECT DATE(timestamp) AS data_transaccion, SUM(amount) AS total_ventas
FROM transaction
WHERE declined = 0
GROUP BY data_transaccion
ORDER BY total_ventas DESC
LIMIT 5;
```

The results grid displays the following data:

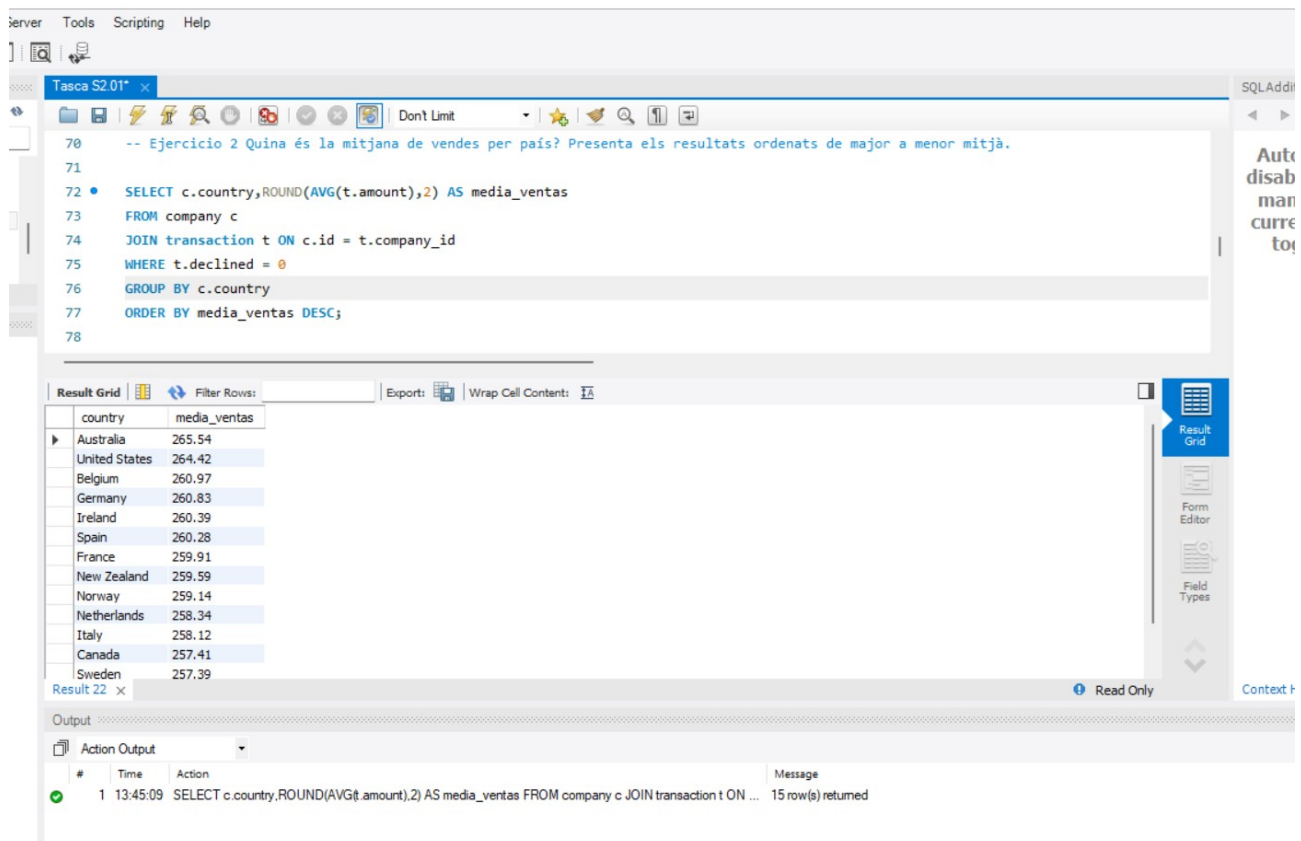
data_transaccion	total_ventas
2022-12-13	14337.44
2019-11-18	13591.32
2023-02-20	13332.59
2017-12-20	13318.43
2019-03-18	12680.95

The output pane shows the execution of the query, indicating that 3653 rows were returned for the first execution and 5 rows for the second execution.

En este ejercicio usaremos en el SELECT la función DATE para obtener las fechas sin el registro horario de las transacciones y la función SUM para obtener el total de los amount de la tabla transaction donde en el condicional WHERE solicitamos que la transacción haya sido realizada y a posterior realizamos un agrupamiento por las fechas y una ordenación por el total de las ventas en orden descendente para que las mayores aparezcan primero y finalmente lo limitamos a 5 ya que el ejercicio nos solicita los 5 días con mas ventas.

Exercici 2

Quina és la mitjana de vendes per país? Presenta els resultats ordenats de major a menor mitjà.



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

```
-- Ejercicio 2 Quina és la mitjana de vendes per país? Presenta els resultats ordenats de major a menor mitjà.  
  
SELECT c.country, ROUND(AVG(t.amount), 2) AS media_ventas  
FROM company c  
JOIN transaction t ON c.id = t.company_id  
WHERE t.declined = 0  
GROUP BY c.country  
ORDER BY media_ventas DESC;
```

The results grid displays the following data:

country	media_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

The output pane at the bottom shows the execution message: "1 13:45:09 SELECT c.country, ROUND(AVG(t.amount), 2) AS media_ventas FROM company c JOIN transaction t ON ... 15 row(s) returned".

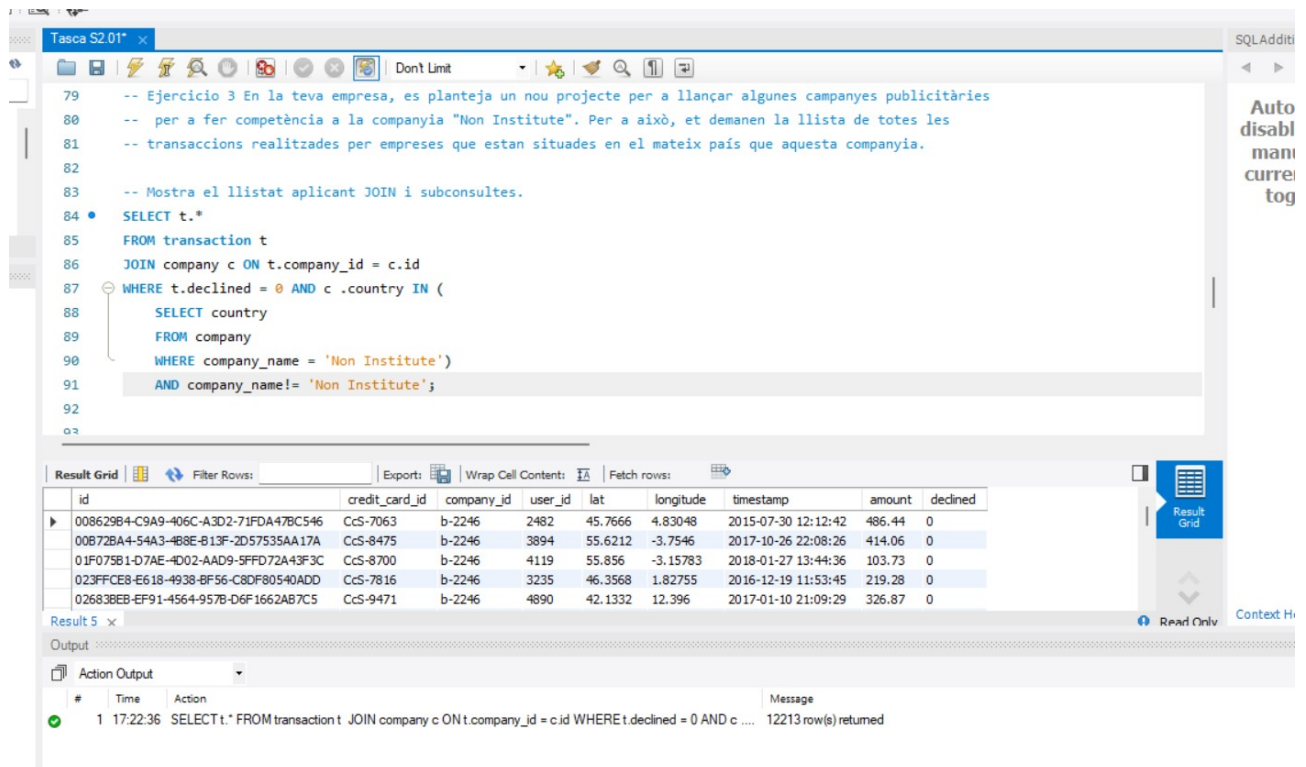
En este ejercicio seleccionamos mediante el **SELECT** los países que están en la tabla **company** y mediante la Función **AVG** le solicitamos la media de ventas de la tabla **transaction** con un **ROUND** para que nos redondee a 2 decimales en esta ocasión, para ello necesitamos concatenar las tablas mediante un **JOIN** y sus columnas coincidentes.

Una vez hecho en el **WHERE** ponemos la condición de la transacción se haya realizado y no sea nula y seguidamente realizamos un agrupamiento por países y lo ordenamos por la media de ventas en orden descendente (de mayor a menor) y obtenemos el resultado que es la media de ventas por país.

Exercici 3

En la teva empresa, es planteja un nou projecte per a llançar algunes campanyes publicitàries per a fer competència a la companyia "Non Institute". Per a això, et demanen la llista de totes les transaccions realitzades per empreses que estan situades en el mateix país que aquesta companyia.

1-Mostra el llistat aplicant JOIN i subconsultes.



The screenshot shows a SQL IDE window titled 'Tasca S2.01*'. The query editor contains the following SQL code:

```
-- Ejercicio 3 En la teva empresa, es planteja un nou projecte per a llançar algunes campanyes publicitàries
-- per a fer competència a la companyia "Non Institute". Per a això, et demanen la llista de totes les
-- transaccions realitzades per empreses que estan situades en el mateix país que aquesta companyia.

-- Mostra el llistat aplicant JOIN i subconsultes.
SELECT t.*
FROM transaction t
JOIN company c ON t.company_id = c.id
WHERE t.declined = 0 AND c.country IN (
    SELECT country
    FROM company
    WHERE company_name = 'Non Institute')
AND company_name != 'Non Institute';
```

The 'Result Grid' shows the following data:

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-B13F-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
023FFCE8-E618-4938-BF56-C8DF80540ADD	CcS-7816	b-2246	3235	46.3568	1.82755	2016-12-19 11:53:45	219.28	0
026838EB-EF91-4564-957B-D6F1662AB7C5	CcS-9471	b-2246	4890	42.1332	12.396	2017-01-10 21:09:29	326.87	0

The 'Output' section shows the following message:

```
1 17:22:36 SELECT t.* FROM transaction t JOIN company c ON t.company_id = c.id WHERE t.declined = 0 AND c ... 12213 row(s) returned
```

En este ejercicio realizamos una subconsulta para obtener las compañías que son del mismo país que NON INSTITUTE

En la consulta principal hemos realizado un join entre las dos tablas y en el where hemos puesto que la transaccion este realizada, hemos excluido los datos de la compañía Non Institute y que el país de la compañía sea el mismo que el de la subconsulta.

2- Muestra el llistat aplicant solament subconsultes.

The screenshot shows a SQL query editor with a query that uses subqueries to filter transactions. The query is as follows:

```
92
93 -- Mostra el llistat aplicant solament subconsultes.
94 SELECT *
95 FROM transaction
96 WHERE declined = 0 AND company_id IN
97     (SELECT id
98      FROM company
99      WHERE country IN (
100          SELECT country
101          FROM company
102          WHERE company_name = 'Non Institute')
103          AND company_name != 'Non Institute'
104     );
105
```

Below the query, the 'Result Grid' displays the results of the query. The grid has 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
0013DBE3-3898-4DC3-B705-CE6723CC1F71	CcU-3736	b-2310	161	55.2114	-3.40245	2024-04-25 16:42:21	425.10	0
001A60EA-DC9C-4E5A-9460-6628B100E7E1	CcS-6225	b-2326	1644	51.7125	19.0674	2018-05-20 02:06:39	354.02	0
001E5D06-A391-4735-88D8-748F16C061A6	CcS-7425	b-2310	2844	59.9165	18.5518	2021-09-10 02:39:32	418.20	0
0022377F-4447-432B-B01A-CFE5416E336C	CcS-8582	b-2522	4001	50.7865	10.6173	2019-10-24 06:52:23	171.13	0
00285171-6887-4E96-9787-BD580BE4515D	CcS-8358	b-2310	3777	55.7751	-3.8232	2022-09-21 21:18:51	192.60	0
0031C741-BE1F-43F3-954F-4F3389953F54	CcS-6854	b-2374	2273	51.6159	19.069	2015-09-22 04:36:39	381.37	0
0031FD57-439C-4412-BAB8-51812F0BD2AD	CcS-6023	b-2522	1442	59.7896	18.7997	2022-08-27 23:04:07	118.22	0

The 'Output' section shows the execution of the query, with the following messages:

```
1 13:49:35 SELECT t.id, c.company_name, c.country FROM transaction t JOIN company c ON t.company_id = c.id ... 12213 row(s) returned
2 13:51:03 SELECT * FROM transaction WHERE declined = 0 AND company_id IN (SELECT id FROM company W... 12213 row(s) returned
```

Para realizar el mismo ejercicio solamente usando subconsultas solicitamos en la consulta principal todas las columnas de la tabla transaction y en el where le pedimos las transacciones realizadas con el declined , que no incluya la empresa nombrada y que company_id este dentro de las dos subconsultas que tenemos que integrar para llegar al resultado final.

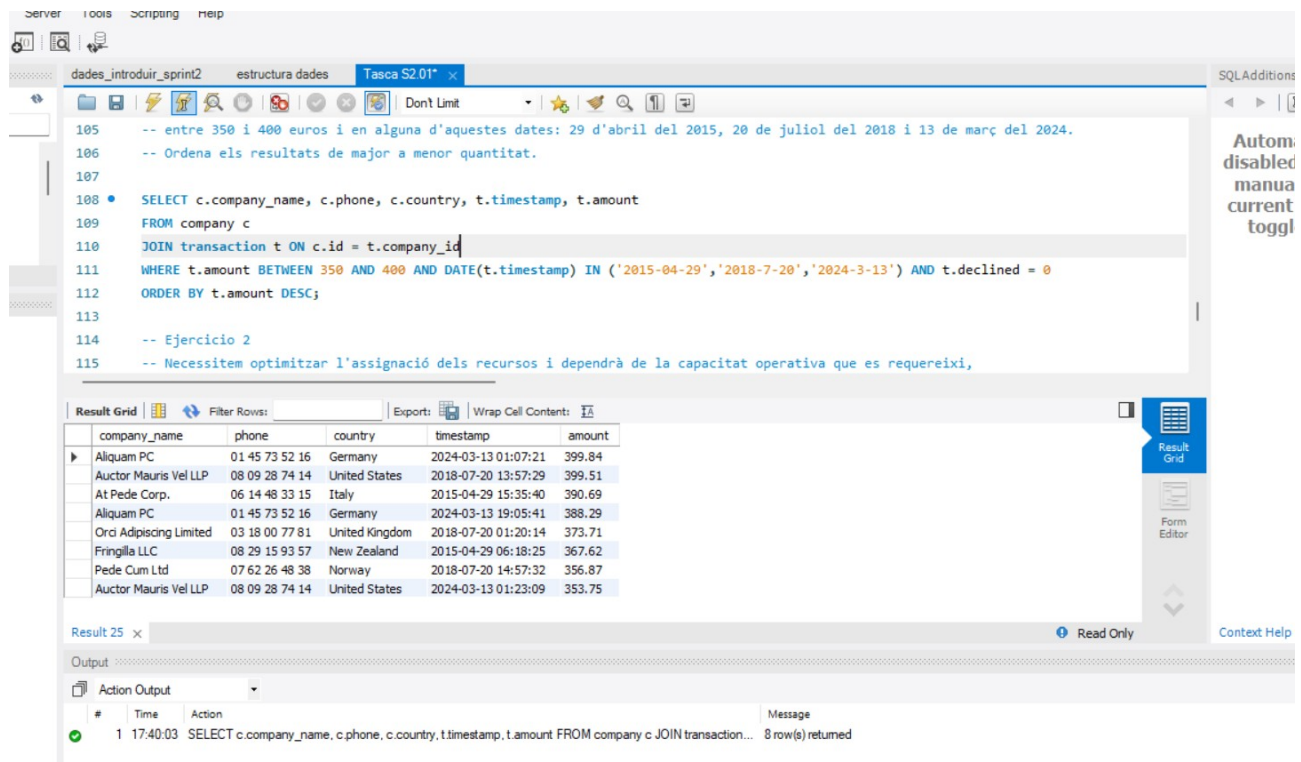
La segunda subconsulta es influente en la primera para llegar a obtener el resultado de la consulta principal.

En la 2 subconsulta solicitamos que el pais sea el mismo que el de NON INSTITUTE y en la primera subconsulta pedimos que el id de la tabla company sea igual que el company_id de la tabla de transactions de esta forma todos los company_id que nos da de resultado final coinciden con el pais de NON INSTITUTE y obtenemos el resultado.

Nivell 3

Exercici 1

Presenta el nom, telèfon, país, data i amount, d'aquelles empreses que van realitzar transaccions amb un valor comprès entre 350 i 400 euros i en alguna d'aquestes dates: 29 d'abril del 2015, 20 de juliol del 2018 i 13 de març del 2024. Ordena els resultats de major a menor quantitat.



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

```
105 -- entre 350 i 400 euros i en alguna d'aquestes dates: 29 d'abril del 2015, 20 de juliol del 2018 i 13 de març del 2024.
106 -- Ordena els resultats de major a menor quantitat.
107
108 • SELECT c.company_name, c.phone, c.country, t.timestamp, t.amount
109 FROM company c
110 JOIN transaction t ON c.id = t.company_id
111 WHERE t.amount BETWEEN 350 AND 400 AND DATE(t.timestamp) IN ('2015-04-29', '2018-7-20', '2024-3-13') AND t.declined = 0
112 ORDER BY t.amount DESC;
113
114 -- Ejercicio 2
115 -- Necesitem optimitzar l'assignació dels recursos i dependrà de la capacitat operativa que es requereixi,
```

The results grid shows the following data:

company_name	phone	country	timestamp	amount
Aliquam PC	01 45 73 52 16	Germany	2024-03-13 01:07:21	399.84
Auctor Mauris Vel LLP	08 09 28 74 14	United States	2018-07-20 13:57:29	399.51
At Pede Corp.	06 14 48 33 15	Italy	2015-04-29 15:35:40	390.69
Aliquam PC	01 45 73 52 16	Germany	2024-03-13 19:05:41	388.29
Orci Adipiscing Limited	03 18 00 77 81	United Kingdom	2018-07-20 01:20:14	373.71
Fringilla LLC	08 29 15 93 57	New Zealand	2015-04-29 06:18:25	367.62
Pede Cum Ltd	07 62 26 48 38	Norway	2018-07-20 14:57:32	356.87
Auctor Mauris Vel LLP	08 09 28 74 14	United States	2024-03-13 01:23:09	353.75

The output pane shows the following message:

```
1 17:40:03 SELECT c.company_name, c.phone, c.country, t.timestamp, t.amount FROM company c JOIN transaction... 8 row(s) returned
```

En este ejercicio En la clausula SELECT seleccionamos los campos que se no han solicitado, seguidamente realizamos un join entre las tablas company y transaction y en el WHERE realizamos las 3 condiciones que se nos solicitan, un BETWEEN entre las 2 cantidades, un DATE para las 3 fechas solicitadas y el DECLINED = 0 para saber que la transacción se habia realizado correctamente. Y para ordenarlo de mayor a menor como nos pide el ejercicio hemos realizado un ORDER BY del AMOUNT en orden descendente y en este caso el resultado nos devuelve 8 registros.

Exercici 2

Necessitem optimitzar l'assignació dels recursos i dependrà de la capacitat operativa que es requereixi, per la qual cosa et demanen la informació sobre la quantitat de transaccions que realitzen les empreses, però el departament de recursos humans és exigent i vol un llistat de les empreses on especifiquis si tenen més de 400 transaccions o menys.

The screenshot shows the SQL Server Enterprise Manager interface. The query editor contains the following SQL code:

```
-- Necessitem optimitzar l'assignació dels recursos i dependrà de la capacitat operativa que es requereixi,
-- per la qual cosa et demanen la informació sobre la quantitat de transaccions que realitzen les empreses,
-- però el departament de recursos humans és exigent i vol un llistat de les empreses on especifiquis
-- si tenen més de 400 transaccions o menys.

SELECT c.company_name, COUNT(t.id) AS num_transactions,
CASE
    WHEN COUNT(t.id) > 400 THEN 'Mas de 400'
    ELSE '400 o Menos'
END AS clasificación
FROM company c
JOIN transaction t ON c.id = t.company_id
WHERE t.declined = 0
GROUP BY c.company_name
ORDER BY num_transactions DESC;
```

The Results pane shows the following data:

company_name	num_transactions	clasificación
Ac Fermentum Incorporated	2400	Mas de 400
Nunc Interdum Incorporated	1599	Mas de 400
Donec Fringilla PC	1590	Mas de 400
Mauris Institute	1583	Mas de 400
Aliquet Vel Vulputate Incorporated	1581	Mas de 400

The Output pane shows the following message:

```
1 13:53:16 SELECT c.company_name, COUNT(t.id) AS num_transactions, CASE WHEN COUNT(t.id) > 400 THEN 'Mas de 400' ELSE '400 o Menos' END AS clasificación FROM company c JOIN transaction t ON c.id = t.company_id WHERE t.declined = 0 GROUP BY c.company_name ORDER BY num_transactions DESC; 100 row(s) returned
```

The screenshot shows the SQL Server Enterprise Manager interface. The query editor contains the following SQL code:

```
-- Necessitem optimitzar l'assignació dels recursos i dependrà de la capacitat operativa que es requereixi,
-- per la qual cosa et demanen la informació sobre la quantitat de transaccions que realitzen les empreses,
-- però el departament de recursos humans és exigent i vol un llistat de les empreses on especifiquis
-- si tenen més de 400 transaccions o menys.

SELECT c.company_name, COUNT(t.id) AS num_transactions,
CASE
    WHEN COUNT(t.id) > 400 THEN 'Mas de 400'
    ELSE '400 o Menos'
END AS clasificación
FROM company c
JOIN transaction t ON c.id = t.company_id
WHERE t.declined = 0
GROUP BY c.company_name
ORDER BY num_transactions DESC;
```

The Results pane shows the following data:

company_name	num_transactions	clasificación
Magna A Neque Industries	406	Mas de 400
Dui Quis Institute	400	400 o Menos
Nec Luctus LLC	399	400 o Menos
Fringilla LLC	396	400 o Menos
Lorem Eu Incorporated	378	400 o Menos

The Output pane shows the following message:

```
1 13:53:16 SELECT c.company_name, COUNT(t.id) AS num_transactions, CASE WHEN COUNT(t.id) > 400 THEN 'Mas de 400' ELSE '400 o Menos' END AS clasificación FROM company c JOIN transaction t ON c.id = t.company_id WHERE t.declined = 0 GROUP BY c.company_name ORDER BY num_transactions DESC; 100 row(s) returned
```

Para la realización de este ejercicio en el SELECT solicitamos el nombre de la compañía , un recuento de todas las transacciones e introducimos la expresión condicional CASE para crear una columna nueva basada en las condiciones que propone el ejercicio que llamaremos CLASIFICACION.

Realizaremos un JOIN entre la tabla COMPANY y la tabla TRANSACTION en el WHERE solicitaremos que las que las transacciones esten realizadas, seguidamente realizamos un agrupación por los nombres de las compañías y finalmente ordenamos según el numero de transacciones en orden DESC y obtenemos el resultado.