

**UNIVERSIDAD NACIONAL DE TRUJILLO**

Facultad de Ingeniería

Escuela Profesional de Ingeniería Mecatrónica

DESARROLLO DE UNA APLICACIÓN PARA LA GESTIÓN DE HISTORIALES MÉDICOS DE LOS PACIENTES EN LOS CENTROS MÉDICOS



**Trabajo de Investigación formativa**

Programación I

**AUTOR (es):**

* **Sánchez Rojas, Jhonatan Artemio**
* **Vigo Villar, Cristhian Aaron**
* **Valdiviezo Jiménez, Víctor Javier**

**DOCENTE :**

**Asto Rodriguez, Emerson Máximo**

**CICLO :**

**III**

Trujillo, Perú 2022

# Resumen

El presente informe consiste en la elaboración de un proyecto de programación que busca la implementación de historiales clínicos digitales en aras de la agilización en el servicio médico, mediante la creación de un programa de escritorio, el cuál contribuya al almacenamiento y consultoría de historiales clínicos, donde el medico tendría acceso rápido a la ‘hoja de vida’, referida a asuntos netamente clínicos, del paciente, como atenciones, diagnósticos, prescripciones, tratamientos, sus datos, entre otros. El desarrollo del software requerirá del uso de herramientas de programación como como Python, visual studio code, biblioteca gráfica Tkinter y el sistema de gestión para una base de datos SQLite. La motivación que sigue el proyecto es la busca de la mejora progresiva en la atención médico-paciente, evitando así, el desgaste de los procesos internos administrativos en los centros médicos como el uso de hojas de papel y archivos físicos.

***Palabras Claves: Python, Tkinter, SQLite, historial médico***

# Abstract

This report consists of the elaboration of a programming project that seeks the implementation of digital medical records in order to streamline the medical service, through the creation of a desktop program, which contributes to the storage and consulting of clinical records, where the doctor would have quick access to the 'resume', referred to purely clinical matters, of the patient, such as care, diagnoses, prescriptions, treatments, their data, among others. The development of the software will require the use of programming tools such as Python, visual studio code, Tkinter graphic library and the management system for a SQLite database. The motivation followed by the project is the search for progressive improvement in medical-patient care, thus avoiding the wear and tear of internal administrative processes in medical centers such as the use of sheets of paper and physical files.

***Keywords: Python, Tkinter, SQLite, medical historial***

**Tabla de Contenidos**

Contenido

[Resumen 2](#_Toc110327401)

[Abstract 3](#_Toc110327402)

[Capítulo 1 Introducción 5](#_Toc110327403)

[1.1. Realidad Problemática 5](#_Toc110327404)

[1.2. Formulación del Problema 10](#_Toc110327405)

[1.4 Objetivos 10](#_Toc110327406)

[1.5 Justificación 10](#_Toc110327407)

[Capítulo 2 Marco Teórico 11](#_Toc110327408)

[Capítulo 3 Materiales y Métodos 14](#_Toc110327409)

[3.1. Descripción general de los procedimientos 14](#_Toc110327410)

[3.2. Desarrollo de los procedimientos 15](#_Toc110327411)

[3.2.2. (Objetivo 2) 17](#_Toc110327412)

[3.2.3. (Objetivo 3) 23](#_Toc110327413)

[Capítulo 4 Resultados 24](#_Toc110327414)

[4.1. (Objetivo 1) 24](#_Toc110327415)

[4.2. (Objetivo 2) 25](#_Toc110327416)

[4.3. (Objetivo 3) 25](#_Toc110327417)

[Capítulo 5 Conclusiones y Recomendaciones 26](#_Toc110327418)

[5.1. Conclusiones 26](#_Toc110327419)

[5.2. Recomendaciones 26](#_Toc110327420)

[Referencias Bibliográficas 27](#_Toc110327421)

[Bibliografía 27](#_Toc110327422)

[Anexos 28](#_Toc110327423)

[Anexo 1: 28](#_Toc110327424)

# Capítulo 1 Introducción

Aun estando en un siglo donde la tecnología es lo que se prioriza, siguen existiendo centros médicos que tramitan las historias clínicas manualmente, en otras palabras, las tienen almacenadas de manera física. Y en la mayoría de circunstancias, este hecho provoca un gran retraso en el tiempo de servicio que se le da a los pacientes que llegan para ser atendidos en dicho centro. El hecho de hacer la búsqueda de una historia clínica en un almacén y luego llevarlo hasta el consultorio médico, aumenta notoriamente el tiempo de atención al paciente. Este proyecto busca dar una solución al diseñar un sistema de control para historiales médicos, el cual permita una gestión más organizada y automatizada de las historias clínicas en un centro de atención médica, para perfeccionar el servicio de atención a sus pacientes.

## Realidad Problemática

* **Trabajos Previos:**

Sanunga y Pérez (2019) realizaron un proyecto con el objetivo de implementar un sistema que permitiera el registro y la gestión del historial del paciente, la entrada del tratamiento y las citas de los pacientes para acudir a la clínica. Se implementó como herramienta de apoyo al Centro Odontológico Grupo Dental. Los creadores del proyecto concluyeron que la implementación de un sistema de gestión de registros médicos permitió una mejor gestión y organización de los registros médicos de los pacientes y fue respaldado por el Centro Dental de Dental Group.

Por otro lado, dirigiéndonos al sector público, tenemos como antecedentes del trabajo, a la institución EsSalud, la cual, puso en marcha en el año 2019, un nuevo Sistema de Gestión de Servicios de Salud que se denominó EsSI (Servicio de Salud Inteligente).

En el cual comprende la digitalización de los historiales médicos mediante una interfaz virtual, en la cual los médicos podrían acceder a la toda la información médica del paciente: consultas, exámenes médicos, recetas y entre otros datos. Con la finalidad de reducir el tiempo y mejorar la atención en centros de salud, beneficiando a más de 11 millones usuarios del seguro.

Y, además, significo un gran acontecimiento en el sistema de Salud del Perú, siendo un primer paso, a gran escala, en la era de la digitalización integral de los procesos y servicios de Salud a nivel nacional; y sirviendo como base para el desarrollo de futuros proyectos en busca de la masificación de los historiales médicos digitales, como el presente proyecto. Y como un proyecto de desarrollo sostenible amigable con el medio ambiente, se sabe que, con la plataforma digital, la institución economizará cerca de cuatro millones de soles al no tener que invertir en papel. (EsSalud, 2019)

* **Datos Estadísticos:**

Perú ocupa el puesto 71 en el nivel de digitalización de servicios en el Índice de Desarrollo de Gobierno Electrónico de las Naciones Unidas (ONU) 2020, lo que lo convierte en uno de los últimos en América Latina en su conjunto. En cuanto a la adopción de la historia clínica electrónica, según la Organización Panamericana de la Salud (OPS), solo el 52,6% de los países que la conforman cuentan con un sistema nacional de historia clínica electrónica, y solo el 26,3% cuenta con leyes que avalan su uso.

Según datos del Ministerio de Salud (Minsa) al 2021, el porcentaje de establecimientos de atención primaria con historia clínica electrónica en Lima no llega al 40%, e incluso al 10% en zonas como Cajamarca. Lo ideal es cambiar esta situación con la digitalización y modernización del sector salud, integrando en un solo lugar la información generada por los profesionales de la salud, los pacientes y los sistemas de información.

Además, Según la Ley N°30024 y las normas de la Organización Mundial de la Salud, la historia clínica digital debe tener al menos con estas cuatro características: ser única, es decir, cada peruano debe tener un historial clínico único; estar integrada, lo que significa que toda prestación o servicio de salud que se le brinde al paciente debe figurar en su historia clínica; debe ser acumulativa, todo evento pasado o futuro debe de quedar archivado en su historia; y debe ser portable, pudiendo acceder a la historia clínica desde cualquier parte del territorio nacional. Y de las iniciativas o aplicativos desarrollados en el Perú, tanto en el sector público como privado, solo pocos cuentan con algunas de estas características, y hasta el año 2019 solo EsSalud cumplía en su totalidad con estos parámetros. (EsSalud, 2019)

Según Medigest Consultants, la digitalización de las historias clínicas traerá importantes beneficios a la gestión de las instalaciones médicas, entre ellos, la tasa de retorno de la inversión en espacio físico al evitar grandes cantidades de registros físicos y reducir el tiempo dedicado a buscar registros médicos y posibles errores de archivo.

De lo expuesto en párrafos anteriores es evidente que el proyecto planteado puede contribuir a la solución del problema, el implementar una aplicación de escritorio que permita almacenar el historial médico del paciente, puede ayudar a tener un mejor seguimiento del estado del paciente, además, permite economizar en cuanto a gastos de oficina.

En el marco donde se rige una historia clínica, se halla la Norma Técnica de Salud (NTS 022, 2006), la cual brinda los métodos tradicionales y convencionales de archivamiento de una historia clínica, incluyendo los formatos que ésta debe contener, tales como: las fichas familiares, formatos de emergencia, formatos de consulta externa, etc.

El 22 de mayo del 2013, el Congreso aprobó la (Ley N°30024, 2014), la cual instituye el Registro Nacional de Historias Clínicas Electrónicas (RENHICE), cuyo fin es recolectar datos de toda persona que haya sido atendida en cualquier centro de salud del Perú (público o privado). Además, el usuario o su presentante legal tienen permitido, además de los médicos, contar con acceso a los datos correspondientes a su respectiva historia clínica electrónica. (Ministerio, 2014)

En octubre del año 2015, se realizó la Jornada Internacional de Integración de Sistemas e Historia Clínica Electrónica, la cual tuvo como objetivo capacitar a todo el personal médico sobre las nuevas tecnologías de información (TI), las cuales se encuentran íntimamente relacionadas con sector Salud, debido a que estos serán las bases del desarrollo de la HCE y posteriormente para su implementación.

Durante el mes de mayo del 2014 “Lolimsa”, una empresa que desarrolla softwares para el sector salud, expresó que, hasta ese momento en el Perú, más del 89% de los datos de los pacientes no se encontraban en historias clínicas virtuales, y un pequeño porcentaje usaba algunos medios electrónicos y los demás centros médicos seguían empleando el archivado manual mediante el uso del papel. (Comercio, 2016)

En el mes de diciembre, el 17 del 2015 fue promulgado por el gobierno peruano el Decreto Supremo N° 039-2015-SA con respecto el reglamento de la (Ley N°30024, 2014), Ley que instituye el Registro Nacional de Historias Clínicas Electrónicas. Mediante este decreto, las clínicas y hospitales deben seguir labores para adecuarse a la nueva realidad.

EsSalud últimamente hace uso de historias clínicas electrónicas, alineadas a los objetivos de la Política Nacional de Gobierno Electrónico 2013-2017. Este sistema permite que los procesos en la atención sean más rápidos y eficaces hacia los usuarios. Osmeli Navarro, Gerente de Procesos Asistenciales en IBTgroup, Lima Perú (sociedades de operadoras de salud), define a este sistema como una herramienta que agiliza y mejora el trabajo. (Yrinna Benites, 2016)

## Formulación del Problema

¿Cómo desarrollar una aplicación para la gestión de historiales médicos que mejore la calidad de servicio de un centro médico?

## Objetivos

* + 1. **Objetivo General**

Desarrollar una aplicación para la gestión de historiales médicos de los pacientes para mejorar la calidad de servicio en los centros médicos.

* + 1. **Objetivos Específicos**
       1. Implementar la base de datos.

2.Implementar una base de datos.

3.Elaborar la interfaz en Python

4. Implementar el acceso directo al software.

5.Pruebas del sistema

## Justificación

* **Social:** Mejorar la calidad de servicio y atención a los pacientes.
* **Económica:** Disminuirá la inversión en los materiales de oficina, en este caso, el uso del papel bond.
* **Tecnología:** Implementar este tipo de sistema, habré las a futuro de mejorar las interfaces tecnológicas.

# Capítulo 2 Marco Teórico

**Python:**

Python es un lenguaje de programación, el cual tiene mucha similitud con el lenguaje humano. Por otro lado, se define como un lenguaje de código abierto y lo mejor de todo, “gratuito”, lo cual permite desarrollar software sin limitaciones.

El usar Python está muy extendido en dos áreas que son las más desarrolladas actualmente: el análisis de datos y el big data. La simplicidad de este lenguaje y su gran número de bibliotecas para procesar datos, la hacen ideal a la hora de hacer un análisis y gestión de una gran cantidad de datos.

**Tkinter:**

Es una librería que viene preinstalada en Python, la cual podemos usar en cualquier momento, utilizando el comando “import tkinter”. Además, es un conjunto de herramientas GUI de Tcl/Tk (Tcl: Tool Command Language), que proporciona una gran variedad de usos, en las cuales está el desarrollo de aplicaciones de escritorio, aplicaciones web, también se desempeña en la administración, entre otros. Tkinter no es solo la única librería para python especializada en la creación de interfaces gráficas, entre las más utilizadas están wxPython, PyQt y PyGtk.

**SQLite:**

SQlite es un software gratuito, el cual permite almacenar información en diferentes dispositivos, tales como una laptop, PC, Tablet o celular. Dicho software es compatible con múltiples plataformas, lo cual hace que la integración sea perfecta. SQlite posee su propio lenguaje de programación, además de contar con muchos componentes, bibliotecas y controladores que nos permiten interactuar con diferentes lenguajes de programación y plataformas.

**Visual Studio Code:**

Visual Studio Code es un editor de código de fuente propiedad de Microsoft. Es un software libre y además es compatible con muchos sistemas operativos, tales como: Windows, Linux y macOs. VS Code posee una gran cantidad de extensiones gratuitas, las cuales facilitan escribir y ejecutar códigos en cualquier lenguaje de programación.

**Historia clínica:**

El historial médico se puede obtener desde una variedad de perspectivas, desde una perspectiva gramatical, una perspectiva legal, un concepto médico o una perspectiva forense, que es un documento médico requerido por ley para registrar toda la información: Relaciones cuidador-paciente, todas las acciones y actividades de higiene médica realizadas sobre ellos, y todos los datos relacionados con su salud elaborados para facilitar su cuidado.

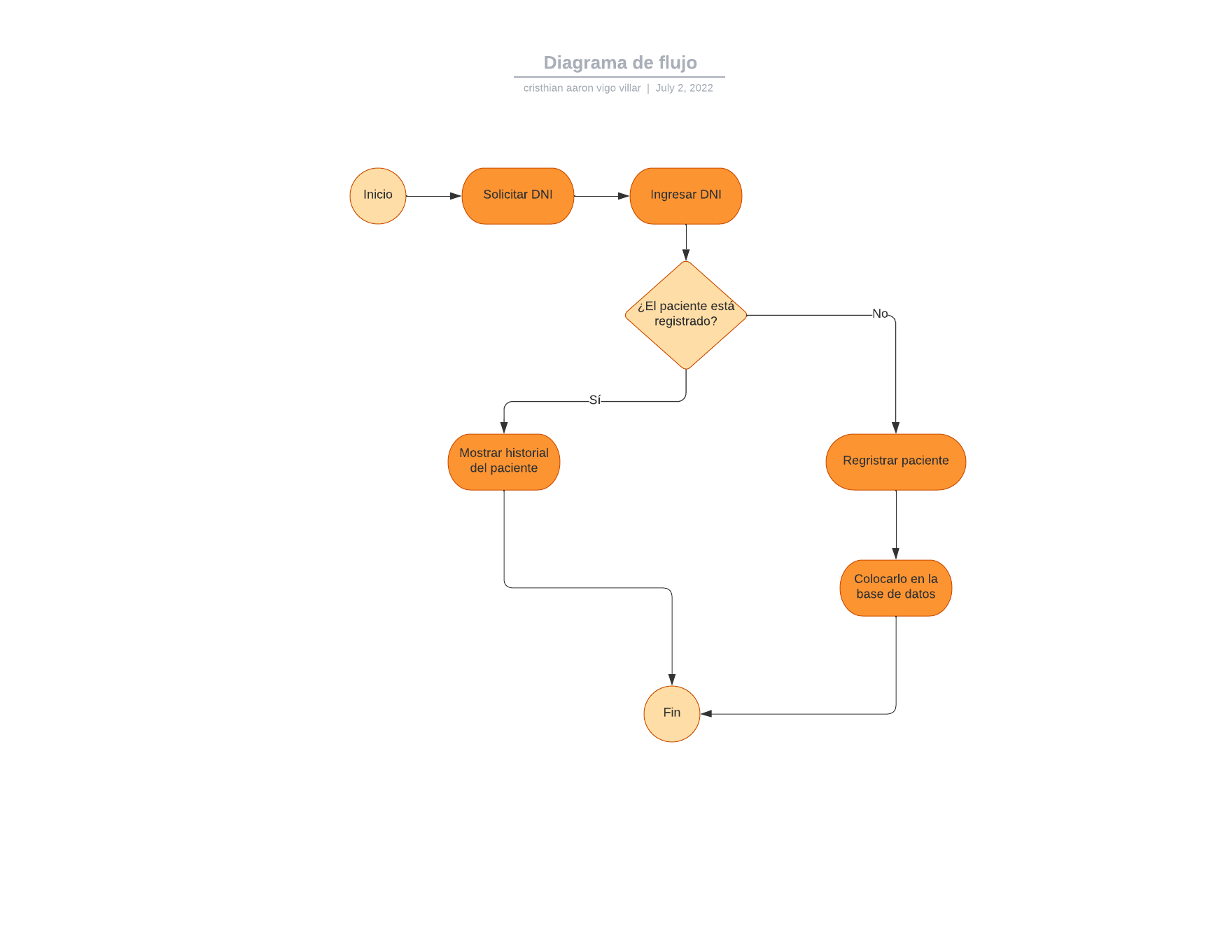
El objetivo principal de la historia clínica es recopilar datos sobre la salud del paciente y promover la atención médica. La razón por la que los médicos comienzan a obtener un historial médico y lo motivan a continuar durante mucho tiempo es la necesidad de servicios médicos por parte del paciente.

Usando las herramientas anteriores y el conocimiento del historial médico, se desarrollará una aplicación de escritorio que pueda recopilar datos médicos del paciente durante una visita al centro de salud. El historial médico del paciente es muy importante porque le da al paciente un antecedente.

# Capítulo 3 Materiales y Métodos

Para el desarrollo de este programa de escritorio, se hará uso del lenguaje de programación Python con la librería Tkinter, cuyo caso de estudio son los historiales médicos

## Descripción general de los procedimientos



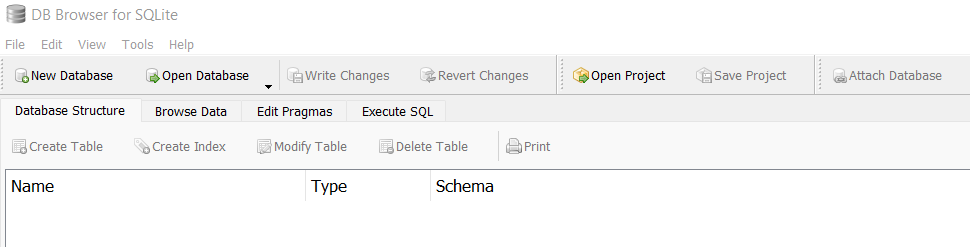
## Desarrollo de los procedimientos

* + 1. **(Objetivo 1)**

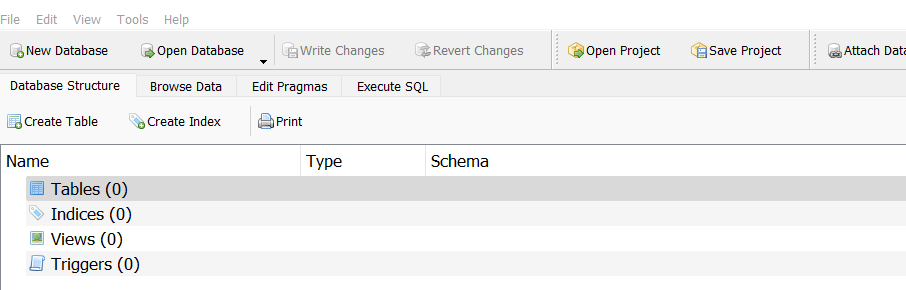
Para la implementación de la base de datos, se usó el software DB Browser for SQlite, en dicho software creamos las tablas que posteriormente se importaran al código general:

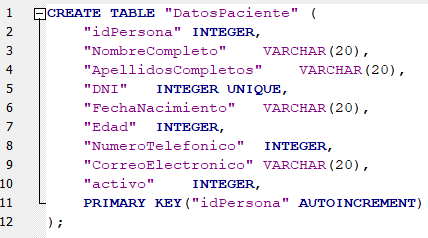
* Iniciamos creando la tabla que contendrá los datos del paciente, los datos que se tomaran del paciente están expresados en la siguiente imagen (ver [Anexo 1](#_Anexo_1:)). El software DB Browser posee su propio lenguaje de programación, a continuación, se explicará paso a paso:

1. Abrimos la aplicación y damos click en donde dice: New Database.



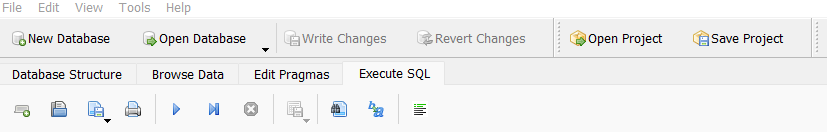
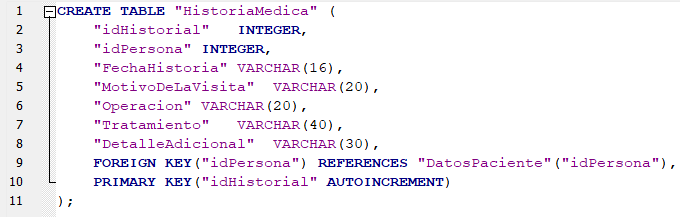
1. Automáticamente se crea la sección de tablas, pero están vacias, por lo tanto, le damos click en “Execute SQL” para ingresar los códigos correspondientes y poder crear las tablas.



1. Se digitó el siguiente código para crear la tabla que contenga los datos del paciente:

Donde:

* CREATE TABLE: Este comando nos permite crear la tabla.
* INTEGER: Este comando se usa para guardar valores numéricos enteros.
* INTEGER UNIQUE: Similar al comando INTERGER, solo que al tener el UNIQUE, garantiza que no se duplique en las columnas o secciones establecidas.
* VARCHAR: Nos permite colocar el número de caracteres permitidos.
* PRIMARY KEY: Esto básicamente es una restricción para que cada se identifique de forma exclusiva.
* AUTOINCREMENT: Este comando es para generar un número único cuando se inserta algún nuevo registro en la tabla.

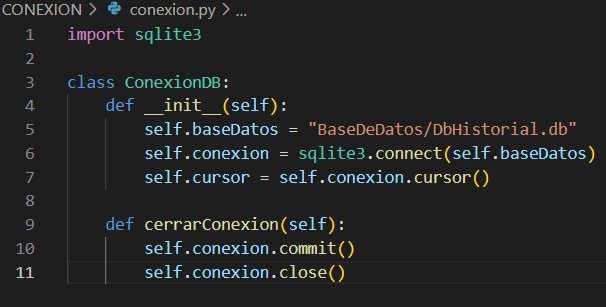
1. Posteriormente al haber colocado el código, le daremos click al siguiente para que el código se ejecute y encuentre algún de ser el caso:
2. De manera análoga se repitió los pasos anteriores para crear la tabla de Historia Médica:

Dos nuevos comandos aparecen en este caso:

* FOREIGN KEY: Sirve para señalar la clave primaria (PRIMARY KEY) de otra tabla.
* REFERENCES: Esto acompaña a FOREIGN KEY para señalar a que tabla está haciendo referencia.

## (Objetivo 2)

Para poder elaborar la interfaz en Python, es necesario crear archivos por separado, en los cuales este la conexión, la GUI, datos del paciente y uno que junte lo anterior mencionado.

* Comenzamos estableciendo la conexión con la base de datos que anteriormente se implementó, para lo cual se creó un archivo Python con el nombre de conexión.py, en el cual se digitó el siguiente código:

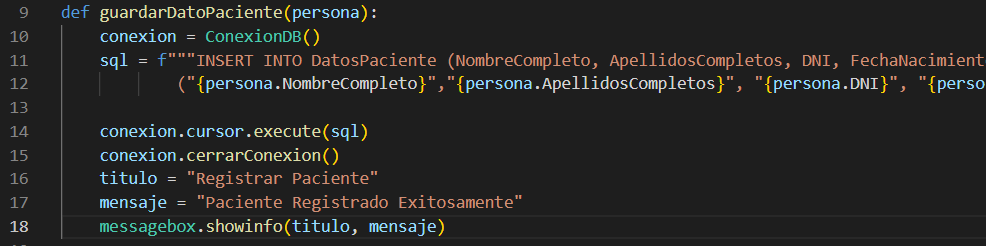
Importamos el módulo sqlite3, el cual nos permitirá establecer la conexión con la base de datos anteriormente implementada. Después colocamos la clase ConexionDB, en la cual se definirán, primero la función \_\_init\_\_(self), el self es un parámetro que nos permite hacer referencia al nombre de algún objeto (en la imagen puede apreciarse). En el renglón 5 se establece la ruta en donde se encuentra el archivo .db (base de datos), en el renglón 6 se usó una función del módulo sqlite3 para hacer referencia al renglón 5 y así poder conectarnos a la base de datos. En el renglón 7 se utilizó el método cursor el cual nos permite modificar los datos.

Luego se definirá la función cerrarConexion(self) en el renglón 9, en el renglón 10 hacemos uso del método commit que nos permitirá subir los datos insertados en las entradas, además asegura que los cambios realizados sean coherentes en la base de datos. Por último, en el renglón 11 usamos el método close para cerrar la conexión luego de insertar los datos.

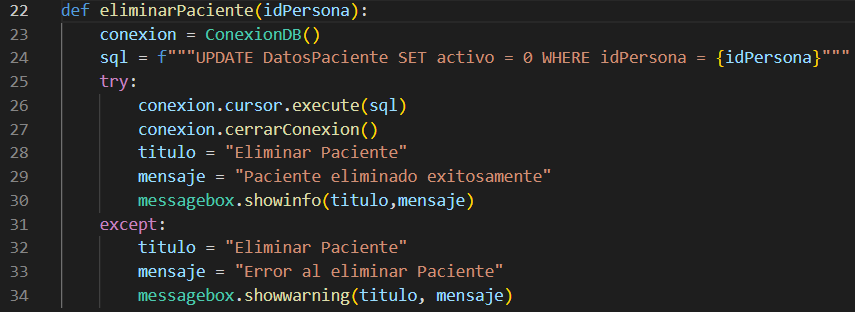
* Luego de haber creado el archivo conexión.py, se creará otro archivo Python con el nombre PacienteDao.py, en el cual se establecerán funciones que permitan vincular a la base de datos, a la GUI y la clase DatosPaciente.

Una vez creado el archivo, se importará la clase ConexionDB del archivo conexión.py y el módulo messagebox de la librería tkinter, el cual servirá para mostrar ventanas emergentes:

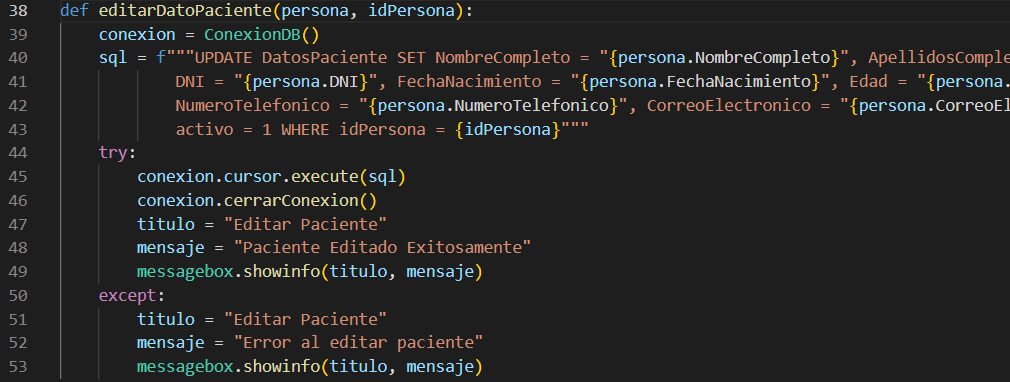
Ahora, se colocará las funciones que vinculen a la base de datos creada en DB Browser for SQlite:

* + - 1. **Función para guardar los datos del paciente:**

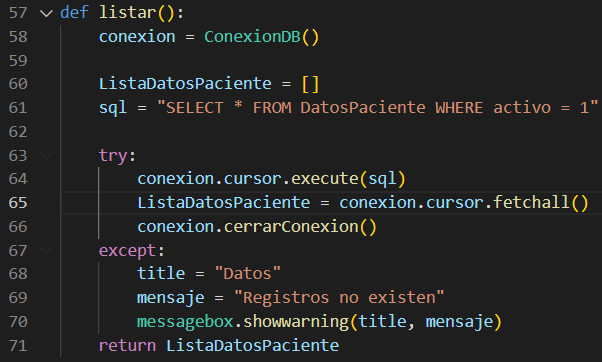
Definimos la función guardarDatoPaciente(persona) (persona es un parámetro) (esta función guardará los datos en la clase DatosPaciente, la cual se implementará más adelante), en el renglón 10 establecemos la conexión con la base de datos (esto se colocará en cada función que se cree). En el renglón 11 establecemos la variable sql (string) que permite almacenar texto, en este caso los datos del paciente (nombre completo, apellidos, etc). En el renglón 14 usamos el método execute para ejecutar el sql (los datos que se agreguen) y en el renglón 15 se cierra la conexión. Por último, en el renglón 18 se aplica el modulo messagebox junto con la función showinfo los cuales permitirán mostrar el titulo y mensaje en la interfaz a la hora de guardar los datos del paciente.

* + - 1. **Función para eliminar los datos del paciente:**

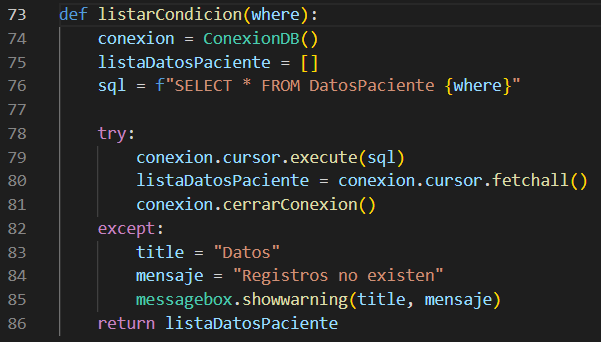
Definimos la función eliminarPaciente(idPersona) (idPersona será el parámetro) (esta función no elimina los datos del paciente de manera permanente, solo los inactiva), en el renglón 23 establecemos la conexión con la base de datos y el renglón 24 establecemos la variable sql la cual tiene almacenada el idPersona (este número se genera automáticamente al registrar un paciente y es único). Se hizo uso de try y except, en caso que el try presente errores, se ejecutará el except. Del renglón 26 al 30 es similar a lo hecho en del renglón 14 al 18. En el except se muestra la función showwarning, la cual es muy similar al showinfo, solo que muestra mas alerta en el mensaje.

* + - 1. **Función para editar los datos del paciente:**

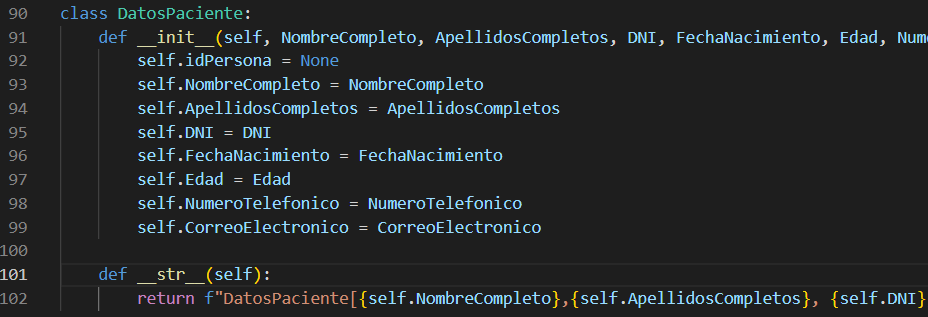
Definimos la función editarDatoPaciente(persona, idPersona) (esta función permitirá sobrescribir los datos en cada objeto persona de la clase DatosPaciente, la cual se implementará posteriormente), en el renglón 39 establecemos la conexión como anteriormente se ha hecho, en el renglón 40 se establece la variable sql, la cual contiene los datos. Por último, del renglón 44 al 53 se hizo lo mismo que en la función anterior, solamente se cambio el titulo y mensaje que se mostrará.

* + - 1. **Funciones para mandar los datos insertados a la tabla que aparece en al GUI:**
* **Función Listar:**

Definimos la función lista(), la cual nos va a permitir mandar los datos a una lista, en el renglón 58 establecemos la conexión, en el renglón 60 establecemos la variable LisraDatosPaciente, la cual representa la lista en donde se guardaran los datos. En el renglón 61 se establece la variable sql para seleccionar los pacientes que se encuentren activos (activo = 1). Del renglón 63 al 71 se repite casi mismo que en las anteriores funciones, salvo en el renglón 65 en donde se añade el método fetchall, el permitirá añadir los datos a la lista, además también se cambió el titulo y mensaje mostrado en el except, al final todo retorna a la ListaDatosPaciente.

* **Función listarCondicion:**

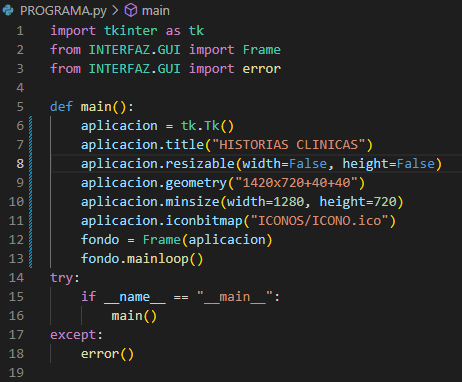
Definimos la funcion listarCondicion(where) (where es un parámetro) (esta función mandará los datos de la lista a la tabla que aparece en la GUI), en el renglon 74 establecemos la conexión, en el renglón 75 colocamos la variable listaDatosPaciente, la cual representa a la lista, después en el renglón 76 establecemos la variable sql, la cual seleccionará los datos de paciente activos. Por último, del renglón 78 al 86, es similar a la función anterior.

Ahora se creará la clase DatosPaciente, la cual estuvo vinculada a las funciones anteriores:

Definimos la clase DatosPaciente, luego en el renglón 91 definimos el método \_\_init\_\_ con sus respectivos parámetros (dichos parámetros son las columnas que están presentes en la tabla DatosPaciente que se creó en la base de datos), porteriormente del renglon 92 al 99 se hizo uso del self para hacer referencia a los nombres de las columnas que están presentes en la tabla DatosPaciente. Por último, definimos en el renglón 101 el método \_\_str\_\_(self), el cual mostrará como objetos en la GUI lo colocado en el renglón 102.

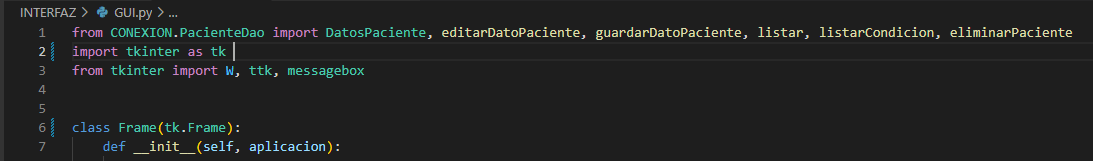
**Interfaz principal**

Creamos la interfaz principal en el archivo “PROGRAMA.py” importando el módulo tkinter y definimos la función principal “main()” que será la ejecutora del programa general. Dentro de la función especificamos los valores para la interfaz como el título, tamaño, el color, entre otros.

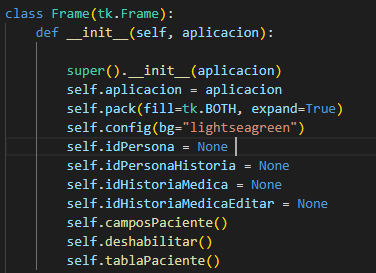


**La GUI**

Creamos la GUI, dentro del archivo “INTERFAZ” e importamos las conexiones creadas en pacienteDao y el módulo tkinter.



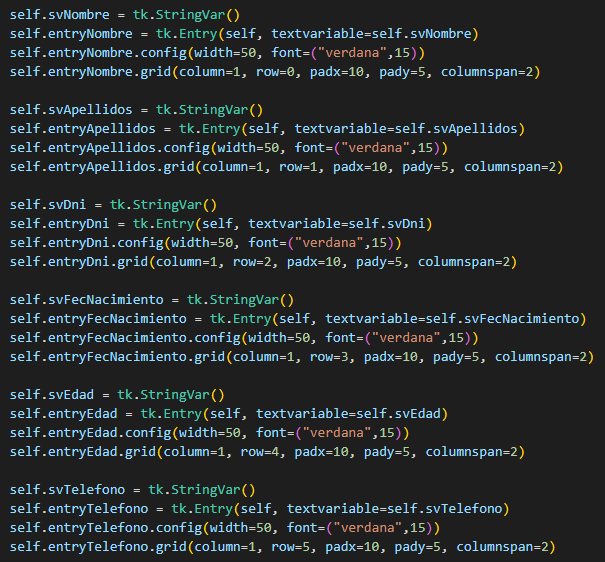
Creamos la clase “frame(tk.Frame)”, dentro de la cual definiremos el constructor “\_\_init\_\_(self, aplicación)”, que será el encargado de ejecutar las funciones que se le pasen, las cuales las crearemos más adelante, como el campo paciente y la tabla paciente.

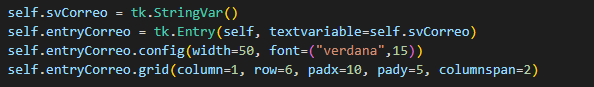


Creamos la función “campoPaciente(self)”, donde irán los ítems de la interfaz principal para ingresar los datos del paciente, llamados “labels”. A los cuales les colocaremos el texto del respectivo ítem con “text=” y lo personalizaremos en el apartado “.config”: su tipo de letra y tamaño con “Font=”, su color con “background=” y su ubicación dentro del widgets con “anchor=”. Además de la ubicación del widget en general, en el apartado “.grid” con los comandos “column=” para especificar columna y “row=” para especificar columna.

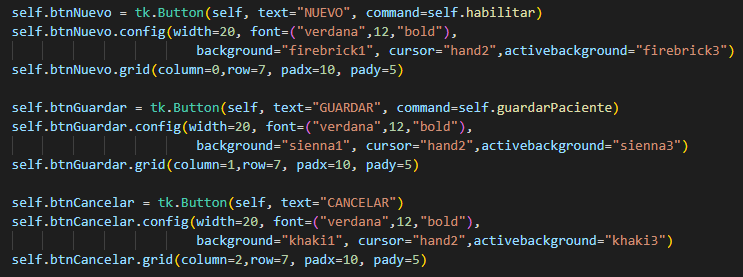


Creamos los espacios de inserción de los datos pedidos por los labels, los cuales son llamados entrys. Los definimos mediante la función StringVar() y les agregamos tipo de letra, tamaño y ubicación de la misma forma que los labels. Además, de algunas opciones de edición como el comando “columnspan=” que define las columnas que ocupara el entry, en este caso, 2.

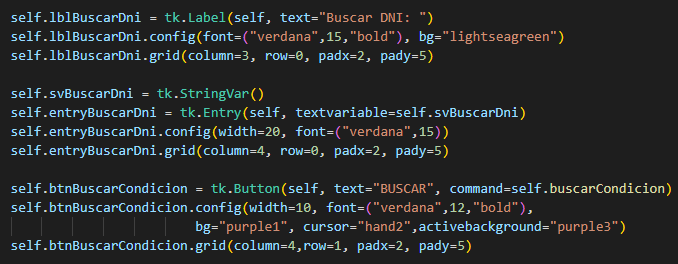




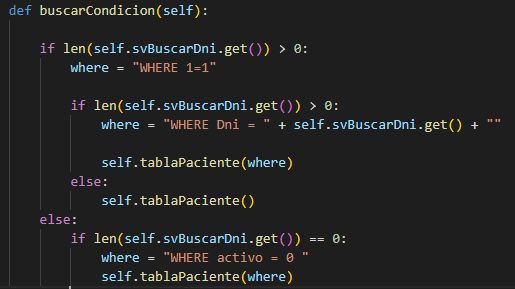
Creamos los botones, necesarios para el campo paciente como el “NUEVO”, “GUARDAR” y “CANCELAR”. Y también les agregamos los valores de tamaño, tipo de letra, color y ubicación como con los anteriores elementos.



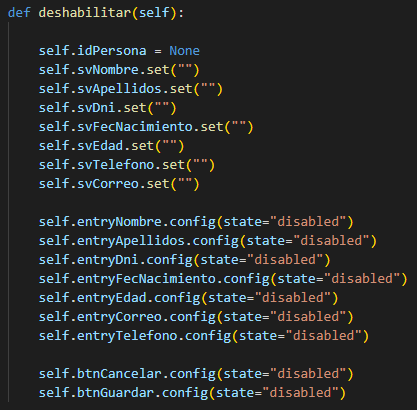
Creamos la función de buscador. Empezando por la creación de su respectivo label, entry y button. Y también les agregamos los valores de tamaño, tipo de letra, color y ubicación como con los anteriores elementos.



Luego, definimos la función “buscarCondicion(self)” donde le asignaremos el funcionamiento de búsqueda al apartado compuesto por el “lblBuscarDni”, “entryBuscarDni” y “btnBuscarCondicion”, la cual trabajara con un condicional que restringe la búsqueda a los pacientes mediante su Dni.



Creamos la función “deshabilitar()”, la cual mantendrá los entrys bloqueados y en blanco con la finalidad de que no se pueda escribir nada hasta que se seleccione el botón nuevo. Al cual, más adelante, le agregaremos una función que contrarreste este bloqueo.



Creamos la función “habilitar()”, la cual habilita en blanco los entrys para que se puedan colocar los datos. Esta función es inversa a “deshabilitar()” y se le agregara al botón NUEVO, para habilitar el ingreso de los datos del paciente con este botón, mediante el comando “command=self.habilitar”.

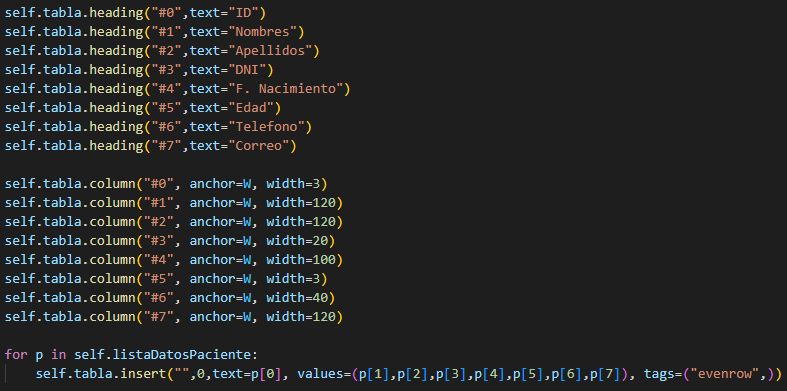


Creamos la función “tablaPaciente(self, where=”n”):” para insertar la tabla de datos de los pacientes desde el Sqlite a la interfaz principal. Acá, hacemos el llamado a la función “listar()” y “listarCondicion()” desde la Dao, para establecer el condicional que permita el listado de datos de los pacientes activos, caso contrario se muestren los pacientes con la condición requerida.

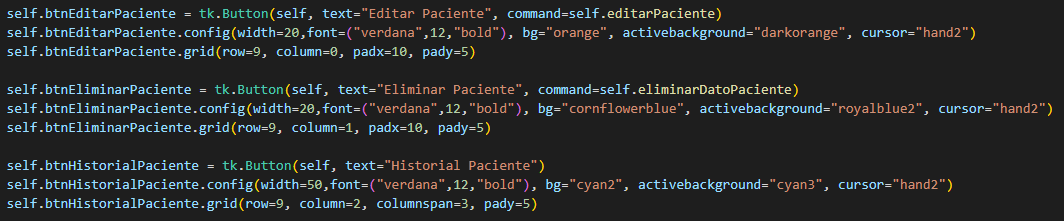
Creamos la tabla con usando el comando “ttk.Treeview”, definiendo las columnas que tendrá, su ubicación, su tipo de letra y su color de filas, de manera similar a como se configuraron estos parámetros para elementos anteriores. Como comando adicional usamos el “ttk.Scrollbar” el cual crea el parámetro para desplazarse (subir y bajar) en la interfaz de la tabla.



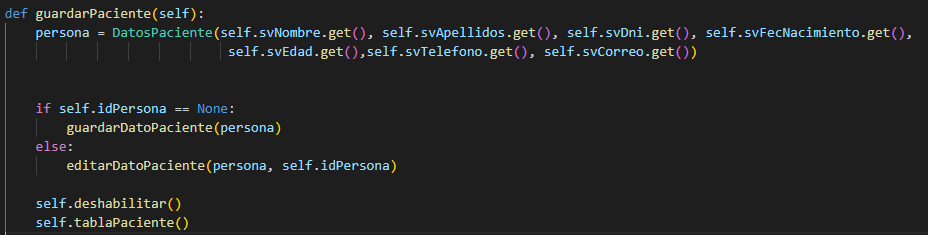
Luego, definimos los títulos de cada columna mediante la función “.heading()” y algunas configuraciones de detalle de estas mismas como el anchor y el width con la función “.column”. Insertamos los datos desde el Sqlite hacia la tabla, asignándole a cada valor una columna en especial e iterando en filas mediante el “for”.



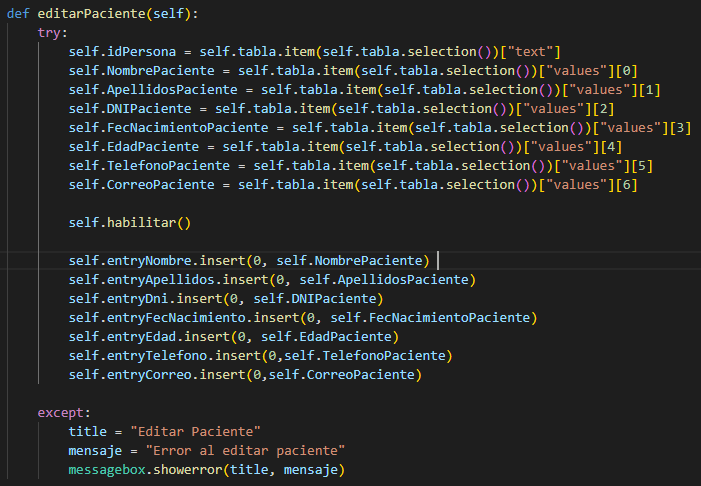
Creamos los botones que trabajaran con la tabla de datos de los pacientes: “Editar Paciente” “Eliminar Paciente” y “Historial Paciente”. Los cuales fueron configurados de forma similar a los botones antes vistos.



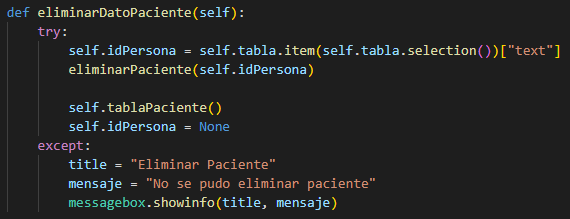
Creamos la función “guardarPaciente(self):” que trabajara con el botón “GUARDAR” y conectara los datos ingresados en los entrys del campo paciente, con la tabla en Sqlite, agregando cada paciente a la tabla. Todo esto, con la ayuda de la función “guardarDatoPaciente(persona)” llamada desde el PacienteDao. Además, crea una funcionalidad para el caso de la edición de un historial de paciente, mediante el condicional, donde si el idPersona es nuevo, se guarda los datos como nuevo, en caso contrario se editarán los datos del idPersona puesto, evaluándolo con la función “editarDatoPaciente()” tambien llamada desde el PacienteDao.



Creamos la función “editarPaciente(self):” la cual lleva los datos de la tabla hacia los entrys del campo paciente. Esto consiste solo en traslado de datos, una vez los datos se ubiquen en los entrys, se ejecutará la función de “guardarPaciente()” en la cual también se definió la función editar según sea el caso. Luego, la función se agregará al botón “Editar Paciente” para su posterior ejecución mediante este botón.



Creamos la función “eliminarDatoPaciente(self):” la cual llamara a la función “eliminarPaciente(idPersona)” desde la Dao, para usarlo directamente como cambio de activo en la función. Luego, la función se agregará al botón “Eliminar Paciente” para su posterior ejecución mediante este botón.



Definimos la función “error()” para diferentes casos donde posiblemente pueda ocurrir errores que convengan en la paralización del programa y la ejecutamos en el código principal mediante un “try except”



## (Objetivo 3)

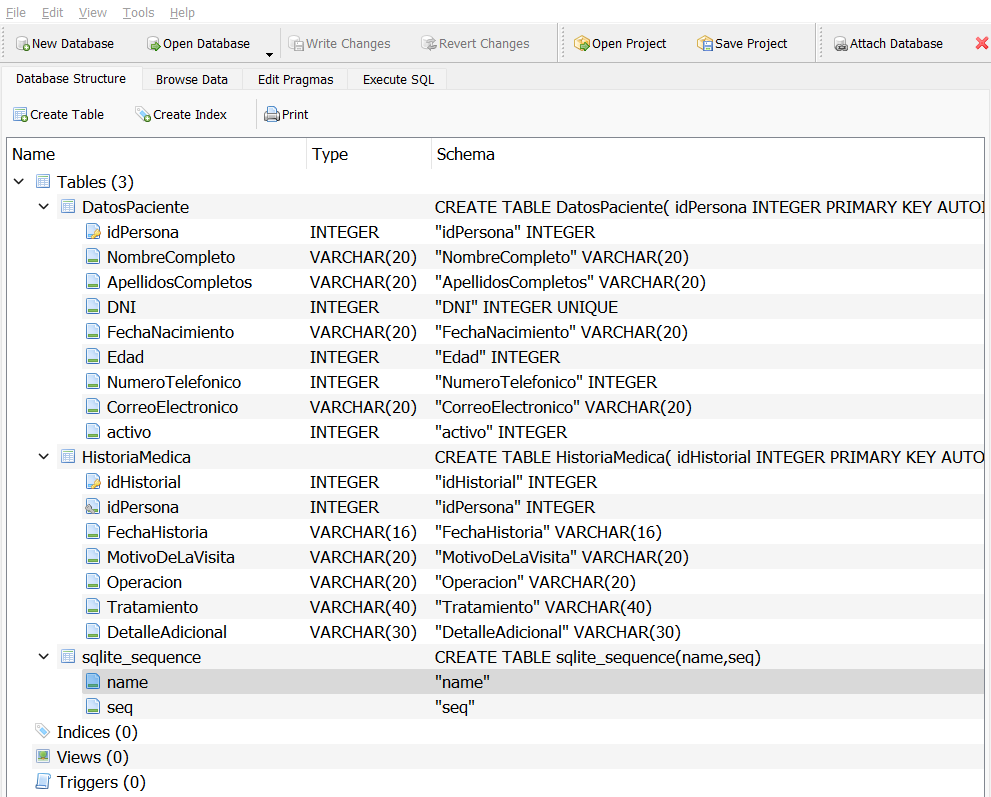
Descripción detallada de los procedimientos realizados para lograr el objetivo 1

..

# Capítulo 4 Resultados

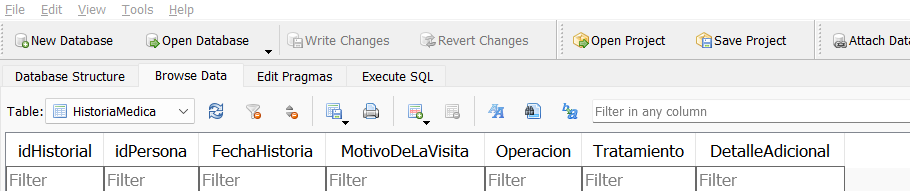
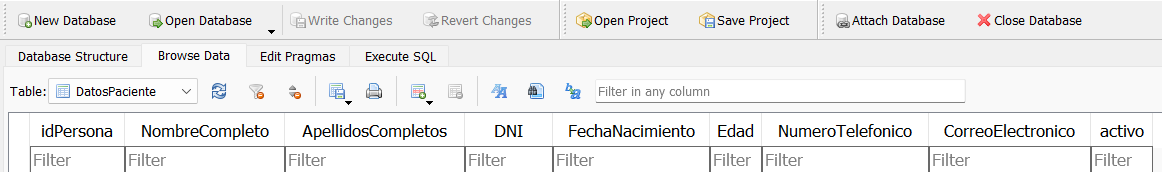
En este capítulo, se discutirán los resultados obtenidos del proyecto de investigación o del caso de estudio analizado. Es recomendable presentar los resultados respetando las etapas mostradas en el capítulo anterior.

## (Objetivo 1)

Una vez ejecutado los códigos para la creación de tablas, nos queda lo siguiente:

La tabla “sqlite\_sequence” se genera de manera automática al ejecutar los códigos de creación de tablas.

Dando click en “Browse Data” se visualiza las columnas de las tablas:



## (Objetivo 2)

Síntesis de los resultados obtenidos para el objetivo específico 2.

## (Objetivo 3)

Síntesis de los resultados obtenidos para el objetivo específico 3.

...

# Capítulo 5 Conclusiones y Recomendaciones

## Conclusiones

* + - Conclusión del objetivo 1
    - Conclusión del objetivo 2
    - Conclusión del objetivo 3
    - ...

## Recomendaciones

Sugerencias relacionadas a las dificultades encontradas en la elaboración del proyecto

# Referencias Bibliográficas

EsSalud. (26 de Septiembre de 2019). *Implementa historia clínica digital para atención de asegurados*. Obtenido de EsSalud: http://noticias.essalud.gob.pe/?publicacion=implementa-historia-clinica-digital-para-atencion-de-asegurados

Fernandez, R. (4 de Enero de 2021). *¿Qué es Tkinter y por qué utilizarlo?* Obtenido de Unipython: https://unipython.com/tkinter-introduccion/

Flores, F. (13 de Mayo de 2022). *¿Qué es Visual Studio Code y qué ventajas ofrece?* Obtenido de OpenWebinars: https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/

RPP Noticias. (2 de Febrero de 2022). *Historia clínica eléctronica: Cinco ventajas de tener un sistema digital en todos los centros de salud*. Obtenido de RPP Noticias: https://rpp.pe/campanas/valor-compartido/historia-clinica-electronica-cinco-ventajas-de-tener-un-sistema-digital-en-todos-los-centros-de-salud-el-pais-que-queremos-noticia-1383397?ref=rpp

Santander Universidades. (1 de Marzo de 2022). *¿Qué es Python?* Obtenido de Becas Santander: https://www.becas-santander.com/es/blog/python-que-es.html

Sanunga Totoy, J. E., & Pérez Palma, K. N. (28 de Enero de 2019). *Repositorio Institucional de la Universidad Politécnica Salesiana: Implementación del sistema para el control de historia clínica de pacientes en centro odontológico Dental Group*. Obtenido de DSpace: https://dspace.ups.edu.ec/handle/123456789/16767

Silvia Velito, A., & Tejada Soriano, S. R. (2010). *La historia clinica como instrumento de calidad*. Obtenido de autoriamedicahoy.com: http://www.auditoriamedicahoy.com/biblioteca/La%20historia%20cl%C3%ADnica%20como%20instrumento%20de%20calidad%20Tejada%20Velito.pdf

Yrinna Benites, K. A. (19 de Noviembre de 2016). ANÁLISIS Y DISEÑO DE PROTOTIPO DE SOFTWARE PARA LA AUTOMATIZACIÓN DE HISTORIAS CLÍNICAS DEL POLICLÍNICO UDEP. Piura, Perú.

**REFERENCIAS LINKOGRÁFICAS**

ANACONDA. (2018). Understanding Conda and Pip. Retrieved from https://www.anaconda.com/blog/understanding-conda-and-pip

Python. (2022). tkinter — Interface de Python para Tcl/Tk. Retrieved from https://docs.python.org/es/3/library/tkinter.html

Estrada web group. (2022). ¿Qué es el tipo de dato varchar de SQL Server y cuando utilizarlo? Retrieved from https://estradawebgroup.com/Post/-Que-es-el-tipo-de-dato-varchar-de-SQL-Server-y-cuando-utilizarlo-/20355

SALVADOR, U. D. EL. (2019). Introducción a SQLITE. Retrieved from https://eisi.fia.ues.edu.sv/materialpublico/pdm115/2019/labs/PDM115\_guia\_lab03\_SQLite.pdf

Runebook.dev. (n.d.). Tipos de datos en SQLITE. Retrieved from https://runebook.dev/es/docs/sqlite/datatype3

TUTORIAL, P. (2022). Tkinter Object-Oriented Window. Retrieved from https://www.pythontutorial.net/tkinter/tkinter-object-oriented-window/

Guia Tkinter. (2015). Interfaz gráfica con Tkinter. Retrieved from https://guia-tkinter.readthedocs.io/es/develop/index.html

FJSevilla. (2020). Fuentes disponibles en Tkinter. Retrieved from https://es.stackoverflow.com/questions/330481/fuentes-disponibles-en-tkinter

Python. (2022). Envoltorio de fuente Tkinter. Retrieved from https://docs.python.org/es/3.9/library/tkinter.font.html

Colors-symbolic color names recognized by Tk. (2018). Retrieved from https://www.tcl.tk/man/tcl8.5/TkCmd/colors.html

Patricia. (2019). Tabla de colores Tkinter. Retrieved from http://patriciaemiguel.com/python/2019/08/01/python-tkinter-colores.html

SQLite. (2022). SQLite. Retrieved from https://www.sqlite.org/index.html

Anaconda / packages / sqlite. (2022). Retrieved from https://anaconda.org/anaconda/sqlite

Python 3 para impacientes. (2016). Retrieved from https://python-para-impacientes.blogspot.com/2016/02/variables-de-control-en-tkinter.html

Mizipzor. (2009). Understanding Python super() with \_\_init\_\_() methods [duplicate]. Retrieved from https://stackoverflow.com/questions/576169/understanding-python-super-with-init-methods

Educative.io. (n.d.). What is super() in Python? Retrieved from https://www.educative.io/answers/what-is-super-in-python

Dev Prakash, S. (2021). How to Disable / Enable a Button in TKinter? Retrieved from https://www.tutorialspoint.com/how-to-disable-enable-a-button-in-tkinter#:~:text=Tkinter Button widgets can be,and disabling the button%2C respectively.

tutorialesprogramacion. (n.d.). Tipos de datos básicos para definir los campos de una tabla. Retrieved from https://www.tutorialesprogramacionya.com/sqliteya/detalleconcepto.php?punto=4&codigo=4&inicio=0

JC Chouinard. (2020). if \_\_name\_\_ == ‘\_\_main\_\_’: What does it mean (Python). Retrieved from https://www.jcchouinard.com/python-if-name-equals-main/#:~:text=When you start working with,What does it mean%3F&text=What this function do is,the module from another one.%0A

Python. (2022). tkinter.messagebox. Retrieved from https://docs.python.org/3/library/tkinter.messagebox.html

Python. (n.d.). tkinter.ttk — Tk widgets temáticos. Retrieved from https://docs.python.org/es/3/library/tkinter.ttk.html%0A

Python. (n.d.). tkinter.ttk — Tk widgets temáticos. Retrieved from https://stackoverflow.com/questions/19561727/what-is-the-difference-between-the-widgets-of-tkinter-and-tkinter-ttk-in-python

recursos python. (2017). Posicionar elementos en Tkinter. Retrieved from https://recursospython.com/guias-y-manuales/posicionar-elementos-en-tkinter/

Manuel, G. (2022). Python-Tkinter. Retrieved from https://www.youtube.com/watch?v=Yvs7YFmKwuE&list=PLh7JzoyIyU4LKz9h3KC7VNrKPdkm0o8N4&index=20&ab\_channel=ManuelGonzález%0A

RIP Tutorial. (n.d.). Difference between Tk and Toplevel. Retrieved from https://riptutorial.com/tkinter/example/22130/difference-between-tk-and-toplevel

# Anexos

## Anexo 1:

