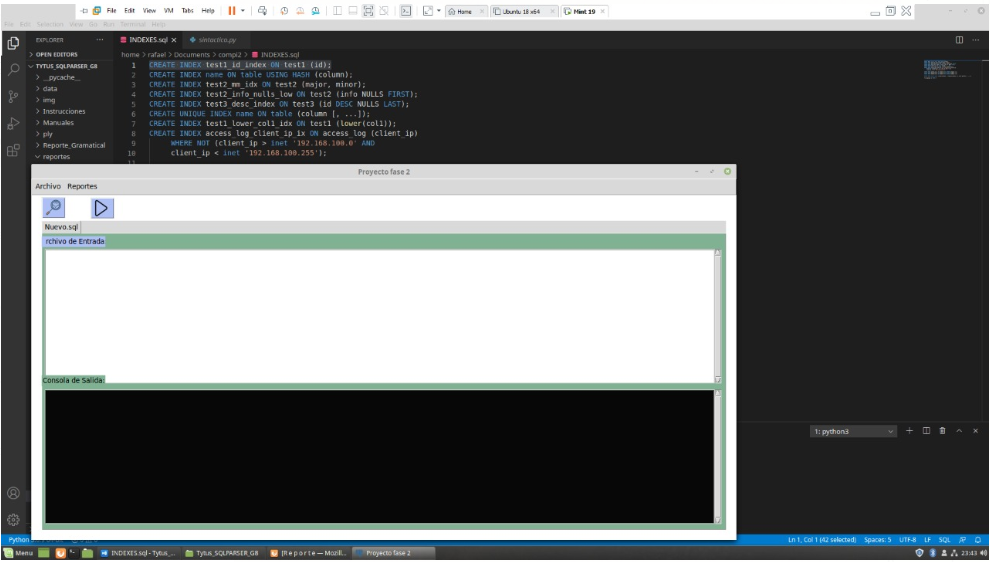
MANUAL DE USUARIO FASE2

\*.- GUI

La interfaz gráfica de usuario cuenta con dos opciones siguien la primera El toolbars para Archivo en la cual se encuentra “Abrir”: en esta parte se permite cargar un archivo de entrada siendo un Query cualquiera, la opción “Guardar” permite salvar un archivo que ya se encuentre en funcionamiento en curso, “Guardar Como” permite efectuar las respectivas validaciones de tal forma que efectúa y ejecuta una nueva versión de funcionamiento en ejecución.

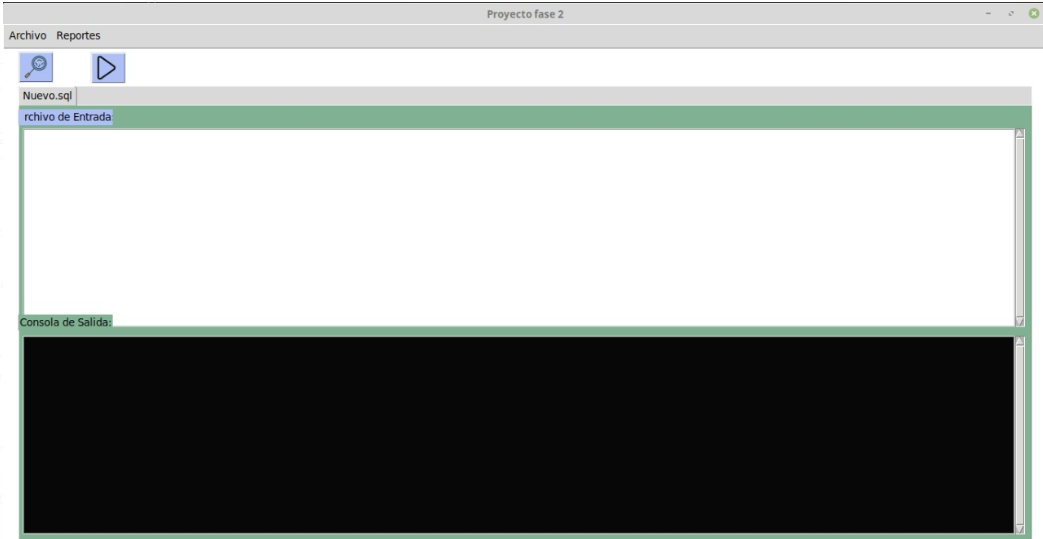


También podemos validar en la GUI distintos iconos los cuales se muestran la “lupa” en el cual genera código en “3d” según el Query que se inserte en tal caso también se cuenta con el “play” que permite efectuar las respectivas validaciones por cada uno entre las cuales cuenta con dos áreas entre las cuales vienen siendo el “Archivo de Entrada” y la “Consola de Salida” donde se puede apreciar cada uno según su funcionalidad.

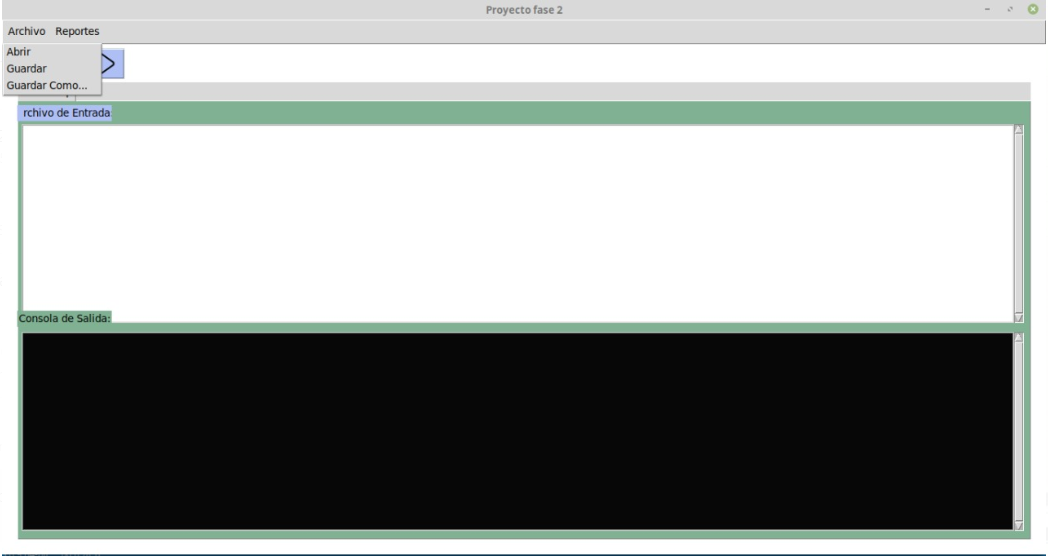
componentes de scr = Scrollbar() para validar la parte de vistas mediante “command=tex.yview”.

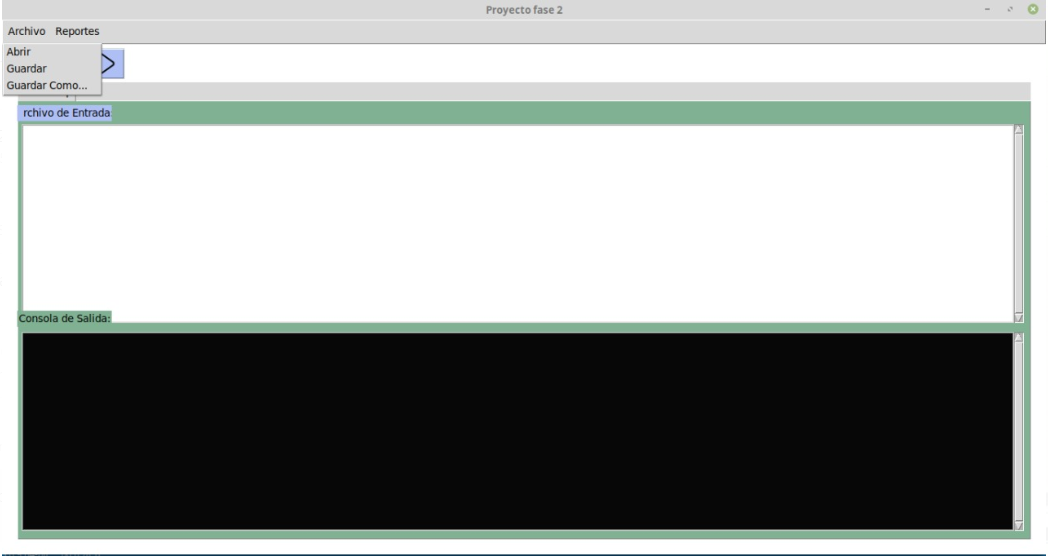
\*.- Estructura de Compilacion:

En esta parte se puede apreciar cada funcionamiento de su utilidad por dos áreas respectivas las cuales vienen siendo tanto la consola de analizador sintáctico, semántico así como el analizador léxico por código en tres direcciones.



Para la validación de comentarios Simples en el editor se tendrá las respectivas validaciones siendo que mediante el analizador léxico vaya concatenando uno por uno según se requiera ò asi mismo que reciba comentarios multiples mediante la expresión regular “r'//.\*\n'” la cual permite validar y llevar un contador por línea que se vaya agregando al editor de sql.

La estructura de la opción en “Archivo” se puede apreciar las opciones que permite efectuar mediante cada selección por opción que se tome.

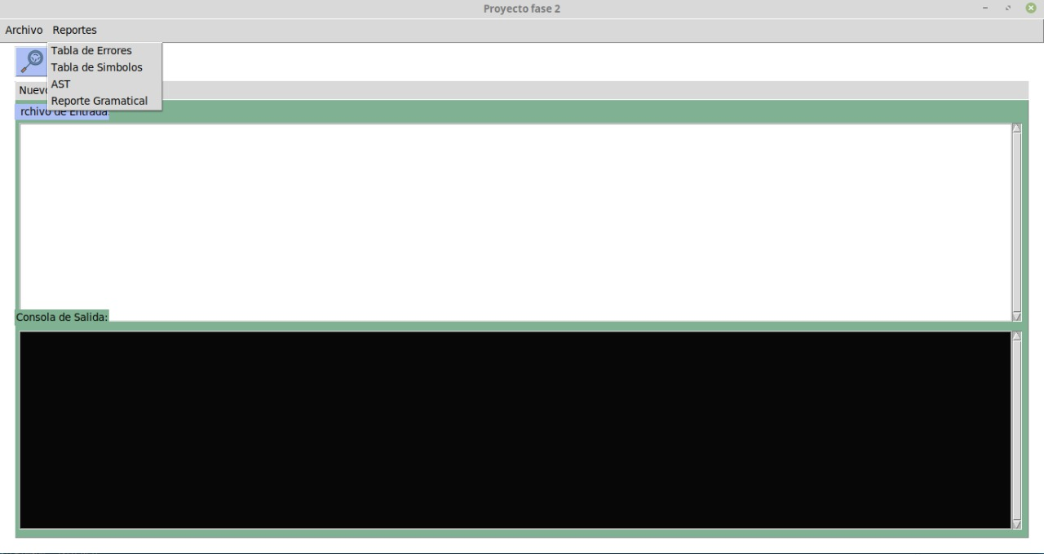
Se presenta en la consola si se obtiene una salida de un Query siendo el posible ejemplo “créate datatable base2;” en este caso se podría apreciar el código en 3 direcciones la implementación de cada salida del cual se genere.

En el apartado de reportes es el cual se encarga de generar el valor de símbolos por cada uno según se determine el error por el árbol.

Se valida que sean comentarios simples y mediante la siguiente expresión regular “r'//.\*\n'” se valida que sean comentarios de forma más extendida siendo multi-comentarios.

\*.- Líneas de validación:

Metodo que define: ”def t\_error(t)” si se agrega un valor que no sea léxico en la línea de lectura en el editor de texto que valida sentencias sql.

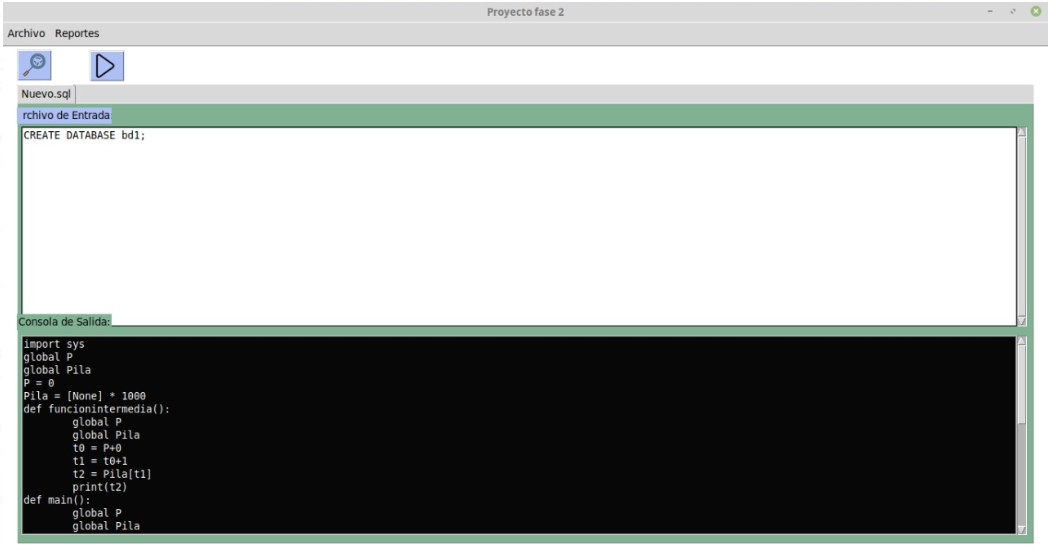
Mediante el parametro de instancia p Se agregan al método del constructor “"LEXICO","Error lexico”, “simbolo "+t.value[0]+" no valido”, “t.lexer.lineno,” y “find\_column(t)” siendo por cada parametro el primero como String, String, y por cada valor para la tabla de símbolos se agrega ”find\_column(t)” el cual valida si recibe los parámetros por cada uno.

Se agrega el valor por cada parametro en la conlumna del método “find\_column(token)” el cual permite validar mediante el testeo de “line\_start = texto.rfind('\n', 0, token.lexpos) + 1” y el valor permitente return tanto para el token como para el valor de lexpos por cada valor que se obtenga para posteriormente se construye mediante:

Así mismo el valor de construcción para el analizador se carga siendo en cada valor así como ply.lex en cada valor que se importa la librería en el valor de texto.

\*Tablas:

El método “reportetabla(self)” crea y obtiene instancia respecto a la tabla en la cual se obtiene dos parámetros siendo la entrada y el valor del ast.



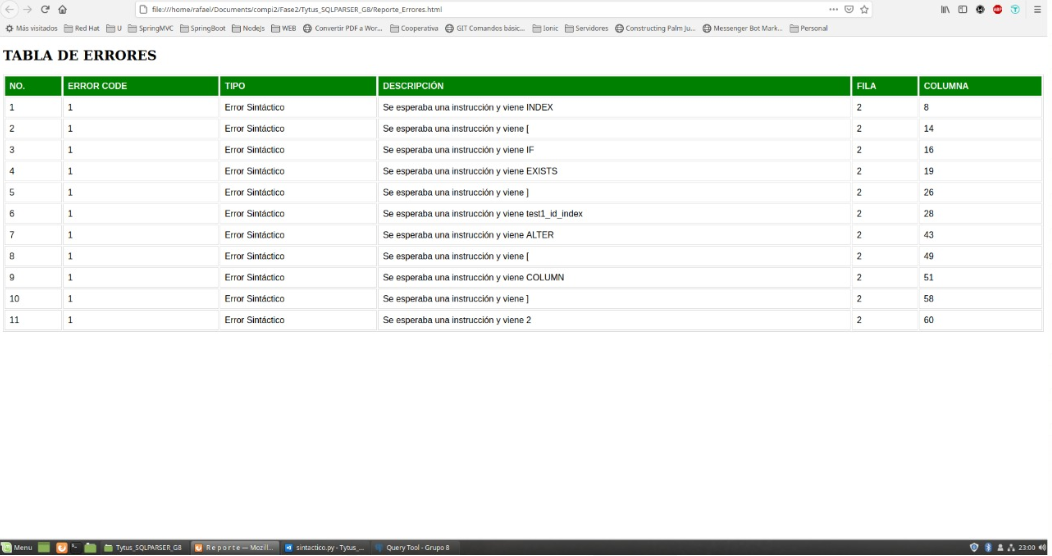
Al segundo método “reporterr(self)” se envía el parámetro el cual va a escribir a la instancia del método.

Por ultimo del archivo gui.py en la línea 20 del método “def reporteR(self)” se crea la setencia y asigna a “Sentencias.ReportedD()” para posteriormente enviar los dos objetos como parámetros quedando asi “SentenciasR.Mostrar(self.ent, self.ast)” en tal caso ir a mostrarlos por cada método en el árbol.

Para ambos método se utiliza el objeto de instancia siendo self el cual hace referencia a dicho parámetro del solicitado.

\*.- TABLA DE ERRORES

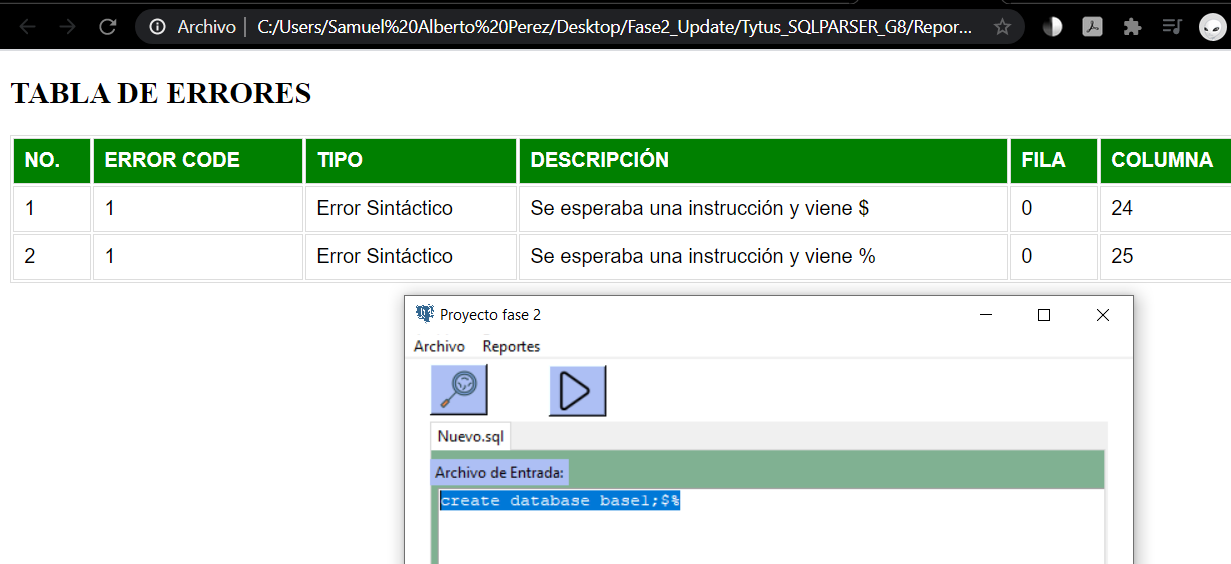
En cada método se agrega su instancia respectiva por reporteR(self), reporteerr(self) y reportetabla(self), por cada método también se tienen las instancias asigandas y parametrizadas.



Para este caso podemos apreciar los posibles errores que se puedan generar al momento de ejecutar un Query a la Tabla según sea el requerimiento necesario.

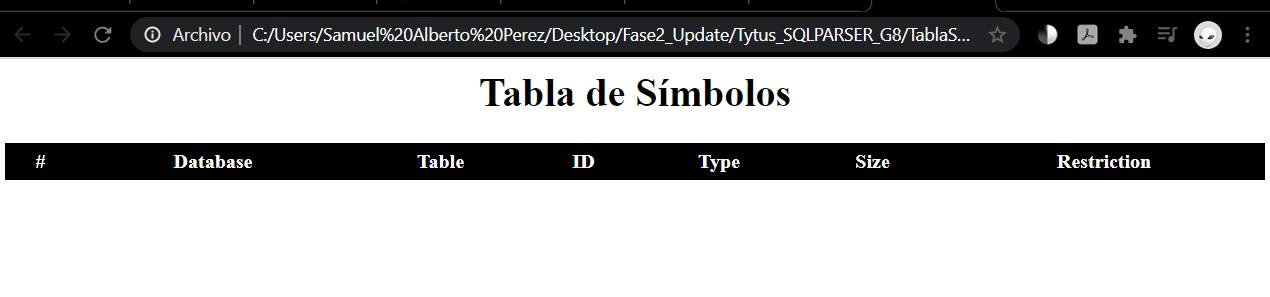
\*.- Sintaxis de Error:

Se puede apreciar la sintaxis de error al momento de crear una tabla o del cual se desee generar un Query del que se permite visualizar sus modificaciones o parte del código del cual se logre visualizar su funcionalidad.



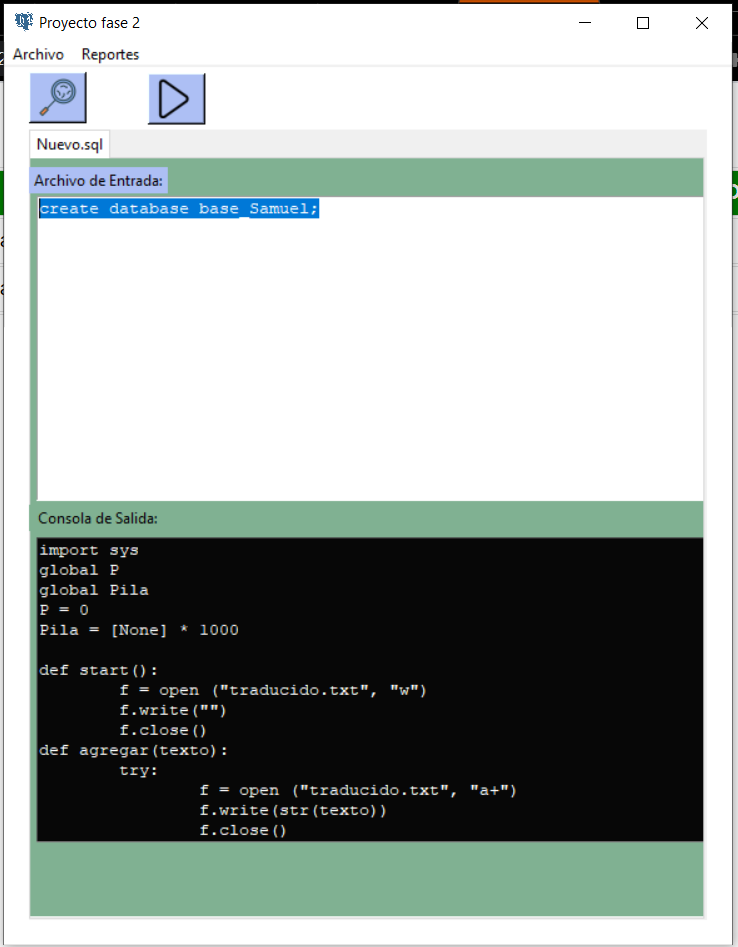
\*.- TABLA DE SIMBOLOS

En este caso se permite apreciar el valor tanto de la base de datos como la tabla, el identificador de la tabla, el tipo de tabla, el tamaño y asi como el valor de su restricción por cada valor que se permita.



\*.- Generación de 3D:

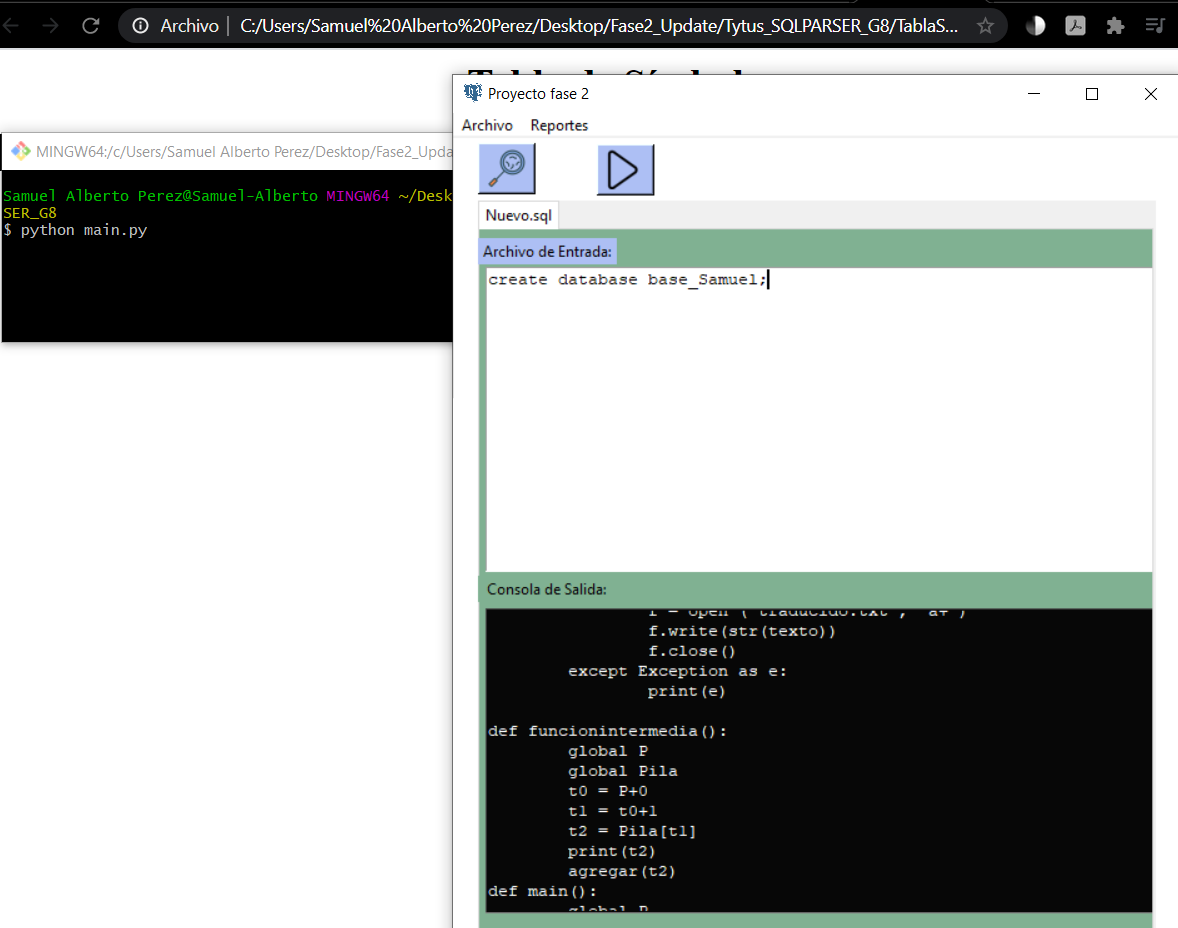
Al efectuar una Query con código tres direcciones el cual permita visualizar tanto el valor de la pila como cada asignación asi como su funcionamiento intermedio.



En este caso la consola permite apreciar cada funcionamiento requerido por valor y la generación del código en tres direcciones que se permita.

\*.- Funcion de Consola

En este apartado se permite apreciar desde el inicio del funcionamiento de la consola siendo el los comandos pertinentes para efectuarse y lograr correr cada valor indicado.



En este apartado se permite visualizar la funcionalidad de cada valor que permita generar la consola en la cual se logre efectuar su funcionalidad por cada valor que se permita ejecutar de tal forma se genera el código de tres direcciones.