

He creado una interfaz gráfica utilizando la librería Tkinter con el propósito de interactuar de manera efectiva con una base de datos. Esta aplicación tiene como objetivo principal facilitar la ejecución de consultas SQL, permitiendo a los usuarios realizar operaciones como SELECT, INSERT, DELETE y UPDATE de manera intuitiva y eficiente.

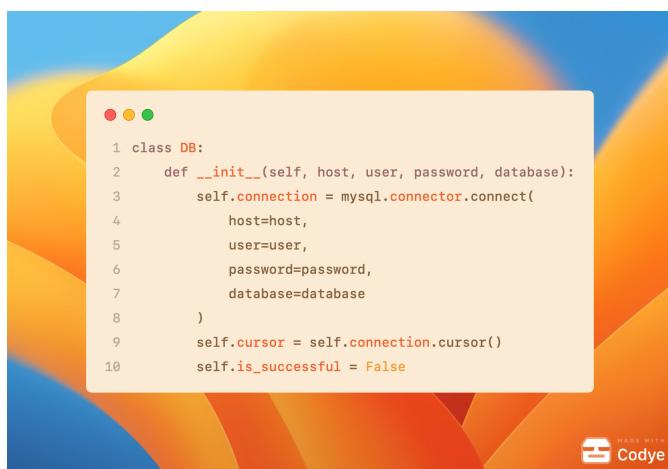
El uso de Tkinter como herramienta para la creación de interfaces gráficas proporciona una experiencia amigable para el usuario, mientras que la integración de consultas SQL brinda flexibilidad y potencial para la gestión de datos de manera efectiva. A lo largo de este informe, explicaré los componentes clave de la aplicación, desde la conexión con la base de datos hasta la implementación de consultas y la presentación de resultados.

Esta aplicación no solo busca cumplir con los requisitos básicos de interacción con bases de datos, sino también proporcionar una experiencia fluida y efectiva, minimizando la complejidad de las consultas y ofreciendo retroalimentación clara al usuario.

## Conexión con la Base de Datos

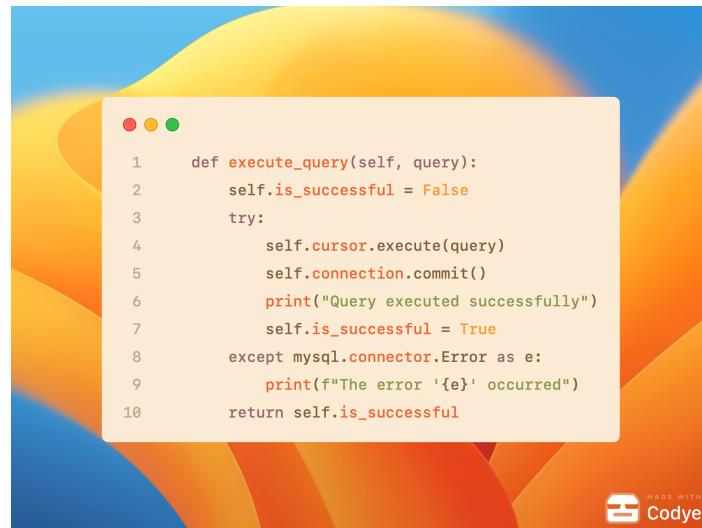
En este proyecto, he optado por utilizar la librería mysql-connector para facilitar la conexión y la ejecución de consultas SQL de manera eficiente.

Para gestionar la conexión con la base de datos, he implementado una clase denominada "DB". En el constructor de esta clase, se inicializa la conexión utilizando los parámetros necesarios, como la dirección del servidor, el nombre de la base de datos, el nombre de usuario y la contraseña. Este enfoque permite centralizar la configuración de conexión y facilita su mantenimiento.



La clase "DB" consta de dos métodos principales:

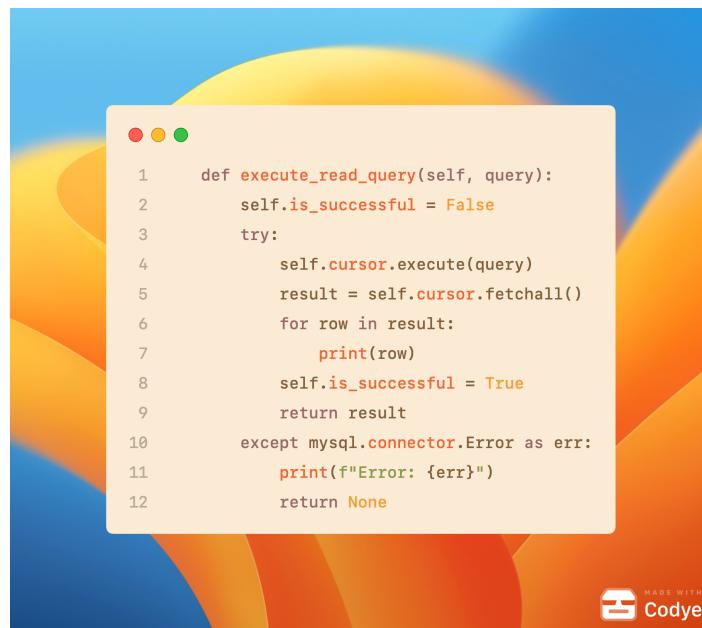
**execute\_query:** Este método se encarga de ejecutar comandos SQL que no devuelven resultados, como INSERT, DELETE y UPDATE. La función retorna un valor booleano que indica si la consulta se ejecutó con éxito o no.



```
1 def execute_query(self, query):
2     self.is_successful = False
3     try:
4         self.cursor.execute(query)
5         self.connection.commit()
6         print("Query executed successfully")
7         self.is_successful = True
8     except mysql.connector.Error as e:
9         print(f"The error '{e}' occurred")
10    return self.is_successful
```

Made with  Codye

**execute\_read\_query:** Este método está diseñado para ejecutar comandos SELECT que devuelven resultados. Después de ejecutar la consulta, los resultados se recuperan y pueden ser procesados según sea necesario, en este caso se los muestra en una tabla usando Tkinter.

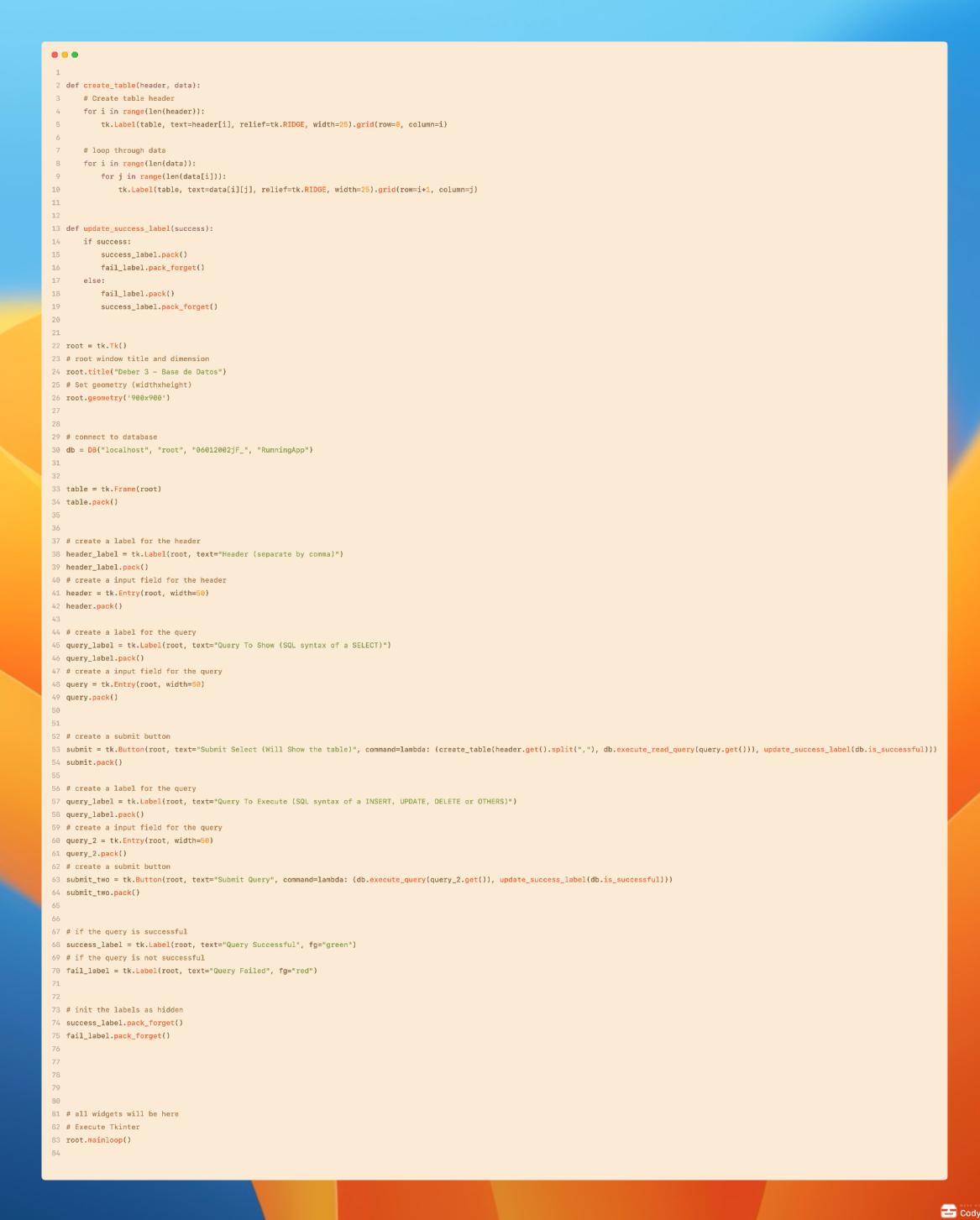


```
1 def execute_read_query(self, query):
2     self.is_successful = False
3     try:
4         self.cursor.execute(query)
5         result = self.cursor.fetchall()
6         for row in result:
7             print(row)
8         self.is_successful = True
9     return result
10 except mysql.connector.Error as err:
11     print(f"Error: {err}")
12     return None
```

Made with  Codye

## Interfaz Gráfica

Debido a que este deber no se centra en la interfaz gráfica, aquí se muestra el código con sus respectivos comentarios sobre cómo se crea la interfaz sin más detalle.



```
1
2 def create_table(header, data):
3     # Create table header
4     for i in range(len(header)):
5         tk.Label(table, text=header[i], relief=tk.RIDGE, width=25).grid(row=0, column=i)
6
7     # loop through data
8     for i in range(len(data)):
9         for j in range(len(data[i])):
10            tk.Label(table, text=data[i][j], relief=tk.RIDGE, width=25).grid(row=i+1, column=j)
11
12
13 def update_success_label(success):
14     if success:
15         success_label.pack()
16         fail_label.pack_forget()
17     else:
18         fail_label.pack()
19         success_label.pack_forget()
20
21
22 root = tk.Tk()
23 # root window title and dimension
24 root.title("Deber 3 - Base de Datos")
25 # Set geometry (widthxheight)
26 root.geometry('980x980')
27
28
29 # connect to database
30 db = DB("localhost", "root", "06012002jF_ ", "RunningApp")
31
32
33 table = tk.Frame(root)
34 table.pack()
35
36
37 # create a label for the header
38 header_label = tk.Label(root, text="Header (separate by comma)")
39 header_label.pack()
40 # create a input field for the header
41 header = tk.Entry(root, width=50)
42 header.pack()
43
44 # create a label for the query
45 query_label = tk.Label(root, text="Query To Show (SQL syntax of a SELECT)")
46 query_label.pack()
47 # create a input field for the query
48 query = tk.Entry(root, width=50)
49 query.pack()
50
51
52 # create a submit button
53 submit = tk.Button(root, text="Submit Select (Will Show the table)", command=lambda: (create_table(header.get().split(","), db.execute_read_query(query.get())), update_success_label(db.is_successful)))
54 submit.pack()
55
56
57 # create a label for the query
58 query_label = tk.Label(root, text="Query To Execute (SQL syntax of a INSERT, UPDATE, DELETE or OTHERS)")
59 query_label.pack()
60 # create a input field for the query
61 query_2 = tk.Entry(root, width=50)
62 query_2.pack()
63 # create a submit button
64 submit_two = tk.Button(root, text="Submit Query", command=lambda: (db.execute_query(query_2.get()), update_success_label(db.is_successful)))
65 submit_two.pack()
66
67
68 # if the query is successful
69 success_label = tk.Label(root, text="Query Successful", fg="green")
70 # if the query is not successful
71 fail_label = tk.Label(root, text="Query Failed", fg="red")
72
73 # init the labels as hidden
74 success_label.pack_forget()
75 fail_label.pack_forget()
76
77
78
79
80
81 # all widgets will be here
82 # Execute Tkinter
83 root.mainloop()
```

## Consultas SQL

Aquí ejemplos de consultas Select que se pueden hacer con el app.

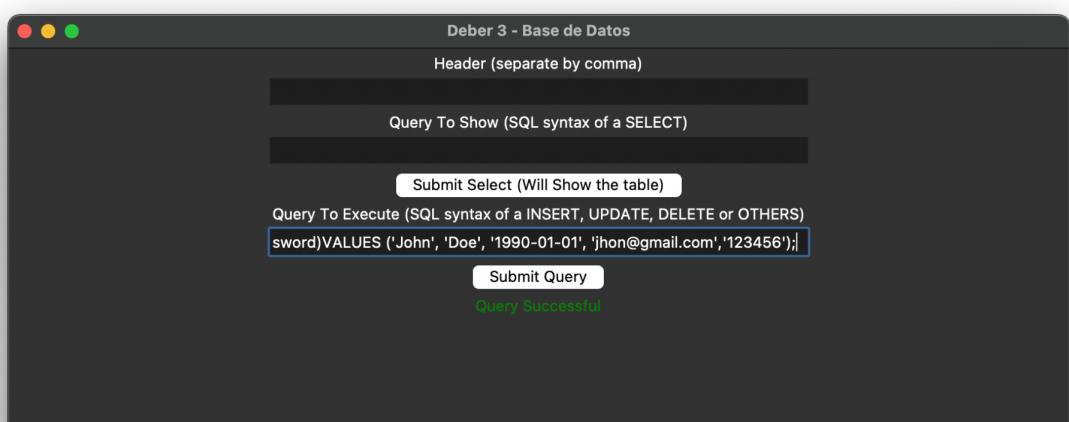
Deber 3 - Base de Datos			
Userid	Name	LastName	Email
1	Juan	Cisneros	juanfrancism2011@icloud.com
2	Elena	Ramirez	elena.ramirez@gmail.com
3	Carlos	Perez	carlos.perez@hotmail.com
4	Sofia	Garcia	sofia.garcia@yahoo.com
5	Luisa	Fernandez	luisa.fernandez@gmail.com
6	Pedro	Lopez	pedro.lopez@gmail.com
7	Miguel	Torres	miguel.torres@yahoo.com
8	Isabel	Sanchez	isabel.sanchez@hotmail.com
9	Javier	Morales	javier.morales@gmail.com
10	Carmen	Diaz	carmen.diaz@yahoo.com
11	Raul	Hernandez	raul.hernandez@gmail.com
12	Laura	Castro	laura.castro@hotmail.com
13	Diego	Gomez	diego.gomez@yahoo.com
14	Natalia	Vargas	natalia.vargas@gmail.com
15	Manuel	Martinez	manuel.martinez@yahoo.com
16	Paula	Santos	paula.santos@hotmail.com
17	Alejandro	Ortega	alejandro.ortega@gmail.com
18	Rosa	Reyes	rosa.reyes@yahoo.com
19	Hector	Cabrera	hector.cabrera@hotmail.com
20	Adriana	Fuentes	adriana.fuentes@gmail.com
21	Julio	Silva	julio.silva@yahoo.com
22	John	Doe	jhon@gmail.com
24	John	Doe	jhon@gmail.com
25	John	Doe	jhon@gmail.com
26	John	Doe	jhon@gmail.com
27	John	Doe	jhon@gmail.com
28	John	Doe	jhon@gmail.com

Header (separate by comma)  
  
 Query To Show (SQL syntax of a SELECT)  
  
  
 Query To Execute (SQL syntax of a INSERT, UPDATE, DELETE or OTHERS)  
  
Query Successful!

Deber 3 - Base de Datos			
AchievementID	Name	Description	
1	Maratón en un Lugar Exótico	un maratón en un lugar exótico o en el extranjero.	
2	Medalla de Oro en Carrera de 5K	medalla de oro en una carrera de 5 kilómetros.	
3	Maratón Completo	completar un maratón de 42.195 kilómetros.	
4	Correr 1000 Kilómetros en un Año	la marca de correr 1000 kilómetros en un año.	
5	Medalla de Oro en Carrera de 10K	medalla de oro en una carrera de 10 kilómetros.	
6	Logro de Velocidad	correr 1 kilómetro en menos de 4 minutos.	
7	Medalla de Finisher en Ultra Maratón	correr una ultra maratón de 50 kilómetros.	
8	Consecución de Maratón en Ciudad	correr un maratón en una ciudad reconocida.	
9	Récord Personal en Media Maratón	un nuevo récord personal en una media maratón.	
10	Desafío de Elevación	correr una ruta con elevación significativa.	
11	Medalla de Oro en Carrera de 21K	una medalla de oro en una carrera de 21 kilómetros.	
12	Cross-Country Runner	correr en una carrera de cross-country y conseguir un ranking.	
13	correr 10 Kilómetros en Menos de 40 Minutos	correr 10 kilómetros en menos de 40 minutos.	
14	Logro de Resistencia	Correr 20 kilómetros sin detenerse.	
15	Consecución de Ruta Escénica	una ruta con vistas panorámicas impresionantes.	
16	Medalla de Oro en Carrera de 42K	ganar una medalla de oro en un maratón de 42.195 km.	
17	Carrera Nocturna Completada	correr una carrera nocturna de 10 kilómetros.	
18	Consecución de Trail Running	correr en una carrera de trail running y conseguir un ranking.	
19	Medalla de Finisher en Carrera de Montaña	ganar una medalla de oro en una carrera de montaña.	
20	correr una Media Maratón en Menos de 1 hora	correr una media maratón en menos de 1 hora.	
21	Entrenamiento Diario por 30 Días	realizar un entrenamiento diario durante 30 días.	

Header (separate by comma)  
  
 Query To Show (SQL syntax of a SELECT)  
  
  
 Query To Execute (SQL syntax of a INSERT, UPDATE, DELETE or OTHERS)  
  
Query Successful!

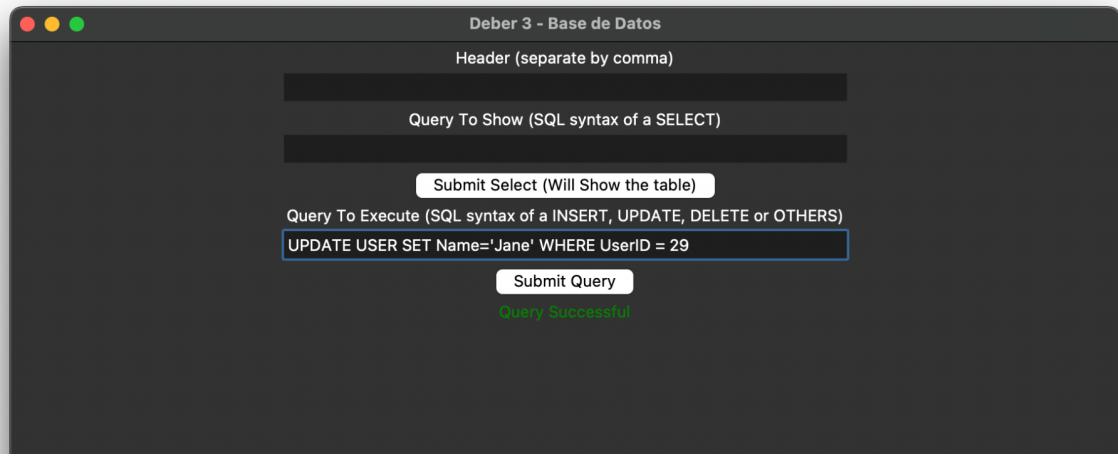
## Ejemplos de querys más complejas.



The screenshot shows the MySQL Workbench interface. The left sidebar is the "Database Explorer" with a tree view of the database schema. The central area is a "console\_1" tab showing the results of a query against the "USER" table. The results are as follows:

	UserID	Name	LastName	BirthDate	Email	Password
18	18	Rosa	Reyes	1983-04-18	rosa.reyes@yahoo.com	18041983_RR
19	19	Hector	Cabrera	2003-12-11	hector.cabrera@hotmail.com	11122003_HC
20	20	Adriana	Fuentes	1992-08-07	adriana.fuentes@gmail.com	07081992_AF
21	21	Julio	Silva	1986-01-24	julio.silva@yahoo.com	24011986_JS
22	29	John	Doe	1990-01-01	jhon@gmail.com	123456

The row with UserID 22 is highlighted with a red border. The right sidebar shows the "Files" and "Structure" panes, which detail the schema of the "User" table.



Database Explorer

localhost

- DATE
- INSTRUMENTS
- RUN\_PLAN
- USER**
- USER\_ACHIEVEMENTS
- WEATHER
- WORKOUT
- WORKOUT\_INSTRUMENTS

console\_1

USER

ACHIEVEMENTS

	User ID	Name	Last Name	Birth Date	Email	Password
18	18	Rosa	Reyes	1983-04-18	rosa.reyes@yahoo.com	18041983_RR
19	19	Hector	Cabrera	2003-12-11	hector.cabrera@hotmail.com	11122003_HC
20	20	Adriana	Fuentes	1992-08-07	adriana.fuentes@gmail.com	07081992_AF
21	21	Julio	Silva	1986-01-24	julio.silva@yahoo.com	24011986_JS
22	29	Jane	Doe	1990-01-01	jhon@gmail.com	123456

Files

EJArtistas.sql

Structure

UserID int  
Name varchar(50)  
LastName varchar(50)  
BirthDate date  
Email varchar(50)