

PROYECTO INTEGRADOR

Análisis de la Industria del Automotor Usado,
Respecto a los Sueldos Argentinos.



COLABORADORES

- Sena Martin Ismael
- Sierra Fernando
- Soria Julio Ezequiel
- Tissera Jorgelina



MANUAL TÉCNICO

Proyecto: Análisis de la Industria del Automotor Usado, respecto a los Salarios Argentinos.

MATERIA

Proyecto Integrador

DOCENTES

Silvia Perotti

Héctor Prado

ISPC

INSTITUTO SUPERIOR POLITÉCNICO CÓRDOBA

WEB & DATA SCRAPING



Web Scraping

Análisis de la Industria del Automotor Usado, respecto a los Salarios Argentinos.

MANUAL TÉCNICO

CONTENIDO

| | |
|---|----|
| OBJETIVOS..... | 4 |
| OBJETIVOS ESPECIFICOS..... | 4 |
| ALCANCE..... | 4 |
| REQUERIMIENTOS TECNICOS..... | 4 |
| REQUERIMIENTOS MINIMOS DE HARDWARE..... | 5 |
| RECOMENDADOS DE HARDWARE..... | 5 |
| REQUERIMIENTOS MINIMOS DE SOFTWARE | 5 |
| HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO | 5 |
| INSTALACIÓN..... | 5 |
| CONFIGURACION..... | 6 |
| ¿QUE ES EL WEB SCRAPING?..... | 6 |
| WEB SCRAPING: Definición. | 7 |
| ¿COMO FUNCIONA EL WEB SCRAPING? | 7 |
| ¿CON QUE FIN SE UTILIZA? | 8 |
| ¿ES LEGAL EL WEB SCRAPING?..... | 8 |
| ANALISIS GENERAL | 9 |
| INTRODUCCIÓN | 9 |
| PROPÓSITO..... | 9 |
| ALCANCE..... | 10 |
| ORGANIZACIÓN ESTRUCTURAL SCRUM..... | 10 |
| PERSONAL INVOLUCRADO | 11 |
| DIAGRAMA DEL SCRIPT DE EXTRACCION DE DATOS..... | 12 |
| DESARROLLO DEL SCRIPT DE EXTRACCIÓN DE DATOS..... | 12 |
| SEGMENTACIÓN DE LOS DATOS OBTENIDOS | 15 |
| INTRODUCCIÓN | 15 |
| DESARROLLO | 15 |
| BASE DE DATOS | 21 |
| CONCLUSIONES FINALES | 28 |
| MATERIAL DE CONSULTA | 31 |

OBJETIVOS

Se ha creado dicho documento con el propósito de mostrar como fue diseñado el sistema, y al mismo tiempo dar referencias de como interactuar con el programa para que sea actualizado o al mismo tiempo se le de un mantenimiento adecuado en caso de un posible fallo.

A grandes rasgos se diseñó con el mero propósito de guiar al programador que este al frente de dicho sistema de como se fue estructurando el mismo, desde su proceso de instalación, código fuente, etc.

OBJETIVOS ESPECIFICOS

- Guía de instalación de las librerías y drivers necesarios.
- Mostrar el código fuente, detallando algunos rasgos para su futura actualización.
- Requisitos para la ejecución.
- Que debe y no debe hacer un web scraper.
- Conclusiones respecto a los datos obtenidos.

ALCANCE

Este documento está dirigido a: programador.

Conocimientos básicos en: Python, HTML, Base de datos.

REQUERIMIENTOS TECNICOS

Software

- Entorno de desarrollo integrado (IDE): PyCharm, Visual Studio Code, Sublime Text3 y Jupyter Notebook; para el desarrollo del script.
- Gestor de bases de datos(MySQL-Sqlite3) para la administración de los datos obtenidos del sitio web.
- Un emulador de servidor: (XAMPP) para el funcionamiento de la base de datos.
- Navegador web(Chrome-Edge) para la ejecución del script que va a interactuar con los datos del sitio.

Hardware

- Una computadora completa (parlantes no es necesario): esto incluye mouse, teclado, CPU, monitor.
- Conexión a internet vía Wifi o ethernet.

REQUERIMIENTOS MINIMOS DE HARDWARE

- Procesador: Intel® Core™3 or AMD Ryzen3 3250u
- RAM: 1GB
- Espacio en Disco: 1-2GB

RECOMENDADOS DE HARDWARE

- Procesador: Intel® Core™ i5 @2.60Ghz or AMD Ryzen5.
- RAM: 4GB
- Solid Disk: 2-3GB

REQUERIMIENTOS MINIMOS DE SOFTWARE

- Sistema operativo: Windows 7 or 10, Linux 64-bit RHEL or Mac OS X 10.11 & up.
- Privilegios de Administrador: Si

HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

- Entorno de desarrollo integrado (IDE): PyCharm, Visual Studio Code, Sublime Text3 y Jupyter Notebook; para el desarrollo del script.
- Librerías de Python: Selenium, Random, Time, Pandas, MySQL.
- Librerías para Grafica: Matplotlib.
- Simulador: ChromeDriver.
- Navegador web(Chrome-Edge) para la ejecución del script que va a interactuar con los datos del sitio.
- Git Bash.
- Gestor de bases de datos(MySQL-Sqlite3) para la administración de los datos obtenidos del sitio web.
- Un emulador de servidor: (XAMPP) para el funcionamiento de la base de datos.

INSTALACIÓN

A fines de que este tramo no sea muy extenso, dejaremos en evidencia las guías a las cuales se pueden consultar para poder instalar las herramientas necesarias para construir el script.

PDF Instalación Python 3.7/3.8

Link: <http://www.frlp.utn.edu.ar/materias/sintaxis/tutorialinstalarPython3.pdf>

Documentación:

Link: <https://www.python.org/downloads/>

Video Instalación Python y VS Code

Link: https://www.youtube.com/watch?v=-lyA_Yvs8IQ

Video Instalación Python y PyCharm

Link: <https://www.youtube.com/watch?v=DHgnFnI2MJU>

Video Instalación ChromeDriver

Documentación

Link: <https://chromedriver.chromium.org/getting-started>

Link: <https://www.youtube.com/watch?v=Q1CiZlYCsWM>

Git y GitHub

Documentación:

Link: <https://git-scm.com/doc>

Link: <https://www.youtube.com/watch?v=cYLapo1FFmA&t=234s>

Link: <https://www.youtube.com/watch?v=3XlZWpLwvvo>

CONFIGURACION

No hay una configuración general de la aplicación, en si ya viene configurada y es necesario contar con las herramientas anteriormente citadas para poder ejecutarlo. Dependiendo del IDE podrá realizarlo de distintas maneras.

Cabe mencionar que el programador podrá modificar la web de la cual quiera extraer los datos, siempre y cuando teniendo en cuenta que los atributos van a variar entre una web y otra.

¿QUE ES EL WEB SCRAPING?

Los motores de búsqueda, como Google, utilizan desde hace tiempo los denominados rastreadores web o crawlers, que exploran Internet en busca de términos definidos por el usuario. Los rastreadores son tipos especiales de

bots, que visitan una página web tras otra para generar asociaciones con términos de búsqueda y categorizarlos. El primer rastreador web se creó ya en 1993, cuando se presentó el primer motor de búsqueda: Jumpstation.

Entre estas técnicas de rastreo se incluye el web scraping o webharvesting. Te explicamos cómo funciona, para qué se utiliza y cómo se puede bloquear en caso necesario.

WEB SCRAPING: Definición.

Durante el web scraping (del inglés scraping = arañar/raspar) se extraen y almacenan datos de páginas web para analizarlos o utilizarlos en otra parte. Por medio de este raspado web se almacenan diversos tipos de información: por ejemplo, datos de contacto, tales como direcciones de correo electrónico o números de teléfono, o también términos de búsqueda o URL. Estos se almacenan en bases de datos locales o tablas.

¿COMO FUNCIONA EL WEB SCRAPING?

Dentro del scraping hay diferentes modos de funcionamiento, aunque en general se diferencia entre el scraping automático y el manual. El scraping manual define el copiado y pegado manual de información y datos, como quien recorta y guarda artículos de periódico y solo se lleva a cabo si se desea encontrar y almacenar alguna información concreta. Es un proceso muy laborioso que raras veces se aplica a grandes cantidades de datos.

En el caso del scraping automático, se recurre a un software o un algoritmo que analiza diferentes páginas web para extraer información. Se utiliza software especializado según el tipo de página web y el contenido. Dentro del scraping automático, se diferencian varios modos de proceder:

Analizador sintáctico: los analizadores sintácticos (o parsers) se utilizan para convertir un texto en una nueva estructura. Por ejemplo, en los análisis de HTML, el software lee un documento HTML y almacena la información. Un analizador DOM utiliza la representación de contenidos del lado del cliente en el navegador para extraer datos.

Bots: un bot es un software dedicado a realizar determinadas tareas y automatizarlas. En el caso del web harvesting, los bots se utilizan para examinar páginas web automáticamente y recopilar datos.

Texto: aquellos que tienen experiencia con la línea de comandos pueden aprovechar la función grep de Unix para buscar en la web determinados términos en Python o Perl. Este es un método muy sencillo para extraer datos, aunque requiere más trabajo que la utilización de un software

¿CON QUE FIN SE UTILIZA?

El web scraping se utiliza para una gran variedad de tareas, por ejemplo, para recopilar datos de contacto o información especial con gran rapidez. En el ámbito profesional, el scraping se utiliza a menudo para obtener ventajas respecto a la competencia. De esta forma, por medio del harvesting de datos, una empresa puede examinar todos los productos de un competidor y compararlos con los propios. El web scraping también resulta valioso en relación con los datos financieros: es posible leer datos desde un sitio web externo, organizarlos en forma de tabla y después analizarlos y procesarlos.

Google es un buen ejemplo de web scraping. El buscador utiliza esta tecnología para mostrar información meteorológica o comparaciones de precios de hoteles y vuelos. Muchos de los portales actuales de comparación de precios también utilizan el scraping para representar información de diferentes sitios web y proveedores.

¿ES LEGAL EL WEB SCRAPING?

El scraping no siempre es legal. En primer lugar, los scrapers deben tener en cuenta los derechos de propiedad intelectual de los sitios web. El web scraping tiene consecuencias muy negativas para algunas tiendas online y proveedores, por ejemplo, si el posicionamiento de su página se ve afectado debido a los agregadores. No es raro, por lo tanto, que una empresa se querelle contra un portal de comparaciones para impedir el web scraping. En uno de estos casos, el Tribunal Superior Regional de Fráncfort dictaminó en 2009 que una compañía aérea debía permitir el scraping por parte de portales comparativos porque, al fin y al cabo, su información es de libre acceso. No obstante, la aerolínea tenía la posibilidad de recurrir a medidas técnicas para evitarlo.

El scraping es legal, por lo tanto, siempre y cuando los datos recabados estén disponibles libremente para terceros en la web. Para garantizar la legalidad del web scraping, hay que tener en consideración lo siguiente:

Observar y cumplir con los derechos de propiedad intelectual. Si los datos están protegidos por estos derechos, no se pueden publicar en ninguna otra parte.

Los operadores de las páginas tienen derecho a recurrir a procesos técnicos para evitar el web scraping que no pueden ser eludidos.

Si, para la utilización de datos, se requiere el registro de usuarios o un contrato de utilización, estos datos no podrán ser aprovechados mediante scraping.

No se permite la ocultación de publicidad, de términos y condiciones, ni de descargos de responsabilidad mediante tecnologías de scraping.

Aunque el scraping está permitido en muchos casos, puede utilizarse con fines destructivos o incluso ilegales. Por ejemplo, esta tecnología se utiliza a menudo

para enviar spam. Los emisores pueden aprovecharla para, por ejemplo, acumular direcciones de correo electrónico y enviar mensajes de spam a estos destinatarios.

ANALISIS GENERAL

¿Qué es lo solicitado de dicha aplicación?

¿Qué es lo que debería hacer la aplicación?

¿Qué información es la que debemos rescatar de la web?

¿El programa guarda archivos? ¿de qué tipo?

¿Qué tipo de interfaz se necesita para manejarlo?

¿Qué es lo que pretendemos analizar?

¿La información es importante para poder compartirla?

¿Qué conocimientos necesitaríamos adquirir para poder concretarlo?

INTRODUCCIÓN

Este sistema ha sido analizado, diseñado, desarrollado, testeado y al mismo tiempo, dándonos soporte entre los colaboradores a posibles fallos de cualquier tipo, con el fin de que el programa brinde la mejor calidad y rendimiento posible a cualquier persona que pueda tener acceso a este.

Siempre se tuvo en mente la meta de poder optimizar, mejorar y estar avanzando todos los requerimientos que este debe cumplir, desarrollando un competente programa, demostrando así la calidad y compromiso de este equipo de trabajo.

Se ha ajustado este sistema de manera tal que el mismo pueda, extraer los datos solicitados, hacer un filtro de rangos, realizar gráficos estadísticos y administrar la información en una base de datos.

PROPÓSITO

Se detallará a continuación como se fue planteando y elaborando este sistema, para evitar cualquier tipo de confusión que se tenga en base a su desarrollo, funcionamiento y los alcances que tiene el mismo.

ALCANCE

La idea surge de poder realizar una comparación de los valores actuales de los vehículos usados en relación al sueldo Argentino de varias profesiones, tomando como referencia la web: https://www.olx.com.ar/autos_c378 y para los datos de los salarios, los mismos serán recolectados sin el proceso de web scraping. Para esto vamos a tener en cuenta varios factores para poder llegar a realizar una predicción para su compra.

Dicho sistema se ha programado para que se obtengan con mayor efectividad todos los datos que creemos que son importantes para el análisis, para luego ser clasificados dentro de un rango y sometidos a un análisis que nos brindara un panorama actual.

ORGANIZACIÓN ESTRUCTURAL SCRUM

Esta aplicación se basa en el método Kanban, inventado en los años 40 para gestionar y procesar el trabajo de manera visual. A nivel de gestión de grupos se trata de saber, con un simple vistazo, qué está haciendo cada persona, cuánto tiempo le queda y que va a ser lo siguiente. De esta manera se puede controlar el proyecto y mejorar las rutinas de trabajo del equipo.

Si usas Trello en el ámbito personal, por ejemplo, para organizar tus estudios, o en tu profesión a nivel de persona autónoma, facilita saber qué cosas faltan por hacer, cuáles estás haciendo y las que ya has hecho.

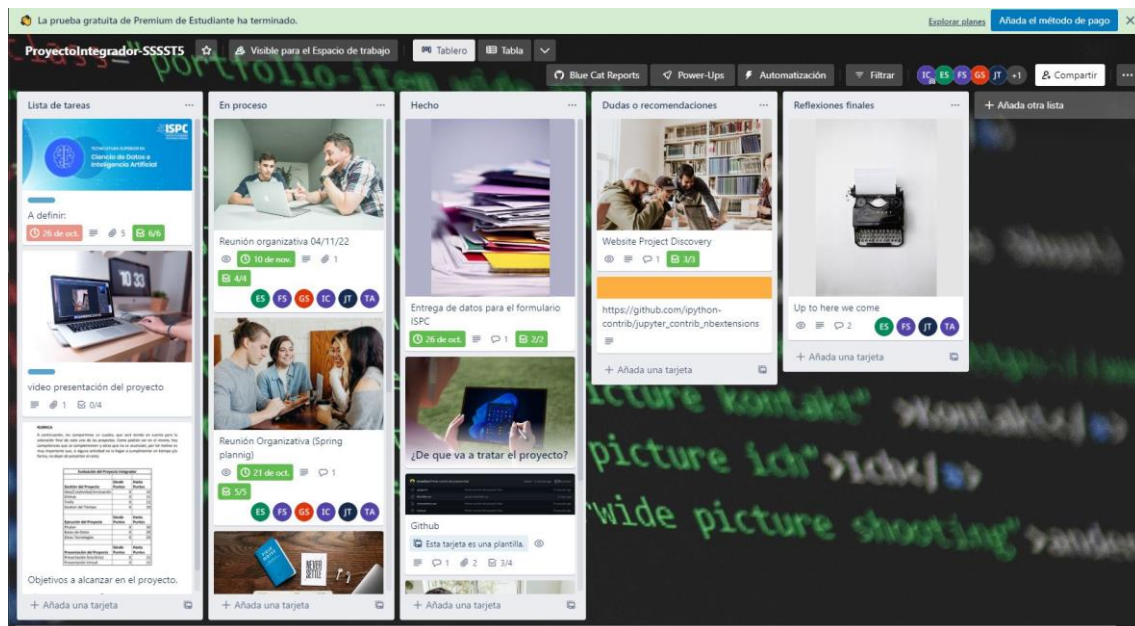
El método Kanban, al igual que Trello, se basa en 3 columnas:

- To Do: Columna en la que se indica lo que queda por hacer.
- Doing: Columna en la que se indica lo que se está haciendo.
- Done: Columna en la que se indica lo que ya se ha hecho.

La manera en la cual asumimos roles, fue a través de una primera reunión por la aplicación MEET, en donde en un primer principio hubo una suerte de "Lluvia de ideas" respecto al enfoque del proyecto. La misma la trabajamos por casi 7 días, en donde luego de analizar conjuntamente, pudimos llegar a la idea de que la misma no colmaba nuestras expectativas en cuanto a su finalidad. Queríamos lograr una producción que sea, no solo buena respecto al código fuente, sino que también, que los análisis que se obtengan de acuerdo a los datos, sea información relevante para la sociedad.

Toda información relevante la compartíamos en los ficheros que se fueron armando dentro de la plataforma, para así de esta manera la información no se pierda y pueda ser utilizada en el momento que sea más conveniente.

A continuación, se pondrá una muestra no concluida de los procesos de elaboración, ya que aún faltan procesos por concluir, no en cuanto a la aplicación sino más bien, relacionados a la formalidad con la que se presenta este software a los docentes y compañeros de la catedra. La idea es ver la dinámica de trabajo que se estuvo aplicando dentro del mismo.



En la primera fila de la izquierda “LISTA DE TAREAS”, son los objetivos que aún no pudimos finalizar, por ejemplo, la presentación de la aplicación mediante un video. La misma disponemos de tiempo según los plazos establecidos por los docentes.

En segundo lugar, se encuentra “EN PROCESO” en donde registramos tareas las cuales han comenzado pero que restan detalles para poder concretarlos, como lo es este manual, el cual, si bien no estaba contemplado como un requisito de proyecto, creemos que es una manera más “didáctica” de adoptar una buena practica como futuros profesionales en data.

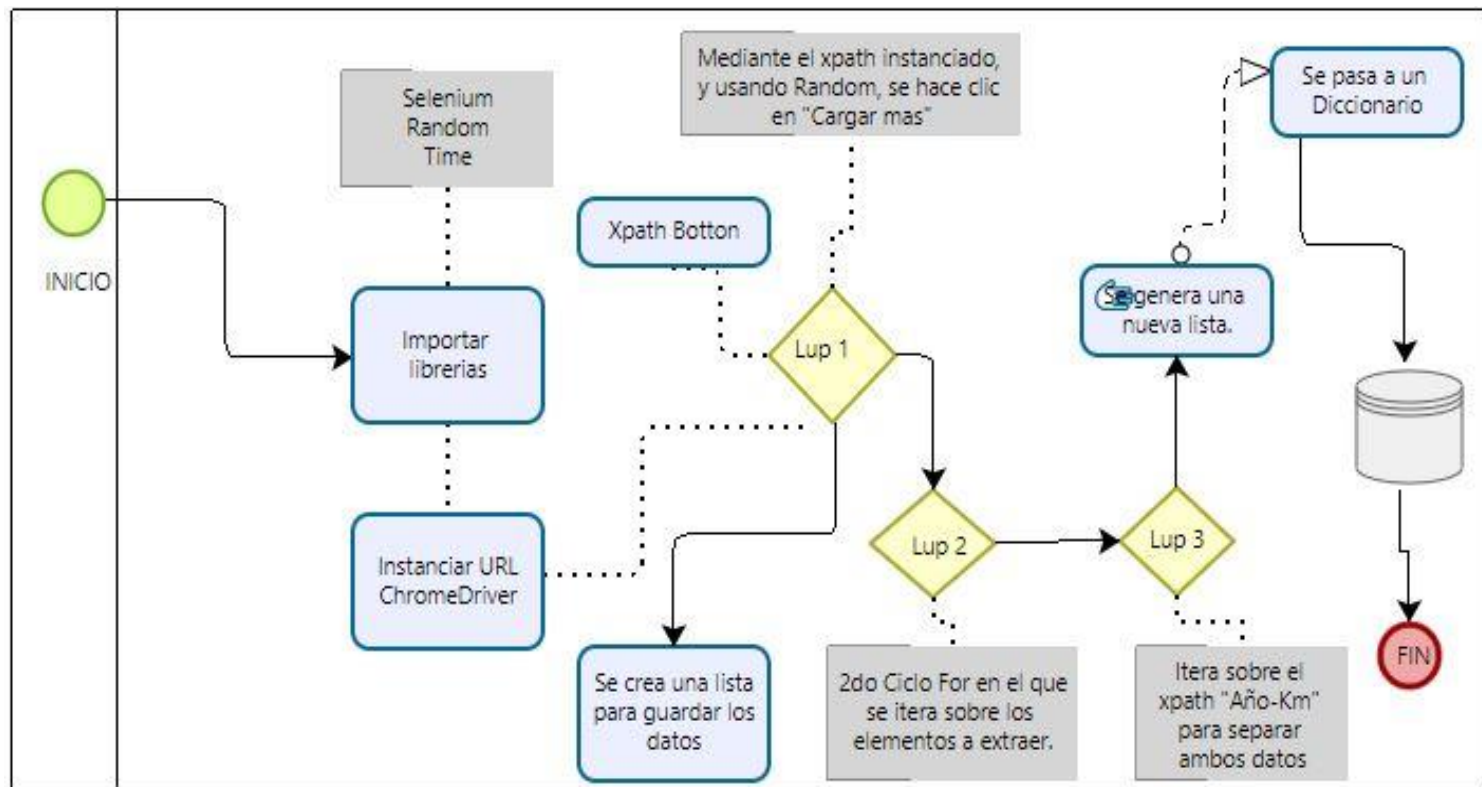
A continuación, esta la columna “HECHO”, el cual aloja los procesos que ya se cumplieron de manera efectiva. Es necesario manifestar que, antes de poder ser declarados como finalizados, se sometieron a diversas pruebas para constatar que los resultados sean los esperados.

Por último, nos encontramos con “DUDAS O RECOMENDACIONES” y “REFLEXIONES FINALES”, en la primera, sus palabras ya nos indican de que trata, allí alojamos las dudas que se iban presentando a lo largo del desarrollo y en el siguiente, quisimos de alguna manera poder dejar un registro de la experiencia personal que tuvimos al momento de concretar el proceso. Para la mayoría, este fue su primer trabajo de manera colaborativa, en la cual surgieron muchas dudas, frustraciones, mucha dedicación y por sobre todas las cosas, compromiso y aprendizaje.

PERSONAL INVOLUCRADO

| Nombre y Apellido | Rol |
|----------------------|--|
| Sena Ismael | Scrum Master Y Desarrollador |
| Sierra Fernando | Desarrollador/ Desarrollador de Bases de Datos |
| Soria Julio Ezequiel | Desarrollador/ Desarrollador de Bases de Datos |
| Tissera Jorgelina | Desarrollador y Analista de Datos |

DIAGRAMA DEL SCRIPT DE EXTRACCION DE DATOS



DESARROLLO DEL SCRIPT DE EXTRACCIÓN DE DATOS

En primer lugar, tuvimos que observar la web para verificar que la misma no esté con código JavaScript, al corroborar que si utiliza debimos optar por trabajar con una librería que es un poco mas lenta que la que se propuso por partes de los docentes. Si bien Selenium está mas destinada para su uso en Testing, también se la puede utilizar para automatizar funciones y poder realizar una paginación de manera automática en la web.

```

1
2 import random
3 from time import sleep
4 from selenium import webdriver
5
6

```

De igual manera lo hacemos con las demás librerías que vamos a utilizar mas adelante: Random y Time.

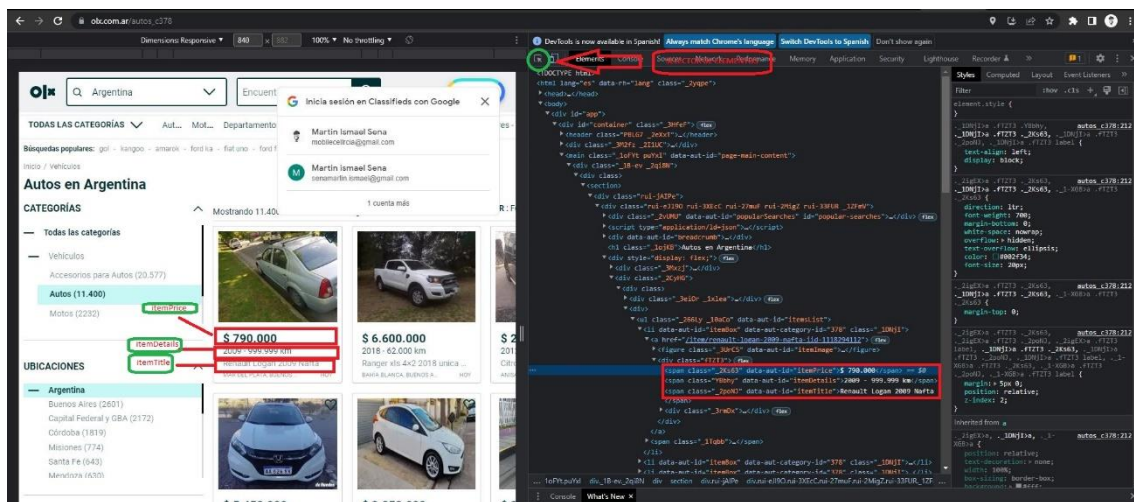
Mas adelante, es necesario instanciar la ubicación del simulador *ChromeDriver* dejando en evidencia la ubicación dentro de nuestro archivo local. Por otro lado en ese mismo nivel a través del método "get.", se hace la referencia al sitio web al cual vamos a manipular. El método "driver.maximize_window()" es solo para poder ampliar el navegador a lo largo y ancho de la pantalla.

```

7 driver = webdriver.Chrome('./chromedriver.exe') # NOMBRE DE TU CHROME DRIVER
8
9 driver.get('https://www.olx.com.ar/autos_c378')
10 driver.maximize_window()

```

Ahora nos toca inspeccionar la web para poder detectar cuales son las referencias que vamos a necesitar para poder extraer la información. Investigamos en varios sitios webs, documentación, tutoriales y en la mayoría coinciden que la mejor manera de poder acceder a estos atributos es por el "xpath" el cual esta menos propensos a errores.



Al hacer scroll sobre la web, nos topamos con que la misma no sigue cargando más artículos, sino que tiene un botón "CARGAR MAS" la cual permite traer más resultados en la búsqueda. Es por eso que se tuvo que detectar a través de una inspección de la web y traer en código el xpath del mismo, para luego automatizar el clic con el método ".click()".

La función "sleep" cumple un papel de suma importancia a la hora de la automatización, ya que el mismo permite darle un tiempo de espera a nuestro código y a la web, antes de seguir corriendo el script. Esta función debe ser lo más parecida a un "humano", ya que, si siempre damos un mismo valor, la web podría detectar que es un "robot", y de esta manera es que evadir si la misma tiene un detector de "Bot". Es más que necesario "randomizar" ese intervalo de tiempo "sleep(random.uniform(8.0, 10.0))" Uniform lo que hace es darnos un valor aleatorio entre 8" y 10".

Al principio del lup "for i in range(0)" se debe ajustar la cantidad de veces que vamos a realizar la paginación, es decir, la cantidad de veces que el programa hará un scroll hasta el final y hará clic en "CARGAR MAS".

```

11 sleep(3)
12 driver.refresh() # Solucion donde los anuncios solo cargan al hacerle refresh o al darle click a algun elemento
13 sleep(5) # Esperamos que cargue el boton
14 # Busco el boton para cargar mas informacion
15 boton = driver.find_element("xpath", '//button[@data-aut-id="btnLoadMore"]')
16 for i in range(20):
17     try:
18         # le doy click
19         boton.click()
20         # espero que cargue la informacion dinamica lo hago random para que no crea q es un robot
21         sleep(random.uniform(8.0, 10.0))
22         # busco el boton nuevamente para darle click en la siguiente iteracion
23         boton = driver.find_element("xpath", '//button[@data-aut-id="btnLoadMore"]')
24     except:
25         # si hay algun error, rompo el lazo.
26         break
27

```

Este paso es importante y fue uno de los cuales más complicaciones nos trajo al momento de poder extraer la data. En una primera instancia debemos identificar los elementos que queremos extraer de la web y cual o cuales son los atributos que la representan dentro del código HTML. Decidimos identificarlo a través de un "xpath".

Los elementos a obtener son: PRECIOS, DESCRIPCION, AÑO Y KM. Las mismas ya identificadas y almacenadas en una variable distintiva para cada una, y a su vez tenían la función "append" el cual cumplirá con el almacenamiento de los datos extraídos.

En la últimas dos, nos surgió un problema, ambos elementos estaban alojados dentro de un mismo xpath: '//*[@data-aut-id="itemDetails"]', el cual tenía tanto AÑO como KM y se las necesitaba de manera separada para poder luego hacer la parte estadística.

Por este motivo, seguido del segundo ciclo for, se agrega uno más el cual va a recorrer el ultimo xpath y se le da como instrucción a través del método "lista=i.split('-')" que va a detectar el separador y va a almacenar los datos por separado "año.append(lista[0]), km.append(lista[1])". De esta manera pudimos comprobar a través de un "print" que los datos ya estaban en columnas por separado.

Por último, se almacenan los datos en una variable **"mi_diccionario"** para luego ser alojado en un ***DataFrame***.

```

27
28 # Encuentro cual es el XPATH de cada elemento donde esta la informacion que quiero extraer
29 # Esto es una LISTA. Por eso el metodo esta en plural
30 lista_precios=[]
31 lista_descripcion=[]
32
33 lista_año_km=[]
34
35 autos = driver.find_elements("xpath", '//li[@data-aut-id="itemBox"]')
36
37 # Recorro cada uno de los anuncios
38 for auto in autos:
39     # Precio por cada anuncio
40     precio = auto.find_element("xpath", '//*[@data-aut-id="itemPrice"]').text
41     lista_precios.append(precio)
42     #print (precio)
43     # Descripción de cada anuncio
44     descripcion = auto.find_element("xpath", '//*[@data-aut-id="itemTitle"]').text
45     lista_descripcion.append(descripcion)
46     #print (descripcion)
47     año_km=auto.find_element("xpath", '//*[@data-aut-id="itemDetails"]').text
48     lista_año_km.append(año_km)
49
50 driver.quit()
51 año=[]
52 km=[]
53 for i in lista_año_km:
54     lista=i.split('-')
55     año.append(lista[0])
56     km.append(lista[1])
57 #print(lista_precios)
58 #print(lista_descripcion)
59 mi_diccionario={'Precio':lista_precios, 'Descripcion':lista_descripcion, 'Año':año, 'Km':km}
60 #print(mi_diccionario)
61

```


En este último paso importamos la librería “Pandas” y realizamos el pasaje de los datos a un archivo CSV la cual luego nos va a permitir poder trabajar con herramientas de visualización.

```
63 import pandas as pd
64 df=pd.DataFrame(mi_diccionario,columns=['Precio','Descripcion','Año','Km'])
65 print(df)
66 df.to_csv('salida_productos.csv',index=False)
```

SEGMENTACIÓN DE LOS DATOS OBTENIDOS

INTRODUCCIÓN

A continuación, especificaremos los procesos que se llevaron adelante para poder filtrar toda la información que obtuvimos de la URL. Al ser muchos datos, contemplamos que solo tomaríamos en cuenta los autos y sueldos en determinado rango y para esto era necesario construir un script que nos arroje los resultados deseados.

DESARROLLO

Una vez obtenido el archivo CSV realizado con la técnica de Scraping se procedió a hacer la búsqueda de los autos a partir de intervalos de valores los cuales fueron implementados en el archivo “Precio_año_auto.py” que se detalla a continuación.

```
conclusiones.py U Precio_año_auto.py U X
Precio_año_auto.py > ...
1 import csv
2 import matplotlib.pyplot as plt
3
4 def promedios(suma, cantidad):
5     promedios=suma//cantidad
6     return promedios
7
8
9
10 separador=","
11 with open('salida_productos.csv', encoding="utf-8") as archivo:
12     next(archivo) #Omito el encabezado del archivo
13     Productos=[]
14     for linea in archivo:
15         linea=linea.rstrip("\n") #quito el salto de línea
16         columnas=linea.split(separador)
17         precio=int(columnas[0])
18         descripcion=columnas[1].lower() #pongo la descripción todo en minúsculas
19         año=int(columnas[2])
20         km=int(columnas[3])
21
22         Productos.append({
23             "precio":precio,
24             "descripcion":descripcion,
25             "año": año,
26             "km":km,
27         })
28
29
30 #print(Productos)
31
32 suma1=0
33 suma2=0
34 suma3=0
35 suma4=0
36 suma5=0
37 suma6=0
38
```


Se importan las librerías “csv” y “matplotlib”, este último tendrá la finalidad de luego poder graficar los resultados obtenidos.

Se define la función promedios, que nos va a devolver el valor promedio de un auto en un rango de precios. Se hace la apertura del archivo salida_productos.csv y se comienza a filtrar en el rango de precios:

1. Autos con precios mayores a \$500000 y menores o iguales a \$1500000
2. Autos con precios mayores a \$1500000 y menores o iguales a \$2500000
3. Autos con precios mayores a \$2500000 y menores o iguales a \$3500000
4. Autos con precios menores a \$3500000 y menores o iguales a \$5000000
5. Autos con precios menores a \$5000000 y menores o iguales a \$6500000
6. Autos con precios mayores a \$6500000 y menores o iguales a \$8000000

```
Precio_año_auto.py > ...
46
47 for producto in Productos:
48     precio_auto = producto["precio"]
49     if precio_auto > 500000 and precio_auto <= 1500000:
50         suma1 += precio_auto
51         autos_menor_15 += 1
52     elif precio_auto > 1500000 and precio_auto <= 2500000:
53         suma2 += precio_auto
54         autos_15_25 += 1
55     elif precio_auto > 2500000 and precio_auto <= 3500000:
56         suma3 += precio_auto
57         autos_25_35 += 1
58     elif precio_auto > 3500000 and precio_auto <= 5000000:
59         suma4 += precio_auto
60         autos_35_50 += 1
61     elif precio (variable) precio_auto: Any to <= 6500000:
62         suma5 += precio_auto
63         autos_50_65 += 1
64     elif precio_auto > 6500000 and precio_auto <= 8000000:
65         suma6 += precio_auto
66         autos_mayor_65 += 1
67
68 total_autos = autos_menor_15 + autos_15_25 + autos_25_35 + autos_35_50 + autos_50_65 + autos_mayor_65
69
70
71
72 #Muestra cantidad de autos por rango de precio:
73 print("La cantidad de autos cuyo valor es menor a 1500000 son: ", autos_menor_15)
74 print("La cantidad de autos con precio entre 1500000 y 2500000 son: ", autos_15_25)
75 print("La cantidad de autos con precio entre 2500000 y 3500000 son: ", autos_25_35)
76 print("La cantidad de autos con precio entre 3500000 y 5000000 son: ", autos_35_50)
77 print("La cantidad de autos con precio entre 5000000 y 6500000 son: ", autos_50_65)
78 print("La cantidad de autos con precio mayor a 6500000 son: ", autos_mayor_65)
79 print("\nEl scrapping se realizo sobre 500 autos ")
80 print("Algunos datos no sirvieron ya que no reflejaban el precio que es un dato crucial para nuestro análisis")
81 print("La cantidad total de autos que sirvieron para el análisis son: ", total_autos)
82
83 #Notas: algunas veces no funciona
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL JUPYTER
PS C:\Users\gelit\Grupo proyecto>
```

Se imprimen los resultados en la consola como así también el cálculo de los valores promedios por intervalo analizado:

```

conclusiones.py U Precio_año_auto.py U
Precio_año_auto.py > ...
64     autos_50_65+=1
65
66     elif precio_auto>6500000 and precio_auto <=8000000:
67         suma6+=precio_auto
68         autos_mayor_65+=1
69 total_autos=autos_menor_15+autos_15_25+autos_25_35+autos_35_50+autos_50_65+autos_mayor_65
70
71
72 #Muestra cantidad de autos por rango de precio:
73 print("La cantidad de autos cuyo valor es menor a 1500000 son: ",autos_menor_15)
74 print("La cantidad de autos con precio entre 1500000 y 2500000 son: ",autos_15_25)
75 print("La cantidad de autos con precio entre 2500000 y 3500000 son: ",autos_25_35)
76 print("La cantidad de autos con precio entre 3500000 y 5000000 son: ",autos_35_50)
77 print("La cantidad de autos con precio entre 5000000 y 6500000 son: ",autos_50_65)
78 print("La cantidad de autos con precio mayor a 6500000 son: ",autos_mayor_65)
79 print("\nEl scrapping se realizo sobre 500 autos ")
80 print("Algunos datos no sirvieron ya que no reflejaban el precio que es un dato crucial para nuestro análisis")
81 print("La cantidad total de autos que sirvieron para el análisis son: ",total_autos)
82
83 #Valores promedios por rangos:
84
85 print("\nEl precio promedio de los autos menores o iguales a $1500000 es: $",promedios(suma1, autos_menor_15))
86 print("\nEl precio promedio de los autos entre $1500000 y $2500000 es: $",promedios(suma2, autos_15_25))
87 print("\nEl precio promedio de los autos entre $2500000 y $3500000 es: $",promedios(suma3, autos_25_35))
88 print("\nEl precio promedio de los autos entre $3500000 y $5000000 es: $",promedios(suma4, autos_35_50))
89 print("\nEl precio promedio de los autos entre $5000000 y $6500000 es: $",promedios(suma5, autos_50_65))
90 print("\nEl precio promedio de los autos mayores a $6500000 es: ",promedios(suma6,autos_mayor_65))
91
92
93 #Análisis en función del modelo:
94
95 modelo_99_05=0
96 modelo_06_10=0
97 modelo_11_15=0
98 modelo_16_22=0
99 modelo_xx_98=0
100
101 for producto in Productos:

```

Se realiza a su vez un análisis de la cantidad de autos que se pudieron analizar, esta vez teniendo en cuenta un rango de año o modelo de auto. Los intervalos que se analizaron son:

1. modelo mayor o igual a 1999 y menor que 2006
2. modelo mayor o igual 2006 y menor que 2011
3. modelo mayor o igual que 2011 y menor que 2016
4. modelo mayor o igual que 2016 y menor o igual a 2022

Se muestran por consola los resultados de este análisis:

```

Precio_año_auto.py > ...
93 #Análisis en función del modelo:
94
95 modelo_99_05=0
96 modelo_06_10=0
97 modelo_11_15=0
98 modelo_16_22=0
99 modelo_xx_98=0
100
101 for producto in Productos:
102     precio_auto = producto["precio"]
103     año_auto=producto["año"]
104     if (precio_auto>500000) and (precio_auto<=8000000):
105         if año_auto>=1999 and año_auto<2006:
106             modelo_99_05+=1
107         elif año_auto>=2006 and año_auto<2011:
108             modelo_06_10+=1
109         elif año_auto>=2011 and año_auto<2016:
110             modelo_11_15+=1
111         elif año_auto>=2016 and año_auto<=2022:
112             modelo_16_22+=1
113         else:
114             modelo_xx_98+=1
115
116 #print("Cantidad de autos analizados: ",modelo_xx_98+modelo_99_05+modelo_06_10+modelo_11_15+modelo_16_22)
117 print("La cantidad de autos menores al año 99 son: ",modelo_xx_98)
118 print("La cantidad de autos modelo 99-05 son: ", modelo_99_05)
119 print("La cantidad de autos modelo 06-10 son: ", modelo_06_10)
120 print("La cantidad de autos modelo 11-15 son: ", modelo_11_15)
121 print("La cantidad de autos modelo 16-22 son: ", modelo_16_22)
122

```

Se realizan los gráficos con los resultados obtenidos y se guardan en archivos con extensión “.png”:

```

122
123 #Gráfico porcentajes según Los rangos precios de Autos
124 cantidad_autos_por_rango=[autos_menor_15,autos_15_25,autos_25_35,autos_35_50,autos_50_65,autos_mayor_65]
125 nombres=["<=1.5M",">1.5M-<=2.5M",">2.5M-<=3.5M",">3.5M-<=5M",">5M-<=6.5M",">6.5M"]
126 plt.pie(cantidad_autos_por_rango, labels=nombres, autopct="%0.1f%%")
127 plt.title("Porcentajes cantidad de autos según los rangos de precios: \n Análisis con respecto a 417 autos ")
128 plt.axis("equal")
129 plt.savefig("Porcentajes autos.png")
130 plt.show()
131
132 #Gráfico Capacidad de Ahorro
133 Modelo_año=[modelo_xx_98,modelo_99_05,modelo_06_10,modelo_11_15,modelo_16_22]
134 nombres=["1998","1999-2005","2006-2010","2011-2015","2016-2022"]
135 fig, ax=plt.subplots()
136 #Etiqueta en eje Y
137 ax.set_ylabel("Cantidad")
138 #Etiqueta en eje X
139 ax.set_title("Autos por rango de modelo")
140 plt.bar(nombres, Modelo_año, color='turquoise')
141 plt.savefig("Autos_por_año.png")
142 plt.show()
143
144

```

Se guarda la información obtenida en un archivo dictPrecios_año.csv:

```

145
146 dictPrecios_Año={'cantidad_autos':total_autos,
147                 "precio_menor_1.5":autos_menor_15,
148                 "precio_1.5-2.5":autos_15_25,
149                 "precio_2.5-3.5":autos_25_35,
150                 "precio_3.5-5":autos_35_50,
151                 "precio_5-6.5":autos_50_65,
152                 "precio_6.5":autos_mayor_65,
153                 "prom_menor1.5":promedios(suma1, autos_menor_15),
154                 "prom_1.5-2.5":promedios(suma2, autos_15_25),
155                 "prom_2.5-3.5":promedios(suma3, autos_25_35),
156                 "prom_5-6.5":promedios(suma5, autos_50_65),
157                 "prom_mayor6.5":promedios(suma6, autos_mayor_65),
158                 "mod_menor_99":modelo_xx_98,
159                 "mod_99_05":modelo_99_05,
160                 "mod_06_10":modelo_06_10,
161                 "mod_11_15":modelo_11_15,
162                 "mod_16_22":modelo_16_22,
163                 }
164
165
166
167 #Creación del archivo CSV con Los datos del análisis de Los modelos de
168 with open('dictPrecios_año.csv','w') as f:
169     writer=csv.writer(f)
170     for k,v in dictPrecios_Año.items():
171         writer.writerow([k,v])
172

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL JUPYTER

PS C:\Users\gelit\Grupo proyecto>

Se realiza el análisis del archivo “escalaSalarial.csv”, en donde se busca filtrar los salarios por intervalos de sueldos, se realiza la apertura de dicho archivo:

```

er Ir Ejecutar Terminal Ayuda
conclusiones.py U salarios.py U x
salarios.py > ...
1 import csv
2 import matplotlib.pyplot as plt
3
4
5
6 def promedios(suma, cantidad):
7     promedios=suma//cantidad
8     return promedios
9
10 def ahorro (promedio):
11     canasta=128214
12     return (promedio - canasta)
13
14
15 separador=","
16 with open('escala_salarial.csv', encoding="utf-8") as archivo:
17     next(archivo) #Omito el encabezado del archivo
18     Salarios=[]
19     for linea in archivo:
20         linea=linea.rstrip("\n") #quito el salto de línea
21         columnas=linea.split(separador)
22         cargo=columnas[0]
23
24
25         liquidacion=int(columnas[1])
26
27         Salarios.append({
28             "cargo":cargo,
29             "liquidacion":liquidacion,
30         })
31
32
33 #Análisis de intervalos de Los sueldos:
34 cant_menor_igual_150=0
35 cant_150_y_300=0
36 cant_300_y_500=0
37 cant_mayor_500=0
38 suma1=0

```

y se procede a analizar los intervalos de sueldos, siendo estos:

1. Salarios menores e iguales a \$150000
2. Salarios mayores a \$150000 y menores o iguales a \$300000
3. Salarios mayores a \$300000 y menores o iguales a \$500000
4. Salarios mayores a \$500000

Una vez realizado esto, se procedió a calcular un sueldo promedio de cada intervalo, y la capacidad de ahorro de esa persona se calculó haciendo sueldo promedio menos canasta básica. Esto evidenció que los sueldos analizados en el primer intervalo (<\$150000) tenían capacidad de ahorro nula, por lo tanto fueron descartados para nuestro futuro análisis.

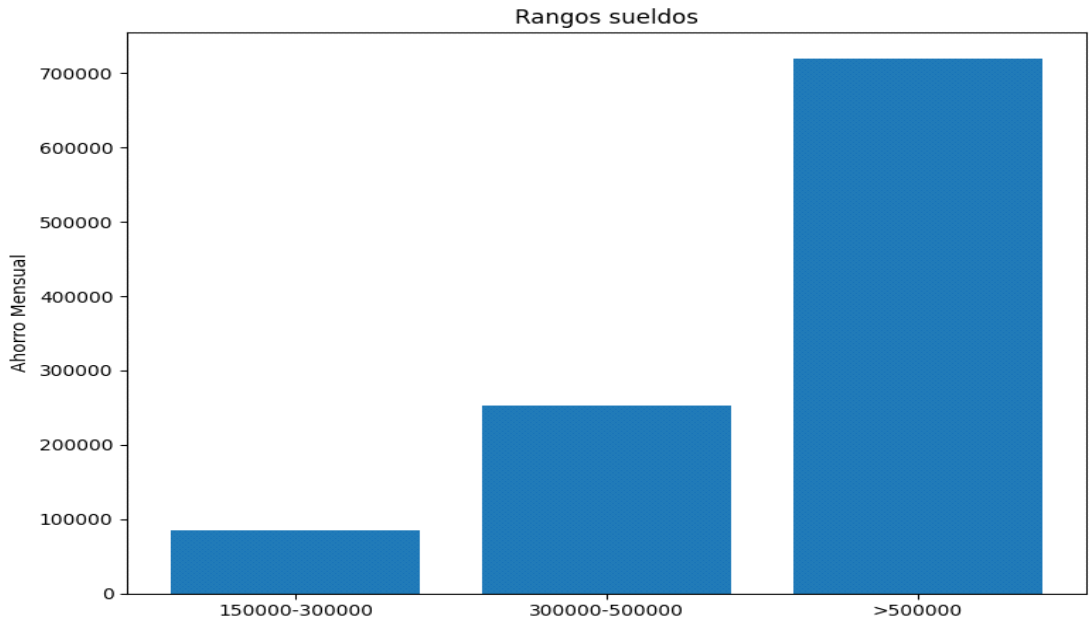
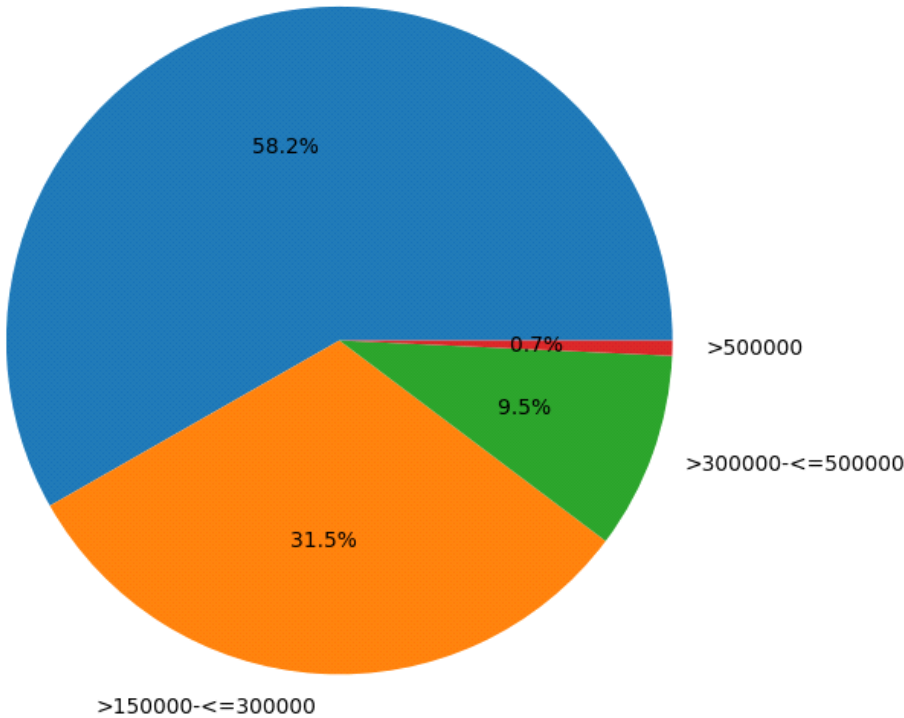
```
conclusiones.py U  salarios.py U X
salarios.py > ...
59
60     else:
61         cant_mayor_500+=1
62         suma+=val_salario
63
64 total_sueldos=cant_menor_igual_150+cant_150_y_300+cant_300_y_500+cant_mayor_500
65 print("\n LA CANTIDAD TOTAL DE SALARIOS ANALIZADOS ES: ", total_sueldos,"\n")
66
67
68 #Intervalo sueldos menores a $150000
69 print("\nLa cantidad de sueldos menores a 150000 son: ",cant_menor_igual_150)
70
71 print("\nEl sueldo promedio entre los sueldos menores o iguales a $150000 es: $",promedio(suma1, cant_menor_igual_150), "por lo que su capacidad de ahorro es nula, ya que no cubre el mont")
72
73 #Intervalo sueldos comprendidos entre 150000 y 300000
74 print("\nLa cantidad de sueldos mayores a 150000 y menores o iguales a 300000 son: ",cant_150_y_300)
75
76 prom_150_300=promedio(suma1, cant_150_y_300)
77 print("\nEl sueldo promedio entre los sueldos entre $150000 y $300000 es: $",prom_150_300)
78 print("\nLa capacidad de ahorro mensual de este grupo es: $",ahorro(prom_150_300))
79
80 #Intervalo sueldos comprendidos entre 300000 y 500000
81 print("\nLa cantidad de sueldos mayores a 300000 y menores o iguales a 500000 son: ",cant_300_y_500)
82
83 prom_300_500=promedio(suma1,cant_300_y_500)
84 print("\nEl sueldo promedio entre los sueldos mayores a $300000 y menores a $500000 es: $",prom_300_500)
85 print("\nLa capacidad de ahorro mensual de este grupo es: $",ahorro(prom_300_500))
86
87 #Intervalo sueldos mayores a 500000
88 print("\nLa cantidad de sueldos mayores a 500000 son: ",cant_mayor_500)
89 prom_500=promedio(suma1, cant_mayor_500)
90 print("\nEl sueldo promedio entre los sueldos mayores a $500000 es: $", prom_500)
91 print("\nLa capacidad de ahorro de este grupo es: $",ahorro(prom_500))
92
93
94
95 #ordifico porcentajes según los rangos de sueldos
96 cantidad_sueldos_por_rango=cant_menor_igual_150+cant_150_y_300+cant_300_y_500+cant_mayor_500
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
```

Estos resultados se representaron gráficamente, y se guardaron los datos en el archivo "dictSalarios.csv"

```
conclusiones.py U  salarios.py U X
salarios.py > ...
92
93
94
95 #Gráfico porcentajes según los rangos de sueldos
96 cantidad_sueldos_por_rango=[cant_menor_igual_150,cant_150_y_300,cant_300_y_500,cant_mayor_500]
97 nombres=["<150000","150000-<300000","300000-<500000",">500000"]
98 plt.pie(cantidad_sueldos_por_rango, labels=nombres,autopct="%0.1f%%")
99 plt.title("Porcentajes según los rangos de sueldos: \n Análisis con respecto a 273 salarios ")
100 plt.axis("equal")
101 plt.savefig("Porcentajes sueldos.png")
102 plt.show()
103
104
105 #Gráfico Capacidad de Ahorro
106
107 capacidad_ahorro=[ahorro(prom_150_300),ahorro(prom_300_500),ahorro(prom_500)]
108 nombres=["150000-300000","300000-500000",">500000"]
109 fig, ax=plt.subplots()
110 #Etiqueta en eje Y
111 ax.set_ylabel("Ahorro Mensual")
112 #Etiqueta en eje X
113 ax.set_title("Rangos sueldos")
114 plt.bar(nombres, capacidad_ahorro)
115 plt.savefig("Capacidad ahorro.png")
116 plt.show()
117
118
119
120 dictSalarios={"sueldos":total_sueldos,
121               "sueldos_menores_150000":cant_menor_igual_150,
122               "sueldo_promedio_150000":promedio(suma1, cant_menor_igual_150),
123               "sueldos_150_300":cant_150_y_300,
124               "sueldos_promedio_150_300":prom_150_300,
125               "ahorros_150_300":ahorro(prom_150_300),
126               "sueldos_300_500":cant_300_y_500,
127               "sueldos_promedio_300_500":prom_300_500,
128               "ahorros_300_500":ahorro(prom_300_500),
129               "sueldos_500":cant_mayor_500,
130               "sueldos_promedio_500":prom_500,
```

Muestra de las gráficas obtenidas:

Porcentajes según los rangos de sueldos:
Análisis con respecto a 273 salarios
<=150000



BASE DE DATOS INTRODUCCIÓN

Si bien en la consigna del proyecto brindada por el grupo de docentes, no estaba contemplado de manera específica que se trabaje sobre distintos sistemas operativos, a modo de demostración este grupo decidió plasmar el desarrollo de una base de datos tanto en LINUX como en WINDOWS. En la mismas están plasmadas el paso a paso para poder llegar a concluir las en cada una de las dos y su código fuente con el nombre "conexión_v2.py"

DESARROLLO BASE DE DATOS EN LINUX

Software utilizado:

Sistema Operativo: linux
Lenguaje python3 Version: 3.10.6
Base de Datos: mySQL 8.0.31

Primero ingresamos a una terminal de Linux y realizamos lo siguiente:

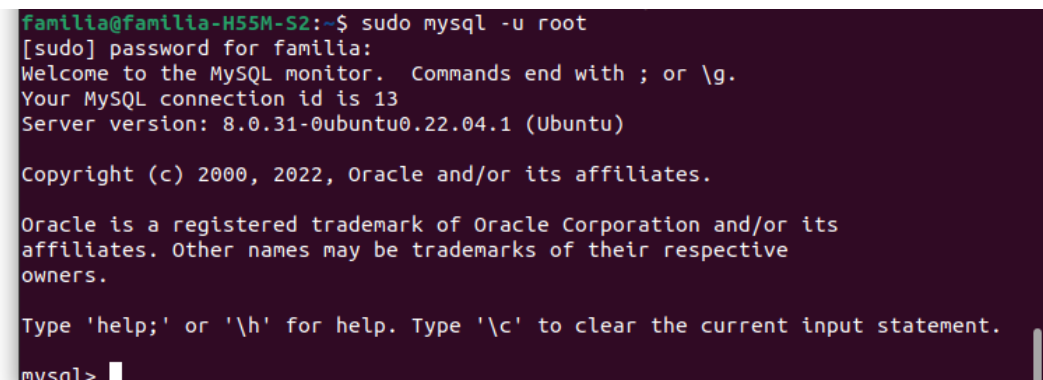
Instalar la librería para conectarse desde Python a la base de datos:

```
pip install mysql-connector-python
```

Instalación de MySQL:

```
sudo apt install mysql-server
```

Luego ingresar a MySQL como root:



```
Familia@familia-H55M-S2:~$ sudo mysql -u root
[sudo] password for familia:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 8.0.31-0ubuntu0.22.04.1 (Ubuntu)

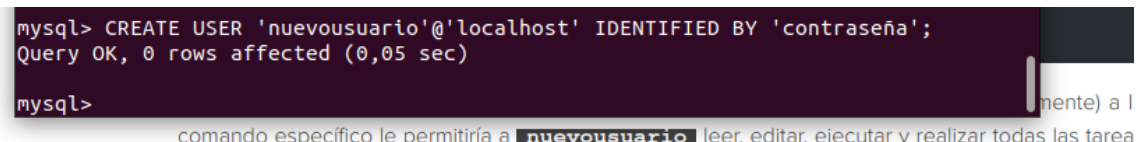
Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Creamos un usuario nuevo:



```
mysql> CREATE USER 'nuevousuario'@'localhost' IDENTIFIED BY 'contraseña';
Query OK, 0 rows affected (0,05 sec)

mysql>
```

comando específico le permitiría a **nuevousuario** leer, editar, ejecutar y realizar todas las tareas

Le asignamos los privilegios para poder crear un nuevo BD:

```
mysql> CREATE TABLE autos (descripcion VARCHAR(200), precio BIGINT , anio VARCHAR(20), km BIGINT);
Query OK, 0 rows affected (0,04 sec)

mysql>
```

use proyecto;

```
CREATE TABLE autos (descripcion VARCHAR(200), precio BIGINT , anio VARCHAR(20), km BIGINT
```

Los datos que vamos a usar son los que se habían extraído en una instancia y almacenados en un CSV denominado “salida_productos.csv” y el script: conexion.py.

Se modifica las líneas donde se configura el usuario:

```
import mysql.connector
from mysql.connector import Error
import re
import csv
;
try:
    connection = mysql.connector.connect(host='localhost',
                                         database='proyecto2',
                                         user='nuevousuario',
                                         password='contraseña')
    if connection.is_connected():
        db_info = connection.get_server_info()
        print("Connected to MySQL Server version ", db_info)
        cursor = connection.cursor()
        with open('salida_productos.csv', newline='') as csvfile:
            reader = csv.DictReader(csvfile)
            for row in reader:
                rr=re.findall(r'\d+\.\d+',row['Precio'])
                cant = len(rr)
                if cant:
                    if "USD" in row['Precio'] :|
                        precio=int(rr[0].replace(".", ""))*158
                    else:
                        precio=int(rr[0].replace(".", ""))
                print(precio,row['Descripcion'],row['Año'],row['Km'])
                mysql_insert_query = """INSERT INTO autos (precio, descripcion, anio,km
                record = (precio,row['Descripcion'], row['Año'],row['Km'])
                cursor.execute(mysql_insert_query, record)
                connection.commit()
except Error as e:
    print("Error while connecting to MySQL", e)
finally:
    if connection.is_connected():
        cursor.close()
```

Al ejecutar el mismo, se puede constatar que no genero ningún error.

```
Welcome to the MySQL Monitor. Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 8.0.31-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> GRANT ALL PRIVILEGES ON *.* TO nuevousuario@localhost;
Query OK, 0 rows affected (0,02 sec)

mysql>
```

Comprobamos que se pueda conectar con el nuevo usuario:

comando específico le permitiría a nuevousuario leer, editar, ejecutar y realizar todas las tareas c

```
byc
familia@familia-H55M-S2:~$ mysql -u nuevousuario -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.31-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Se crea una nueva base de datos la cual va a contener la data extraída y alojada en el archivo CSV.

```
mysql> create database proyecto;
ERROR 1007 (HY000): Can't create database 'proyecto'; database exists
mysql> create database proyecto2;
Query OK, 1 row affected (0,02 sec)

mysql> use proyecto2;
Database changed
mysql>
```

Creamos las tablas donde se va a guardar los datos:

```
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use proyecto2;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from autos;
+-----+-----+-----+-----+
| description | precio | anio | km |
+-----+-----+-----+-----+
| VOYAGE CONFORT 16 / 2013 / 135MIL | 2300000 | 2013 | 135000 |
| VW GOLF 14 TSI TURBO CONFORTLINE MANUAL 2016 FULL | 5250000 | 2016 | 69000 |
| GOL TREND 16 PACK 1 SPTAS / 2016 / 75MIL | 335000 | 2020 | 33500 |
```

Ingresamos a la base de datos y realizamos un SELECT a la tabla para ver los resultados:


```
2250000 Suran impecable titular 2011 118000
1850000 Vendo Renault Sandero modelo 2012 2012 87400
7000000 CRUZE PREMIER 2022 2022 4000
3300000 Kwid intense 2020 con 33500 reales 2020 34000
4200000 MITSUBISHI OUTLANDER 24 GLS CVT AT6 CON LEVAS 4WD 2013 SEGUNDO DUEÑO 117
000 TITULAR 2013 117000
5600000 Impecable Nissan Kicks 2018 nueva 17333 nafta 2018 17333
9250000 Mercedes Benz Sprinter 21 415 Combi 3665 150cv 15+1 Te 2018 Unico dueño A
cepto Vehiculo de menor valor 2018 240000
8000000 2021 Chevrolet S10 LS 4x2 Linea 2021 2021 32000
4400000 Citroen C4 Lounge 16 Hdi 2018 Linea nueva 48000 Unico dueño Acepto meno
r valor 2018 48000
16500 Mini Cooper Countryman 2012 2012 52000
3350000 Volkswagen Fox mod 2017 34700 2017 34700
2750000 Fiorino 2018 GNC 2018 97000
2500000 FOCUS 20 EXE TREND 2013 TITULAR 2DO DUEÑO VTV EXCELENTE 2013 1700000
5500000 2013 Toyota Hilux DX Pack 4x2 25TDI 2013 186000
2700000 Ford Ranger limited 4x4 doble cabina año 2007 2007 310000
7300000 VOLKSWAGEN AMAROK ULTIMATE 4X4 MANUL 2016 2016 140000
2955555 Citroen c4 2009 250000
1900000 PEUGEOT 207 FELINE 4 PTAS 2013 88600
1500000 Logan Nafta/GNC Full 08 PERMUTO 2008 143000
MySQL connection is closed
familia@familia-H55M-S2:~/proyecto/sss2/SSSST5$ python3 conexion.py
```

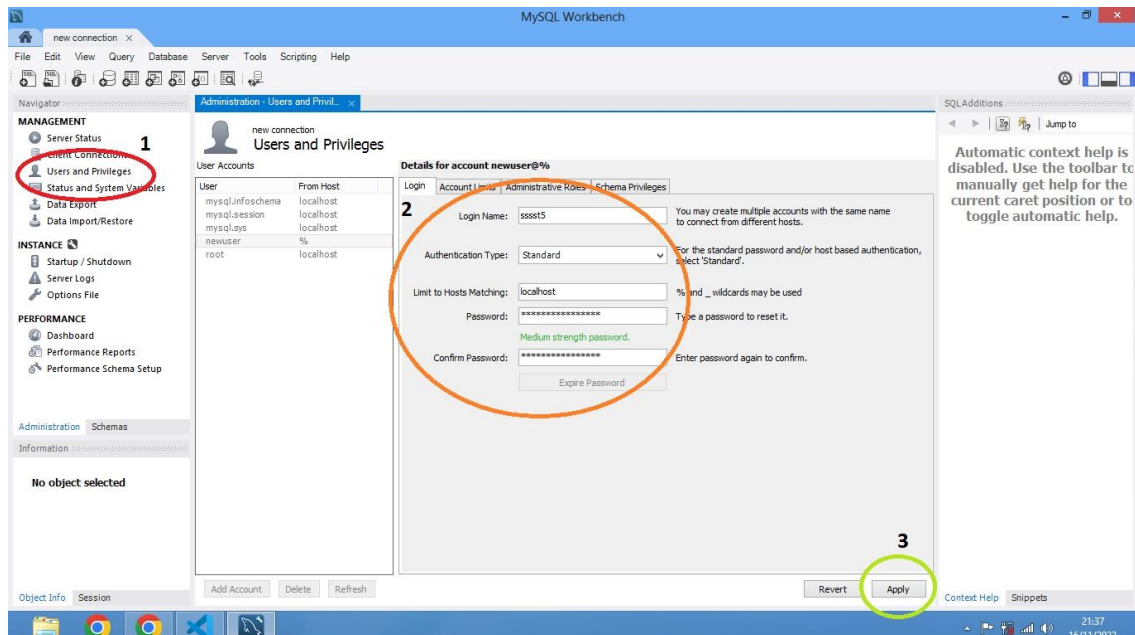
DESARROLLO BASE DE DATOS EN WINDOWS

Para etapa se utilizó MySQL Workbench 8.0 CE, junto con Python 3.8.

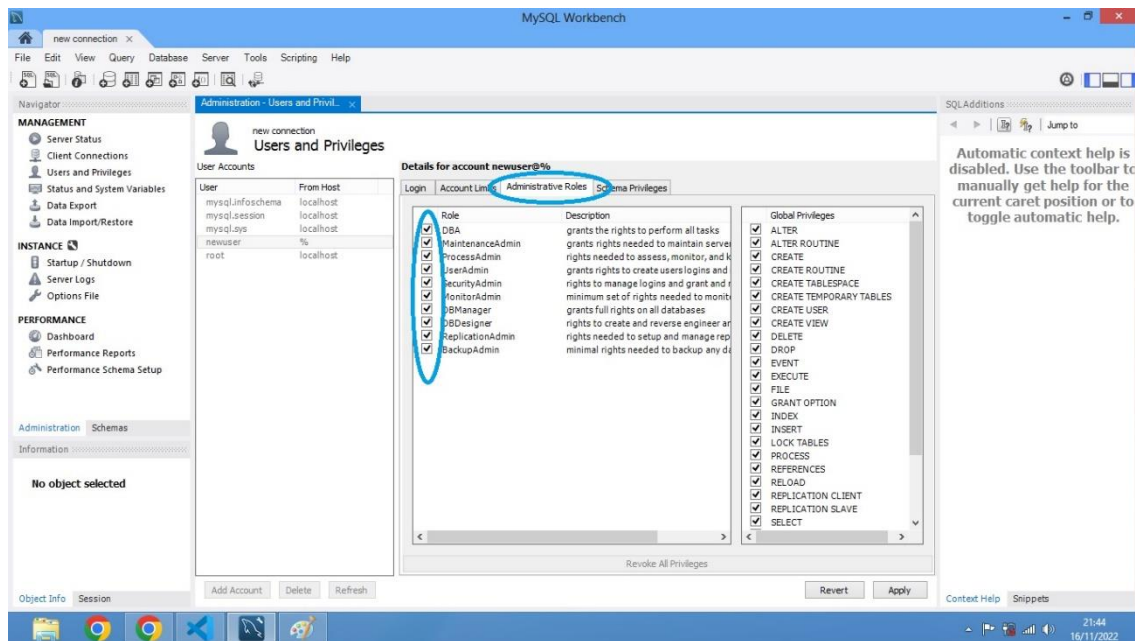
Crear un usuario.

Para lograr esto, en la página principal de “WORKBENCH” sobre el borde izquierdo, encontraremos la opción “Users and Privileges”.

Al hacer clic nos aparecerá un formulario donde podremos poner el nombre del usuario, su “password” y la dirección del host, en este caso, localhost.

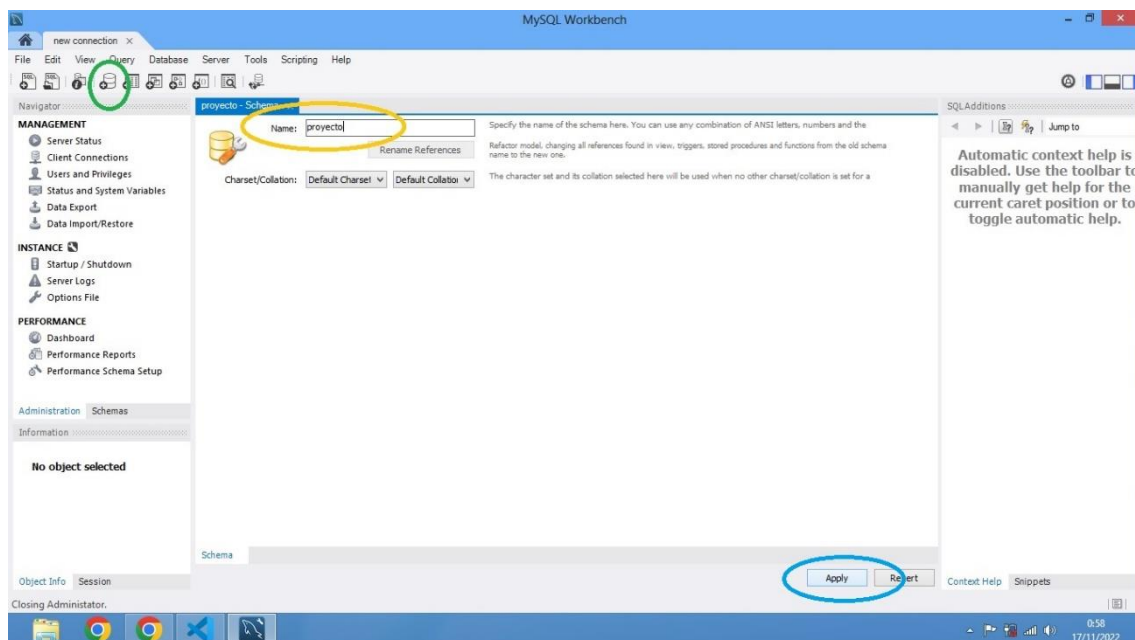


En la solapa “Administrative Roles” le daremos acceso total.



CREAR UN NUEVO SCHEMA

Seleccionamos el botón de nuevo “schema” en la barra de tareas señalado de verde. Ingresamos el nombre que tendrá la base de datos, aplicamos los cambios y nos aparecerá una ventana de confirmación. Aceptamos y listo tendremos la base creada.



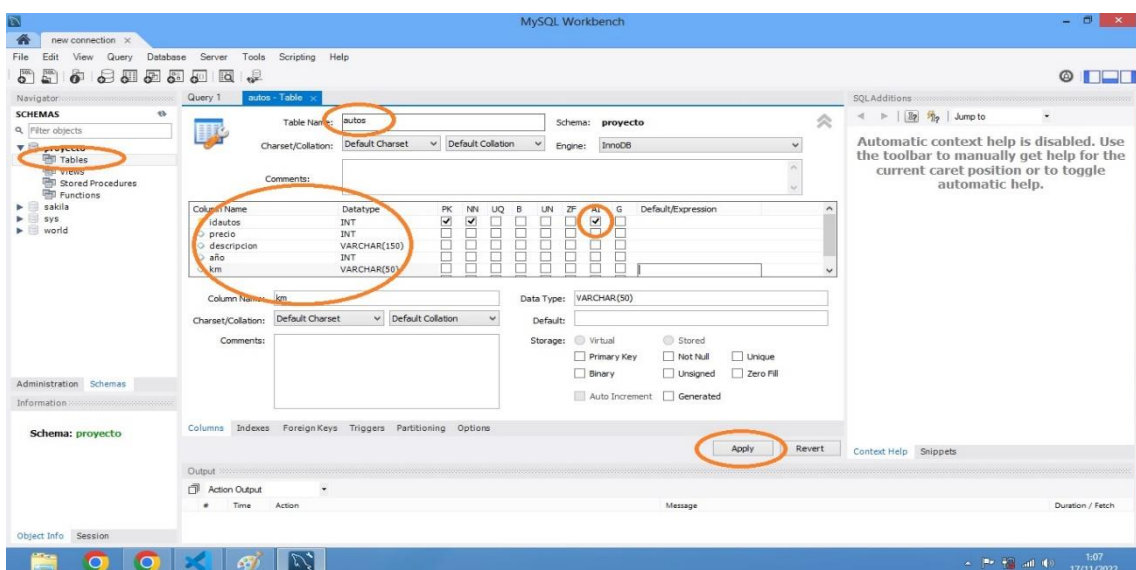
CREANDO UNA TABLA

Una vez creado el proyecto debemos crear la tabla con la que trabajaremos.

Para eso nos ubicamos en la ventana izquierda superior donde dice el nombre de nuestra base de datos, en este caso proyecto.

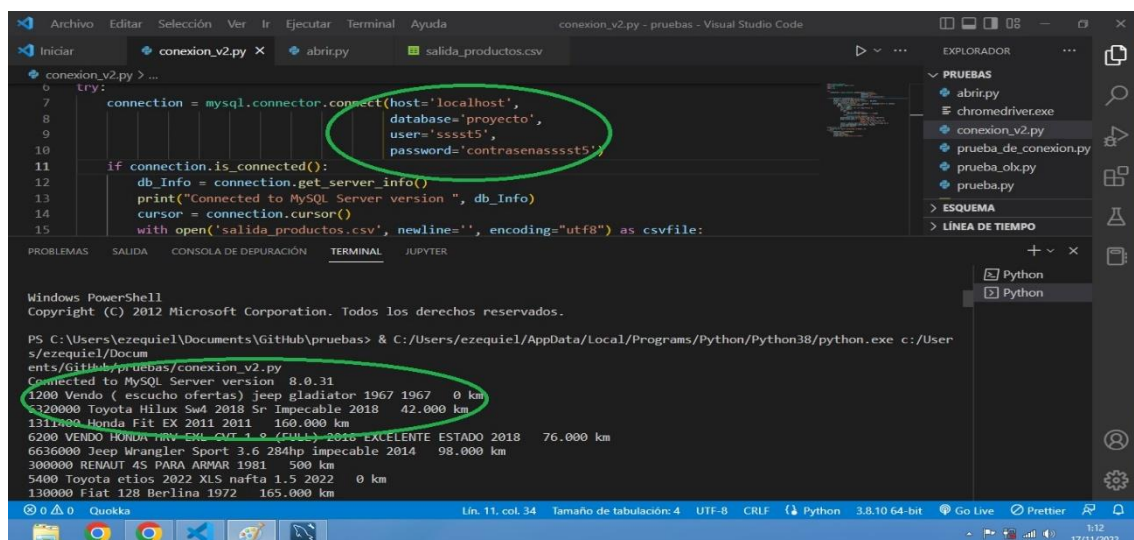
Se despliega un menú, entre ella "Tables", con botón derecho sobre "tables" elegimos crear una tabla.

Se abrirá un formulario en el que debemos completar el nombre de la tabla, indicar el nombre y tipo de datos de las columnas. En mi caso agregue una columna con el nombre "idautos" con el tipo de dato INT y la casilla AI (autoincremento) seleccionado.



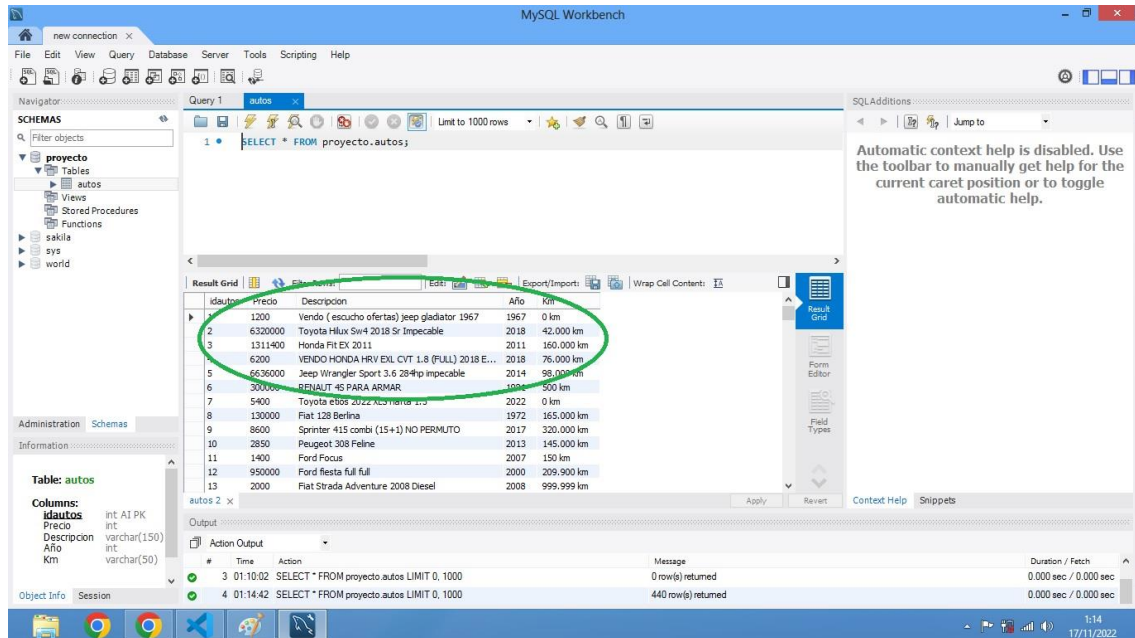
Hacemos uso del comando `SELECT * FROM proyecto.autos;` vemos que la tabla esta vacía.

Hora es necesario ejecutar nuestro programa en Python.



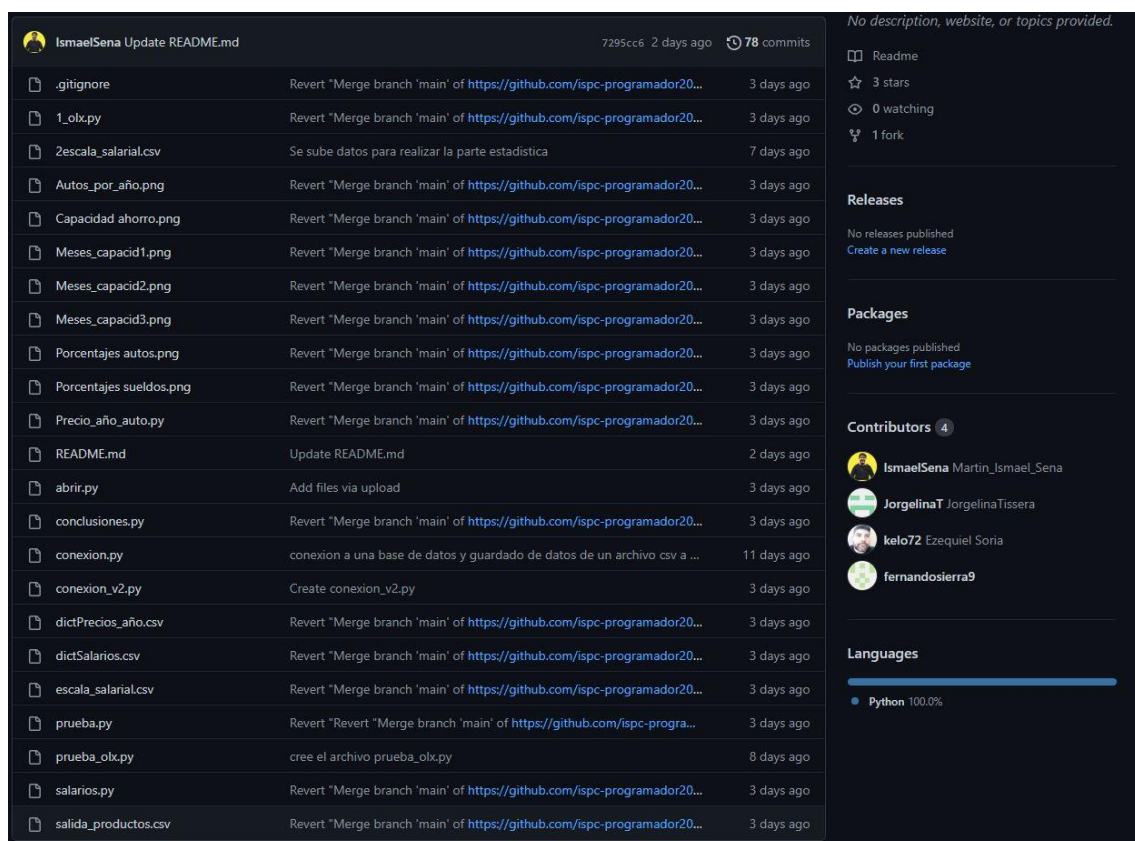
Como podemos ver en el programa de Python se aprecia los datos para poder conectarnos a la base de datos.

En la salida del terminal se aprecia los datos que han sido tomados de un “CSV” producto del Scraping que se realizó de la URL y importados a nuestra base de datos del proyecto.



REPOSITORIO

Toda la información que se obtuvo al igual que el código fuente, los archivos CSV y las gráficas se subieron a un repositorio que se creó dentro del ISPC a través del alias “SSSST5”



Link: <https://github.com/ispc-programador2022/SSSST5>

CONCLUSIONES FINALES

Con toda la información recabada se analiza finalmente cuantos meses demorará una persona en poder adquirir un vehículo dependiendo de su capacidad de ahorro como así también del precio promedio de un automóvil según el rango de precios que se estableció a continuación:

```
print("\nConclusiones de nuestro análisis:")
print("\n Capacidad de ahorro de $85717, sueldos entre 150000y 300000")
tiempo_primer_intervalo1 = round(1113762/85171)
print("Para adquirir un auto de valor menor a 1.5M se demorará: ", tiempo_primer_intervalo1, " meses")
tiempo_primer_intervalo2 = round(1998217/85171)
print("Para adquirir un auto de valor comprendido entre 1.5M y 2.5M se demorará: ", tiempo_primer_intervalo2, " meses")
tiempo_primer_intervalo3 = round(3008308/85171)
print("Para adquirir un auto de valor comprendido entre 2.5M y 3.5M se demorará: ", tiempo_primer_intervalo3, " meses")
tiempo_primer_intervalo4 = round(4228707/85171)
print("Para adquirir un auto de valor comprendido entre 3.5M y 5M se demorará: ", tiempo_primer_intervalo4, " meses")
tiempo_primer_intervalo5 = round(5809771/85171)
print("Para adquirir un auto de valor comprendido entre 5M y 6.5M se demorará: ", tiempo_primer_intervalo5, " meses")
tiempo_primer_intervalo6 = round(7232857/85171)
print("Para adquirir un auto de valor mayor 6.5M se demorará: ", tiempo_primer_intervalo6, " meses")

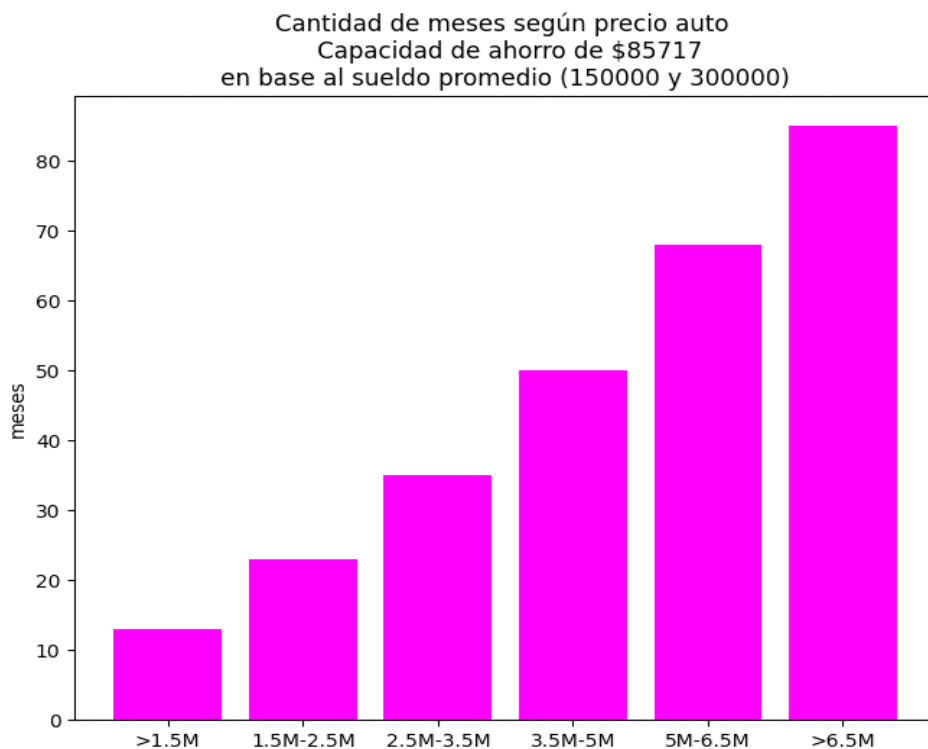
print("\n Capacidad de ahorro de $253467, sueldos entre 300000 y 500000 ")
tiempo_segundo_intervalo1 = round(1113762/253467)
print("Para adquirir un auto de valor menor a 1.5M se demorará: ", tiempo_segundo_intervalo1, " meses")
tiempo_segundo_intervalo2 = round(1998217/253467)
print("Para adquirir un auto de valor comprendido entre 1.5M y 2.5M se demorará: ", tiempo_segundo_intervalo2, " meses")
tiempo_segundo_intervalo3 = round(3008308/253467)
print("Para adquirir un auto de valor comprendido entre 2.5M y 3.5M se demorará: ", tiempo_segundo_intervalo3, " meses")
tiempo_segundo_intervalo4 = round(4228707/253467)
print("Para adquirir un auto de valor comprendido entre 3.5M y 5M se demorará: ", tiempo_segundo_intervalo4, " meses")
tiempo_segundo_intervalo5 = round(5809771/253467)
print("Para adquirir un auto de valor comprendido entre 5M y 6.5M se demorará: ", tiempo_segundo_intervalo5, " meses")
tiempo_segundo_intervalo6 = round(7232857/253467)
print("Para adquirir un auto de valor mayor 6.5M se demorará: ", tiempo_segundo_intervalo6, " meses")

print("\n Capacidad de ahorro de $718947 , sueldos mayores a 500000")
tiempo_tercer_intervalo1 = round(1113762/718947)
print("Para adquirir un auto de valor menor a 1.5M se demorará: ", tiempo_tercer_intervalo1, " meses")
tiempo_tercer_intervalo2 = round(1998217/718947)
print("Para adquirir un auto de valor comprendido entre 1.5M y 2.5M se demorará: ", tiempo_tercer_intervalo2, " meses")
```

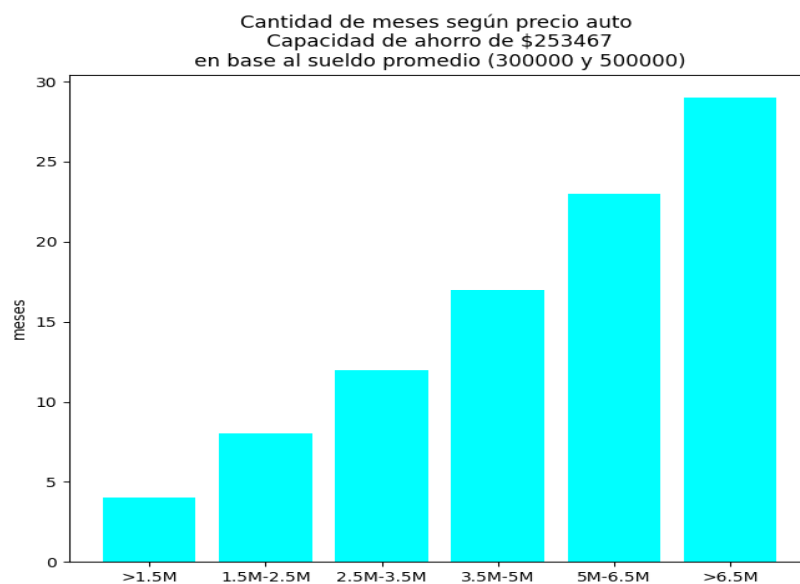
Se realizan las gráficas que permitan ver de forma más amigable las conclusiones a las que arribamos:

```
conclusiones.py > _
51
52 #Módulos:
53
54 Intervalo1=[tiempo_primer_intervalo1,tiempo_primer_intervalo2,tiempo_primer_intervalo3,tiempo_primer_intervalo4,tiempo_primer_intervalo5,tiempo_primer_intervalo6]
55 nombres=["<1.5M","1.5M-2.5M","2.5M-3.5M","3.5M-5M","5M-6.5M",">6.5M"]
56 fig, ax=plt.subplots()
57 #Etiqueta en eje Y
58 ax.set_ylabel("meses")
59 #Etiqueta en eje X
60 ax.set_title("Cantidad de meses según precio auto \n Capacidad de ahorro de $85717\n en base al sueldo promedio (150000 y 300000) ")
61 plt.bar(nombres, Intervalo1, color="magenta")
62 plt.savefig("Meses_capacid1.png")
63 plt.show()
64
65
66 Intervalo2=[tiempo_segundo_intervalo1,tiempo_segundo_intervalo2,tiempo_segundo_intervalo3,tiempo_segundo_intervalo4,tiempo_segundo_intervalo5,tiempo_segundo_intervalo6]
67 nombres=["<1.5M","1.5M-2.5M","2.5M-3.5M","3.5M-5M","5M-6.5M",">6.5M"]
68 fig, ax=plt.subplots()
69 #Etiqueta en eje Y
70 ax.set_ylabel("meses")
71 #Etiqueta en eje X
72 ax.set_title("Cantidad de meses según precio auto \n Capacidad de ahorro de $253467 \n en base al sueldo promedio (300000 y 500000) ")
73 plt.bar(nombres, Intervalo2, color="cyan")
74 plt.savefig("Meses_capacid2.png")
75 plt.show()
76
77 Intervalo3=[tiempo_tercer_intervalo1,tiempo_tercer_intervalo2,tiempo_tercer_intervalo3,tiempo_tercer_intervalo4,tiempo_tercer_intervalo5,tiempo_tercer_intervalo6]
78 nombres=["<1.5M","1.5M-2.5M","2.5M-3.5M","3.5M-5M","5M-6.5M",">6.5M"]
79 fig, ax=plt.subplots()
80 #Etiqueta en eje Y
81 ax.set_ylabel("meses")
82 #Etiqueta en eje X
83 ax.set_title("Cantidad de meses según precio auto \n Capacidad de ahorro de $ 718947\n en base al sueldo promedio (>500000) ")
84 plt.bar(nombres, Intervalo3, color="salmon")
85 plt.savefig("Meses_capacid3.png")
86 plt.show()
87
```

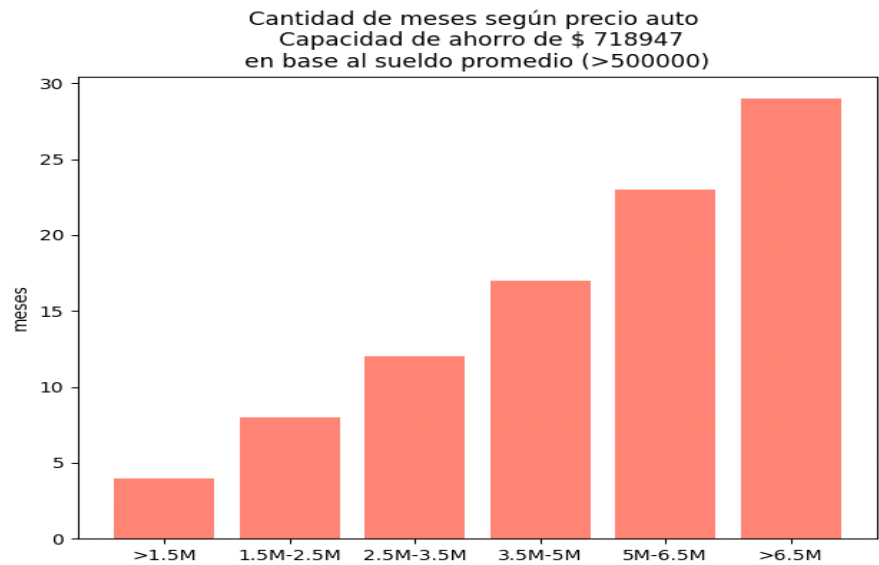
Gráfica1- Muestra la cantidad de meses que demora una persona que tiene una capacidad de ahorro mensual de \$85717, (percibe un salario entre \$150000 y \$300000) según el valor promedio según el rango de precios del vehículo que quiera comprar:



Gráfica2- Muestra la cantidad de meses que demora una persona que tiene una capacidad de ahorro mensual de \$253467, (percibe un salario entre \$300000 y \$500000) según el valor promedio según el rango de precios del vehículo que quiera comprar:



Gráfica3- Muestra la cantidad de meses que demora una persona que tiene una capacidad de ahorro mensual de \$718947, (percibe un salario mayor \$500000) según el valor promedio según el rango de precios del vehículo que quiera comprar:



MATERIAL DE CONSULTA

Links:

GITHUB

 [Como crear un Repositorio y Subir Proyecto a !\[\]\(b89ecf30df3dbaee65fa9f1829524a6e_img.jpg\) GITHUB !\[\]\(12caa8c16ee33cc266cee3a47dfba46b_img.jpg\) Paso a Paso !\[\]\(2137b87161c99f1e992a818823b2a5a3_img.jpg\)](#)

WEB SCRAPING

[Cómo hacer scrape de sitios web con Python 3](#)

[WEB SCRAPING \(MUY FACIL !\[\]\(c694a3ff3b077d76910920a6a1593ab4_img.jpg\) \) | PYTHON. Web scraping con Python en Español.](#)

[Web Scraping with Python - BeautifulSoup Crash Course](#)

[Extrayendo datos de Mercado Libre con Python | Web Scraping PARTE 8](#)

LIBRERIAS

<https://selenium-python.readthedocs.io/>

BASE DE DATOS

<https://docs.python.org/3/library/sqlite3.html>

[Query Language Understood by SQLite](#)

https://www.youtube.com/watch?v=31DuU-98XtY&t=377s&ab_channel=Matoosfe

[Creando una tabla de sql a partir de un csv desde Python](#)