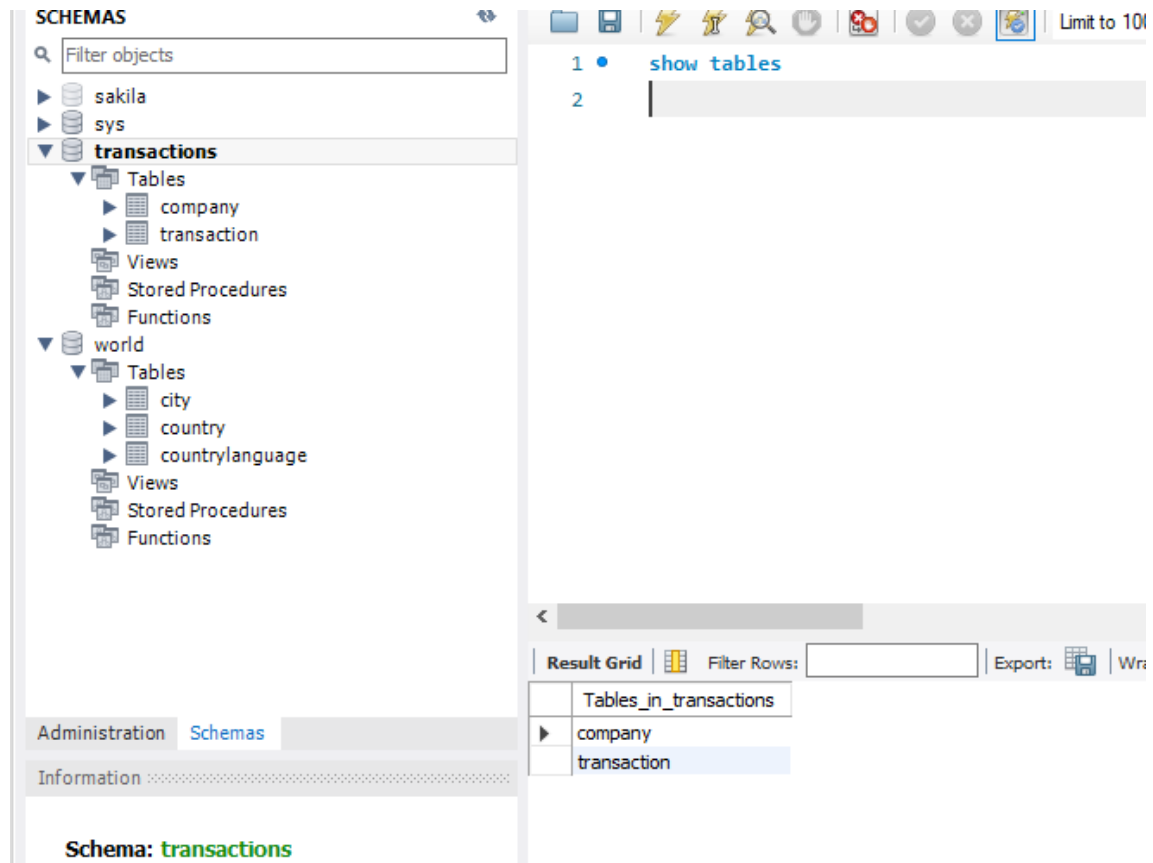


NIVEL 1

EJERCICIO 1

En este ejercicio se pide que nos muestre las tablas que hemos cargado en el programa y hagamos un diagrama en el que se muestren las partes comunes de las dos




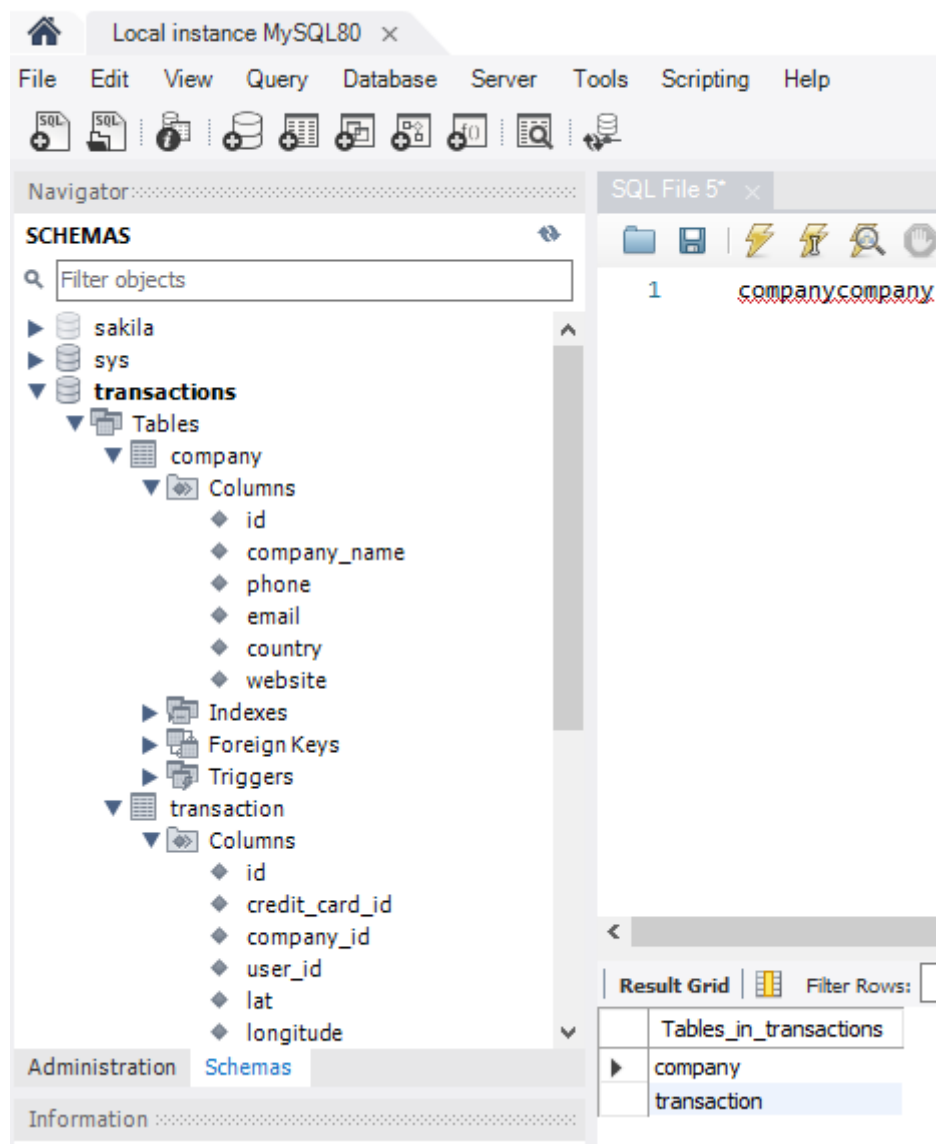
Use la función SHOW TABLES lo que hace que me aparezcan dos tablas una de compañía y otra de Transacciones, ambas están compuesta a su vez por columnas.

RESULTADO: se muestran las tablas en la pantalla Company y transaction

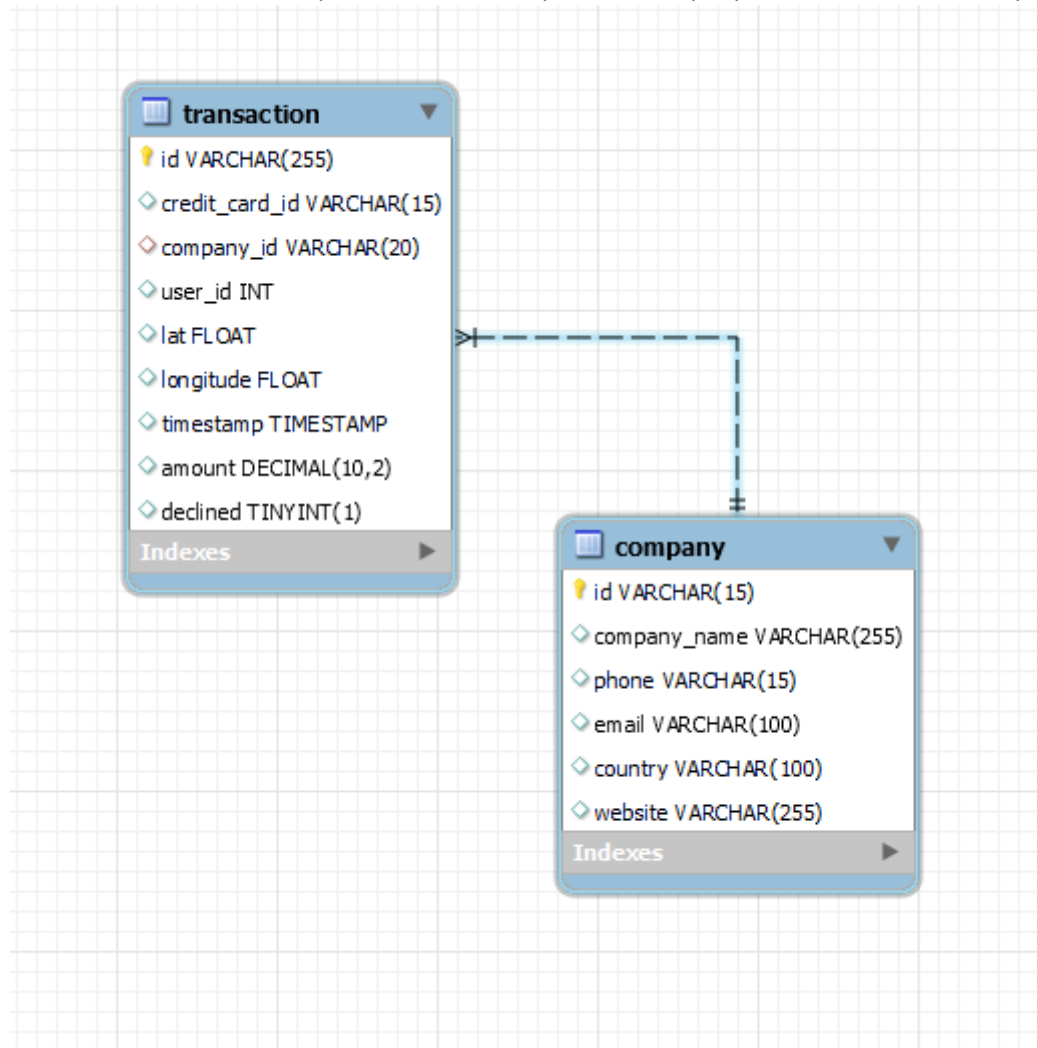
Company tiene por columnas: ID(número de identificación), nombre de la compañía, Phone(número de teléfono), email, country(país) y dirección de sitio web.

Transaction tiene por columnas: ID(numero de identificación),Credit card, company ID(numero de identificación de la empresa), User ID(numero de identificación del usuario),LAT(latitud),longitud,timestamp, amount , declined.

 MySQL Workbench



Luego creamos un diagrama en el cual se muestra las interacciones entre company y transaccion, las dos tienen en común que en el index muestra como columna especial el ID de la compañía la cual es la misma en las dos tablas y en caso de que se cambiase este tendría efecto en las dos tablas, aparte de eso no hay nada más que pueda afectar a las dos por igual



EJERCICIO 2

En este ejercicio nos pide que el programa nos de la base de datos de Company ordenando el nombre de las compañías alfabéticamente y aparte que nos de la información de que país son y su dirección de correo electrónico.

The screenshot shows a database management interface. On the left, a tree view displays the database schema, including tables like 'company'. The main area shows a SQL query: `SELECT company_name, email, country FROM company ORDER BY company_name`. Below the query, a 'Result Grid' displays the data. The table has three columns: 'company_name', 'email', and 'country'. The data is sorted alphabetically by company name.

company_name	email	country
A Institute	metus.aliquam@google.edu	Belgium
Ac Fermentum Incorporated	donec.porttitor.tellus@yahoo.net	Germany
Ac Industries	ipsum@yahoo.com	Germany
Ac Libero Inc.	mollis.lectus@protonmail.ca	United Kingdom
Aliquam Erat Volutpat LLP	pede.nunc@icloud.net	Italy

He usado el comando select sobre los nombres de las compañías dirección de email y país, esta información la sacamos de la tabla Company mediante el comando FROM y aparte le pedimos que nos ordenen los nombres de las compañías alfabéticamente mediante el comando ORDER BY sobre la columna Company_name

Se puede ver en la imagen que los nombres de las compañías están ordenado alfabéticamente

EJERCICIO 3

En este ejercicio lo que tenemos que hacer es que el programa nos de los nombres de los países que están realizando compras, lo que hice fue unir las dos tablas (Company,transaction) mediante **SELECT DISTINCT** para seleccionar valores únicos que en este caso es country de la tabla Company, usamos **FROM** para especificar que estamos usando los datos de la tabla COMPANY y la unión de las dos tablas se produce en el comando **INNER JOIN transaction ON company.id = transaction.company_id**. Lo que hace es tratar solamente los datos que hay en común en las dos tablas que en este caso es **Company_id**. Al ejecutar el programa nos dará el nombres de los países que han tenido alguna transacción con nuestra empresa

The screenshot shows a database management interface. On the left, a 'SCHEMAS' tree displays the database structure, including tables 'company' and 'transaction'. The 'transaction' table is selected, showing its columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The main window displays a SQL query:

```
1 SELECT DISTINCT company.country
2 FROM company
3 INNER JOIN transaction ON company.id = transaction.company_id;
4
```

Below the query, the 'Result Grid' shows the output of the query, which is a list of countries: China, Canada, France, Netherlands, and Spain.

country
China
Canada
France
Netherlands
Spain

RESULTADO: China , Canada, Francia, Países Bajos, Spain Estos son los países que están realizando compras

Ejercicio 4

En este ejercicio lo que nos pide es que el programa nos de los numero de Compañías que han tenido alguna transacción con nosotros.

The screenshot shows a SQL IDE interface. On the left, the 'SCHEMAS' pane displays a tree view with 'sakila' and 'sys' schemas. Under 'sakila', there is a 'transactions' folder containing 'company' and 'transaction' tables. The 'transaction' table's columns are listed: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The main editor pane shows a SQL query:

```
1 • Select count(Distinct company.country) from company
2 inner join transaction on company.id= transaction.company_id
```

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

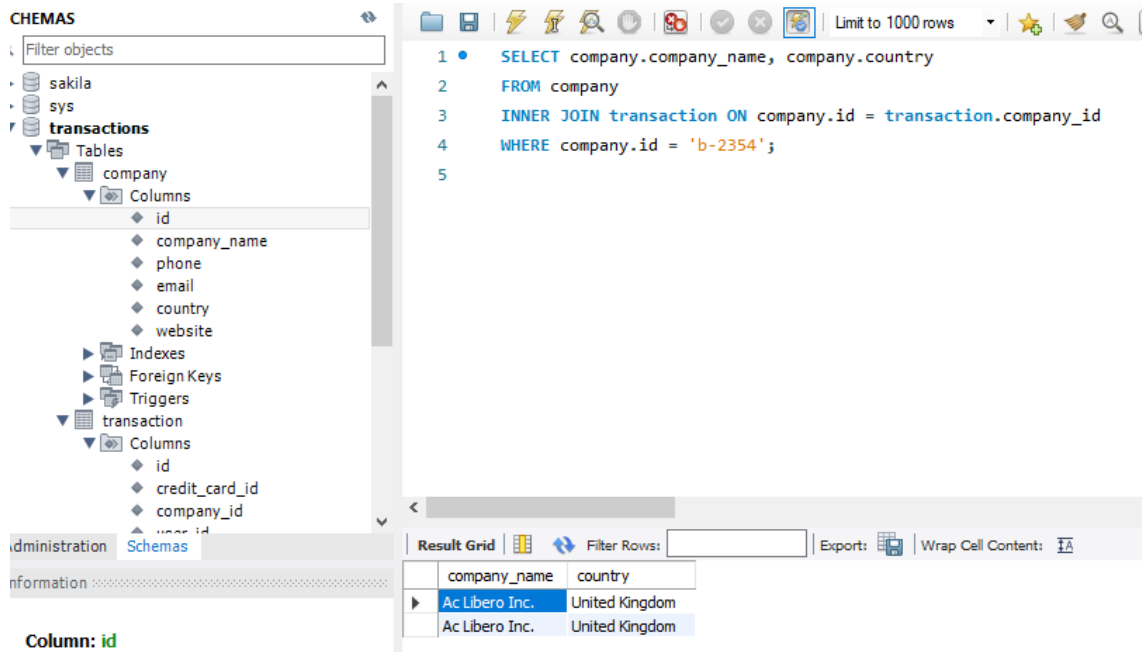
count(Distinct company.country)
15

Para hacerlo necesitamos contar el numero de países que han tenido alguna transacción con nosotros para ello usamos el comando **Select count(Distinct company.country)** lo que hace es seleccionar por contado el numero de países que aparece en la tabla Company mediante el comando FROM luego insertamos el comando **inner join transaction on company.id= transaction.company_id** esto lo que hace es buscar los valores comunes en las dos tablas (Company, transactions) y darnos los valores comunes que en este caso es **Company.id**

El resultado es 15 países

Ejercicio 5

En este ejercicio lo que tenemos que hacer es localizar un ID b-2354 de una empresa dentro de nuestra base de datos y que nos diga el país y el nombre de la empresa que estamos buscando



The screenshot shows a SQL IDE interface. On the left, a tree view displays the database schema, including tables like 'company' and 'transaction'. The 'company' table is selected, showing its columns: id, company_name, phone, email, country, and website. The main query editor on the right contains the following SQL code:

```
1 • SELECT company.company_name, company.country
2 FROM company
3 INNER JOIN transaction ON company.id = transaction.company_id
4 WHERE company.id = 'b-2354';
5
```

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

company_name	country
Ac Libero Inc.	United Kingdom
Ac Libero Inc.	United Kingdom

Lo primero que hacemos es seleccionar la columnas de las que necesitamos información que en este caso es Company.Company_name y Company.country usamos el comando **From Company** para que el programa sepa de que tabla queremos la información también usamos el comando

INNER JOIN transaction ON company.id = transaction.company_id

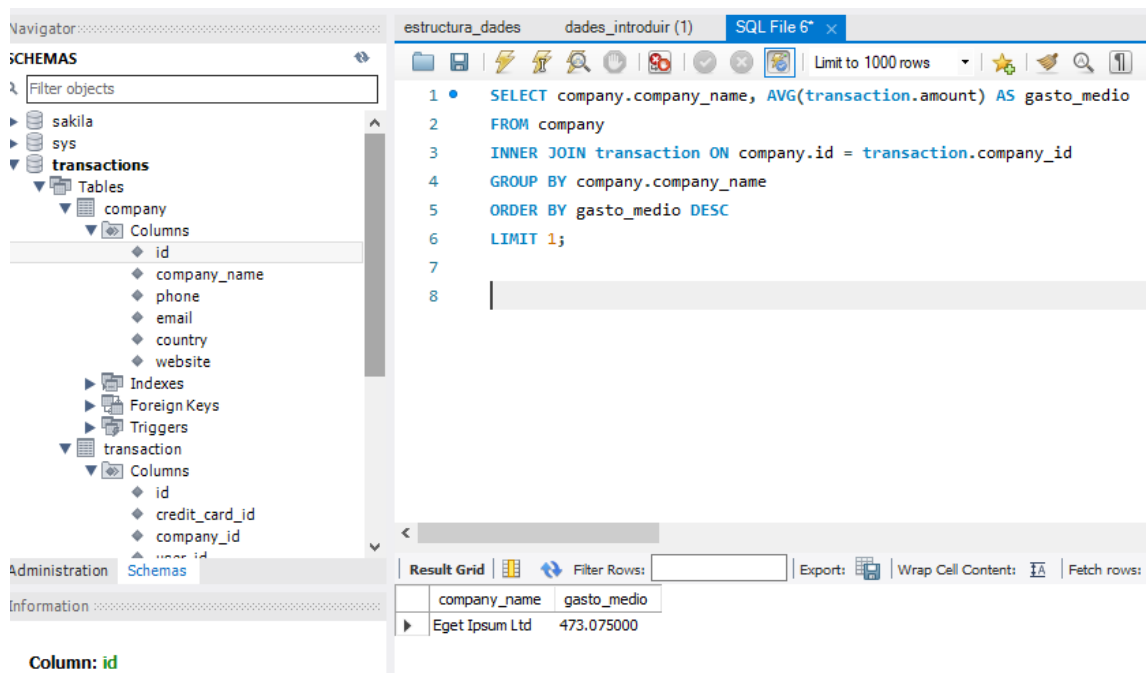
Este commando lo que hace es interconectar las dos tablas para encontrar el ID que estamos buscando solo en los parámetros comunes que en este caso es Company ID

Por ultimo usamos el comando **WHERE Company.id = 'b-2354'** para buscar en especifico este parámetro dentro de la base de datos

El resultado es que el ID b2354 es de la empresa AC Libero Inc y esta en el país United Kingdom

EJERCICIO 6

En este caso lo que nos pide el ejercicio es que saquemos cual es la empresa con más gasto medio



The screenshot shows a database management tool interface. On the left, the 'SCHEMAS' pane displays a tree view of the database structure, including tables 'company' and 'transaction'. The 'company' table has columns: id, company_name, phone, email, country, and website. The 'transaction' table has columns: id, credit_card_id, company_id, and amount. The main pane shows a SQL query in a text editor:

```
1 • SELECT company.company_name, AVG(transaction.amount) AS gasto_medio
2 FROM company
3 INNER JOIN transaction ON company.id = transaction.company_id
4 GROUP BY company.company_name
5 ORDER BY gasto_medio DESC
6 LIMIT 1;
7
8
```

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

company_name	gasto_medio
Eget Ipsum Ltd	473.075000

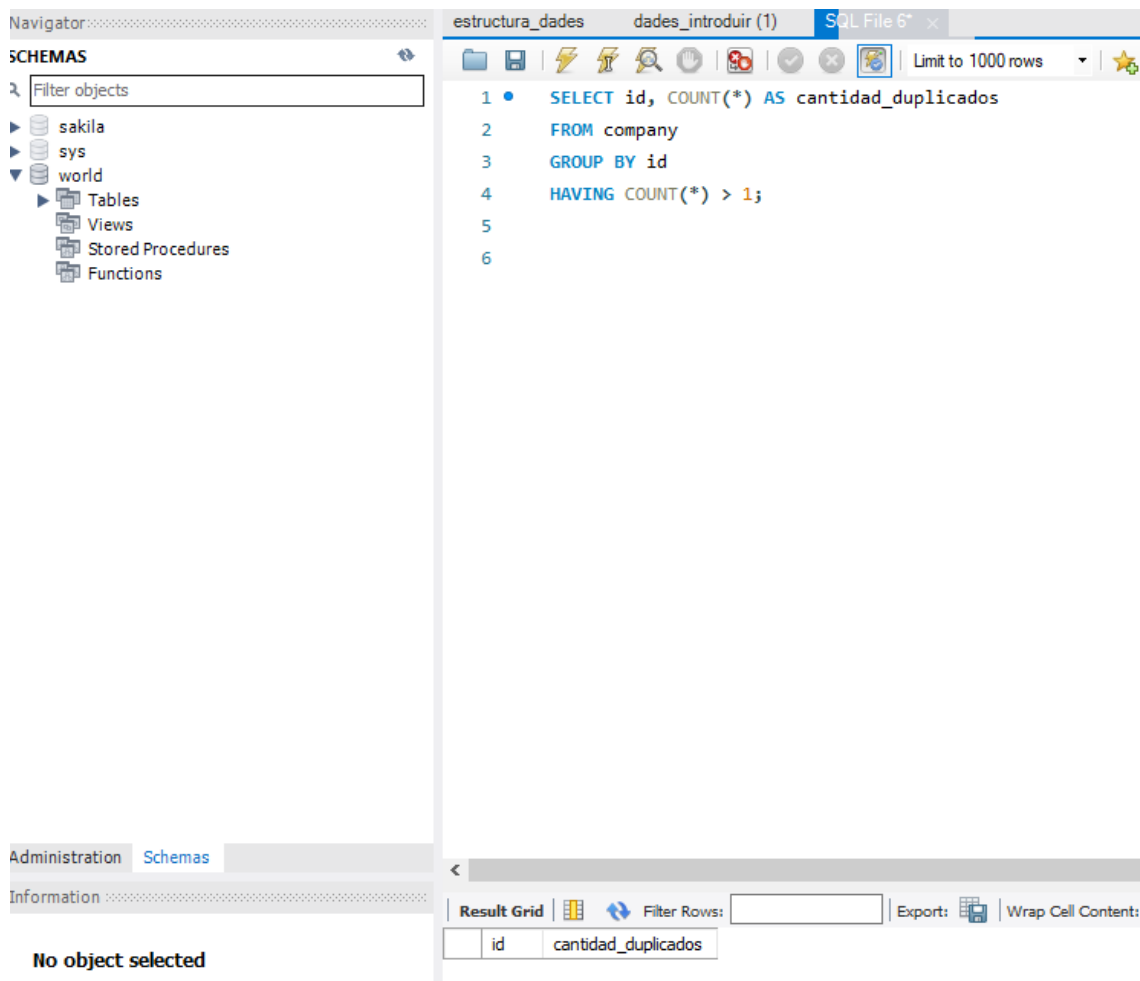
Lo primero que hacemos es hacer que el programa busque nombres de compañías dentro de la tabla **Company** y dentro de la misma la columna **Company_name** luego usamos el comando **AVG** que lo que hace es hacer un cálculo promedio sobre los valores de la tabla trasaction y al resultado le da el Alias de **gasto_medio**, luego mediante la función de **INNER JOIN transaction ON company.id = transaction.company_id**. Este comando lo que haces es buscar en las dos tablas y darnos los valores comunes que haya en las dos tablas que en este caso es **Company_name**, **GROUP BY** lo que hace esta función es agrupar todos los valores de la columna **Company_name** de la tabla **Company**, **ORDER BY** lo que hace es ordenar todos los valores de **gasto_medio** de manera descendente con la función **DESC** luego para que solo nos pueda dar un resultado limitamos los resultados que nos dé a 1

RESULTADO: La compañía que mas gasto promedio tiene es 'Eget Ipsum Ltd' con un gasto promedio de '473.075000'

NIVEL 2

EJERCICIO 1

Este ejercicio lo que nos pide es que comprobemos si se repite o duplica algún nombre de compañía dentro de nuestra base de datos.



The screenshot shows a SQL IDE interface. On the left, the 'SCHEMAS' pane displays a tree view with 'sakila', 'sys', and 'world' schemas. Under 'world', there are sub-items for 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main editor area shows a SQL query:

```
1 • SELECT id, COUNT(*) AS cantidad_duplicados
2 FROM company
3 GROUP BY id
4 HAVING COUNT(*) > 1;
5
6
```

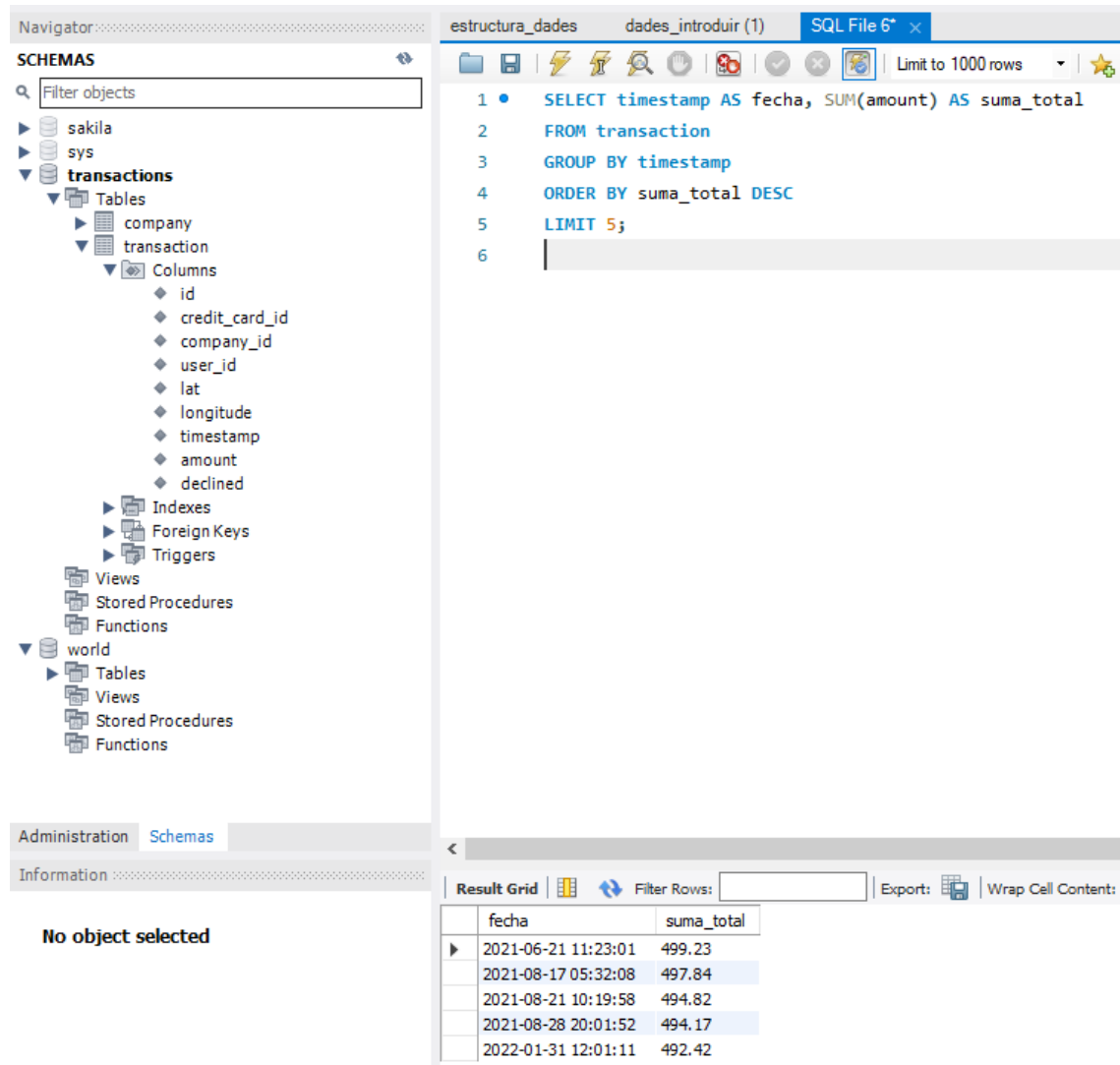
Below the query editor, the 'Result Grid' is visible, showing a table with two columns: 'id' and 'cantidad_duplicados'. The 'Information' pane at the bottom left indicates 'No object selected'.

Para comprobarlo usamos el comando **SELECT id, COUNT(*) AS cantidad_duplicados** lo que hace es buscar los ID y numerar el numero de ID que ahí y ponerle como ALIAS cantidad_duplicados, todo esto dentro de la tabla Company, luego creamos una agrupación de ID y mediante la función **HAVING COUNT(*) > 1**; conseguimos que nos de solo los nombres que salen mas de una vez dentro de nuestra base de datos, la función Having lo que hace es marcar una característica en especifico que en este caso es que las veces que sale el ID sea mayor que 1 que se.

RESULTADO: no hay ningún nombre duplicado

EJERCICIO 2

En este ejercicio se nos pide que sepamos cuales fueron los 5 días con más ventas y el programa debe de mostrar la fecha en la que se hicieron las 5 mayores ventas



The screenshot shows a SQL IDE interface. On the left is a 'SCHEMAS' navigator showing a tree view of databases: sakila, sys, and transactions. The 'transactions' database is expanded, showing tables (company, transaction), columns (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, declined), indexes, foreign keys, and triggers. Below the navigator are tabs for 'Administration' and 'Schemas', and an 'Information' section that says 'No object selected'.

The main editor shows a SQL query in a file named 'SQL File 6*':

```
1 • SELECT timestamp AS fecha, SUM(amount) AS suma_total
2 FROM transaction
3 GROUP BY timestamp
4 ORDER BY suma_total DESC
5 LIMIT 5;
6
```

Below the query editor is a 'Result Grid' showing the results of the query. It has columns 'fecha' and 'suma_total'. The results are:

fecha	suma_total
2021-06-21 11:23:01	499.23
2021-08-17 05:32:08	497.84
2021-08-21 10:19:58	494.82
2021-08-28 20:01:52	494.17
2022-01-31 12:01:11	492.42

Lo que he hecho es usar la función select en la columna timestamp y darle un ALIAS de fecha aparte mediante la función SUM que hace efecto en la columna amount y le da a este un ALIAS de suma_total, estos datos estan en la tabla transaction FROM transaction, luego le pedimos que haga una agrupación de timestamp GROUP BY timestamp y mediante la función ORDER BY suma_total DESC le estamos pidiendo al programa que nos ordene suma_total de manera descendente .

Y como ultima función usamos LIMIT 5 para que solo nos de 5 suma_total.

EJERCICIO 3

Lo que se nos pide en este ejercicio es que sepamos en que días se dieron las peores ventas

The screenshot shows a database management interface with a left sidebar for navigation and a main area for SQL queries and results.

Navigation Panel (Left):

- SCHEMAS
 - Filter objects
 - sakila
 - Tables
 - company
 - transaction
 - Columns
 - id
 - credit_card_id
 - company_id
 - user_id
 - lat
 - longitude
 - timestamp
 - amount
 - declined
 - Indexes
 - Foreign Keys
 - Triggers
 - Views
 - Stored Procedures
 - Functions
 - world
 - Tables
 - Views
 - Stored Procedures
 - Functions

SQL Editor (Top Right):

```
1 • SELECT timestamp AS fecha, SUM(amount) AS suma_total
2 FROM transaction
3 GROUP BY timestamp
4 ORDER BY suma_total ASC
5 LIMIT 5;
6
```

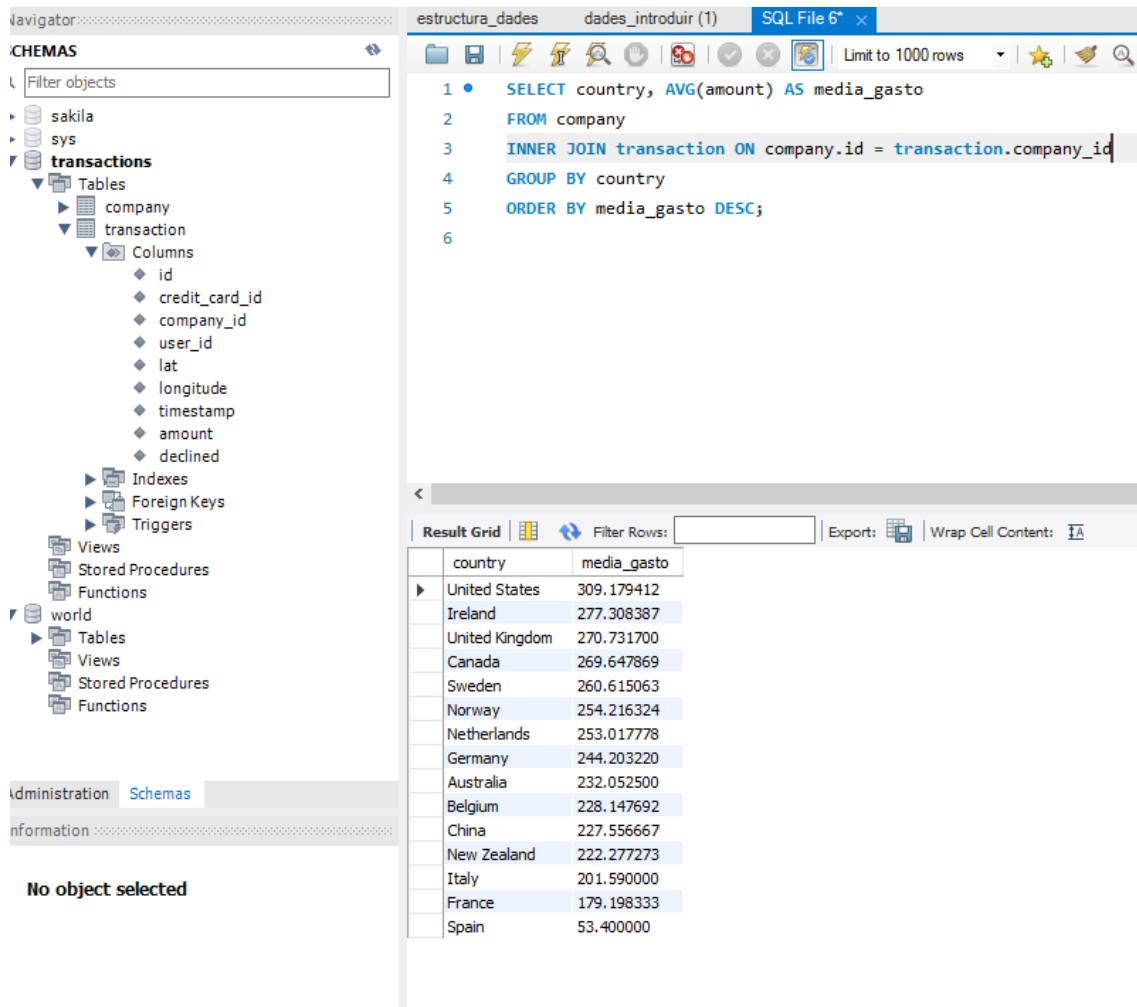
Result Grid (Bottom Right):

fecha	suma_total
2022-01-04 08:21:53	15.05
2021-04-22 12:37:13	15.38
2021-07-25 11:39:28	15.90
2021-12-29 20:38:23	17.97
2021-04-27 08:12:28	18.08

Lo que he hecho es usar la función select en la columna timestamp y darle un ALIAS de fecha aparte mediante la función SUM que hace efecto en la columna amount y le da a este un ALIAS de suma_total, estos datos estan en la tabla transaction FROM transaction, luego le pedimos que haga una agrupación de timestamp GROUP BY timestamp y mediante la función ORDER BY suma_total ASC le estamos pidiendo al programa que nos ordene suma_total de manera ascendente al contrario que el ejercicio anterior.

EJERCICIO 4

Este ejercicio nos pide que demos la media de las ventas ordenados de mayor a menor según país de procedencia



The screenshot shows a SQL IDE interface with a database navigator on the left, a query editor in the center, and a results grid at the bottom.

Database Navigator: The left pane shows a database named 'CHEMAS'. Under the 'transactions' schema, the 'transaction' table is selected. Its columns are: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The 'company' table is also visible under the same schema.

Query Editor: The central pane contains the following SQL query:

```
1 SELECT country, AVG(amount) AS media_gasto
2 FROM company
3 INNER JOIN transaction ON company.id = transaction.company_id
4 GROUP BY country
5 ORDER BY media_gasto DESC;
6
```

Results Grid: The bottom pane displays the results of the query in a table format. The columns are 'country' and 'media_gasto'. The results are ordered from highest to lowest average amount.

country	media_gasto
United States	309.179412
Ireland	277.308387
United Kingdom	270.731700
Canada	269.647869
Sweden	260.615063
Norway	254.216324
Netherlands	253.017778
Germany	244.203220
Australia	232.052500
Belgium	228.147692
China	227.556667
New Zealand	222.277273
Italy	201.590000
France	179.198333
Spain	53.400000

Mediante este commando **SELECT country, AVG(amount) AS media_gasto** lo que hacemos es seleccionar la columna country y pedirle al programa que nos de la media de gasto de la tabla Company y a esta le da el ALIAS de media_gasto, mediante la función **INNER JOIN transaction ON Company.id = transaction.company_id** este código lo que hace es unir las dos tablas por un valor común que en este caso es la columna ID de la tabla Company, en transaction es Company_id. Creamos GROUP BY country y también añadimos la función ORDER BY media_gasto DESC lo que hace es ordenar de mayor a menor las media de gasto que luego al ejecutarse clasifica las medias según nacionalidad

RESULTADO: En la imagen se ve cuáles son las ventas medias de cada país.