

NIVELL 1

Ex1

Se realiza un **LEFT JOIN** de *transactions* con *company* especificando en la cláusula **WHERE** que el país sea Germany y que la transacción no se haya declinado.

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

```
#NIVELL 1
#Ex1
SELECT transaction.*
FROM transaction
LEFT JOIN company ON company.id = transaction.company_id
WHERE country = 'Germany' AND declined = 0;
```

The results grid displays a list of transactions for Germany that have not been declined. The columns include id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The first few rows are:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B1D10-5B23-A76C-55F8-C568E49A03D0	CcU-2938	b-2222	275	83.7839	-178.86	2021-07-07 17:43:16	293.57	0
A069F53-965E-A2A8-CE06-CABC4F9D2501	CcU-2959	b-2234	275	1.64819	-158.007	2021-04-15 13:37:18	60.99	0
0466A42E-47C7-8D24-FD01-C06689713128	CcU-4219	b-2302	170	-43.9695	-117.525	2021-07-26 07:29:18	49.53	0

The output pane shows the execution of the query and subsequent aggregate queries, all returning the expected number of rows.

CORRECCIÓN

Se realiza una *SubQuery* en la clausula **WHERE** donde se especifica que el *bussines_id* a de ser igual a los *company_id* de la tabla que se forma a partir de un select de la tabla *company* donde el country = 'Germany'.

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

```
#Ex1 CORRECCIÓN
SELECT transaction.*
FROM transaction
WHERE declined = 0 AND transaction.bussines_id IN (SELECT company_id FROM company WHERE country = 'Germany');
```

The results grid displays a list of transactions for Germany that have not been declined. The columns include id, card_id, bussines_id, timestamp, amount, declined, product_id, user_id, lat, longitude, product_id1, product_id2, product_id3, and product_id4. The first few rows are:

id	card_id	bussines_id	timestamp	amount	declined	product_id	user_id	lat	longitude	product_id1	product_id2	product_id3	product_id4
108B1D10-5B23-A76C-55F8-C568E49A03D0	CcU-2938	b-2222	2021-07-07 17:43:16	293.57	0	59	275	83.7839152128	-178.860333536	59			
A069F53-965E-A2A8-CE06-CABC4F9D2501	CcU-2959	b-2234	2021-04-15 13:37:18	60.99	0	11, 13, 61, 29	275	1.6481916928	-158.0065729536	11	13	61	29
632BEE9E-4725-E15C-7FC4-58507E58D29	CcU-3078	b-2302	2021-08-27 22:13:25	203.23	0	89, 3	275	45.8950729728	-58.70354432	89	3		

The output pane shows the execution of the query and subsequent aggregate queries, all returning the expected number of rows.

Ex2

En este caso se requiere una **SubQUERY**. En la cláusula **WHERE** se especificará mediante un **SELECT** de la cantidad media (**AVG()**) de cada transacción.

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

```
9 #Ex2
10 SELECT DISTINCT(company_name)
11 FROM company
12 JOIN transaction ON company.id = transaction.company_id
13 WHERE declined = 0 AND amount > (SELECT AVG(amount)
14 FROM transaction);
```

The results grid displays a list of company names. The first few rows are:

company_name
>Lorem Eu Incorporated
Non Institute
Ut Semper Foundation
Nunc Interdum Incorporated
Ac Fermentum Incorporated
Enim Conditimentum Ltd
Tortor Nunc Commodo Company
Mollisuada PC
Arcu LLP
Aliquet Diam Limited
Mauris Institute
Sed Nunc Ltd
Fringilla LLC
Neque Tellus Incorporated
Non Magna LLC
Sed LLC
Quam A Felo Industries
Rutrum Non Inc.
Amet Luctus Vulputate Foundat...
Eget Iosum Ltd
Sapien Nunc Pulvinar LLP
Amet Institute
Ac Industries
Wiverna Donec Foundation
Placerat LLP

The output pane at the bottom shows the execution log with the following messages:

#	Time	Action	Message
10	15:49:34	SELECT AVG(amount), country FROM transaction, company WHERE company.id = transaction.company_id AND declined = 0 GROUP BY company...	15 row(s) returned
11	15:53:10	SELECT company_name, SUM(amount) FROM company, (SELECT * FROM transaction WHERE declined = 0 AND amount BETWEEN 100 AND 20...	23 row(s) returned
12	15:55:24	SELECT company_name, SUM(amount) FROM company, transaction WHERE transaction.company_id = company.id GROUP BY company_name O...	100 row(s) returned
13	15:55:44	SELECT company_name, SUM(amount) FROM company, transaction WHERE transaction.company_id = company.id AND declined = 0 GROUP BY ...	100 row(s) returned
14	16:10:27	SELECT DISTINCT(company_name) FROM company, transaction WHERE company.id = transaction.company_id AND declined = 0 AND timestamp ...	5 row(s) returned
15	16:15:48	SELECT transaction * FROM transaction LEFT JOIN company ON company.id = transaction.company_id WHERE country = 'Germany' AND declined ...	111 row(s) returned
16	16:22:26	SELECT DISTINCT(company_name) FROM company JOIN transaction ON company.id = transaction.company_id WHERE declined = 0 AND amount ...	50 row(s) returned

Ex3

En este caso se recuperarán todas las transacciones realizadas por empresas cuyo nombre empieza por la letra c. Se adjuntará toda la info de las transacciones, así como los nombres de las empresas. Se espera que con esta información contabilidad pueda recuperar los datos perdidos.

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

```

13 WHERE amount > (SELECT AVG(amount)
14 FROM transaction);
15
16 #Ex3
17 SELECT transaction.*, company_name
18 FROM company, transaction
19 WHERE company_id = transaction.company_id AND company_name LIKE 'C%';
20

```

The results grid displays the following data:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined	company_name
0002E608-5C9E-01B3-4999-899F43AD735A	CUJ-2959	b-2234	275	9.68811	130.282	2021-04-17 05:30:17	252.47	1	Convalls In Incorporated
AB069F53-965E-A2A8-CE06-CABC4FD92501	CUJ-2959	b-2234	275	1.64819	-158.007	2021-04-15 13:37:18	60.99	0	Convalls In Incorporated
439F4F6A-57C7-3A59-6038-AD713387522	CUJ-3449	b-2514	268	-13.9867	-104.268	2021-03-18 05:42:48	60.93	0	Cras Consulting
ED1CFB7D-E626-CE54-57A1-A0367788E1F	CUJ-3449	b-2514	268	9.75206	-134.718	2021-11-28 23:27:04	230.41	1	Cras Consulting
5C3AB83D-8974-40A6-ABAF-C1F7AA6B7C94	CUJ-3519	b-2554	267	15.1828	165.662	2021-07-23 15:35:14	158.05	1	Cras Vehicula Aliquet Industries
8D895ADD-5501-249C-24DD-2EAE7AEF1C5	CUJ-3519	b-2554	267	8.58711	-34.6616	2021-10-27 19:43:57	181.87	0	Cras Vehicula Aliquet Industries

The output window shows the execution of the query, with messages indicating the number of rows returned for each step.

CORRECCIÓN

Se puede introducir una *SubQuery* para especificar de donde se ha de recuperar las *bussines_id*. En este caso por eso no me parece muy eficiente ya que yo he decido que, dadas las circunstancias del ejercicio, enseñar en nombre de las compañías es necesario, por lo que se tenía que ir a buscar la tabla *company* de todos modos.

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

```

21
22 #Ex3 #XLL
23 SELECT transaction.*, company_name
24 FROM company, transaction
25 WHERE company.company_id = transaction.bussines_id AND company_name LIKE 'C%';
26
27 #Ex3 CORRECCIÓN
28 SELECT transaction.*, company_name
29 FROM company, transaction
30 WHERE company.company_id = transaction.bussines_id AND transaction.bussines_id IN (SELECT company.company_id FROM company WHERE company_name LIKE 'C%');
31

```

The results grid displays the following data:

id	credit_card_id	bussines_id	timestamp	amount	declined	product_id	user_id	lat	longitude	product_id1	product_id2	product_id3	product_id4	company_name
AB069F53-965E-A2A8-CE06-CABC4FD92501	CUJ-2959	b-2234	2021-04-15 13:37:18	60.99	0	11, 13, 61, 29	275	1.6481916928	-158.005729536	11	13	61	29	Convalls In Incorporated
0002E608-5C9E-01B3-4999-899F43AD735A	CUJ-2959	b-2234	2021-04-17 05:30:17	252.47	1	7, 47, 17	275	9.6881091584	130.282389504	7	47	17	17	Convalls In Incorporated
439F4F6A-57C7-3A59-6038-AD713387522	CUJ-3449	b-2514	2021-03-18 05:42:48	60.93	0	79, 17	268	-13.9867411456	-104.268312576	79	17	17	17	Cras Consulting
ED1CFB7D-E626-CE54-57A1-A0367788E1F	CUJ-3449	b-2514	2021-11-28 23:27:04	230.41	1	59, 71, 73	268	9.7520625024	-134.719451392	59	71	73	73	Cras Consulting
8D895ADD-5501-249C-24DD-2EAE7AEF1C5	CUJ-3519	b-2554	2021-10-27 19:43:57	181.87	0	61, 43, 1	267	8.5871097616	-34.6615640236	61	43	1	1	Cras Vehicula Aliquet Industries
5C3AB83D-8974-40A6-ABAF-C1F7AA6B7C94	CUJ-3519	b-2554	2021-07-23 15:35:14	158.05	1	97, 31	267	15.1828036608	165.6621228032	97	31	31	31	Cras Vehicula Aliquet Industries

The output window shows the execution of the query, with messages indicating the number of rows returned for each step.

Ex4

Se hará un **LEFT JOIN** con la condición en la clausula **WHERE** de que el *company_id* sea **NULL**, de esta forma se obtienen las empresas que quedan fuera del **JOIN**. En este caso el **JOIN** es de las tablas *company* y *transaction*, así que el sentido opuesto del **JOIN** nos proporciona las empresas que no tienen datos en las dos tablas, y por tanto las empresas que no han realizado transacciones. En este caso todas las compañías han realizado alguna transacción.

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

```

22 #Ex4
23 SELECT company_name
24 FROM company
25 LEFT JOIN transaction ON company.id = transaction.company_id
26 WHERE company_id IS NULL;
27
28 #NIVELL 2
29

```

The results pane shows the output of the query, which is a single row with the company name 'company_name'.

#	Time	Action	Message
86	19:30:32	SELECT company_name, COUNT(transaction.id) CASE WHEN COUNT(transaction.id) >= 4 THEN 'TRUE' WHEN COUNT(transaction.id) < 4 TH...	100 row(s) returned
87	19:30:49	SELECT transaction.* FROM transaction LEFT JOIN company ON company.id = transaction.company_id WHERE country = 'Germany'	118 row(s) returned
88	19:39:17	SELECT transaction.* FROM transaction LEFT JOIN company ON company.id = transaction.company_id WHERE country = 'Germany'	118 row(s) returned
89	19:39:55	SELECT DISTINCT(company_name) FROM company JOIN transaction ON company.id = transaction.company_id WHERE amount > (SELECT AVG(...	70 row(s) returned
90	19:40:32	SELECT transaction.*, company_name FROM company, transaction WHERE company.id = transaction.company_id AND company_name LIKE 'C%'	6 row(s) returned
91	19:41:04	SELECT transaction.*, company_name FROM company, transaction WHERE company.id = transaction.company_id AND company_name LIKE 'C%'	6 row(s) returned
92	19:41:09	SELECT company_name FROM company LEFT JOIN transaction ON company.id = transaction.company_id WHERE company_id IS NULL	0 row(s) returned

CORRECCIÓN

En forma de *SubQuery* lo que se podría hacer en este caso es seleccionar todos los *business_id* no nulos de la tabla *transaction* y pedirle los *company_name* de la tabla *company* con la condición que el *company_id* no este en la lista que acabamos de crear.

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

```

31
32 #Ex4 PAL
33 SELECT company_name
34 FROM company
35 LEFT JOIN transaction ON company.company_id = transaction.business_id
36 WHERE transaction.business_id IS NULL;
37
38 #Ex4 CORRECCIÓN
39 SELECT company_name
40 FROM company
41 WHERE company_id NOT IN (
42     SELECT business_id
43     FROM transaction
44 );

```

The results pane shows the output of the query, which is a single row with the company name 'company_name'.

#	Time	Action	Message
26	00:09:37	SELECT transaction.*, company_name FROM company, transaction WHERE company.company_id = transaction.business_id AND transaction.bu...	6 row(s) returned
27	00:16:46	SELECT company_name FROM company LEFT JOIN transaction ON company.company_id = transaction.business_id WHERE transaction.bu...	0 row(s) returned
28	00:16:49	SELECT company_name FROM company LEFT JOIN transaction ON company.company_id = transaction.business_id WHERE transaction.bu...	0 row(s) returned
29	00:18:56	SELECT company_name FROM company WHERE id NOT IN (SELECT company_id FROM transaction WHERE company_id IS NOT NULL)	Error Code: 1054. Unknown column 'id' in 'IN/ALL/ANY subquery'
30	00:19:44	SELECT company_name FROM company LEFT JOIN transaction ON company_id = transaction.business_id	0 row(s) returned
31	00:20:05	SELECT company_name FROM company LEFT JOIN transaction ON company_id = transaction.business_id WHERE transaction.bu...	0 row(s) returned
32	00:20:07	SELECT company_name FROM company LEFT JOIN transaction ON company_id = transaction.business_id WHERE transaction.bu...	0 row(s) returned

NIVELL 2

Ex1

Aquí se realizará una **SubQuery** en la cláusula **WHERE** donde se especifica que el **country** ha de ser el mismo que el de la empresa **Nos Institute**.

```

28 #NIVELL 2
29
30 #Ex1
31 • SELECT transaction.*
32 FROM transaction, company
33 WHERE transaction.company_id = company.id AND country = (SELECT country FROM company WHERE company_name = 'Nos Institute');

```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
28928E1C-EC14-A760-5A75-871477649D6A	CcU-2980	b-2246	275	-41.0496	16.1685	2021-08-10 08:14:49	383.73	0
ACD2011A-A2B1-C365-41E1-2A800C65147A	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
8C2B9A38-77B4-2KCD-1FEB-14CED663E773	CcU-3092	b-2310	275	-78.0295	18.5295	2021-10-18 07:27:35	477.95	1
14785D24-678A-C7B8-4CE3-4D7C2DEB548B	CcU-2994	b-2326	133	66.2672	172.399	2021-08-09 00:58:07	309.45	0
152588C2-028D-D684-4B66-91EDF393EBFF	CcU-2994	b-2326	126	-67.0189	-141.672	2021-07-05 03:10:00	395.43	0
18636858-A2E8-7C69-D9C9-C5453DAFDC3B	CcU-2994	b-2326	131	70.2543	-13.1336	2021-07-06 08:48:46	195.06	0
204188E5-8804-8E9B-8D7A-A95C18FDBF5C	CcU-2994	b-2326	126	-79.1145	1.51481	2022-01-03 15:59:29	479.52	0
23988576-6C0E-137A-C3F6-3180A188A2D3	CcU-2994	b-2326	126	23.6174	137.222	2021-08-25 06:04:05	43.90	0
267C4A86-78A7-1C5E-0738-3824983C70D0	CcU-2994	b-2326	126	-17.5259	104.915	2021-10-01 21:08:53	122.63	0
314DCC3E-4B37-4E64-EE2D-29CA834B18D0	CcU-2994	b-2326	126	-67.8476	-119.578	2021-04-06 17:24:44	91.59	0
3578888E-7B1D-8887-38C7-2088673AA31E	CcU-2994	b-2326	126	87.0665	-22.7339	2021-07-26 22:59:24	303.60	0
360C7814-F74F-843A-0946-AB38D2683C86	CcU-2994	b-2326	116	-7.93005	-79.0733	2021-08-21 10:19:58	494.82	0
391E1CFD-D653-E4B8-A729-F2E893247858	CcU-2994	b-2326	118	60.5512	103.904	2021-10-09 00:50:38	271.27	0
3C4D7CA4-A402-8941-625A-064CA33528E8	CcU-2994	b-2326	117	23.1627	-10.381	2021-04-25 19:11:52	441.27	0
3ED34AC2-01CA-60E9-208D-4B553A7912F7	CcU-4219	b-2326	137	69.0549	96.5023	2021-03-29 11:38:38	478.54	0
3FEC4131-9EED-2E46-1776-82674C8D00B8	CcU-2994	b-2326	126	-7.28469	179.063	2021-07-25 04:59:17	393.42	0
41C13F08-473E-D475-3E14-760F3F79B42	CcU-4219	b-2326	138	-51.7289	55.3294	2022-01-16 16:48:55	45.63	0
41EC591D-808E-085C-F0F9-08BC3DE7A618	CcU-2994	b-2326	126	5.38535	-78.3917	2022-01-06 07:47:39	335.54	0
4217E857-1D51-5D25-8645-EC24D23C8759	CcU-2994	b-2326	129	65.4733	-20.2735	2021-04-23 13:07:58	193.33	0
45CA2E45-4942-CAC4-7683-EBB64D52383E	CcU-4219	b-2326	140	85.7055	36.1497	2021-04-29 06:17:02	149.89	0
49CD105D-E569-3733-C9C9-3908ED7E1838	CcU-3393	b-2326	108	-47.8783	168.275	2021-12-24 23:43:59	411.81	0
4A8E3D77-6F66-1D33-1C55-22663D37C199	CcU-3120	b-2326	273	-63.6087	69.6023	2021-11-13 04:18:28	38.02	1
4B74994B-D123-7C7E-A9CE-7DA26F911F55	CcU-2994	b-2326	128	25.6024	53.4543	2022-02-12 14:27:28	304.43	0
4D8E4891-1711-7D30-E4CE-645ABDF59A43	CcU-2994	b-2326	120	5.71915	-155.369	2021-12-26 01:49:19	365.83	0

Result 13

Output

Action Output

#	Time	Action	Message
24	16:35:00	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned
25	16:35:00	SELECT AVG(amount), country FROM transaction, company WHERE transaction.company_id = company.id GROUP BY country HAVING AVG(amount) > 4;	5 row(s) returned
26	16:35:00	SELECT company_name, COUNT(transaction.id), CASE WHEN COUNT(transaction.id) >= 4 THEN TRUE WHEN COUNT(transaction.id) < 4 THEN FALSE;	100 row(s) returned
27	16:35:04	SELECT transaction.* FROM transaction, company WHERE transaction.company_id = company.id AND country = (SELECT country FROM company WHERE company_name = 'Nos Institute');	0 row(s) returned
28	16:35:25	SELECT * FROM company WHERE company_name = 'Nos Institute';	0 row(s) returned
29	16:35:44	SELECT distinct(company_name) FROM company;	100 row(s) returned
30	16:36:09	SELECT transaction.* FROM transaction, company WHERE transaction.company_id = company.id AND country = (SELECT country FROM company WHERE company_name = 'Nos Institute');	100 row(s) returned
31	16:40:11	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned
32	16:40:20	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned
33	16:40:26	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned

Ex2

Este caso es exactamente igual que el anterior, pero en este caso la **SubQuery** nos condiciona el **amount**, siendo este la cantidad máxima adquirida mediante la función **MAX()**.

```

34
35 #Ex2
36 • SELECT company_name, amount
37 FROM company, transaction
38 WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction);
39

```

company_name	amount
Nunc Interdum Incorporated	499.23

Result 16

Output

Action Output

#	Time	Action	Message
27	16:35:04	SELECT transaction.* FROM transaction, company WHERE transaction.company_id = company.id AND country = (SELECT country FROM company WHERE company_name = 'Nos Institute');	0 row(s) returned
28	16:35:25	SELECT * FROM company WHERE company_name = 'Nos Institute';	0 row(s) returned
29	16:35:44	SELECT distinct(company_name) FROM company;	100 row(s) returned
30	16:36:09	SELECT transaction.* FROM transaction, company WHERE transaction.company_id = company.id AND country = (SELECT country FROM company WHERE company_name = 'Nos Institute');	100 row(s) returned
31	16:40:11	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned
32	16:40:20	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned
33	16:40:26	SELECT company_name, amount FROM company.transaction WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction);	1 row(s) returned

NIVELL 3

Ex1

En este caso, como se ha de usar una función de agrupación como es **AVG()**, se tendrá que usar la cláusula **HAVING** para establecer la condición, que en este caso es que el gasto medio del país sea superior al gasto medio general.

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

```
40 #NIVELL 3
41
42 #Ex1
43 SELECT country
44 FROM transaction, company
45 WHERE transaction.company_id = company.id
46 GROUP BY country
47 HAVING AVG(amount) > (SELECT AVG(amount) FROM transaction);
48
```

The results pane shows a table with the following data:

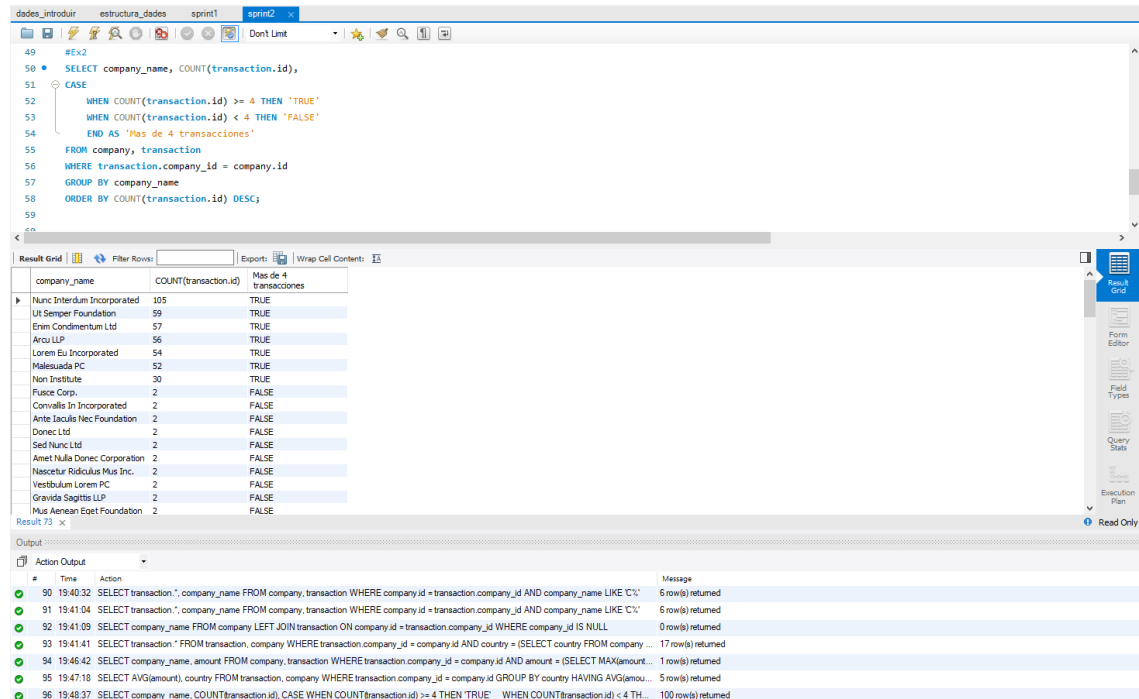
country
United States
United Kingdom
Sweden
Ireland
Canada

The bottom pane shows the execution log with the following messages:

```
32 16:40:20 SELECT company_name, amount FROM company, transaction WHERE transaction.company_id = company.id AND amount = (SELECT MAX(amount) FROM transaction) 1 row(s) returned
33 16:40:26 SELECT company_name, amount FROM company, transaction WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction) 1 row(s) returned
34 16:44:00 SELECT company_name FROM company, transaction WHERE transaction.company_id = company.id AND declined = 0 AND amount = (SELECT MAX(amount) FROM transaction) 1 row(s) returned
35 16:45:21 SELECT country FROM transaction, company WHERE transaction.company_id = company.id AND AVG(amount) > (SELECT AVG(amount) FROM transaction) Error Code: 1111. Invalid use of group function
36 16:47:07 SELECT AVG(amount), country FROM transaction, company WHERE transaction.company_id = company.id GROUP BY country HAVING AVG(amount) > (SELECT AVG(amount) FROM transaction) 5 row(s) returned
37 16:47:33 SELECT country FROM transaction, company WHERE transaction.company_id = company.id GROUP BY country HAVING AVG(amount) > (SELECT AVG(amount) FROM transaction) 5 row(s) returned
38 16:48:02 SELECT country FROM transaction, company WHERE transaction.company_id = company.id GROUP BY country HAVING AVG(amount) > (SELECT AVG(amount) FROM transaction) 5 row(s) returned
```

Ex2

Para proporcionar la información como nos la piden usaremos la cláusula **CASE**, que nos devolverá una nueva columna los valores de la cual dependerán de la condición que nosotros determinemos, en este caso que la empresa hay realizado más de 4 transacciones.



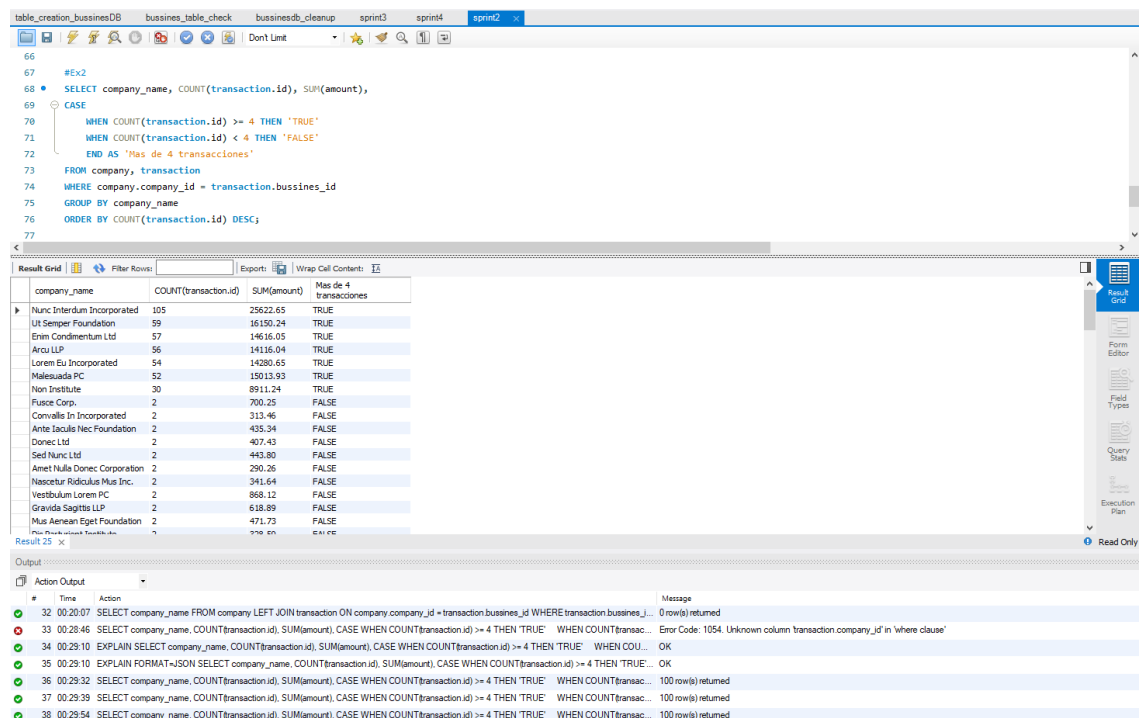
```

49 #Ex2
50 SELECT company_name, COUNT(transaction.id),
51 CASE
52 WHEN COUNT(transaction.id) >= 4 THEN 'TRUE'
53 WHEN COUNT(transaction.id) < 4 THEN 'FALSE'
54 END AS 'Mas de 4 transacciones'
55 FROM company, transaction
56 WHERE transaction.company_id = company.id
57 GROUP BY company_name
58 ORDER BY COUNT(transaction.id) DESC;
59

```

company_name	COUNT(transaction.id)	Mas de 4 transacciones
Nunc Interdum Incorporated	105	TRUE
Ut Semper Foundation	59	TRUE
Enim Conditum Ltd	57	TRUE
Atro LLP	56	TRUE
Lorem Eu Incorporated	54	TRUE
Malesuada PC	52	TRUE
Non Institute	30	TRUE
Fusce Corp.	2	FALSE
Convallis In Incorporated	2	FALSE
Ante Taculis Nec Foundation	2	FALSE
Donec Ltd	2	FALSE
Sed Nunc Ltd	2	FALSE
Amet Nulla Donec Corporation	2	FALSE
Nascetur Ridiculus Mus Inc.	2	FALSE
Vestibulum Lorem PC	2	FALSE
Gravida Sagittis LLP	2	FALSE
Mus Aenean Eget Foundation	2	FALSE

Para mostrar el monto de la empresa se puede poner cualquier función aplicada a *amount* ya que tenemos un **GROUP BY** con *company_name*. En este caso utilizaremos **SUM()**, pero se podría hacer también con **AVG()**.



```

66
67 #Ex2
68 SELECT company_name, COUNT(transaction.id), SUM(amount),
69 CASE
70 WHEN COUNT(transaction.id) >= 4 THEN 'TRUE'
71 WHEN COUNT(transaction.id) < 4 THEN 'FALSE'
72 END AS 'Mas de 4 transacciones'
73 FROM company, transaction
74 WHERE company.company_id = transaction.bussines_id
75 GROUP BY company_name
76 ORDER BY COUNT(transaction.id) DESC;
77

```

company_name	COUNT(transaction.id)	SUM(amount)	Mas de 4 transacciones
Nunc Interdum Incorporated	105	25622.65	TRUE
Ut Semper Foundation	59	16150.24	TRUE
Enim Conditum Ltd	57	14616.05	TRUE
Atro LLP	56	14116.04	TRUE
Lorem Eu Incorporated	54	14280.65	TRUE
Malesuada PC	52	15013.93	TRUE
Non Institute	30	8911.24	TRUE
Fusce Corp.	2	700.25	FALSE
Convallis In Incorporated	2	313.46	FALSE
Ante Taculis Nec Foundation	2	435.34	FALSE
Donec Ltd	2	407.43	FALSE
Sed Nunc Ltd	2	443.80	FALSE
Amet Nulla Donec Corporation	2	290.26	FALSE
Nascetur Ridiculus Mus Inc.	2	341.64	FALSE
Vestibulum Lorem PC	2	868.12	FALSE
Gravida Sagittis LLP	2	618.89	FALSE
Mus Aenean Eget Foundation	2	471.73	FALSE