HITO2 1T SGE

ADRIÁN HERMOSILLA

19/11/2024

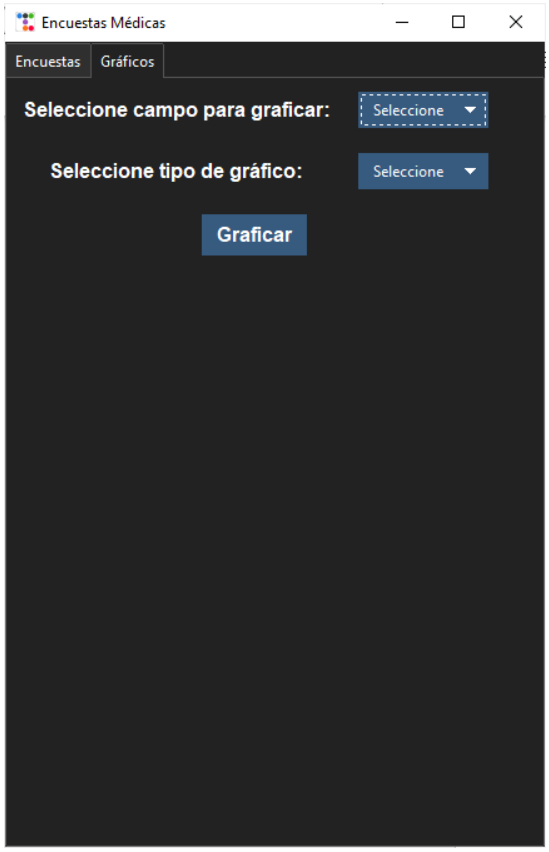
Contenido

INTRODUCCIÓN

Este proyecto se enfoca en la recolección, almacenamiento y análisis de datos relacionados con los hábitos de consumo de alcohol y su impacto en la salud. A través de una interfaz gráfica construida en Python con Tkinter, el sistema permite a los usuarios registrar, visualizar y analizar los resultados de encuestas sobre consumo de bebidas alcohólicas y sus efectos en la salud, proporcionando datos útiles para estudios médicos.

DESCRIPCIÓN DE LA INTERFAZ Y CONEXIÓN

La interfaz gráfica está diseñada con Tkinter y permite a los usuarios registrar información mediante un formulario de encuesta, visualizando los datos almacenados y generando gráficos para análisis. Dispone de dos pestañas para mayor comodidad del usuario.



La conexión con la base de datos MySQL es establecida al inicio del programa, utilizando las credenciales de acceso para interactuar con la base de datos ENCUESTAS.

```python
# Conexión a la base de datos
self.conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="curso",
    database="ENCUESTAS"
)
self.cursor = self.conn.cursor()
```

OPERACIONES CRUD

Aquí debes detallar cómo se implementaron las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) y cómo funcionan dentro de tu aplicación. Incluye capturas de pantalla de los formularios de entrada y ejemplos de interacción.

Crear: Los usuarios pueden agregar una nueva encuesta completando un formulario con los datos necesarios, y posteriormente utilizando el botón de agregar. Al agregar una nueva encuesta, esta aparecerá abajo del todo.

```python
def agregar_dato(self):
    data = {
        "edad": self.edad_entry.get(),
        "sexo": self.sexo_entry.get(),
        "bebidas_semana": self.bebidas_semana_entry.get(),
        "cervezas_semana": self.cervezas_semana_entry.get(),
        "bebidas_fin_semana": self.bebidas_fin_semana_entry.get(),
        "bebidas_destiladas_semana": self.bebidas_destiladas_semana_entry.get(),
        "vinos_semana": self.vinos_semana_entry.get(),
        "perdidas_control": self.perdidas_control_entry.get(),
        "diversion_dependencia_alcohol": self.diversion_dependencia_alcohol_entry.get(),
        "problemas_digestivos": self.problemas_digestivos_entry.get(),
        "tension_alta": self.tension_alta_entry.get(),
        "dolor_cabeza": self.dolor_cabeza_entry.get()
    }

    if all(data.values()):
        self.cursor.execute(
            operation: "INSERT INTO ENCUESTA (edad, Sexo, BebidasSemana, CervezasSemana, BebidasFinSemana, BebidasDest
            data
        )
        self.conn.commit()
        Messagebox.show_info( message: "Éxito",  title: "Dato agregado correctamente", parent=self.root)
    else:
        Messagebox.show_warning( message: "Advertencia",  title: "Todos los campos son obligatorios", parent=self.root)
```

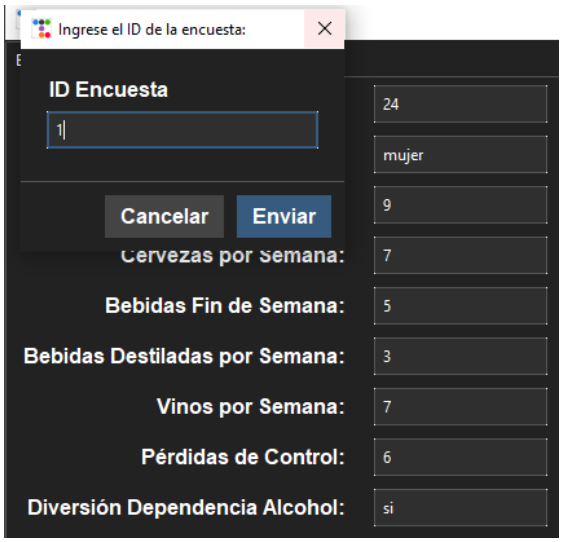
Leer: Los datos se pueden visualizar en una tabla dinámica donde se pueden ordenar por diferentes campos.

idEncuesta	edad	Sexo	BebidasSemana	CervezasSemana	BebidasFinSema	BebidasDestilad	VinosSemana	PerdidasControl	DiversionDepen	ProblemasDiges	TensionAlta	DolorCabeza
1	23	mujer	9	7	5	3	7	6	si	no	si	no
2	55	Mujer	6	4	5	1	2	3	Sí	Sí	No	Alguna vez
3	19	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
4	20	Hombre	0	0	0	0	0	0	No	No	No	Nunca
5	21	Hombre	20	15	10	5	0	12	Sí	No	No	Alguna vez
6	20	Hombre	1	12	45	12	23	45	No	No	No	Alguna vez
7	19	Hombre	2	0	2	2	0	3	No	No	No	Nunca
8	19	Hombre	3	5	5	5	0	1	Sí	No	No lo se	Muy a menudo
9	22	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
10	19	Hombre	2	10	5	1	5	2	No	No	No	Alguna vez
11	19	Hombre	83	0	75	27	56	33	Sí	Sí	Sí	Muy a menudo
12	21	Hombre	0	0	0	0	0	0	No	No	No	Alguna vez
13	19	Hombre	5	5	5	5	0	1	Sí	No	No lo se	Alguna vez
14	24	Hombre	25	18	10	7	0	365	No	No	No	Alguna vez
15	38	Mujer	41	5	25	10	10	365	Sí	Sí	Sí	Muy a menudo
16	53	Mujer	0	0	0	0	0	0	No	No	Sí	Alguna vez
17	19	Hombre	4	0	1	1	1	1	No	No	No	Nunca
18	20	Hombre	3	7	7	2	0	0	Sí	Sí	No lo se	Nunca
19	20	Hombre	3	2	5	3	0	1	No	No	No	Alguna vez
20	21	Mujer	2	0	2	0	2	0	No	No	No	A menudo
21	19	Hombre	20	20	10	7	0	3	No	No	No lo se	Alguna vez
22	20	Hombre	1	0	0	1	1	0	No	No	No	Nunca
23	19	Hombre	0	0	0	0	0	0	No	No	Sí	Alguna vez
24	58	Hombre	12	6	7	4	3	2	No	Sí	Sí	A menudo
25	47	Hombre	0	0	3	3	0	2	No	No	No	Nunca
26	19	Hombre	1	0	5	1	0	5	Sí	No	No	Nunca
27	37	Mujer	10	5	6	3	3	3	Sí	Sí	Sí	Muy a menudo
28	50	Hombre	50	35	65	26	1	100	Sí	Sí	Sí	Muy a menudo
29	19	Hombre	0	0	0	0	0	0	No	Sí	No	Alguna vez
30	19	Hombre	23	0	10	11	12	69	Sí	No	No	A menudo
31	19	Hombre	0	0	0	0	0	3	No	No	No lo se	Nunca
32	60	Mujer	3	2	3	0	1	0	No	Sí	Sí	A menudo
33	25	Hombre	7	2	7	5	0	0	No	No	No	A menudo
34	45	Mujer	10	3	5	3	0	1	Sí	Sí	No lo se	Muy a menudo
35	53	Mujer	3	3	3	0	1	0	No	Sí	No	Muy a menudo
36	55	Hombre	11	5	4	0	2	0	No	No	No	Nunca
37	18	Hombre	0	0	0	0	0	0	No	No	No	Nunca

```python
def mostrar_datos(self):
    if not hasattr(self, "tree"):
        self.data_frame = ttk.Frame(self.root)
        self.data_frame.pack(side="right", fill="both", expand=True, padx=10, pady=10)  # Use pac

        self.tree = ttk.Treeview(self.data_frame, show="headings")
        self.tree.pack(side="left", fill="both", expand=True)  # Use pack with side="left"

        self.scrollbar = ttk.Scrollbar(self.data_frame, orient="vertical", command=self.tree.yvie
        self.scrollbar.pack(side="right", fill="y")  # Use pack with side="right"
        self.tree.configure(yscroll=self.scrollbar.set)

    self.cursor.execute("SELECT * FROM ENCUESTA")
    rows = self.cursor.fetchall()
    columns = [desc[0] for desc in self.cursor.description]

    if not self.tree["columns"]:
        self.tree["columns"] = columns
        for col in columns:
            self.tree.heading(col, text=col, command=lambda c=col: self.ordenar_datos(c, "ASC"))
            self.tree.column(col, width=100)

    for item in self.tree.get_children():
        self.tree.delete(item)

    for row in rows:
        self.tree.insert( parent: "",  index: "end", values=row)
```

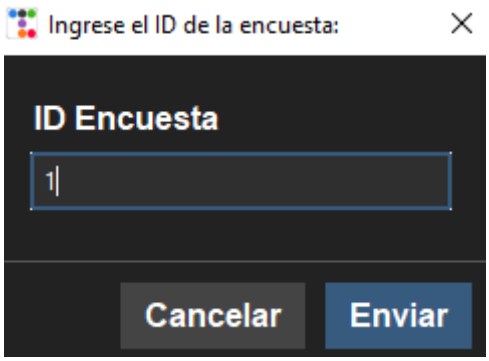
Actualizar: Los usuarios pueden modificar los datos existentes ingresando el ID de la encuesta y editando la información requerida.

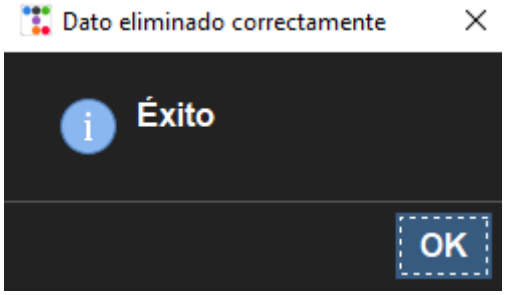
```python
def actualizar_dato(self):
    id_encuesta = self.get_id_encuesta()
    if not id_encuesta:
        return

    data = {
        "edad": self.edad_entry.get(),
        "sexo": self.sexo_entry.get(),
        "bebidas_semana": self.bebidas_semana_entry.get(),
        "cervezas_semana": self.cervezas_semana_entry.get(),
        "bebidas_fin_semana": self.bebidas_fin_semana_entry.get(),
        "bebidas_destiladas_semana": self.bebidas_destiladas_semana_entry.get(),
        "vinos_semana": self.vinos_semana_entry.get(),
        "perdidas_control": self.perdidas_control_entry.get(),
        "diversion_dependencia_alcohol": self.diversion_dependencia_alcohol_entry.get(),
        "problemas_digestivos": self.problemas_digestivos_entry.get(),
        "tension_alta": self.tension_alta_entry.get(),
        "dolor_cabeza": self.dolor_cabeza_entry.get()
    }

    if all(data.values()):
        self.cursor.execute(
            operation: "UPDATE ENCUESTA SET edad=%(edad)s, Sexo=%(sexo)s, BebidasSemana=%(bebidas_semana)s, CervezasS
            params: {**data, "id_encuesta": id_encuesta}
        )
        self.conn.commit()
        Messagebox.show_info( message: "Éxito",  title: "Dato actualizado correctamente", parent=self.root)
    else:
        Messagebox.show_warning( message: "Advertencia",  title: "Todos los campos son obligatorios", parent=self.root)
```



Eliminar: Es posible eliminar encuestas específicas usando el ID correspondiente.

```
def eliminar_dato(self):
    id_encuesta = self.get_id_encuesta()
    if not id_encuesta:
        return

    self.cursor.execute( operation: "DELETE FROM ENCUESTA WHERE idEncuesta=%s",  params: (id_encuesta,))
    self.conn.commit()
    Messagebox.show_info( message: "Éxito",  title: "Dato eliminado correctamente", parent=self.root)
```

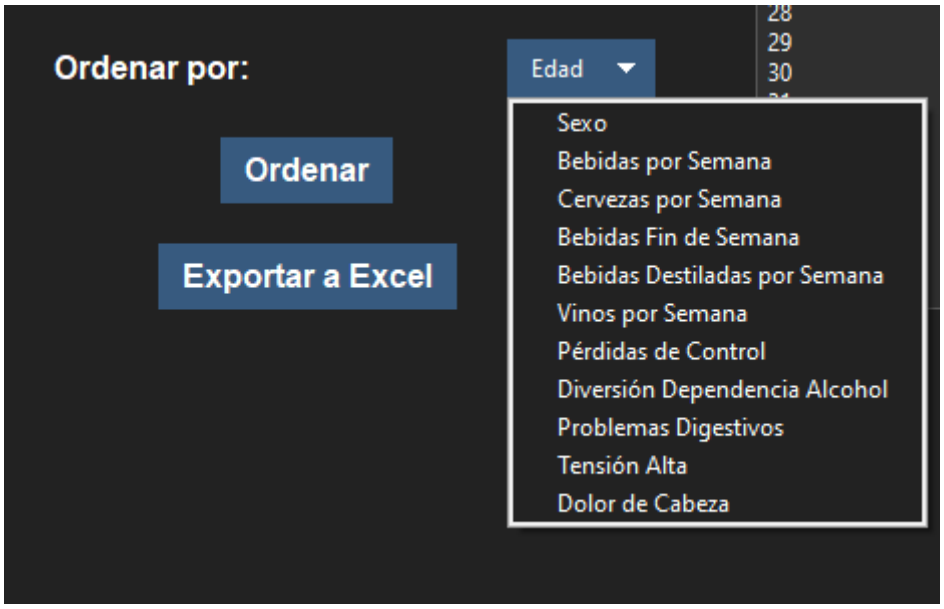




CONSULTAS Y ORDENACIÓN DE DATOS

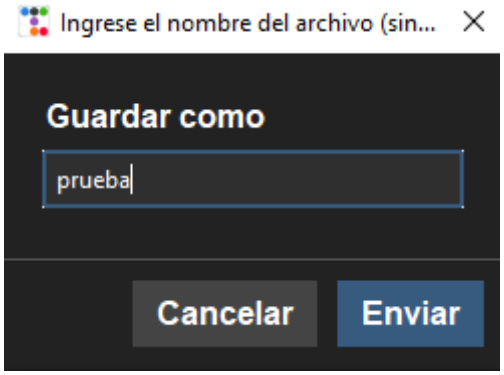
En el proyecto, se implementan varios filtros para organizar los datos. Por ejemplo, los usuarios pueden ordenar las encuestas por edad o frecuencia de consumo de alcohol. Estas consultas se ejecutan directamente en la base de datos y los resultados se presentan en una tabla organizada.

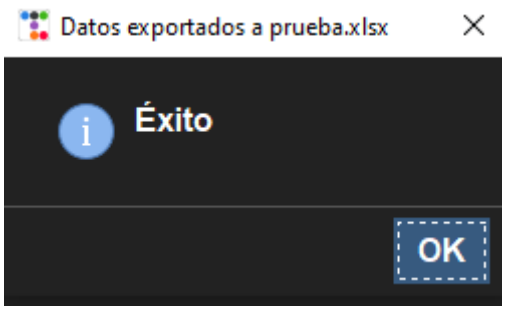
```python
def ordenar_datos(self):
    # Obtener el campo seleccionado
    sort_field = self.sort_by.get()
    if sort_field == "Seleccione campo":
        Messagebox.show_warning( message: "Advertencia",  title: "Seleccione un campo para ordenar o filtrar", parent=self.
        return

    # Opciones de filtro específicas
    if sort_field == "Alta frecuencia de consumo de alcohol":
        query = "SELECT * FROM ENCUESTA WHERE BebidasSemana > 10"
    elif sort_field == "Perdidas de control > 3":
        query = "SELECT * FROM ENCUESTA WHERE PerdidasControl > 3"
    elif sort_field == "Dolor de cabeza y Tensión alta":
        query = "SELECT * FROM ENCUESTA WHERE DolorCabeza = 1 AND TensionAlta = 1"
    else:
        # Mapeo del campo para la base de datos
        field_map = {
            "Edad": "edad",
            "Sexo": "Sexo",
            "Bebidas por Semana": "BebidasSemana",
            "Cervezas por Semana": "CervezasSemana",
            "Bebidas Fin de Semana": "BebidasFinSemana",
            "Bebidas Destiladas por Semana": "BebidasDestiladasSemana",
            "Vinos por Semana": "VinosSemana",
            "Pérdidas de Control": "PerdidasControl",
            "Diversión Dependencia Alcohol": "DiversionDependenciaAlcohol",
            "Problemas Digestivos": "ProblemasDigestivos",
            "Tensión Alta": "TensionAlta",
            "Dolor de Cabeza": "DolorCabeza"
        }
        sort_field_db = field_map[sort_field]
        query = f"SELECT * FROM ENCUESTA ORDER BY {sort_field_db}"

    # Ejecutar la consulta y mostrar resultados
    self.cursor.execute(query)
    rows = self.cursor.fetchall()

    if not rows:
        Messagebox.show_info( message: "Información",  title: "No se encontraron datos para mostrar", parent=self.root)
        return
```

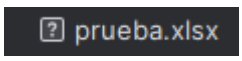


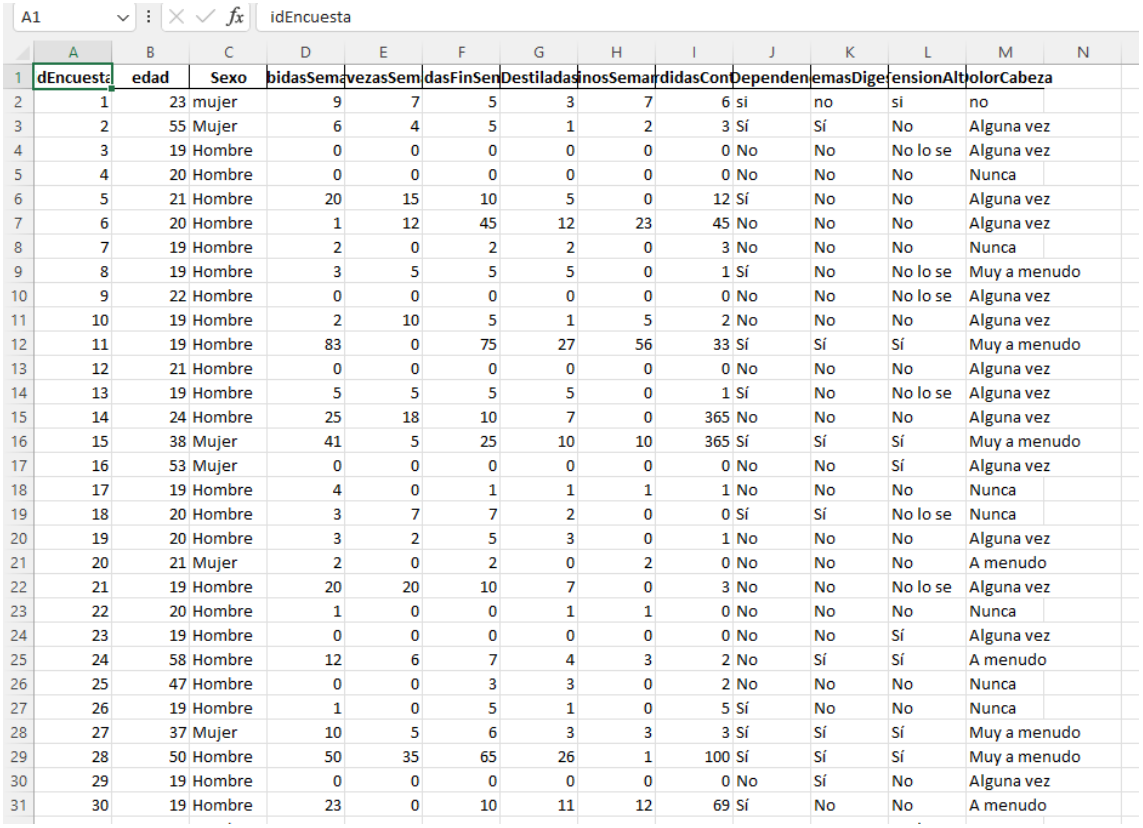
EXPORTACIÓN DE RESULTADOS A EXCEL

La funcionalidad de exportación permite a los usuarios guardar los resultados de la encuesta en un archivo .xlsx, lo que facilita el análisis posterior. Esta opción es accesible a través de un botón en la interfaz, y el usuario puede ingresar el nombre del archivo antes de exportar los datos.









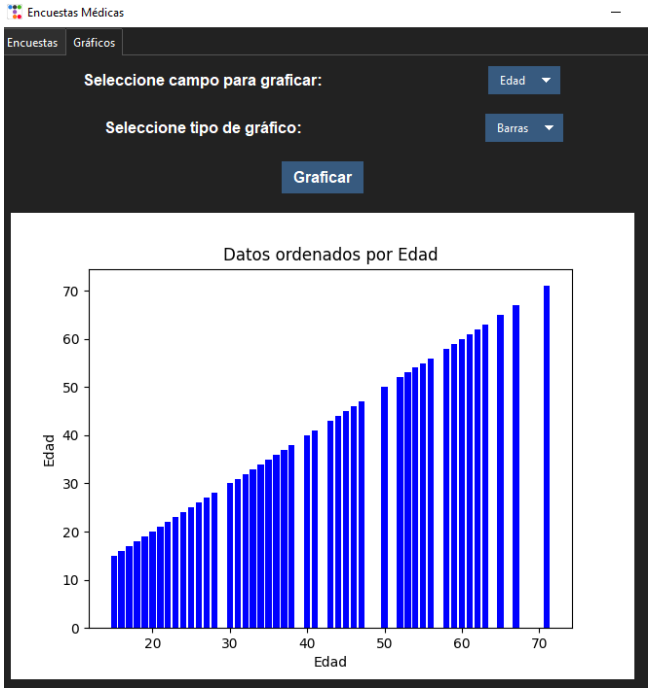
	idEncuesta	edad	Sexo	bidasSem	vezasSem	dasFinSen	Destiladas	inosSemar	didasCon	Dependen	emasDige	ensionAlt	olorCabeza
2	1	23	mujer	9	7	5	3	7	6	si	no	si	no
3	2	55	Mujer	6	4	5	1	2	3	Sí	Sí	No	Alguna vez
4	3	19	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
5	4	20	Hombre	0	0	0	0	0	0	No	No	No	Nunca
6	5	21	Hombre	20	15	10	5	0	12	Sí	No	No	Alguna vez
7	6	20	Hombre	1	12	45	12	23	45	No	No	No	Alguna vez
8	7	19	Hombre	2	0	2	2	0	3	No	No	No	Nunca
9	8	19	Hombre	3	5	5	5	0	1	Sí	No	No lo se	Muy a menudo
10	9	22	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
11	10	19	Hombre	2	10	5	1	5	2	No	No	No	Alguna vez
12	11	19	Hombre	83	0	75	27	56	33	Sí	Sí	Sí	Muy a menudo
13	12	21	Hombre	0	0	0	0	0	0	No	No	No	Alguna vez
14	13	19	Hombre	5	5	5	5	0	1	Sí	No	No lo se	Alguna vez
15	14	24	Hombre	25	18	10	7	0	365	No	No	No	Alguna vez
16	15	38	Mujer	41	5	25	10	10	365	Sí	Sí	Sí	Muy a menudo
17	16	53	Mujer	0	0	0	0	0	0	No	No	Sí	Alguna vez
18	17	19	Hombre	4	0	1	1	1	1	No	No	No	Nunca
19	18	20	Hombre	3	7	7	2	0	0	Sí	Sí	No lo se	Nunca
20	19	20	Hombre	3	2	5	3	0	1	No	No	No	Alguna vez
21	20	21	Mujer	2	0	2	0	2	0	No	No	No	A menudo
22	21	19	Hombre	20	20	10	7	0	3	No	No	No lo se	Alguna vez
23	22	20	Hombre	1	0	0	1	1	0	No	No	No	Nunca
24	23	19	Hombre	0	0	0	0	0	0	No	No	Sí	Alguna vez
25	24	58	Hombre	12	6	7	4	3	2	No	Sí	Sí	A menudo
26	25	47	Hombre	0	0	3	3	0	2	No	No	No	Nunca
27	26	19	Hombre	1	0	5	1	0	5	Sí	No	No	Nunca
28	27	37	Mujer	10	5	6	3	3	3	Sí	Sí	Sí	Muy a menudo
29	28	50	Hombre	50	35	65	26	1	100	Sí	Sí	Sí	Muy a menudo
30	29	19	Hombre	0	0	0	0	0	0	No	Sí	No	Alguna vez
31	30	19	Hombre	23	0	10	11	12	69	Sí	No	No	A menudo

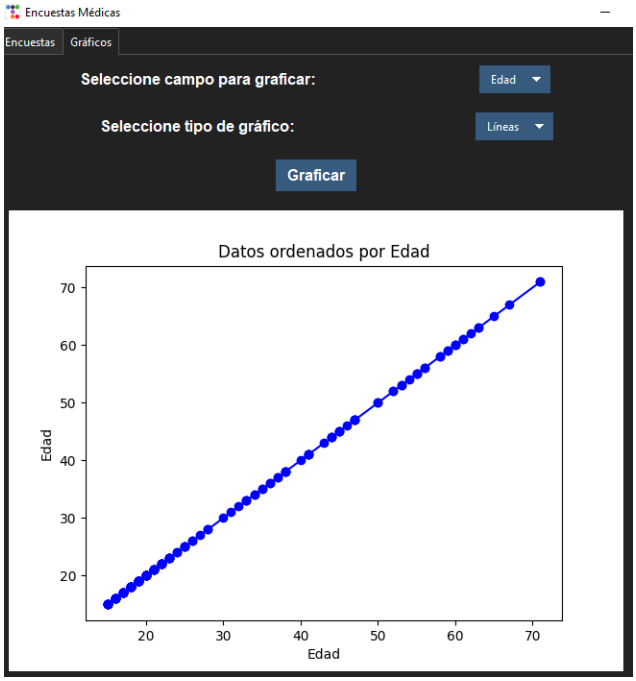
El código utilizado para lograr esta funcionalidad es el siguiente:

```
def exportar_datos(self):
    try:
        self.cursor.execute("SELECT * FROM ENCUESTA")
        rows = self.cursor.fetchall()
        if rows:
            df = pd.DataFrame(rows, columns=[desc[0] for desc in self.cursor.description])
            file_path = Querybox.get_string( prompt: "Guardar como", title: "Ingrese el nombre del archivo (sin extensión):", parent=self.root)
            if file_path:
                df.to_excel( excel_writer: f"{file_path}.xlsx", index=False)
                Messagebox.show_info( message: "Éxito", title: f"Datos exportados a {file_path}.xlsx", parent=self.root)
    except Exception as e:
        Messagebox.show_error( message: "Error", title: f"Error al exportar datos: {e}", parent=self.root)
```

VISUALIZACIÓN DE DATOS EN GRÁFICOS

La visualización de datos se realiza mediante gráficos de barras y líneas, que permiten observar tendencias en los datos, como la relación entre la edad y el consumo de alcohol. Los gráficos se generan utilizando la librería Matplotlib, y se pueden personalizar en función de los campos seleccionados por el usuario.





Como podemos observar, el usuario puede tanto elegir el tipo de campo que quiere graficar, como el tipo de gráfico que quiere utilizar. En este caso puede elegir entre un gráfico de barras o un gráfico de líneas.

CONCLUSIÓN

En este proyecto, se ha desarrollado una herramienta interactiva y funcional que combina el manejo eficiente de datos con una interfaz gráfica amigable. La implementación de las operaciones CRUD, la capacidad de ordenar y filtrar datos, la exportación a Excel y la visualización mediante gráficos confirman que el objetivo principal del proyecto (proporcionar un análisis accesible de los hábitos de consumo de alcohol y su impacto en la salud) se ha cumplido de manera eficaz.

A través de este trabajo, se han afianzado conocimientos en la integración de Python con bases de datos MySQL, el uso de bibliotecas como Matplotlib para la representación gráfica y la creación de interfaces con Tkinter.

BIBLIOGRAFÍA

PEP 8 – style guide for Python code. (n.d.). Python Enhancement Proposals (PEPs). Retrieved November 20, 2024, from https://peps.python.org/pep-0008/

tkinter — Python interface to Tcl/Tk. (n.d.). Python Documentation. Retrieved November 20, 2024, from https://docs.python.org/3/library/tkinter.html

User Guide — pandas 2.2.3 documentation. (n.d.). Pydata.org. Retrieved November 20, 2024, from https://pandas.pydata.org/docs/user_guide/index.html

Using matplotlib — matplotlib 3.9.2 documentation. (n.d.). Matplotlib.org. Retrieved November 20, 2024, from https://matplotlib.org/stable/contents.html

(N.d.-a). Mysql.com. Retrieved November 20, 2024, from https://dev.mysql.com/doc/refman/8.0/en/

(N.d.-b). Realpython.com. Retrieved November 20, 2024, from https://realpython.com/python-excel-spreadsheets/