**Benemerita Universidad Autonoma de Puebla**

Materia: Base de Datos para ingeniería.

Trabajo: Programación del Proyecto Final - (Between|Join) | Descriptor Archivos

Docente: Ambrosio Vázquez Alma Delia.

Integrantes del equipo: Ruiz Lozano Paulo Cesar 201727952

Contreras Aguilar Fernando 201735460

Memije Galván Enrique 201878802

Canalizo Mendoza Isaac André 201728726

Tlahuapa Coraza David 201736875

Periodo: Primavera 2020.

## ÍNDICE

[ÍNDICE 1](#_Toc33536975)

[RESUMEN 2](#_Toc33536976)

[INTRODUCCIÓN 3](#_Toc33536977)

[DESCRIPTOR DE ARCHIVOS 4](#_Toc33536978)

[IMPLEMENTACIÓN 6](#_Toc33536979)

[PROGRAMACIÓN 7](#_Toc33536980)

[INTERFAZ 16](#_Toc33536981)

[REFERENCIAS 21](#_Toc33536982)

## RESUMEN

El descriptor de archivos es una estructura de datos que describe o documenta las características (posición, tipo, valor inicial, etc.) de la información dentro de un archivo. Cada archivo que contiene información posee un descriptor de archivo único. Los descriptores de archivos para una colección de archivos relacionados se almacenan juntos en un archivo denominado descriptores.

## INTRODUCCIÓN

Al aumentar los costos de la programación, es importante encontrar formas de aumentar la productividad de la programación. Una manera de hacerlo es usar lenguajes o anotaciones más apropiados para la tarea en cuestión, con más poder expresivo, menos propensos a errores de los programadores, y donde no se especifiquen o programen acciones entendidas o naturales.

En este documento se discuten este tipo de anotaciones, algunas junto con las herramientas para usarlas. Todos ellos emplean el “descriptor de archivos”, una recopilación de datos sobre un archivo en particular, en este caso con un archivo de texto de “empleados”, el cual contiene varios atributos y con base a ellos vamos a realizar, paso a paso, lo que hace un manejador de datos, consultas y una consulta con condición de Between, la cual nos va a pedir la consulta de empleados, dado el nombre y el salario, solamente de los que cuenten con un salario entre los 10.000 y los 15.000.

Como segunda consulta se implementó con condición de Join (Equijunta), la cual nos va a pedir la consulta de empleados y departamentos, solamente cumpliendo la condición que se junten los que tengan el mismo número de departamento.

## DESCRIPTOR DE ARCHIVOS

Consideremos archivos donde todos los registros son de la misma longitud y donde cada uno contiene información organizada de la misma manera. Esta información es sobre un objeto en particular y un archivo es una colección (sobre información).

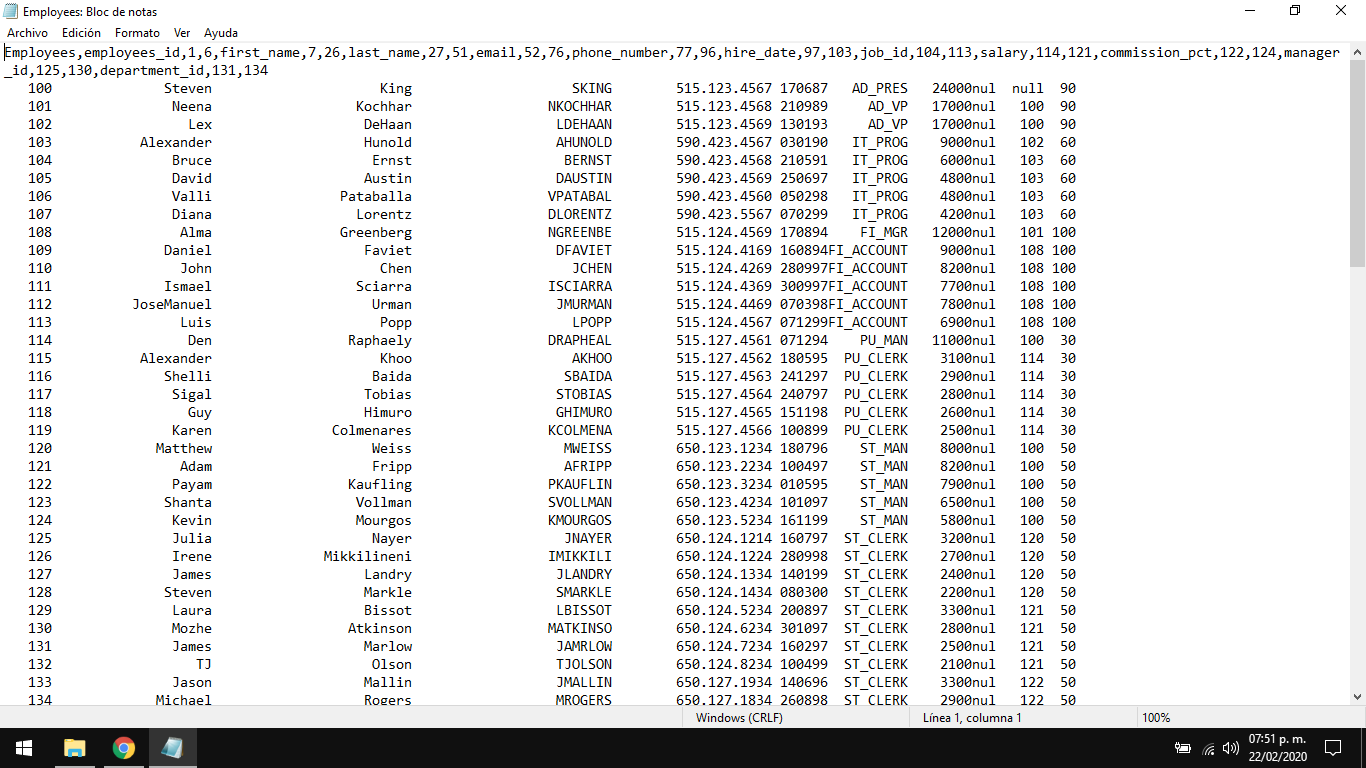
Así como el archivo de texto EMPLEADOS (img. 1), contiene un total de 106 registros, 106 empleados (un empleado es un objeto del archivo EMPLEADOS).

Cada empleado contiene un empleado\_id (1-6 bytes), nombre (7-26 bytes), etc.

Todos estos registros son contenidos en un descriptor de archivos, cada uno de estos comienzan con el nombre del archivo que los describe (en este caso empleados), seguidamente de algo de información acerca de dicho registro.

El tamaño del archivo se basa en sus registros, los tamaños de los registros se basan en sus bytes, etc.

Cada descriptor de archivos contiene el nombre del campo que describe, la posición inicial del objeto (tamaño en bytes), su tipo, y rango de su tamaño (img.1).



img 1

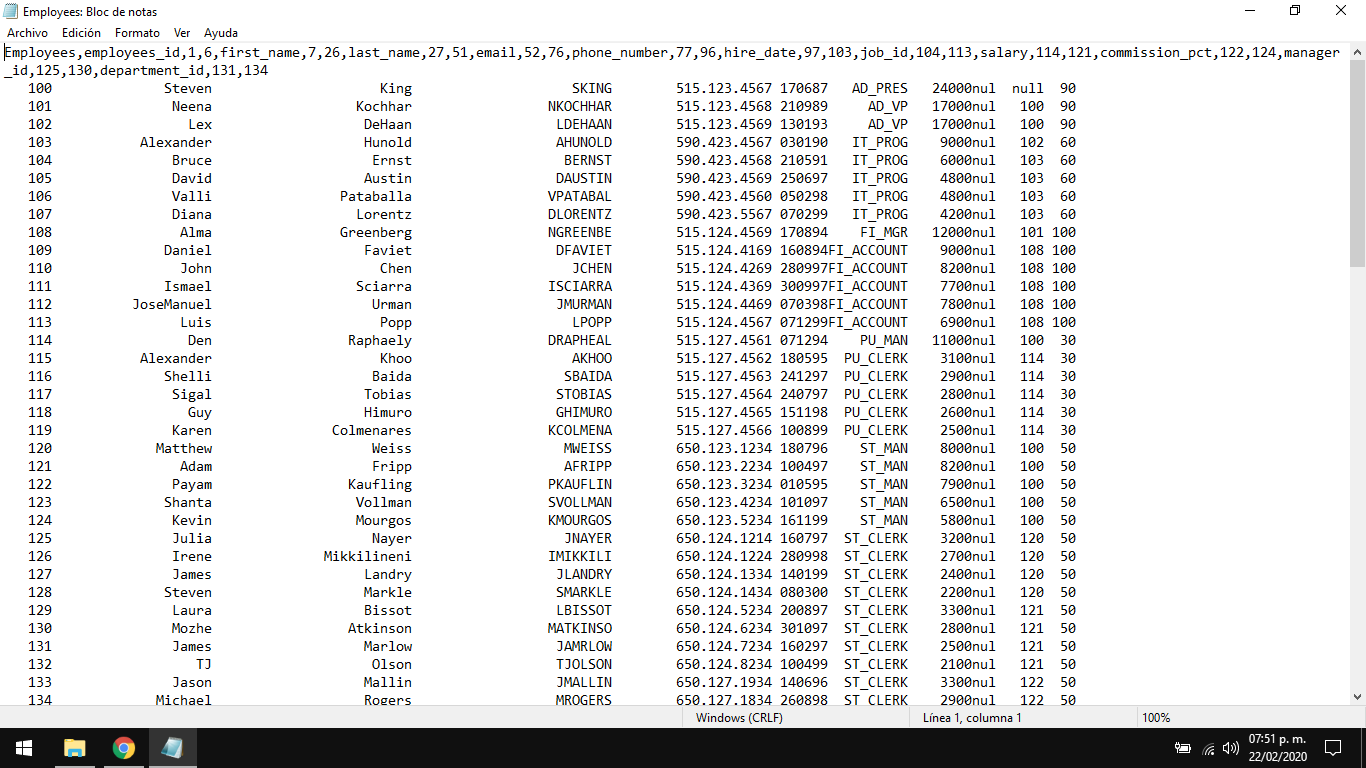
Un descriptor de campo más sofisticado puede contener:

* Descripción de los datos (en el objeto) que no están fijos en su posición.
* Numero de ocurrencias de cada campo, para campos multivaluados.
* Condiciones en las que se pueden eliminar un registro particular; por ejemplo, que ciertos empleados no se pueden eliminar a menos que Salario = 0.

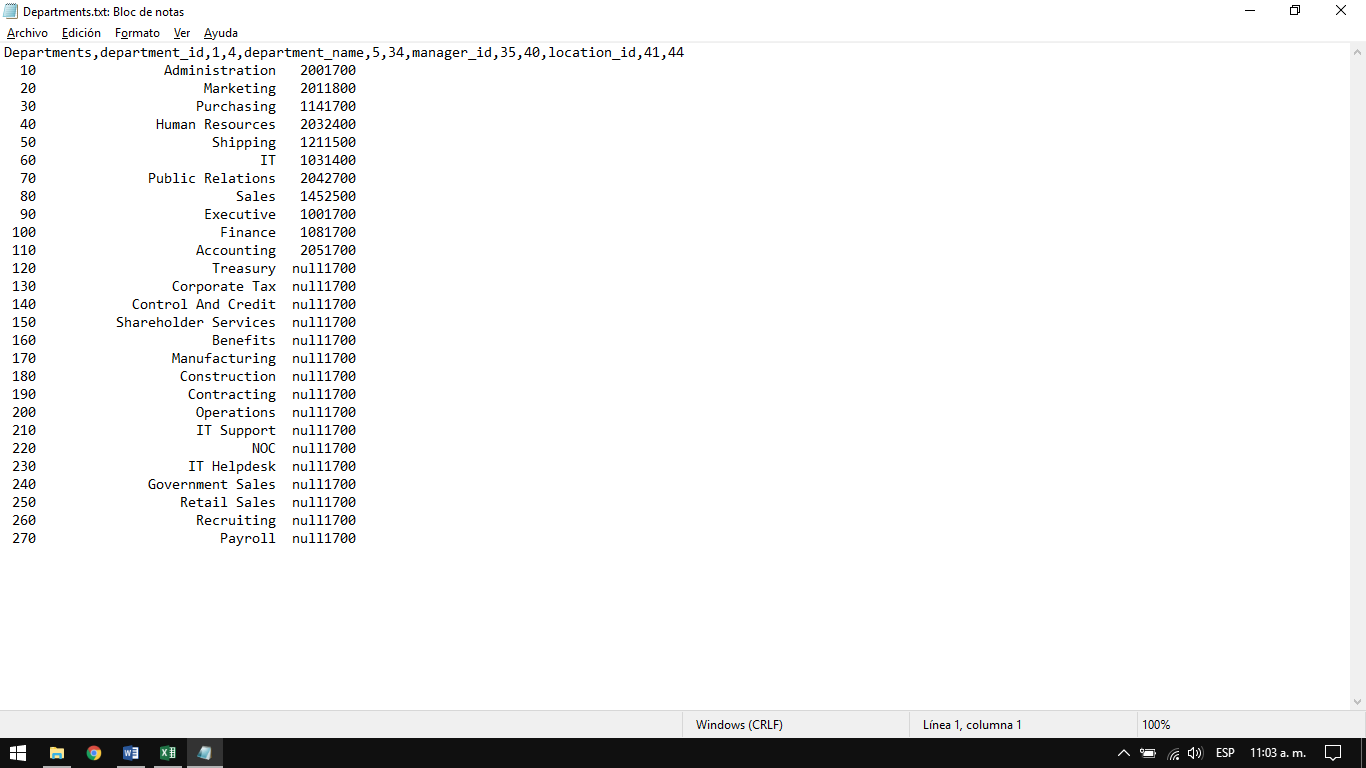
### IMPLEMENTACIÓN

Una vez sabiendo cómo trabaja un manejador de datos, con la selección, proyección y resultado; decidimos implementar el lenguaje Python para realizar dicho trabajo, ya que se nos facilitó más el uso de este, con este trabajo en particular por el uso de archivos.

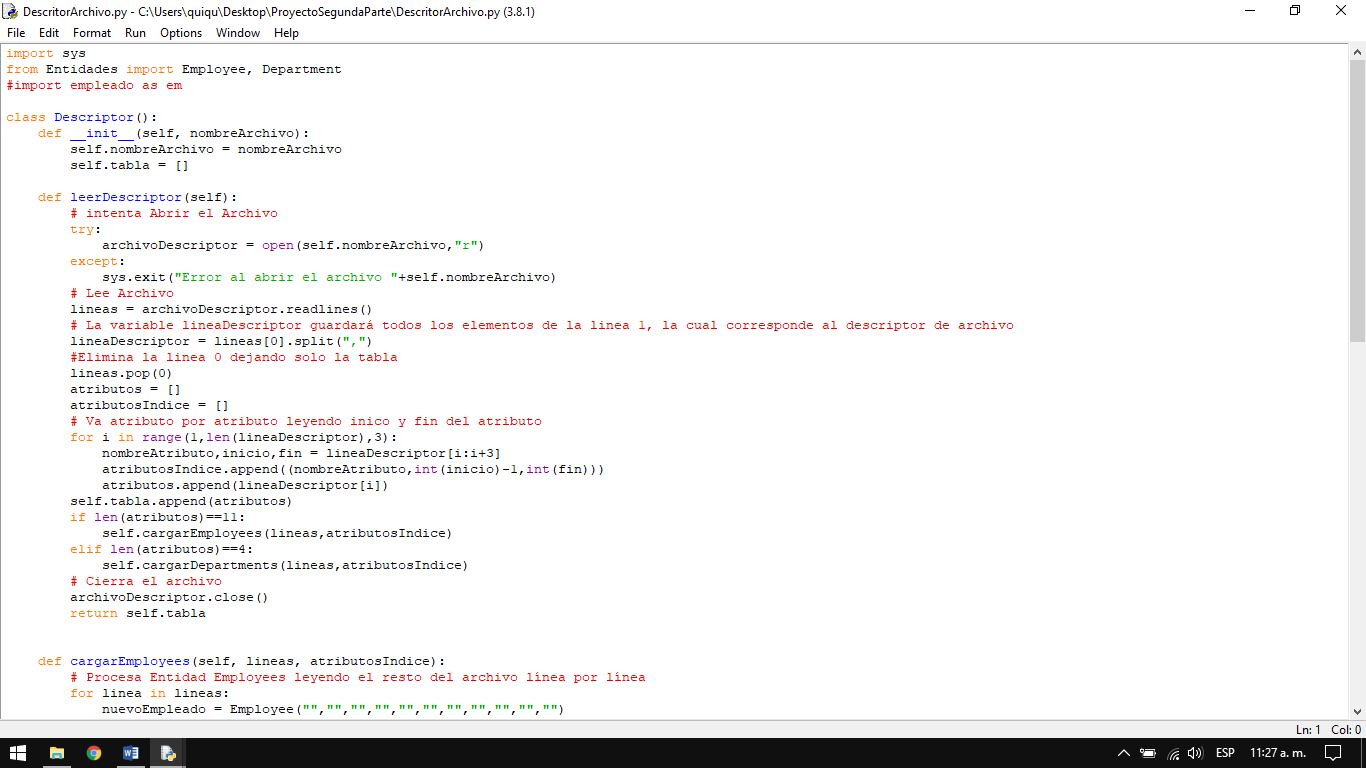
## PROGRAMACIÓN

Del documento de texto de empleados:

Del documento de texto de departamentos:



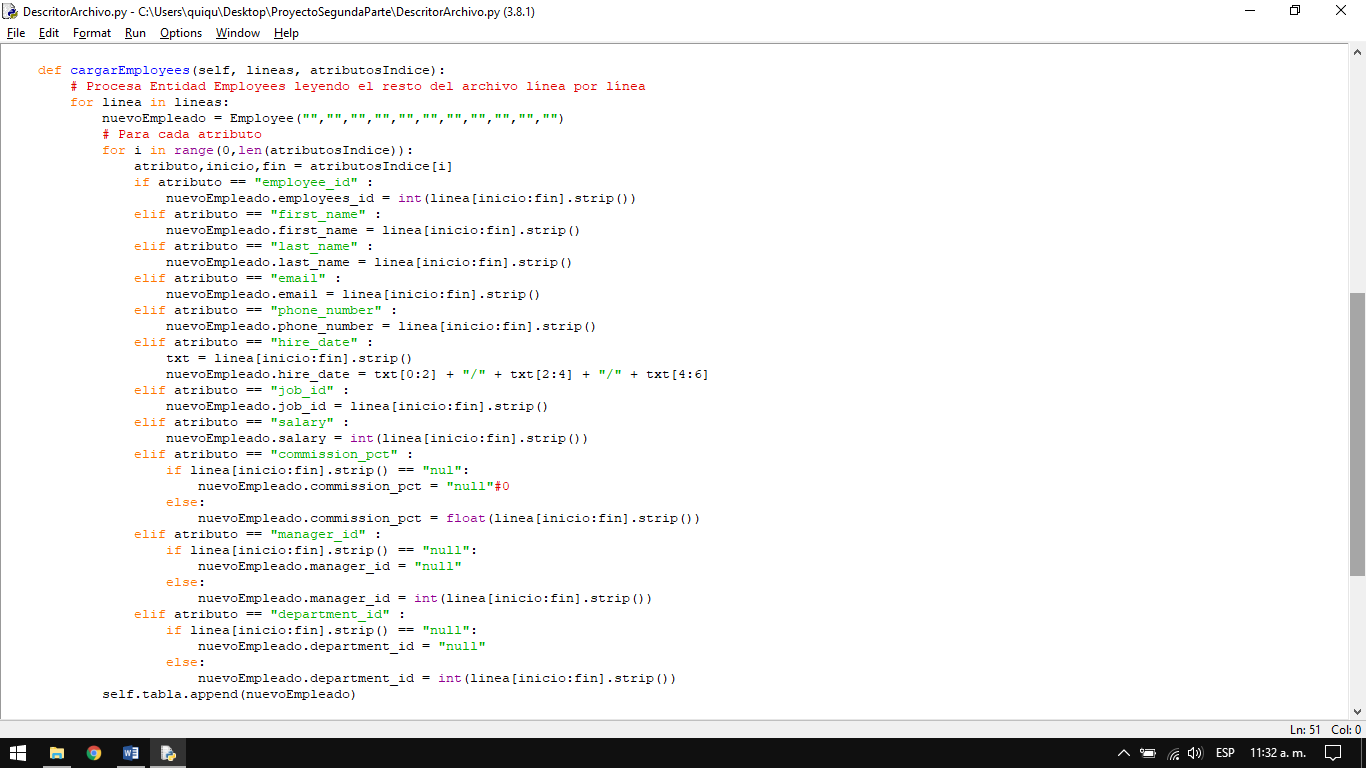
Debíamos de tomar los datos de los registros en él, para ello debíamos de crear una clase más donde se especializará a recopilar los datos dentro del documento de texto para después de ello, proceder a realizar las consultas solicitadas.



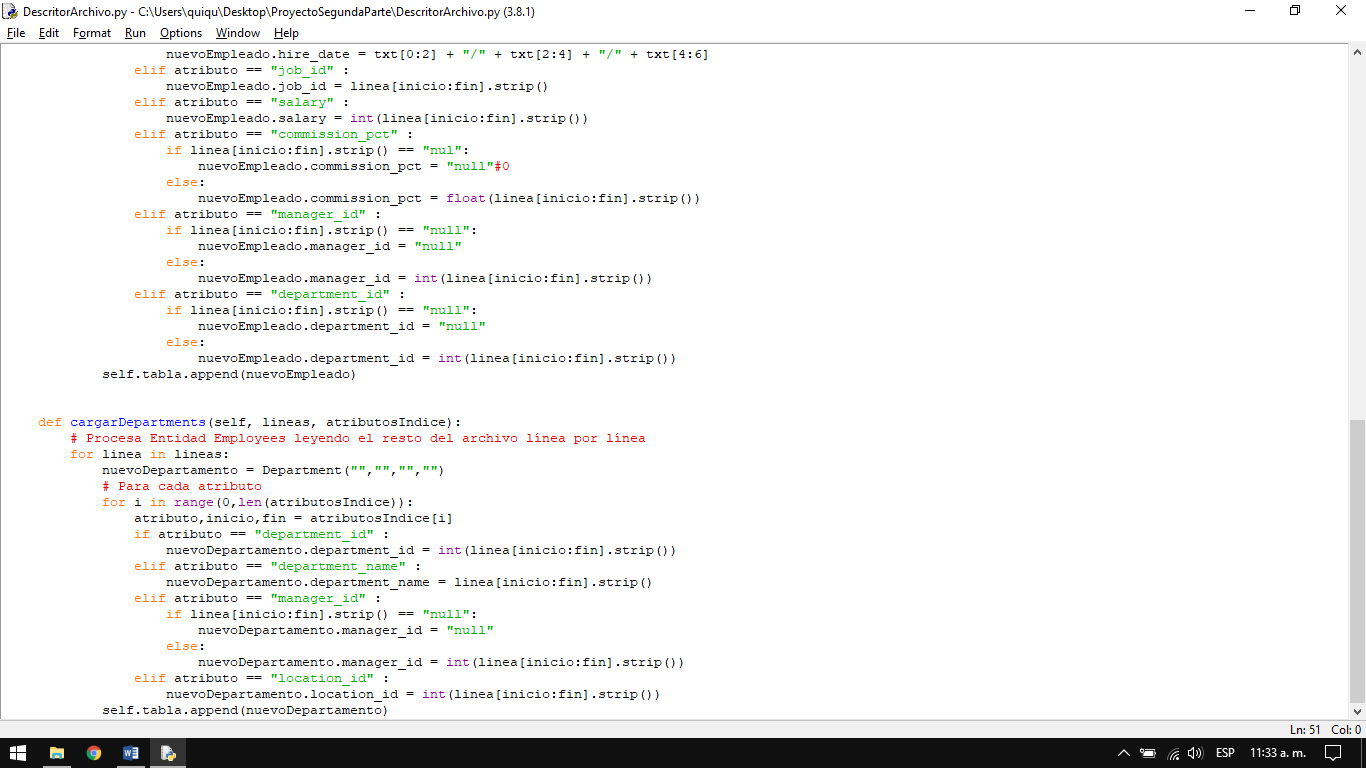
Como se puede observar indicamos el open(self.nombreArchivo, “r”) para decirle que el documento de texto que vayamos a abrir sea nada más en lectura.

Debemos de indicar que leemos el archivo línea por línea, indicamos una longitud para obtener los atributos que en este caso nos serviran para identificar las columnas de información en el documento.

Tanto del documento de empleados:

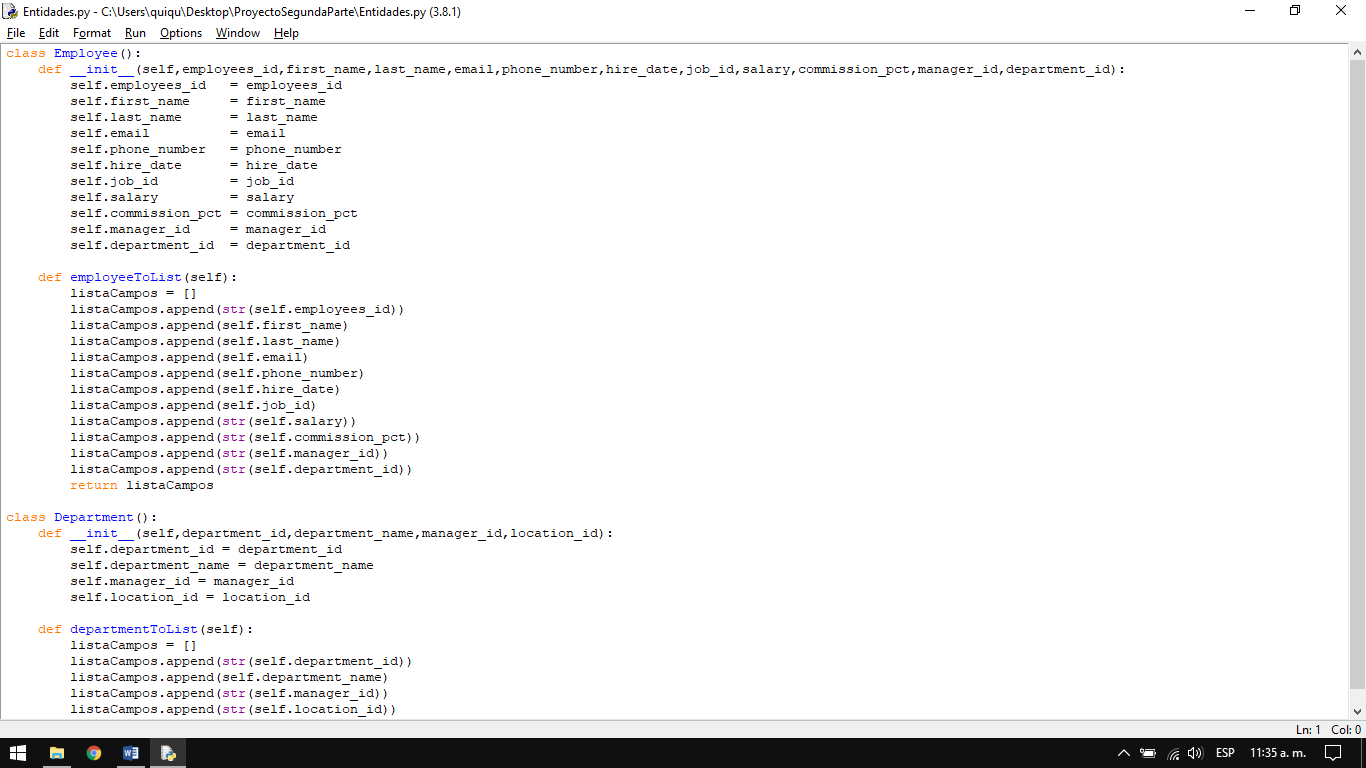


Como del documento departamentos:

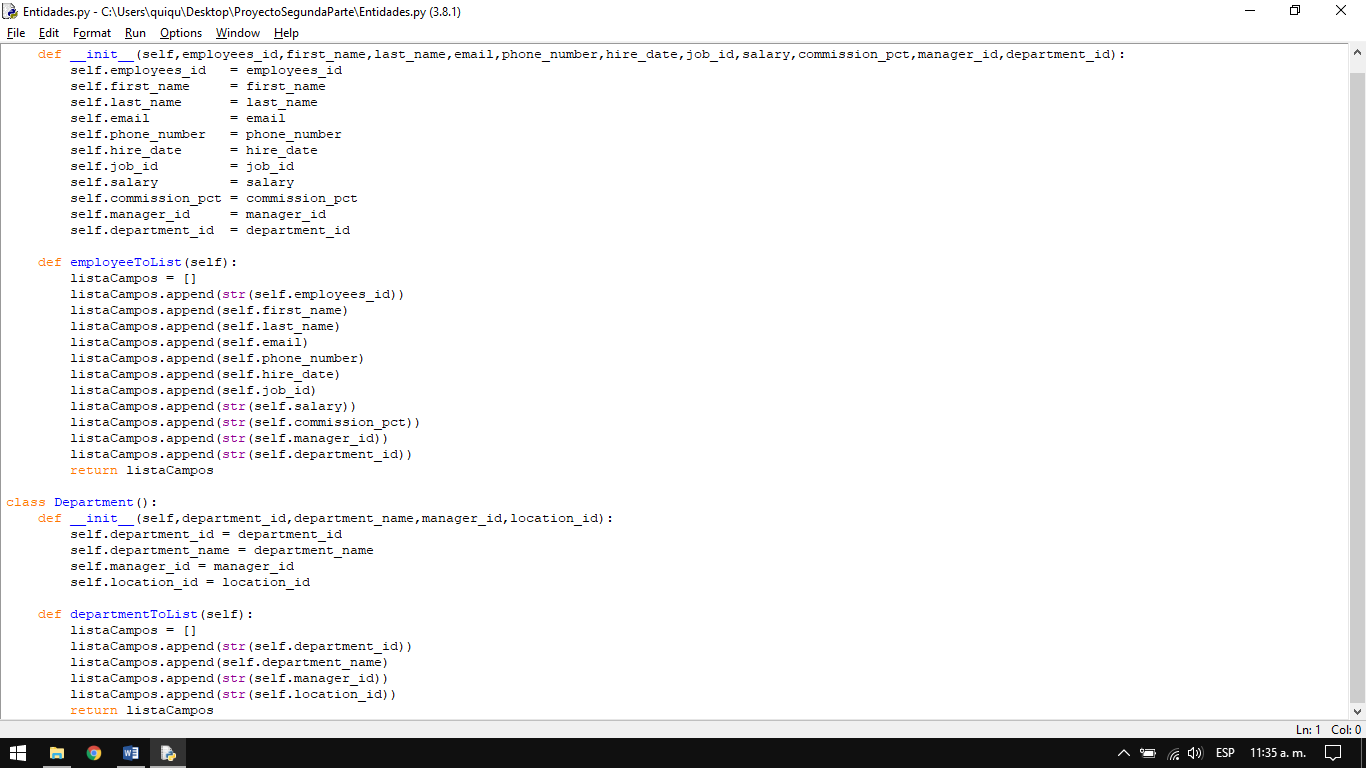


Ya solo queda identificar y agregar los atributos como variables en el programa, para su uso.

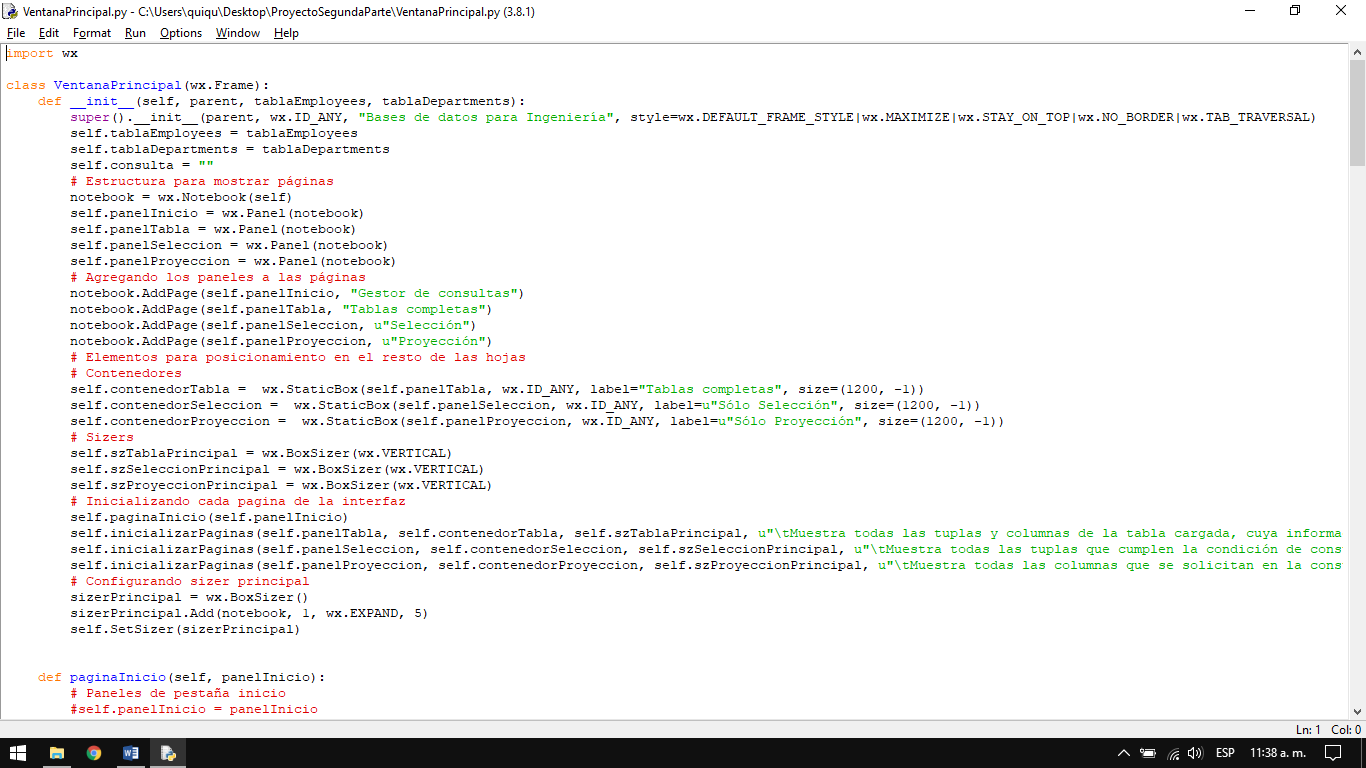
Tanto de los empleados:



Como de los departamentos:



Ahora creamos una clase generadora de consultas, la cual era para crear nuestra interfaz gráfica, para ello teníamos que importar la librera wx, las cuales sirven para poder crear pantallas (frames (interfaces graficas)), configurándola con nombre de ventana, tamaño predeterminado, al igual que su mismo botón para que el usuario interactuará en ella.



Ya nada más queda lo que es programas las consultas:

Tabla completa del Between:

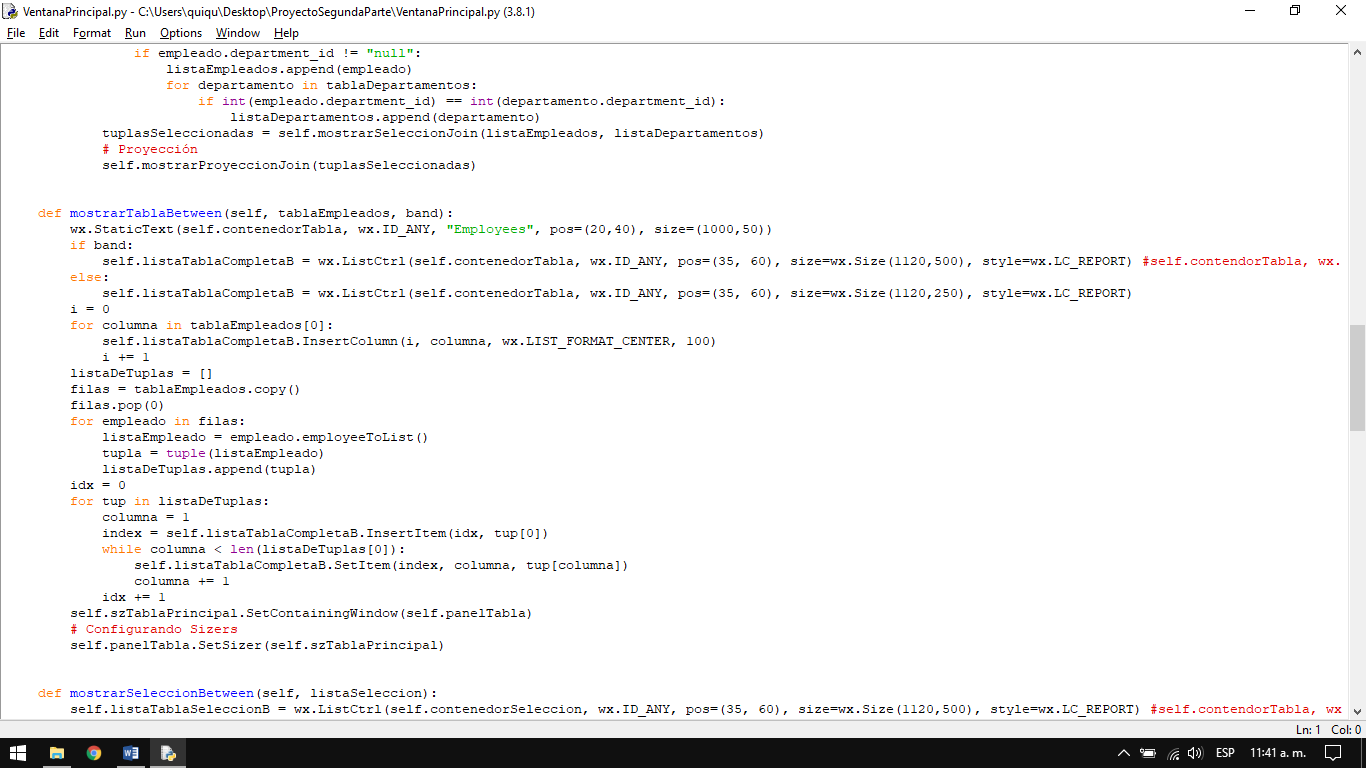


Tabla selección del Between:



Tabla proyección del Between:



Tabla completa de Join(Equijunta):

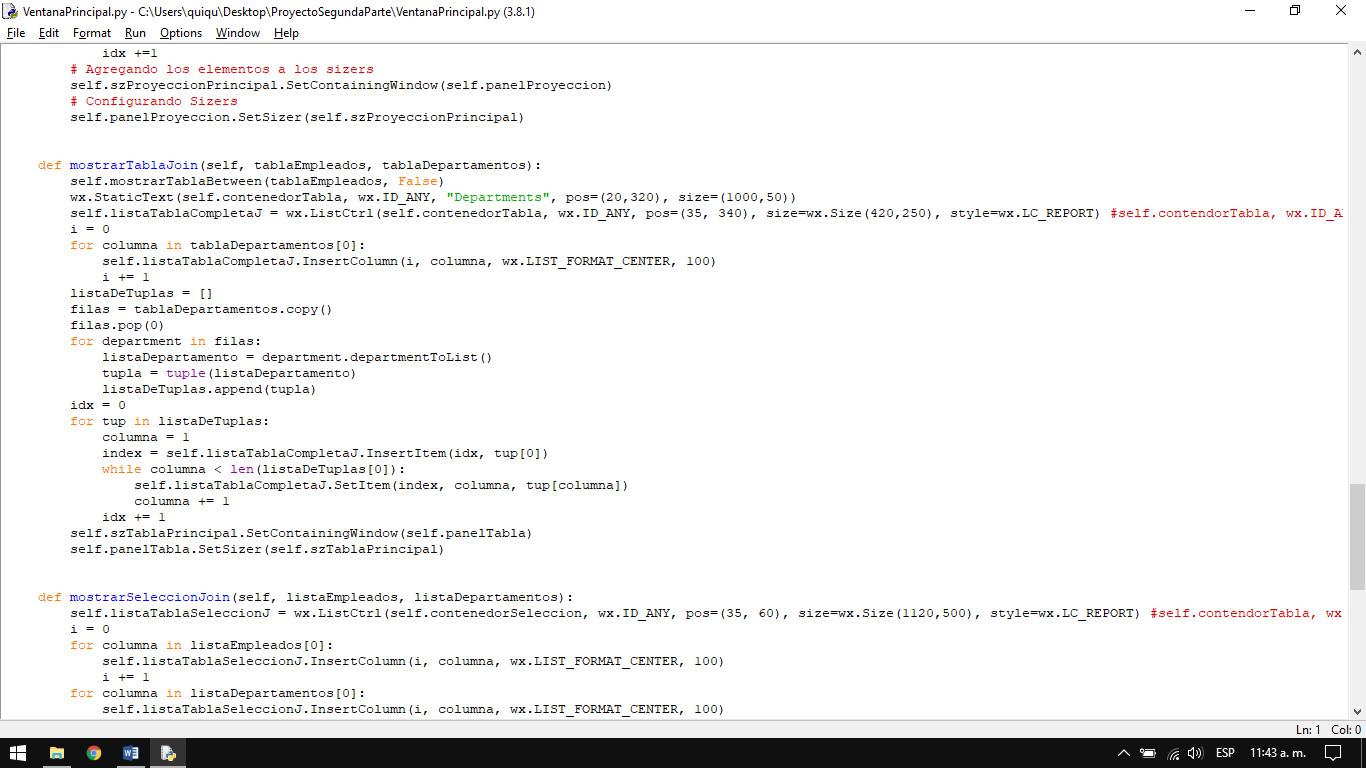


Tabla selección de Join(Equijunta):

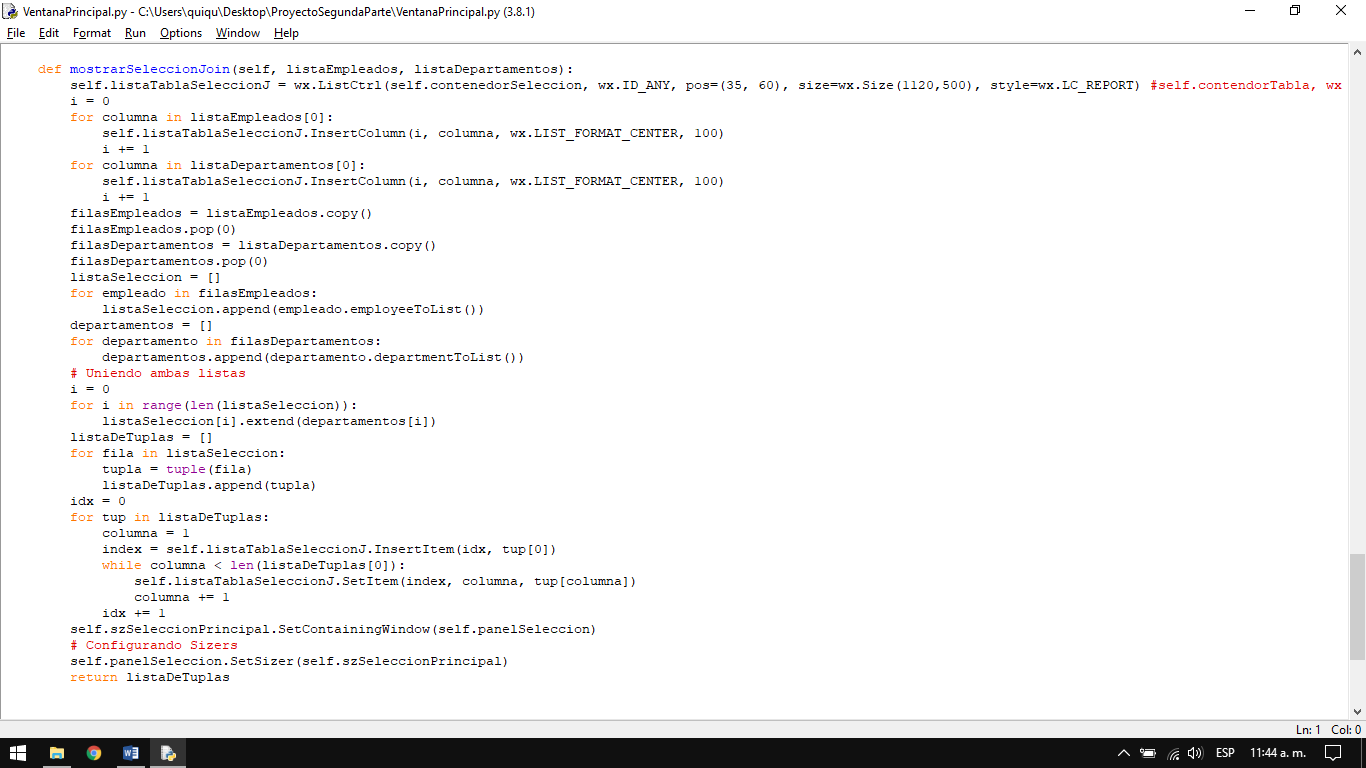
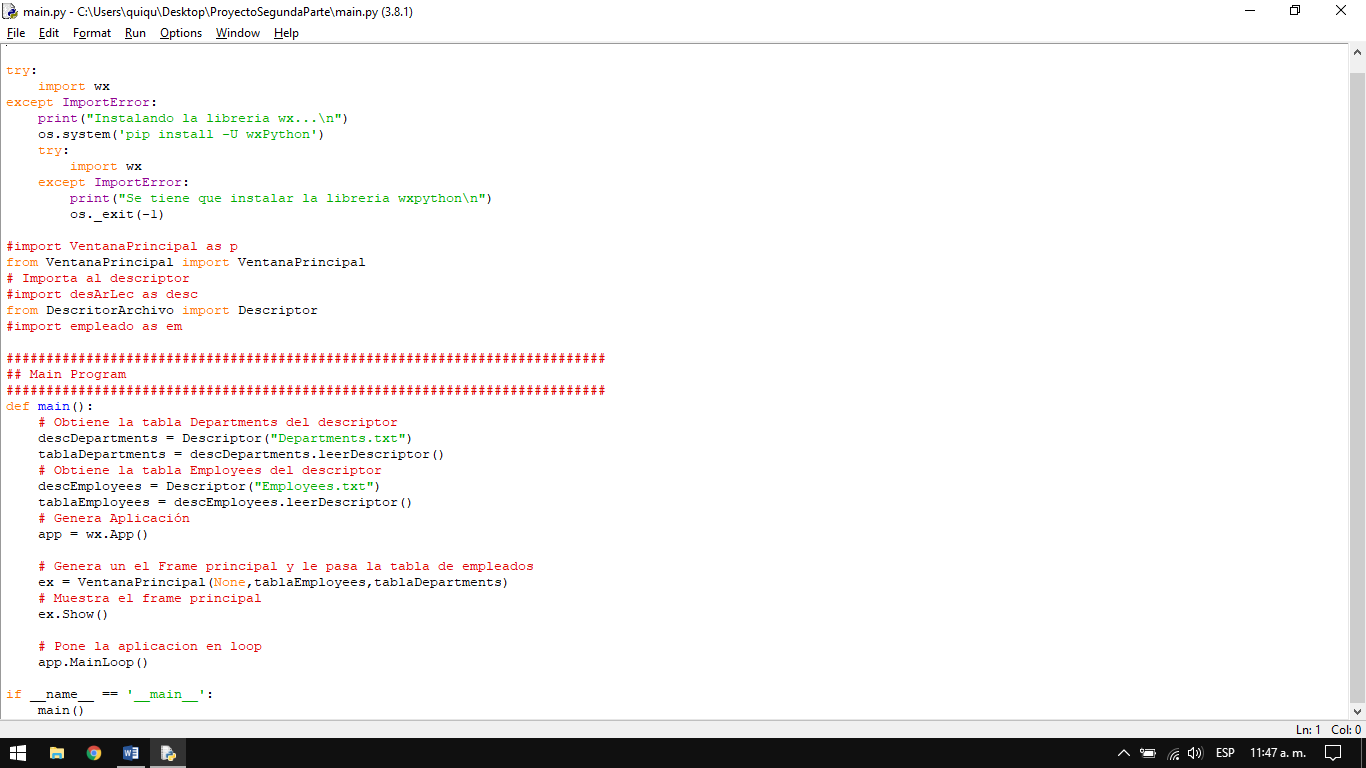


Tabla proyección de Join(Equijunta):

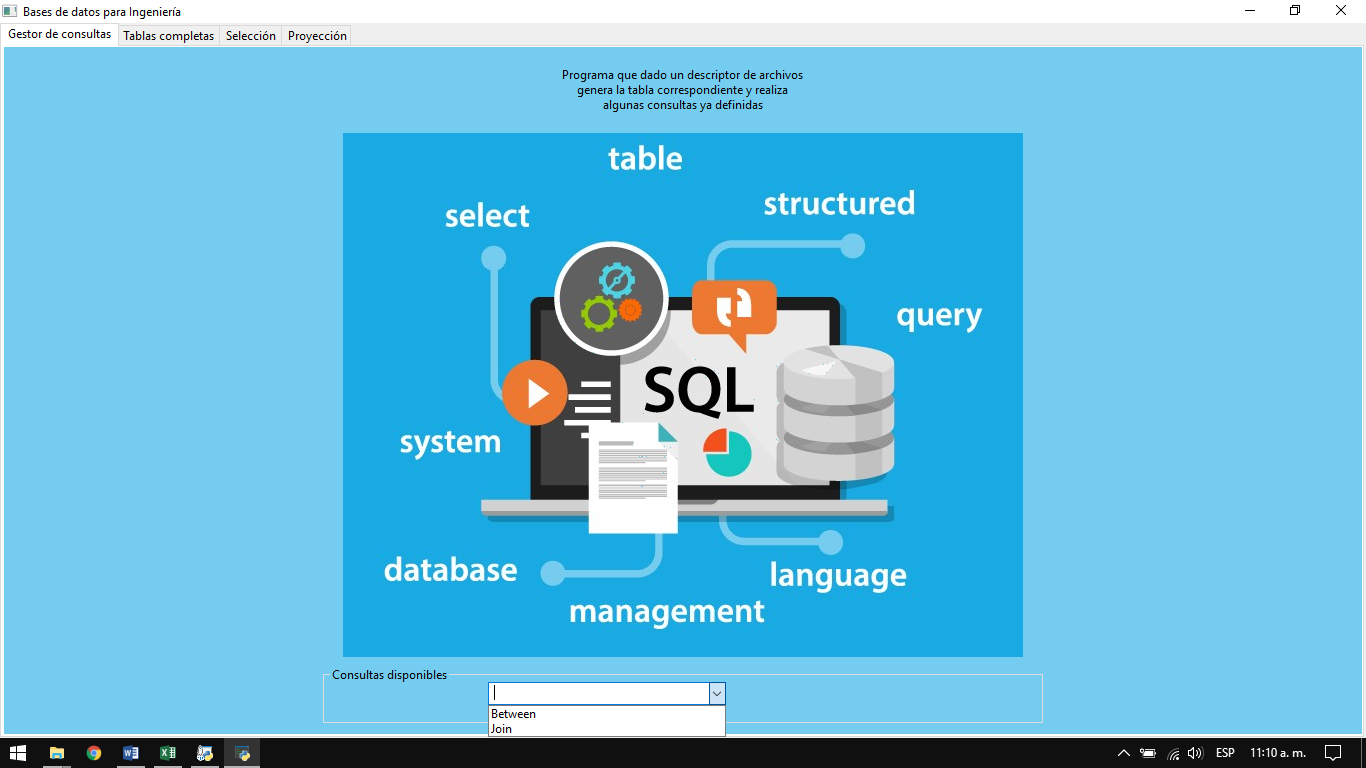


Como ultimo la clase principal o la main class, para que dé inicio a las demás clases creadas y pueda funcionar el programa.



## INTERFAZ

Primero que nada, realizamos una ventana principal, en la cual el usuario pudiera interactuar directamente con el programa como se puede ver en la siguiente imagen:



Y cuando el usuario seleccione la opcion de consulta, tiene para escoger el Between o Join, al seleccionar la consulta que desea mostrar da clin en aceptar.



En este caso la de Between nos mostrará:

* Tabla empleados, con toda la tabla completa.
* Tabla selección, con toda la tabla completa de los empleados cuyo salario sea entre 10.000 y 15.000.
* Tabla proyección, con unicamente los campos de nombre y salario, de los empleados cuyo salario sea entre 10.000 y 15.000.

Para verlas una vez generadas, nos vamos a la parte de arriba en la interfaz y seleccionamos la tabla que queremos mostrar:

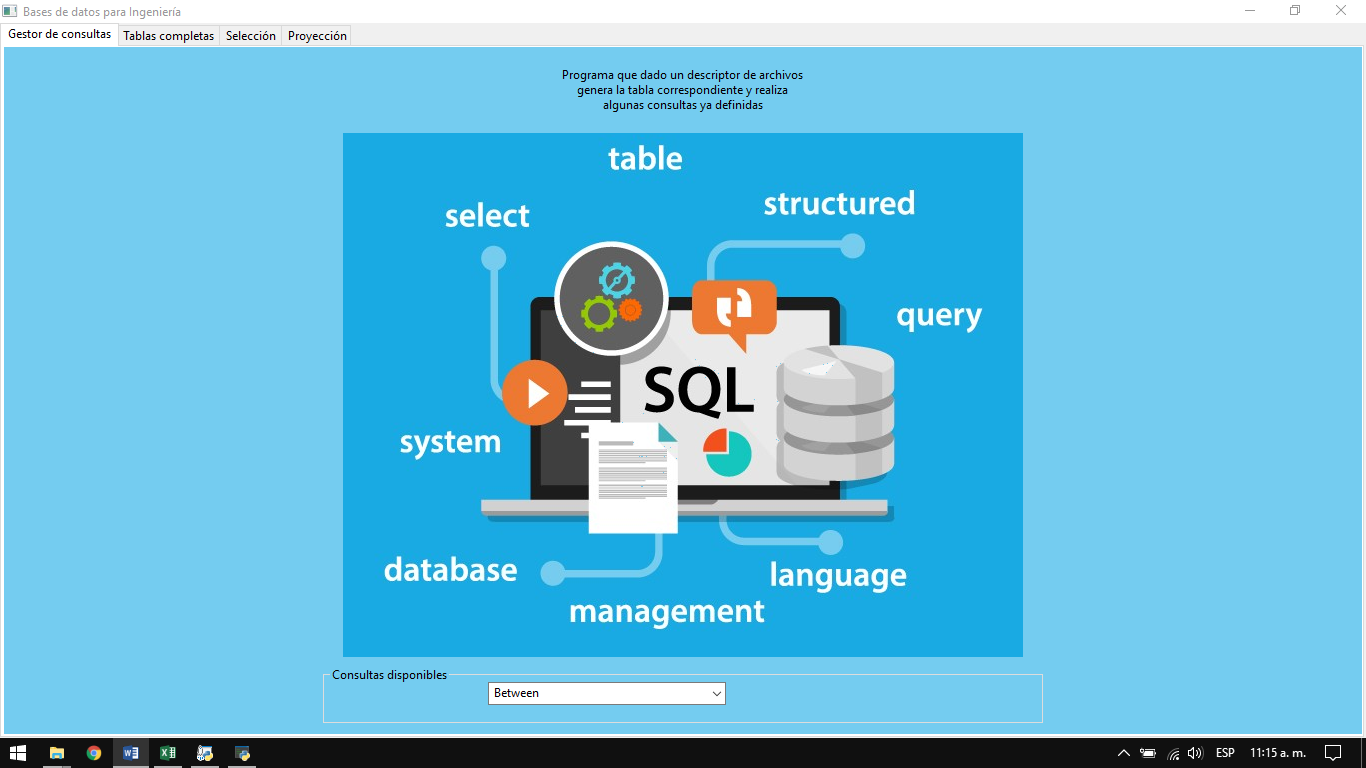


Tabla completa de Between:

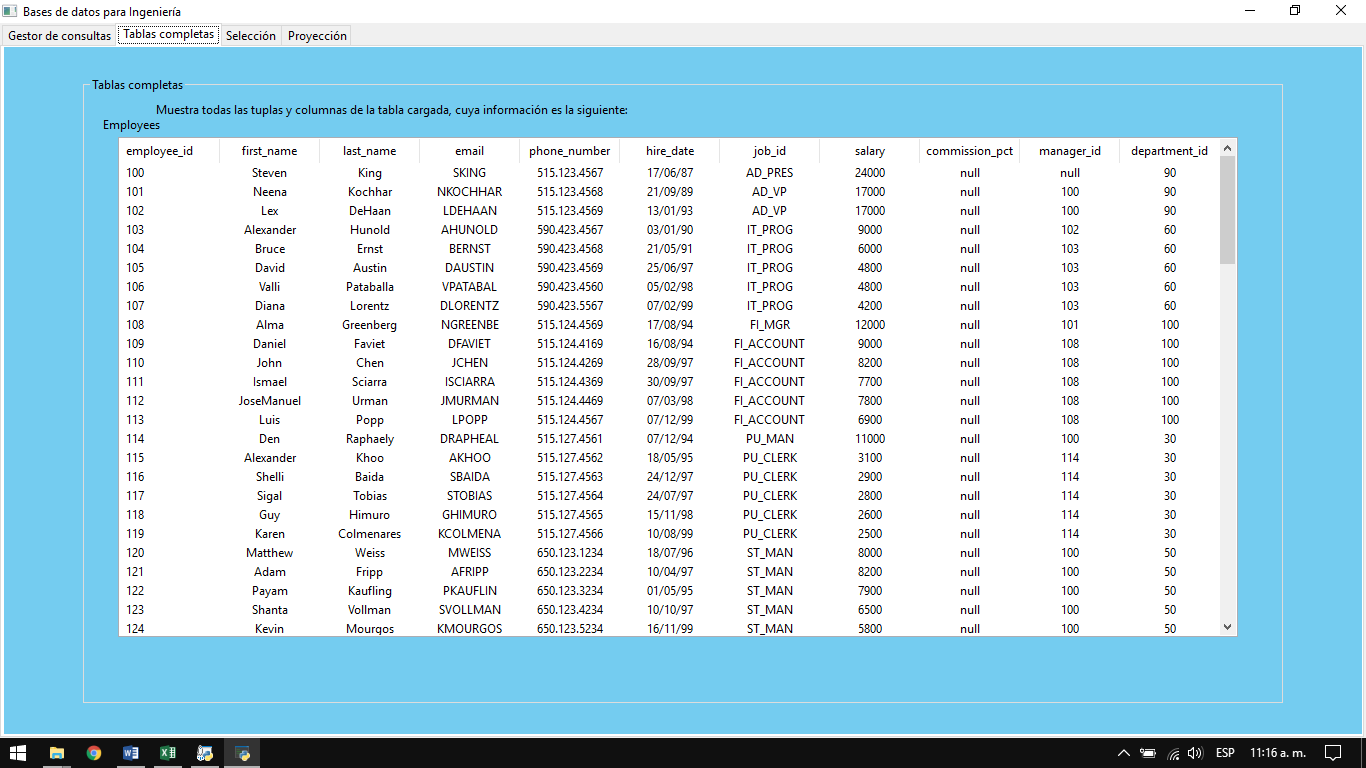


Tabla selección de Between:

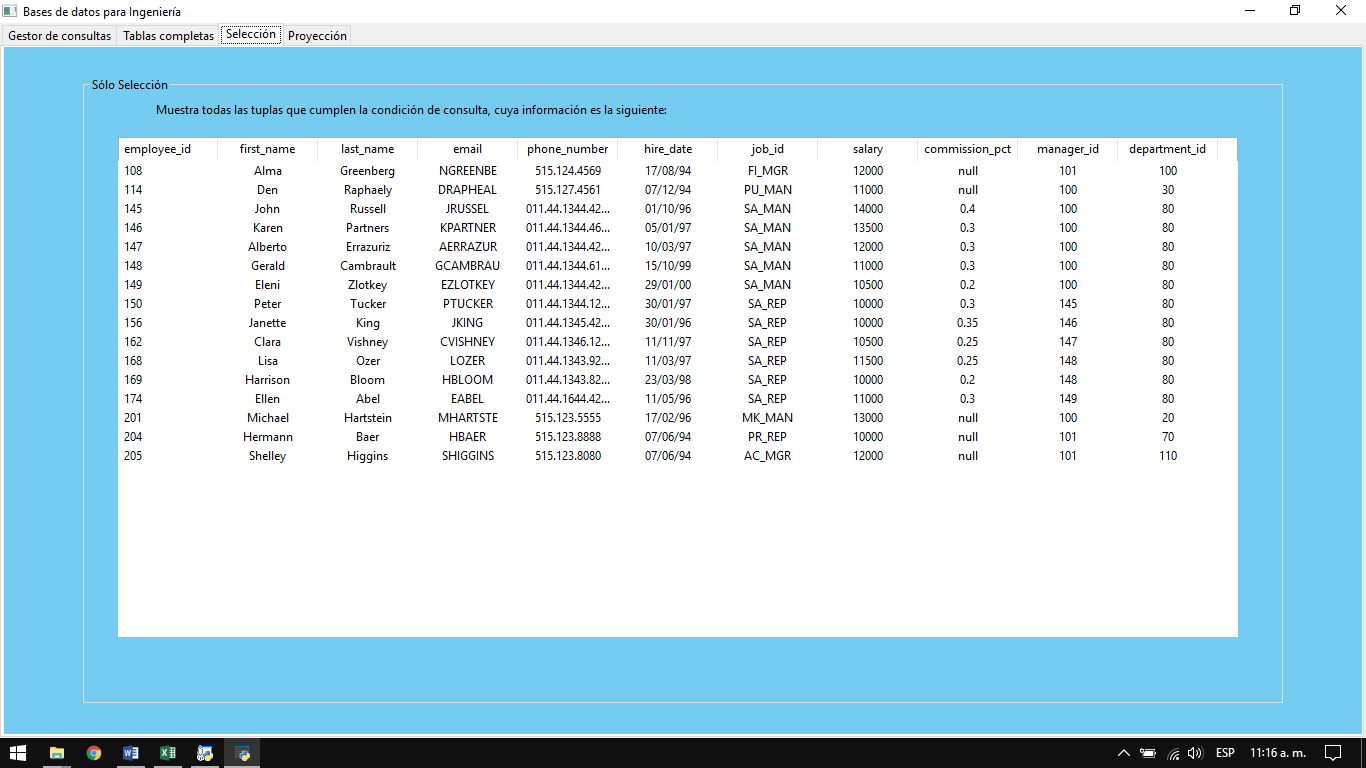
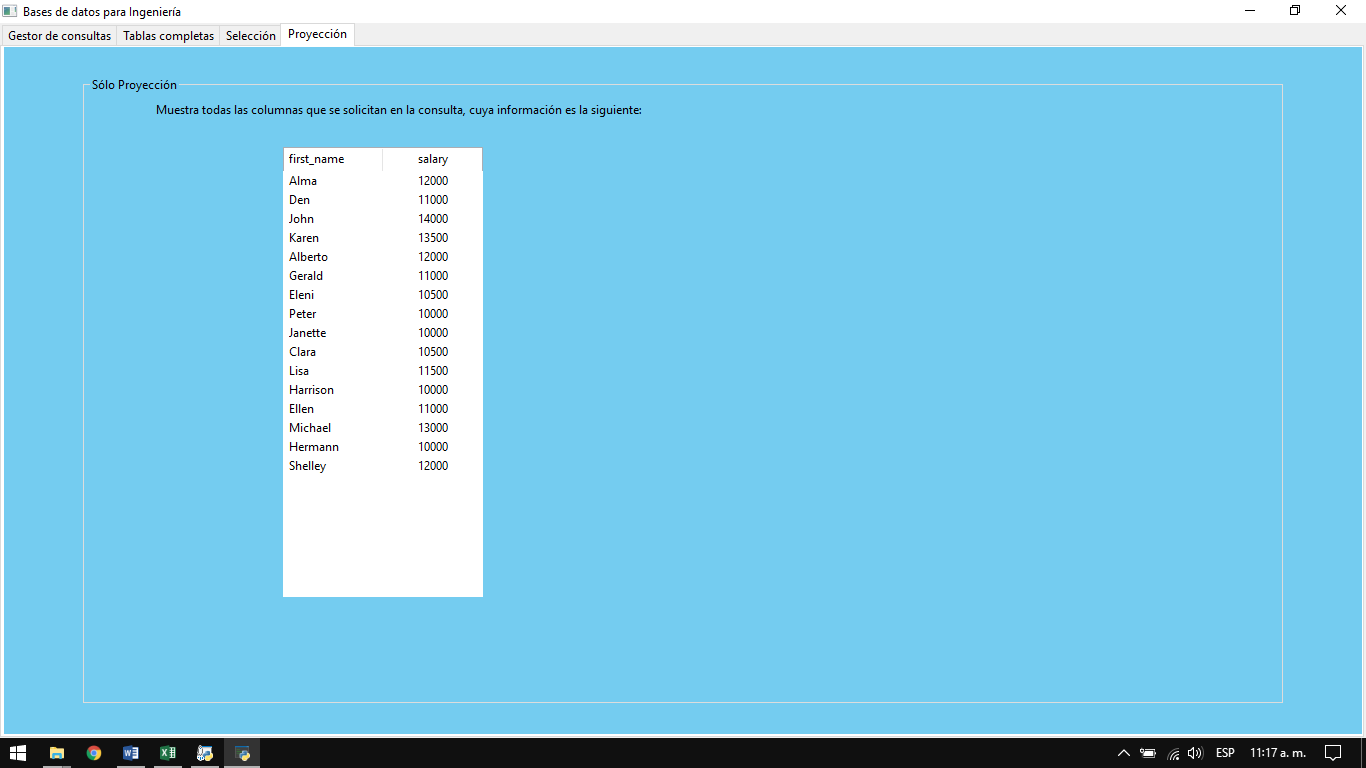
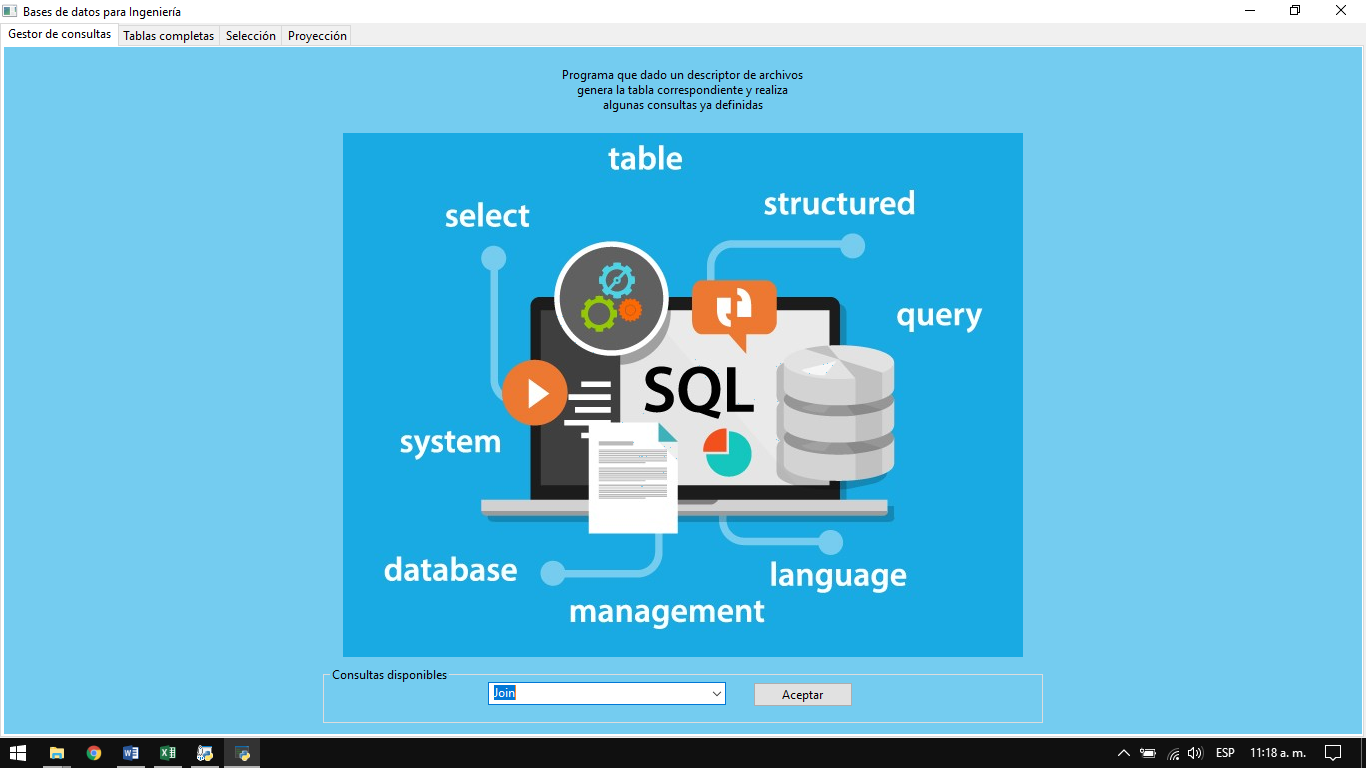


Tabla proyección de Between:



Ahora bien, en el caso de seleccionar Join (Equijunta):



Nos mostrará:

* Tabla empleados y departamentos, con todas las tablas completas.
* Tabla selección, con toda la tabla completa de los empleados que compartan un mismo ID de departamento.
* Tabla proyección, con unicamente los campos de ID empleado, nombre empleado, ID departamento de empleado y ID departamento de departamento, ID localización de departamento.

Tabla completa del Join (Equijunta):

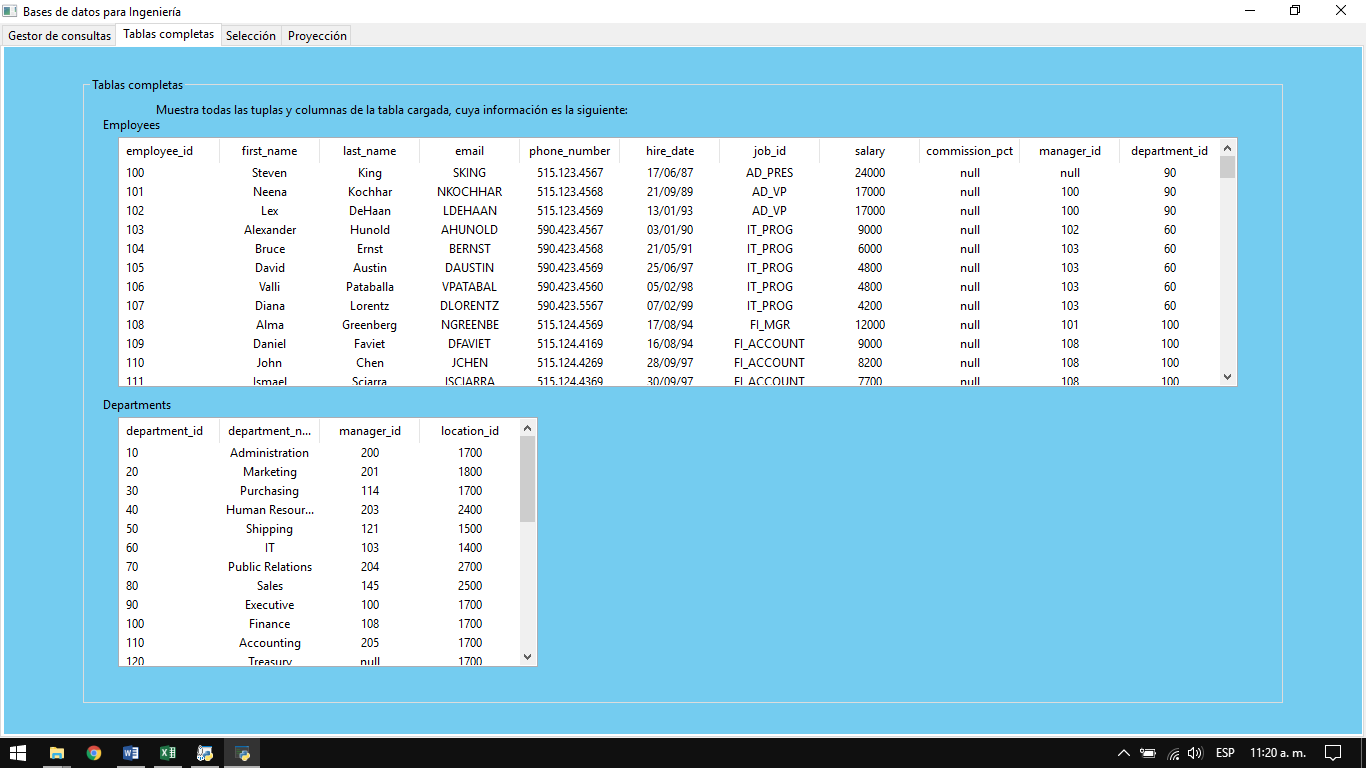


Tabla selección del Join (Equijunta):

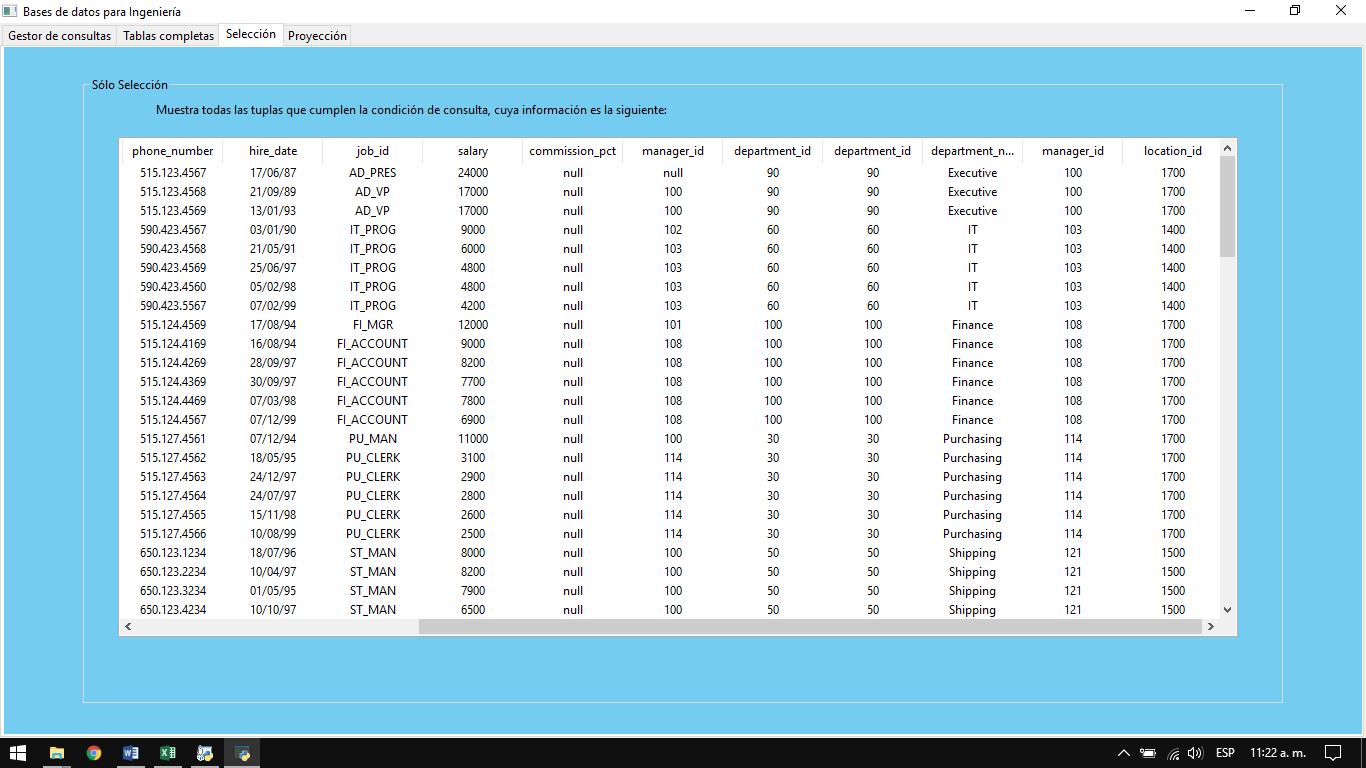
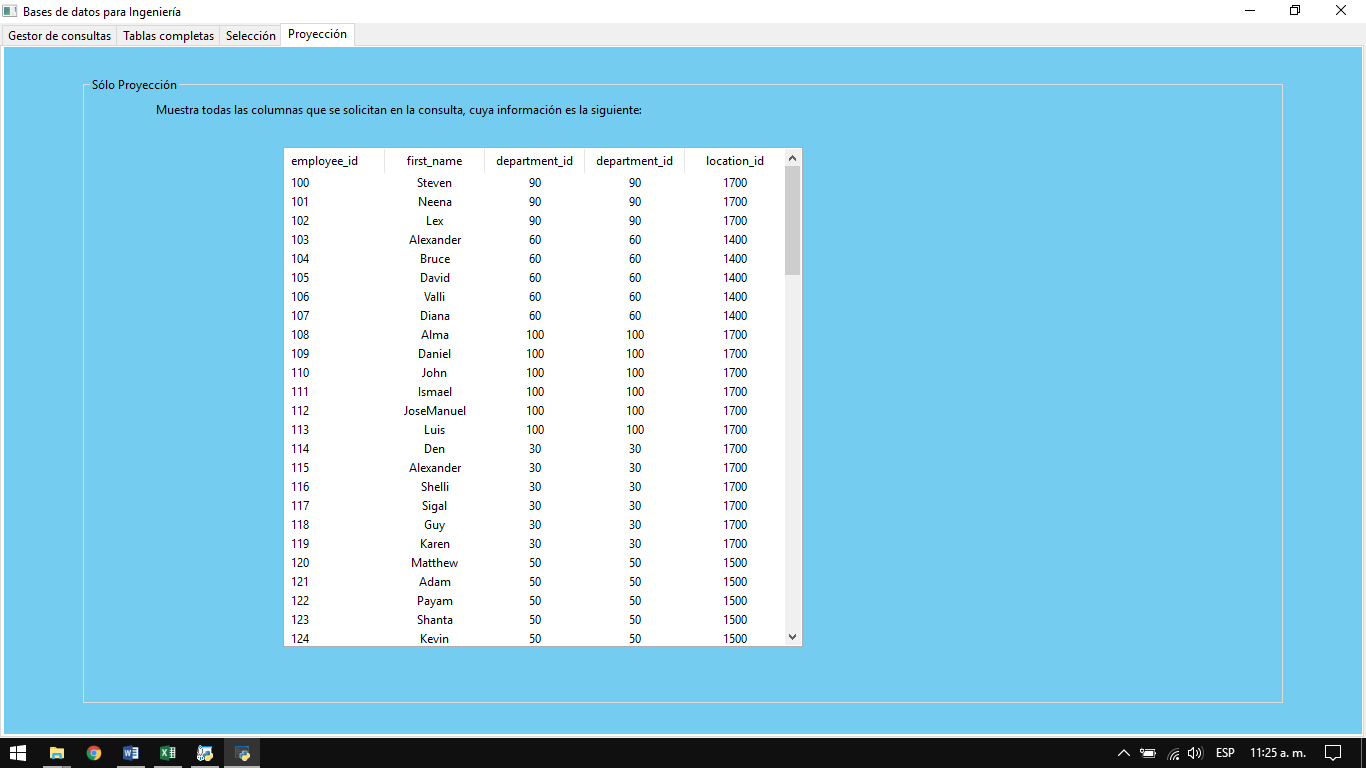


Tabla proyección del Join (Equijunta):



## REFERENCIAS

Acceso de directorios y archivos, Python, <https://docs.python.org/3/library/filesys.html>

Interfaz GUI – Frames, Python, <https://likegeeks.com/es/ejemplos-de-la-gui-de-python/>

The file descriptor, Adolfo Guzmán, <https://dl.acm.org/doi/abs/10.1145/800018.800543>