

SPRINT #2

RELATIONAL DATABASES AND INTRODUCTION TO SQL

student:
Leo Kalugin



Date: 03/10/2024

SUMMARY

In this sprint, I will review the basic concepts of working with relational databases. I will begin hands-on experience with a database containing information from a company dedicated to selling products online. In this activity, I will focus on data related to completed transactions and the corporate information of the companies involved.

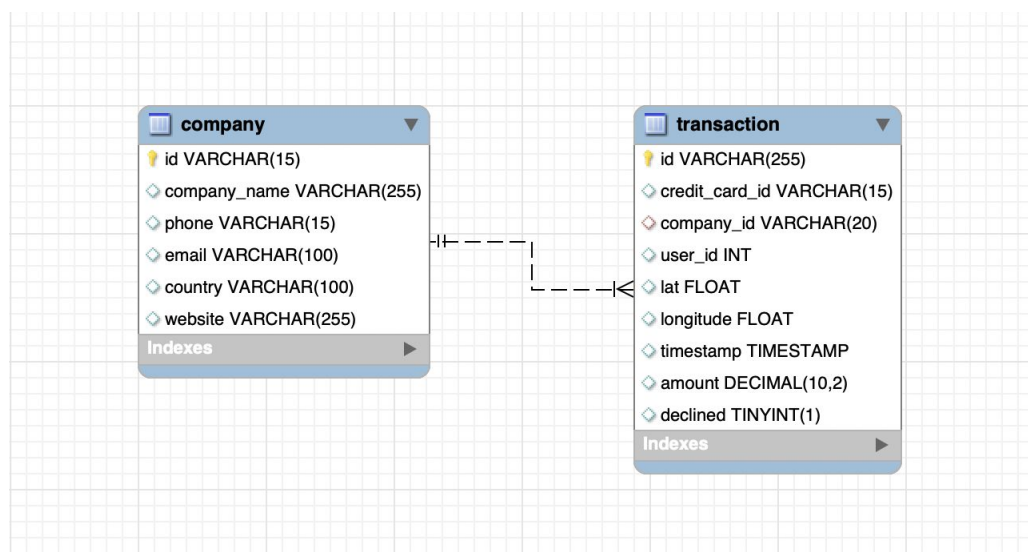
RESULT

In this folder on the GitHub repository, you will find: the file **S2_01.sql**, which contains all the scripts; **transactions_schema.pdf**, which provides the database schema (also shown in this presentation); and this file, **Sprint_2.pdf**, which includes screenshots of the work environment showing the queries I executed and the results obtained for each exercise:

https://github.com/leocareer/DA_specialization/tree/main/Sprint_02

LEVEL 1 EXERCISE 1

From the attached documents ('data_structure.sql' and 'db_data.sql'), import the two tables. Show the main features of the diagram you have created and explain all the tables and variables that exist. Be sure to include a diagram that illustrates the relationships between the various tables and variables.



This database consists of two tables: 'company' and 'transaction'.

1. 'company' contains information about companies involved in the transactions, each company is uniquely identified by an 'id':

- id (varchar(15)): primary key, uniquely identifies each company;
- company_name (varchar(255)): name of the company;
- phone (varchar(15)): contact phone number of the company;
- email (varchar(100)): email address of the company;
- country (varchar(100)): country where the company is based;
- website (varchar(255)): website URL of the company.

This table provides 6 variables, representing company-specific information. To store all variables, a variable-length string data type is used with length of 15, 100 and 255 characters for storage depending on the need. The 'id' field links the company table with the transaction table via a foreign key, establishing a one-to-many relationship ('one' for the 'company' table, 'many' for the 'transaction' table).

2. 'transaction' records individual transactions, with each transaction linked to a company:

- id (varchar(255)): primary key, uniquely identifies each transaction;
- credit_card (varchar(15)): the credit card number used for the transaction;
- company_id (varchar(20)): foreign key referencing the 'id' field in the 'company' table, linking the transaction to the relevant company;
- user_id (int): id of the user who made the transaction;
- lat (float): latitude where the transaction occurred;
- longitude (float): longitude where the transaction occurred;
- timestamp (timestamp): date and time when the transaction occurred;
- amount (decimal(10,2)): the amount of money involved in the transaction;
- decline (tinyint(1)): a flag indicating whether the transaction was declined (1 = declined, 0 = successful).

The data types used here also include variable-length strings, integer and float for storing numbers, timestamp for an exact timestamp, decimal(10,2) with up to 10 digits, including 2 decimal places, and tinyint is binary data type is efficient for storing true/false values.

The 'company' table is linked to the transaction table through the 'company_id' foreign key. This defines a one-to-many relationship: one company can have multiple transactions, but each transaction is associated with only one company. The primary key in both tables ('id' in 'company' and 'id' in 'transaction') ensures uniqueness for records in each respective table.

LEVEL 1 EXERCISE 2

Using JOIN you will perform the following queries:

- List of countries that are shopping;
- From how many countries the purchases are made;
- Identify the company with the highest average sales.

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and settings, along with a 'Limit to 5000 rows' dropdown. The SQL editor contains the following query:

```
4 -- List of countries that are shopping
5 • SELECT DISTINCT country 'Countries that are shopping'
6 FROM company AS t1
7 JOIN transaction AS t2
8 ON t1.id = t2.company_id
9 ORDER BY country;
10
```

Below the editor, the 'Result Grid' is displayed with the title 'Countries that are shopping'. It shows a list of 15 countries: Australia, Belgium, Canada, China, France, Germany, Ireland, Italy, Netherlands, New Zealand, Norway, Spain, Sweden, United Kingdom, and United States. The 'Action Output' pane at the bottom shows the execution details for 'Result 3':

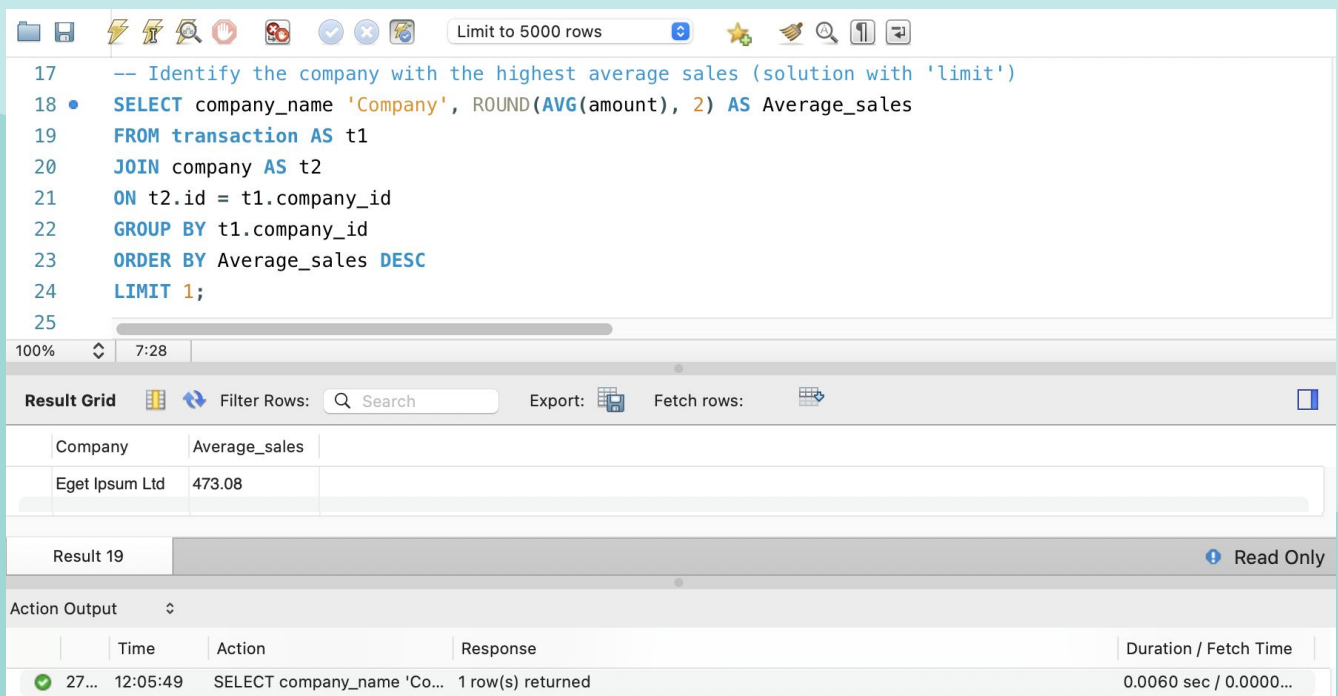
	Time	Action	Response	Duration / Fetch Time
✓ 27...	11:39:42	SELECT DISTINCT country '...	15 row(s) returned	0.0015 sec / 0.00001...

The screenshot shows the same SQL IDE interface with the second query in the editor:

```
12 • SELECT count(DISTINCT country) 'How many countries are shopping'
13 FROM company AS t1
14 JOIN transaction AS t2
15 ON t1.id = t2.company_id;
16
```

The 'Result Grid' is titled 'How many countries are shopping' and displays a single result: 15. The 'Action Output' pane shows the execution details for 'Result 5':

	Time	Action	Response	Duration / Fetch Time
✓ 27...	11:43:44	SELECT count(DISTINCT co...	1 row(s) returned	0.0025 sec / 0.00000...



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 5000 rows' dropdown. The SQL editor contains the following query:

```
17 -- Identify the company with the highest average sales (solution with 'limit')
18 • SELECT company_name 'Company', ROUND(AVG(amount), 2) AS Average_sales
19 FROM transaction AS t1
20 JOIN company AS t2
21 ON t2.id = t1.company_id
22 GROUP BY t1.company_id
23 ORDER BY Average_sales DESC
24 LIMIT 1;
25
```

Below the editor is the 'Result Grid' section, which displays the query results in a table:

Company	Average_sales
Eget Ipsum Ltd	473.08

Below the result grid is the 'Action Output' section, which shows the execution details:

	Time	Action	Response	Duration / Fetch Time
27...	12:05:49	SELECT company_name 'Co...	1 row(s) returned	0.0060 sec / 0.0000...

Using LIMIT in SQL isn't always a good idea because:

1. Performance issues: With large datasets, the query often selects all rows, sorts them, and then trims the result to the specified limit. This can slow down execution for big tables.
2. Inaccurate results: LIMIT can hide important rows, especially without proper sorting, leading to random or non-representative samples.
3. Unpredictable behavior: Without an explicit ORDER BY clause, rows may be returned in a random order, producing unpredictable results.

While LIMIT is useful, it should be used carefully, considering these limitations. So I also implemented a solution without LIMIT:

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a 'Limit to 5000 rows' dropdown. The SQL editor contains the following query:

```
26 -- Identify the company with the highest average sales (solution without 'limit')
27 • SELECT company_name, Average_sales
28 FROM (
29     SELECT company_id, ROUND(AVG(amount), 2) AS Average_sales
30     FROM transaction
31     GROUP BY company_id
32 ) AS t2
33 JOIN company AS t1 ON t1.id = t2.company_id
34 WHERE Average_sales = (
35     SELECT max(avg_amount_t3)
36     FROM (
37         SELECT ROUND(AVG(amount), 2) AS avg_amount_t3 FROM transaction
38         GROUP BY company_id
39     ) AS t3
40 );
41
```

Below the editor, the 'Result Grid' shows the query results:

company_name	Average_sales
Eget Ipsum Ltd	473.08

The 'Action Output' pane at the bottom shows the execution details for the query:

	Time	Action	Response	Duration / Fetch Time
✓ 27...	12:24:58	SELECT company_name, Av...	1 row(s) returned	0.0035 sec / 0.00000...

LEVEL 1 EXERCISE 3

Using only subqueries (without using JOIN):

- Show all transactions made by companies in Germany;
- List the companies that have made transactions for an amount higher than the average of all transactions;
- Companies that do not have registered transactions will be removed from the system, provide the list of these companies.

The screenshot shows a database IDE interface. At the top, there's a toolbar with various icons and a 'Limit to 5000 rows' dropdown. Below the toolbar is a SQL editor with the following code:

```
45 -- Show all transactions made by companies in Germany
46 • SELECT *
47 FROM transaction
48 WHERE company_id IN (
49     SELECT id FROM company
50     WHERE country = 'Germany'
51 );
```

Below the editor is a 'Result Grid' section. It has a search bar and a table with the following data:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B1D1D-5B23-A76C-55...	CcU-2938	b-2222	275	83.7839	-178.86	2021-07-07 17:43:16	293.57	0
EA2C3281-C9C1-A387-44...	CcU-2938	b-2222	275	20.2004	-116.84	2021-05-09 10:25:08	119.36	1
0DD2E608-5C9E-D1B3-4...	CcU-2959	b-2234	275	9.68811	130.282	2021-04-17 05:30:17	252.47	1
AB069F53-965E-A2A8-CE...	CcU-2959	b-2234	275	1.64819	-158.007	2021-04-15 13:37:18	60.99	0
0466A42E-47CF-8D24-FD...	CcU-4219	b-2302	170	-43.9695	-117.525	2021-07-26 07:29:18	49.53	0

Below the result grid is an 'Action Output' section. It shows a table with the following data:

Time	Action	Response	Duration / Fetch Time
27:...	12:27:51	SELECT * FROM transactio...	118 row(s) returned

The 'Response' column shows '118 row(s) returned' and the 'Duration / Fetch Time' column shows '0.0012 sec / 0.00005...'.

Limit to 5000 rows

```
53  -- List the companies that have made transactions for an amount higher than the average of all
54  -- transactions
55  • SELECT company_name 'Companies'
56  FROM company
57  WHERE id IN (
58      SELECT company_id
59      FROM transaction
60      WHERE amount > (
61          SELECT AVG(amount) FROM transaction
62      )
63  ORDER BY company_name;
```

100% 53:60

Result Grid Filter Rows: Search Export:

Companies
A Institute
Ac Fermentum Incorporated
Ac Industries
Aliquam PC
Aliquam PC
Aliquam PC

company 53 Read Only

Action Output

	Time	Action	Response	Duration / Fetch Time
✓ 27...	12:58:01	SELECT company_name 'Co...	70 row(s) returned	0.0027 sec / 0.00002...

Limit to 5000 rows

```
66  -- Companies that do not have registered transactions will be removed from the system, provide the
67  -- list of these companies
68  • SELECT company_name 'Companies'
69  FROM company
70  WHERE NOT EXISTS (
71      SELECT company_id
72      FROM transaction
73      WHERE transaction.company_id = company_id
74  );
```

100% 30:79

Result Grid Filter Rows: Search Export:

Companies

company 62 Read Only

Action Output

	Time	Action	Response	Duration / Fetch Time
✓ 27...	13:07:13	SELECT company_name 'Co...	0 row(s) returned	0.00063 sec / 0.000...

LEVEL 2 EXERCISE 1

Identify the five days that generated the largest amount of revenue for the company from sales. It shows the date of each transaction along with the sales total.

The screenshot shows a SQL IDE interface. At the top, there's a toolbar with various icons and a 'Limit to 5000 rows' dropdown. Below the toolbar, a SQL query is displayed in a text editor:

```
77 -- with 'limit'
78 • SELECT sum(amount) 'Sales total', DATE(timestamp) 'Date'
79 FROM transaction
80 WHERE declined = 0
81 GROUP BY DATE(timestamp)
82 ORDER BY sum(amount) DESC
83 LIMIT 5;
```

Below the query editor, there's a 'Result Grid' section. It includes a 'Filter Rows' search bar, an 'Export' button, and a 'Fetch rows' button. The grid displays the following data:

Sales total	Date
1532.36	2021-12-20
1397.96	2021-04-22
1344.37	2021-05-09
1337.62	2022-02-26
1325.12	2021-03-29

Below the result grid, there's a 'Result 65' tab with a 'Read Only' indicator. At the bottom, there's an 'Action Output' section with a table showing the execution details:

	Time	Action	Response	Duration / Fetch Time
✓ 27...	13:10:02	SELECT sum(amount) 'Sales...	5 row(s) returned	0.0026 sec / 0.00000...

The screenshot shows the DBeaver SQL editor interface. The top toolbar includes icons for file operations, navigation, and execution. A status bar at the top right indicates "Limit to 5000 rows".

```

85 -- the second implementation option with window function without 'limit'
86 • SELECT Sales_total, Sale_date
87 FROM (
88     SELECT sum(amount) AS Sales_total, DATE(timestamp) AS Sale_date,
89     ROW_NUMBER() OVER(ORDER BY sum(amount) DESC) AS ind_amount
90     FROM transaction
91     WHERE declined = 0
92     GROUP BY Sale_date
93 ) AS t
94 WHERE ind_amount <= 5
95 ORDER BY Sales_total;
    
```

Below the editor, the "Result Grid" tab is active, displaying the query results in a table:

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

At the bottom, the "Action Output" tab shows the execution details:

	Time	Action	Response	Duration / Fetch Time
✓ 27 ...	13:18:30	SELECT Sales_total, Sale_d...	5 row(s) returned	0.0035 sec / 0.00000...

LEVEL 2 EXERCISE 2

What is the average sales per country? It presents the results sorted from highest to lowest average.

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and settings, along with a 'Limit to 5000 rows' dropdown. The SQL editor contains the following query:

```
98 -- What is the average sales per country? It presents the results sorted from highest to lowest average.
99 • SELECT country 'Countries', ROUND(AVG(amount), 2) AS Average_sales
100 FROM transactions.company AS t1
101 JOIN transactions.transaction AS t2
102 ON t1.id = t2.company_id
103 WHERE declined = 0
104 GROUP BY country
105 ORDER BY Average_sales DESC;
```

Below the editor is the 'Result Grid' section, which displays the query results in a table:

Countries	Average_sales
United States	287.53
Ireland	285.83
Sweden	276.67
United Kingdom	271.77
Canada	261.94
Belgium	255.22

Below the result grid is the 'Action Output' section, which shows the execution details:

	Time	Action	Response	Duration / Fetch Time
28...	13:53:47	SELECT country 'Countries',...	15 row(s) returned	0.0045 sec / 0.00001...

LEVEL 2 EXERCISE 3

In your company, a new project is being considered to launch some advertising campaigns to compete with the 'Non Institute' company. For this, they ask you for the list of all transactions carried out by companies that are located in the same country as this company.

- Display the list by applying JOIN and subqueries;
- Display the listing by applying only subqueries.

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and settings, along with a 'Limit to 5000 rows' dropdown. The SQL editor contains the following query:

```
110 -- with join
111 • SELECT *
112 FROM transaction AS t1
113 JOIN company AS t2
114 ON t2.id = t1.company_id
115 WHERE country = (
116     SELECT country FROM company
117     WHERE company_name LIKE 'Non Institute'
118 )
119 AND company_name <> 'Non Institute'
120 ORDER BY company_name;
```

Below the editor is the 'Result Grid' section, which includes a search bar and an 'Export' button. It displays a table with 12 columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined, id, and company_n. The table contains 7 rows of data.

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined	id	company_n
2A5A3001-104F-1D1F-78...	CcU-3169	b-2354	272	-67.2525	-142.557	2022-02-27 18:35:15	30.76	0	b-2354	Ac Libero In
9679E769-32DC-2591-B8...	CcU-3169	b-2354	272	47.6643	130.202	2021-06-28 11:22:10	186.34	1	b-2354	Ac Libero In
6D69D98A-F18A-99BD-B...	CcU-3204	b-2374	272	-52.3724	70.1522	2021-05-26 02:33:06	144.33	1	b-2374	Amet Faucit
E5078B1B-9591-E204-CC...	CcU-3204	b-2374	272	69.5434	98.8783	2021-09-06 01:29:42	220.85	0	b-2374	Amet Faucit
1479B3D2-B7BA-C7BB-4...	CcU-2994	b-2326	133	66.2672	172.399	2021-08-09 00:58:07	309.45	0	b-2326	Enim Condi
152598C2-029D-D684-4B...	CcU-2994	b-2326	126	-67.0189	-141.672	2021-07-05 03:10:00	395.43	0	b-2326	Enim Condi

Below the result grid is the 'Action Output' section, which shows a table with 5 columns: Time, Action, Response, and Duration / Fetch Time. It contains one row of data:

Time	Action	Response	Duration / Fetch Time
27... 13:28:08	SELECT * FROM transaction...	70 row(s) returned	0.0021 sec / 0.00005...

The screenshot shows a database query editor interface. The top toolbar includes icons for file operations, a 'Limit to 5000 rows' dropdown, and other utility icons. The SQL editor contains a query starting with a comment '-- with only subqueries'. The query selects all columns from a table named 'transaction' where the company_id is in a subquery. The subquery selects company IDs from the 'company' table where the country is from another subquery. This inner subquery selects countries from the 'company' table where the company name is like 'Non Institute'. The main query also filters for company names not equal to 'Non Institute'. The interface shows the query is executed, and the 'Result Grid' displays the first few rows of the result set. Below the grid, the 'Action Output' panel shows the execution details.

```
122 -- with only subqueries
123 • SELECT *
124 FROM transaction
125 WHERE company_id IN (
126     SELECT id FROM company
127     WHERE country = (
128         SELECT country FROM company
129         WHERE company_name LIKE 'Non Institute'
130     )
131     AND company_name <> 'Non Institute'
132 );
```

100% 42:129

Result Grid Filter Rows: Search Edit: Export/Import:

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
2B928E1C-EC14-A760-0A...	CcU-2980	b-2246	275	-41.0496	161.685	2021-08-10 08:14:49	383.73	0
ACD2011A-A2B1-C365-41...	CcU-2980	b-2246	275	-54.4792	-82.7974	2022-03-05 20:41:20	60.07	1
4334349E-CEB0-3D68-A4...	CcU-3092	b-2310	275	-20.4859	150.87	2021-05-03 22:37:23	458.74	0
BC0B0A28-77B4-28CD-1...	CcU-2980	b-2246	275	-78.0205	18.5205	2021-10-18 07:07:05	477.05	1

transaction 85 Apply

Action Output

	Time	Action	Response	Duration / Fetch Time
✓ 27...	13:39:10	SELECT * FROM transaction...	70 row(s) returned	0.0014 sec / 0.00007...

LEVEL 3 EXERCISE 1

It presents the name, telephone, country, date and amount of those companies that made transactions with a value between 100 and 200 euros and on any of these dates: April 29, 2021, July 20, 2021 and March 13, 2022. Sort the results from highest to lowest amount.

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 5000 rows' dropdown. The SQL editor contains a query with line numbers 135 to 142. Below the editor, the 'Result Grid' shows 5 rows of data with columns: company_name, phone, country, date, and amount. The 'Action Output' pane at the bottom shows a successful execution of the query at 13:43:01, returning 5 rows in 0.0019 seconds.

```
135 -- It presents the name, telephone, country, date and amount of those companies that made transactions
136 -- with a value between 100 and 200 euros and on any of these dates: April 29, 2021, July 20, 2021 and
137 -- March 13, 2022. Sort the results from highest to lowest amount.
138 • SELECT company_name, phone, country, DATE(timestamp) 'date', amount
139 FROM company AS t1
140 JOIN transaction AS t2
141 ON t1.id = t2.company_id
142 WHERE DATE(timestamp) IN ('2021-04-29', '2021-07-20', '2022-03-13')
143 AND amount BETWEEN 100 AND 200
144 ORDER BY amount DESC;
```

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

Result 87 Read Only

Time	Action	Response	Duration / Fetch Time
27... 13:43:01	SELECT company_name, ph...	5 row(s) returned	0.0019 sec / 0.00001...

LEVEL 3 EXERCISE 2

We need to optimize the allocation of resources and it will depend on the operational capacity that is required, so they ask you for the information about the amount of transactions that the companies carry out, but the HR department is demanding and wants a list of the companies where you specify if they have more than 4 transactions or less.

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 5000 rows' dropdown. The SQL editor contains the following code:

```
145 -- We need to optimize the allocation of resources and it will depend on the operational capacity that
    is required, so they ask you for the information about the amount of transactions that the companies
    carry out, but the HR department is demanding and wants a list of the companies where you specify if
    they have more than 4 transactions or less.
146 • SELECT company_name,
147     CASE
148         WHEN count(t2.id) >= 4 THEN '>= 4 transactions'
149         ELSE '< 4 transactions'
150     END AS transaction_count
151 FROM transactions.company AS t1
152 JOIN transactions.transaction AS t2
153 ON t1.id = t2.company_id
154 GROUP BY t1.id
155 ORDER BY transaction_count DESC;
```

Below the editor is the 'Result Grid' showing the query results. It has a 'Filter Rows' search bar and an 'Export' button. The results are as follows:

company_name	transaction_count
Non Institute	>= 4 transactions
Malesuada PC	>= 4 transactions
Lorem Eu Incorporated	>= 4 transactions
Ut Sempur Foundation	>= 4 transactions
Enim Condimentum Ltd	>= 4 transactions
Nunc Interdum Incorporated	>= 4 transactions
Arcu LLP	>= 4 transactions
Amet Nulla Donec Corporation	< 4 transactions
Necetur Biddulus Mus Inc	< 4 transactions

At the bottom, the 'Action Output' panel shows the execution details:

	Time	Action	Response	Duration / Fetch Time
28...	13:47:51	SELECT company_name, C...	100 row(s) returned	0.0028 sec / 0.00003...