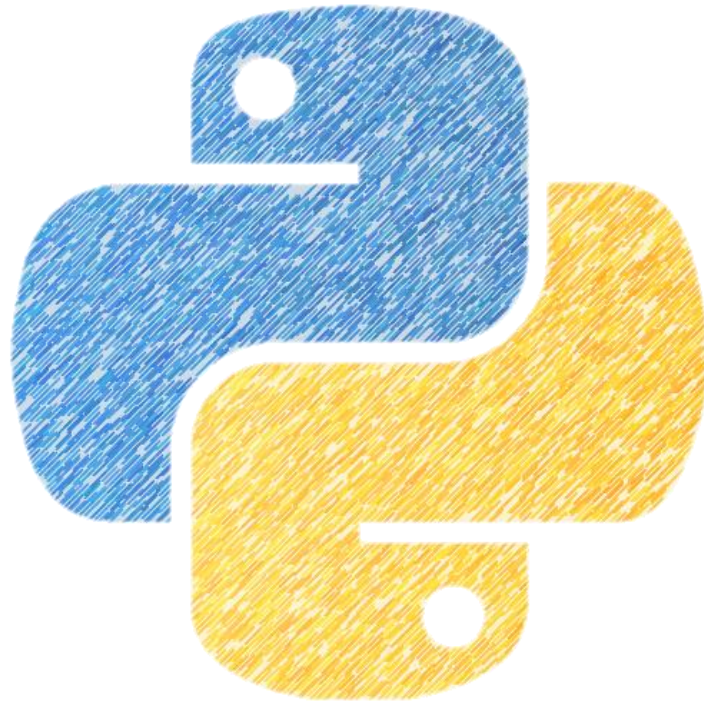


Proyecto 1

Introducción a Python

Francisco Morales Figueroa



05/09/2020



Contenido

Introducción	3
Definición del código	4
Librerías	4
Menú principal	4
Inicio de sesión	5
Menú secundario	6
Ventas de productos	8
Conteo de ventas	8
Exportación	9
Ordenamiento	9
Impresión	10
Búsquedas de productos	11
Mejores y peores opiniones	13
Finanzas	16
Análisis por categoría	19
Menú categorías	19
Solución al problema	22
Conclusiones	26



Introducción

Este reporte es producto de un análisis de caso muy cercano a uno real con el cuál tras analizar datos se pueden identificar puntos a mejorar en este caso de un negocio.

El caso estudiado es una tienda virtual que se dedica a la venta de productos de computación y electrónica. Tras un periodo establecido se ha recopilado información sobre los productos como su id, nombre, precio, etc., información sobre las ventas y también sobre la búsqueda sobre los mismos.

Existen dos grandes razones que dan sustento a la creación del presente proyecto y su reporte técnico. El primer motivo es poner en práctica los conocimientos adquiridos en el curso y desarrollar habilidades que permitan mejorar nuestra interpretación de problemas para llevarlos a nivel de código. Y la segunda es dimensionar la aplicabilidad de los conocimientos en casos de negocio reales que pueden ser llevados de la interpretación de negocios a la conversión de sus datos en activos tangibles para el negocio.

El reporte se estructura de manera que es posible atender las dos necesidades generales del proyecto. Se da espacio para detallar el proceso técnico de solución describiendo la lógica de programación y la finalidad de cada elemento empleado. Y también se da espacio a plantear las soluciones a nivel de negocio presentando resultados que pueden ser considerados gerenciales y que dan pie a la toma de decisiones en pro del negocio.

Para la solución del problema es necesario ir de lo general (problema grande) a lo particular (problema pequeño) para que la solución individual de tareas específicas al momento de integrarse den solución a una tarea general cumpliendo las necesidades globales de la tarea.



Definición del código

Librerías

El código comienza importando 3 librerías que sirven para la visualización de la información mas no implican una intervención en la lógica de solución del problema. Estas tres librerías y las funciones utilizadas respectivamente son:

- Os
 - `os.system("clear")`: Funciona para limpiar pantalla.
- Time
 - `time.sleep(#)`: Funciona para hacer una pausa de # segundos.
- Csv
 - `open()`: Abre un archivo csv para su posible edición.
 - `writer()` y `writerows()`: Sirven para escribir en el archivo

El código comienza con un bucle while que permite la ejecución repetitiva del programa hasta que se cumpla la condición de finalizado (3 errores al iniciar sesión o la instrucción del usuario para terminar la ejecución). Figura 1

```
#Inicio de codigo
ejecutar = 3
while (ejecutar > 0): ...

os.system("clear")
print("Vuelve pronto!")
```

Figura 1

Menú principal

Dentro del while se imprime un menú que indica las opciones de selección para el usuario. Se hace la verificación de la variable ingresada "sesión" en formato `str()` para identificar si se ingresó cualquier otro caracter e indicar el error. La figura 2 muestra el código y la Figura 3 muestra la visualización en la terminal Al ingresar como usuario o administrador se entra a la parte de inicio de sesión.

- Opción 1: Ingresar como administrador
- Opción 2: Ingresar como usuario
- Opción 0: Salir



```
time.sleep(.3)
os.system('clear')
print("Bienvenido. Presione: \n 1--Para ingresar como
administrador \n 2--Para ingresar como usuario \n
0--Para salir")
sesion = input()
+ if str(sesion) == "1": ...
+ elif str(sesion) == "2": ...
+ elif str(sesion) == "0": ...

else:
    print("\n\nOpción incorrecta. Elija una opción
    valida")
    time.sleep(0.8)
```

Figura 2

```
Bienvenido. Presione:
 1--Para ingresar como administrador
 2--Para ingresar como usuario
 0--Para salir
█
```

Figura 3

Inicio de sesión

En el inicio de sesión se hacen dos verificaciones de ingreso. Primero se solicita ingresar un correo, para que el correo sea válido debe contener un @, si no se cumple la condición se indican las oportunidades restantes y se regresa al menú principal.

Si el correo ingresado es válido se procede a el ingreso y validación de una contraseña guardada en la variable "contraseña". El requisito para que la contraseña sea válida es que tenga una longitud entre 4 y 10 caracteres.

Para estas validaciones simplemente se utilizan dos if (uno dentro de otro) y al no cumplirse ya sea el primer paso o el segundo se hace una disminución en el número de intentos disponibles durante la ejecución del programa. Figuras 4 y 5

Si el usuario y contraseña son correctas se pasa al menú 2 del programa.



```
os.system('clear')
print("Bienvenido. Ingrese su correo para iniciar
sesión (Debe contener un @ para ser válido):")
usuario = input()

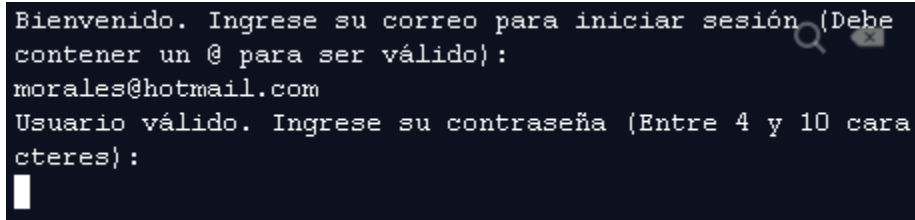
if "@" not in usuario:
    print("Correo incorrecto (tiene " + str
(ejecutar-1) + " intentos más)\n\n")
    ejecutar -= 1

else:
    print("Usuario válido. Ingrese su contraseña
(Entre 4 y 10 caracteres):")
    contraseña = input()

    if len(contraseña) < 4 or len(contraseña) > 10:
        print("Contraseña incorrecta (tiene " + str
(ejecutar-1) + " intentos más)\n\n")
        ejecutar -= 1

    else:
```

Figura 4



```
Bienvenido. Ingrese su correo para iniciar sesión (Debe
contener un @ para ser válido):
morales@hotmail.com
Usuario válido. Ingrese su contraseña (Entre 4 y 10 carac
teres):
█
```

Figura 5

Menú secundario

Nota: Al menú secundario se ingresa solo como administrador. Si se ingresa como usuario simplemente se muestra un texto y se redirige al menú inicial

La primera característica importante de este segundo menú es que tiene un propio ciclo iterativo while que permite al usuario mantenerse en el tras realizar alguna interacción sin tener que volver al menú inicial e iniciar sesión nuevamente. La condición que interrumpe la iteración del flujo es la interrupción del usuario al ingresar un "0". Figura 6.



```
opcion = 1
while (opcion != 0): ...
ejecutar = 0
```

Figura 6

Se hace una bienvenida personalizada gracias a extraer todos los elementos antes del “@” en el usuario y posteriormente se imprimen las opciones que tiene el usuario para ingresar. Figura 7 y Figura 8.

Se tiene la misma validación que en el menú principal por si el usuario ingresa algún carácter inválido.

```
time.sleep(0.5)
os.system('clear')
print("Bienvenido " + usuario[0:usuario.index("@")])
print("\nElige una opción:\n 1--Ventas de productos.
\n 2--Busquedas de productos. \n 3--Mejores y peores
opiniones. \n 4--Finanzas. \n 5--Análisis por
categoria. \n 0--Salir. \n\n")

opcion = input("Ingresa el numero de la opción
deseada:")

if str(opcion) == "0":
    | opcion = 0

elif str(opcion) == "1": ...

elif str(opcion) == "2": ...

elif str(opcion) == "3": ...

elif str(opcion) == "4": ...

elif str(opcion) == "5": ...

else: ...
```

Figura 7



```
Bienvenido morales

Elige una opción:
1--Ventas de productos.
2--Busquedas de productos.
3--Mejores y peores opiniones.
4--Finanzas.
5--Análisis por categoría.
0--Salir.

Ingresa el número de la opción deseada: █
```

Figura 8

Ventas de productos

Para el análisis de productos con mayores y menores ventas el código se divide en 4 secciones que solucionan tareas específicas. Estas son:

1. Conteo de ventas
2. Exportación
3. Ordenamiento
4. Impresión

Conteo de ventas

En esta sección la problemática a resolver es lograr unir la lista de productos con la de ventas. Para esto el elemento clave es el id del producto que se encuentra en ambas listas, con este se puede atribuir una venta a determinado producto.

Esta sección cuenta con dos for anidados y un condicional dentro de ellos. El primer for sirve para recorrer toda la lista de productos, dentro de este for se declara una variable auxiliar con las propiedades de la lista iterada (id, nombre, precio, categoría, 0, 0, 0, stock), además de algunos campos con valor de "0" a los cuales se les agregan características de la lista de compras.

El segundo ciclo for sirve para iterar la lista de ventas. Dentro de estos dos for se encuentra un condicional que compara los campos id de producto en ambas listas. En caso de coincidir se hace un incremento unitario al registro de cantidad de ventas, además como un incremento de valor calificación en otro registro y finalmente otro incremento de valor devoluciones en la variable temporal.

Al finalizar el primer ciclo for se agrega el valor final de la variable temporal a la lista final de ventas y se le hace una reasignación para repetir el proceso. Figura 9.



```
#Calculo de ventas (Conteo de ventas)
ventas = []
for producto in lifestore_products:
    registro = [producto[0],producto[1],producto[2],
                producto[3],0,0,0,producto[4]]
    for venta in lifestore_sales:
        if producto[0] == venta[1]:
            registro[4] += 1
            registro[5] += venta[2]
            registro[6] += venta[4]
    ventas.append(registro)
    registro=[]
```

Figura 9

Exportación

Una vez generada esta lista de ventas se procede a exportarla a un archivo csv para poder generar los gráficos de la siguiente sección. En esta parte del código se abre un archivo csv para su edición "w" y se sobrescribe con la variable que contiene nuestra información de ventas. Figura 10.

```
#exportar csv con ventas
mi_archivo = open("datos.csv","w")
with mi_archivo:
    writer = csv.writer(mi_archivo)
    writer.writerows(ventas)
print("Exportacion completa")
```

Figura 10

Ordenamiento

Para ordenar la lista generada de ventas se utilizó el código visto en las asesorías el cual mediante un ciclo while asigna a dos variables temporales el valor de todo el elemento y la cantidad de ventas del producto. Con estas variables y con ayuda de un condicional dentro de un ciclo for se recorre toda la lista de ventas para descubrir si existe un producto con una venta menor y hacer la reasignación. Al encontrar el producto con menor venta se guarda en una nueva lista y se elimina de la lista inicial. El proceso se repite hasta quedar todos los elementos ordenados y la lista inicial vacía, saliendo así del ciclo while. Figura 11.



```
#Ordenamiento de ventas por producto
ventas_producto = ventas
ventas_ordenadas = []
while ventas_producto:
    minimo = ventas_producto[0][4]
    lista_actual = ventas_producto[0]
    for grupo in ventas_producto:
        if grupo[4] < minimo:
            minimo = grupo[4]
            lista_actual = grupo

    ventas_ordenadas.append(lista_actual)
    ventas_producto.remove(lista_actual)
```

Figura 11

Impresión

La ultima parte hace la impresión de los elementos ordenados. Para esto se le solicita al administrador ingresar el numero de productos que desea ver (mejores y peores).

Para la impresión se utiliza un ciclo for para recorrer la lista con los productos ordenados. Dentro de este for se encuentra un condicional que limita la impresión al “numero” de productos que desea ver el usuario. La condición cambia es distinta para las mayores ventas que para las menores ventas.

Una vez determinado que se encuentra en el rango de impresión se hace un barrido del nombre del producto con un ciclo while hasta encontrar una “,”. Esto sirve para obtener únicamente el nombre del producto sin sus características.

Finalmente se hace la impresión que incluye la posición general de ventas, el nombre del producto y la cantidad de ventas. Figura 12 y 13.

```
n = 0
for venta in ventas_ordenadas:
    n += 1
    if (len(ventas_ordenadas)- n) < int(numero):
        nombre = venta[1]
        i = 0
        while nombre[i] != ",":
            i += 1
        print("\t" + str(len(ventas_ordenadas)- n + 1) +
              ".-" + str(nombre[0:i]) + " con " + str(venta[4])
              + " ventas" )
```

Figura 12



```
Los 3 productos mas vendidos son:

3.-Procesador Intel Core i3-9100F con 20 ventas
2.-Procesador AMD Ryzen 5 2600 con 42 ventas
1.-SSD Kingston A400 con 50 ventas

Los 3 productos menos vendidos son:

96.-Procesador Intel Core i3-8100 con 0 ventas
95.-Tarjeta de Video EVGA NVIDIA GeForce GT 710 con 0 ventas
94.-Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gam
ing con 0 ventas

Presione cualquier tecla para regresar
```

Figura 13

Al haberse impreso los “n” productos mas vendidos y los “n” menos vendidos se deja al usuario mantenerse en pantalla hasta presionar alguna tecla. Esto simplemente se hace mediante un input a una variable que no se utiliza en otra parte del programa.

```
regresar= input("\nPresione cualquier tecla para regresar")
```

Búsquedas de productos

Para conocer los productos más y menos buscados se hacen procesos análogos a los realizados en el subtema anterior. Por lo que describiré a grandes rasgos los pasos y se anexan figuras que muestren el código de los pasos indicados.

1. Primero se realiza una unión entre la lista de productos con la de búsquedas mediante el id del producto. Figura 14.
2. Se hace la exportación de la lista anterior para poder visualizar los datos y obtener conclusiones más rápido. Figura 15.
3. Se ordenan los productos de menor a mayor búsqueda utilizando el mismo método que en la sección anterior, siendo la variable de comparación el campo que tiene el número de búsquedas por producto. Figura 16.
4. Se imprimen los # productos más buscados y los # productos menos buscados, según la instrucción del usuario. La impresión contiene la posición general del producto, su nombre recortado (sin características) y el número de búsquedas. Figura 17.

Un ejemplo de la impresión final se muestra en la Figura 18.



```
#Calculo de busquedas (Conteo de busquedas)
busquedas = []
registro = []
for producto in lifestore_products:
    registro = [producto[0],producto[1],producto
    [2],producto[3],producto[4],0]
    for busqueda in lifestore_searches:
        if producto[0] == busqueda[1]:
            registro[5] += 1
    busquedas.append(registro)
    registro=[]
```

Figura 14

```
#exportar csv con busquedas
mi_archivo = open("datos.csv","w")
with mi_archivo:
    writer = csv.writer(mi_archivo)
    writer.writerow(busquedas)
    print("Exportacion completa")
```

Figura 15

```
#Ordenamiento de busquedas por producto
busquedas_producto = busquedas
busquedas_ordenadas = []
minimo_busqueda = 0
lista_actual_busqueda = []
while busquedas_producto:
    minimo_busqueda = busquedas_producto[0][5]
    lista_actual_busqueda = busquedas_producto[0]
    for grupo in busquedas_producto:
        if grupo[5] < minimo_busqueda:
            minimo_busqueda = grupo[5]
            lista_actual_busqueda = grupo

    busquedas_ordenadas.append(lista_actual_busqueda)
    busquedas_producto.remove(lista_actual_busqueda)
```

Figura 16



```
#impresion de campos
os.system("clear")
numero = input("¿Cuántos productos desea ver?")
print ("\nLos " +str(numero) + " productos mas
buscados son:\n")

n = 0
for busqueda in busquedas_ordenadas:
    n += 1
    if (len(busquedas_ordenadas)- n) < int(numero):
        nombre = busqueda[1]
        i = 0
        while nombre[i] != ",":
            i += 1
        print("\t" + str(len(busquedas_ordenadas)- n + 1)
            + ".-" + str(nombre[0:i]) + " con " + str
            (busqueda[5]) + " busquedas" )

print ("\nLos " + str(numero) + " productos menos
buscados son:\n")
```

Figura 17

```
¿Cuántos productos desea ver?3
Los 3 productos mas buscados son:

3.-Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING con 60 busquedas
2.-SSD Adata Ultimate SU800 con 107 busquedas
1.-SSD Kingston A400 con 263 busquedas

Los 3 productos menos buscados son:

96.-Tarjeta de Video EVGA NVIDIA GeForce GT 710 con 0 busquedas
95.-Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming con 0 busquedas
94.-Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile con 0 busquedas

Presione cualquier tecla para regresar
```

Figura 18

Mejores y peores opiniones

Para conocer los productos mayor y menor calificación se hacen procesos análogos a los realizados en los subtemas anteriores. A grandes rasgos los para la solución de este punto son.

1. Primero se realiza una unión entre la lista de productos con la de ventas mediante el id del producto. Este paso es exactamente el mismo que el de ventas, sin embargo, se repite en esta sección porque el cálculo se hace



hasta haber ingresado a la opción de ventas, por lo que si se ingresara antes a las calificaciones esta lista no estaría creada aún. Figura 19.

2. En este caso no se hace la exportación porque sería idéntica a la generada en ventas y como se realiza únicamente para fines del reporte es innecesario obtener dos extracciones idénticas.
3. Un paso que precede al ordenamiento es la obtención de la calificación promedio. Debido a que en el paso 1 las calificaciones únicamente se sumaron es necesario realizar un promedio de calificación considerando el número de ventas. Esto se hace mediante un ciclo for que recorre todo el registro, también se utiliza un condicional para no hacer divisiones en productos con 0 ventas. Figura 20.
4. Una vez obtenida la calificación promedio se puede hacer el ordenamiento mediante este mismo campo. El proceso es exactamente el mismo que en las secciones anteriores y se obtiene un ordenamiento de menor a mayor. Figura 21.
5. Finalmente se imprimen los # productos mejor calificados y los # productos peor calificados, según la instrucción del usuario. La impresión contiene la posición general del producto, su nombre recortado (sin características) y la calificación promedio. Figura 22.

Un ejemplo de la impresión final se muestra en la Figura 23.

```
#Calculo de ventas (Conteo de ventas)
ventas = []
registro = []
for producto in lifestore_products:
    registro = [producto[0],producto[1],producto[2],
    producto[3],0,0,0]
    for venta in lifestore_sales:
        if producto[0] == venta[1]:
            registro[4] += 1
            registro[5] += venta[2]
            registro[6] += venta[4]
    ventas.append(registro)
    registro=[]
```

Figura 19



```
#Cálculo de calificación promedio
resenas_producto = []
for resena in ventas:
    if resena[4] != 0:
        resena[5] = round(resena[5]/resena[4],2)
    resenas_producto.append(resena)
```

Figura 20

```
#Ordenamiento de reseñas por producto
resenas_ordenadas = []
while resenas_producto:
    minimo_resenas = resenas_producto[0][5]
    lista_actual_resenas = resenas_producto[0]
    for grupo in resenas_producto:
        if grupo[5] < minimo_resenas:
            minimo_resenas = grupo[5]
            lista_actual_resenas = grupo

    resenas_ordenadas.append(lista_actual_resenas)
    resenas_producto.remove(lista_actual_resenas)
```

Figura 21

```
#impresion de campos
os.system("clear")
numero = input("¿Cuántos productos desea ver?")
print ("\nLos " + str(numero) + " productos con mejor
calificación son:\n")

n = 0
for resena in resenas_ordenadas:
    n += 1
    if (len(resenas_ordenadas)- n) < int(numero):
        nombre = resena[1]
        i = 0
        while nombre[i] != ",":
            i += 1
        print("\t" + str(len(resenas_ordenadas)- n + 1) +
            ".-" + str(nombre[0:i]) + " con " + str(resena[5])
            + " de calificacion y " + str(resena[6]) + "
            devoluciones.")

print ("\nLos " + str(numero) + " productos con menor
calificacion son:\n")
```

Figura 22



```
¿Cuántos productos desea ver?2
Los 2 productos con mejor calificación son:
    2.-Logitech Audifonos Gamer G332 con 5.0 de calificacion y 0 devoluciones.
    1.-Logitech Audifonos Gamer G635 7.1 con 5.0 de calificacion y 0 devoluciones.
Los 2 productos con menor calificacion son:
    96.-Procesador Intel Core i3-8100 con 0 de calificacion y 0 devoluciones
    95.-Tarjeta de Video EVGA NVIDIA GeForce GT 710 con 0 de calificacion y 0 devoluciones
Presione cualquier tecla para regresar
```

Figura 23

Finanzas

Para obtener las finanzas de la tienda primero se hace el registro de ventas como en la sección de ventas y búsquedas. La diferencia con las agrupaciones anteriores es que en esta lista se agregan campos que mediante el slicing de la fecha de venta separa los meses y los años en campos independientes. Figura 24.

```
#Calculo de finanzas (Conteo de ventas)
finanzas = []
for venta in lifestore_sales:
    registro = [venta[0],venta[1],venta[3],0,[],[]]
    for producto in lifestore_products:
        if producto[0] == venta[1]:
            fecha = registro[2]
            registro[3] += producto[2]
            registro[4] = fecha[3:5]
            registro[5] = fecha[6:]
    finanzas.append(registro)
    registro=[]
```

Figura 24

Posteriormente se hace la extracción de la lista de finanzas en un archivo csv para poder graficar los resultados. Figura 25.

```
#exportar csv con finanzas
mi_archivo = open("datos.csv","w")
with mi_archivo:
    writer = csv.writer(mi_archivo)
    writer.writerows(finanzas)
print("Exportacion completa")
```

Figura 25



Posteriormente se hace una agrupación de ventas por mes. Para hacer esto defino una lista “meses” que contiene listas con el número de mes y con un campo con valor 0 que guardará el monto de venta. Se emplea un for anidado con una comparación dentro para saber el mes a que corresponde cada venta y agregar el monto de venta al campo en la lista “meses”. Figura 26.

```
#Agrupacion de ventas por mes
meses = [ ["01",0], ["02",0], ["03",0], ["04",0], ["05",0],
          ["06",0], ["07",0], ["08",0], ["09",0], ["10",0], ["11",0],
          ["12",0]]

n = 0
for mes in meses:
    for finanza in finanzas:
        if mes[0] == finanza[4]:
            meses[n][1] += finanza[3]
    n += 1
```

Figura 26

Para la impresión de las ventas por mes se generaron 3 listas para distinguir el número de días por mes y asociar un nombre al número de mes. Después se recorre la lista que contiene a los meses y el monto total de venta por mes para hacer el promedio dependiendo del número de días que tiene cada uno de los meses. La impresión indica el nombre del mes y la venta promedio por día respectivamente. Figura 27 y 28.

Para calcular las finanzas por año se utiliza una forma que permite identificar los distintos años existentes en nuestro registro. Esto pudo haberse aplicado para los meses también, sin embargo, como es una técnica que se encontró durante el proceso se decidió dejar ambas para mostrar la evolución del pensamiento durante el proyecto. Primero se crea una lista únicamente con los valores de año de cada venta posteriormente se recorre la lista con un ciclo for y con una condición se determina si el año no está contenido en una lista vacía llamada “años_unicos”, de ser así el valor se agrega a esta lista. El proceso se repite hasta recorrer todo el registro de años y encontrar todos los años distintos. Figura 29.

Una vez obtenidos los años distintos se hace una lista que contiene el acumulado de ventas por cada año. Mediante un ciclo que recorre las finanzas y acumula la venta dependiendo del año. Finalmente, para la impresión se utiliza un ciclo for que recorre todos los años únicos y va obteniendo el promedio de venta mensual en dicho año. Se imprime el año y su venta mensual promedio. Figura 30 y 31.



```
mes31 = ["01","03","05","07","08","10","12"]
mes30 = ["04","06","09","11"]
mes_nombre = ["Enero", "Febrero", "Marzo", "Abril",
"Mayo", "Junio","Julio", "Agosto", "Septiembre",
"Octubre", "Noviembre", "Diciembre"]

#Impresion de valores
n = 0
for mes in meses:
    if mes[0] in mes31:
        dinero = round(float(mes[1]/31) , 2)
        print("La venta diaria promedio en " + str(mes_nombre
[n]) + " fue de: $" + str(dinero))
    elif mes[0] in mes30:
        dinero = round(float(mes[1]/30) , 2)
        print("La venta diaria promedio en " + str(mes_nombre
[n]) + " fue de: $" + str(dinero))
    elif mes[0] == "02":
        dinero = round(float(mes[1]/28) , 2)
        print("La venta diaria promedio en " + str(mes_nombre
[n]) + " fue de: $" + str(dinero))
    else:
        print("numero raro")
    n += 1
```

Figura 27

```
Estas son las finanzas de la tienda

La venta diaria promedio en Enero fue de: $3878.61
La venta diaria promedio en Febrero fue de: $3933.54
La venta diaria promedio en Marzo fue de: $5313.84
La venta diaria promedio en Abril fue de: $6443.17
La venta diaria promedio en Mayo fue de: $3109.48
La venta diaria promedio en Junio fue de: $1231.63
La venta diaria promedio en Julio fue de: $869.32
La venta diaria promedio en Agosto fue de: $99.26
La venta diaria promedio en Septiembre fue de: $139.97
La venta diaria promedio en Octubre fue de: $0.0
La venta diaria promedio en Noviembre fue de: $140.3
La venta diaria promedio en Diciembre fue de: $0.0
```

Figura 28



```
#Calculo de finanzas anual
n = 0
años = []
años_unicos = []
for año in finanzas:
    años.append(año[5])
for año in años:
    if año not in años_unicos:
        años_unicos.append(año)
```

Figura 29

```
suma_año = [0]*len(años_unicos)
for año in finanzas:
    suma_año[(años_unicos.index(año[5]))] += año[3]

n = 0
print("\n")
for año in años_unicos:
    promedio = round(int(suma_año[n])/12,2)
    print("La venta mensual promedio el " + str(año) + "
    fue de: $" + str(promedio))
    n += 1
```

Figura 30

```
La venta mensual promedio el 2020 fue de: $62975.75
La venta mensual promedio el 2019 fue de: $350.75
La venta mensual promedio el 2002 fue de: $21.58

Presione cualquier tecla para regresar
```

Figura 31

Análisis por categoría

Menú categorías

El primer paso en ese menú es identificar las categorías diferentes existentes para esto se utiliza el mismo proceso empleado en la detección de años en el apartado de finanzas. Una vez detectados se hace la impresión del menú para que el usuario pueda seleccionar la opción de su interés. El menú como los dos anteriores tiene un ciclo iterativo que permite al usuario permanecer en el hasta dar la instrucción de salida. Figura 32 y 33.



```
#Selección de categoría
opcion_categorias = 1
while opcion_categorias != 0:
    os.system('clear')
    print("\nElige una opción:\n 1--Procesadores. \n
    2--Tarjetas de video. \n 3--Tarjetas madre. \n 4--Discos
    duros. \n 5--Memorias usb. \n 6--Pantallas. \n 7--Bocinas
    \n 8--Audifonos. \n 0--Regresar.\n")
    opcion_categorias = input("Ingresa el número de la opción
    deseada:")
```

Figura 32

```
Elige una opción:
 1--Procesadores.
 2--Tarjetas de video.
 3--Tarjetas madre.
 4--Discos duros.
 5--Memorias usb.
 6--Pantallas.
 7--Bocinas
 8--Audifonos.
 0--Regresar.

Ingresa el número de la opción deseada: 0
```

Figura 33

Al ingresar una opción con la ayuda de condicionales se determina si la opción ingresada es una categoría y se guarda el número seleccionado para realizar los cálculos a esa categoría en específico. Figura 34.

```
if (str(opcion_categorias) == "1" or str(opcion_categorias)
    == "3" or str(opcion_categorias) == "4" or str
    (opcion_categorias) == "5" or str(opcion_categorias) ==
    "6" or str(opcion_categorias) == "7" or str
    (opcion_categorias) == "8"): ...

elif str(opcion_categorias) == "0": ...

else: ...
```

Figura 34

Se realizan los cálculos de mayor venta, menor venta, mayor búsqueda y menor búsqueda para la categoría específica mediante las soluciones empleadas en las secciones correspondientes. Las principales diferencias con los cálculos a nivel producto son:



- Se establece un número fijo de visualización a 5 productos por categoría para que la visualización sea adecuada.
- Se hace un filtro posterior al ordenamiento para obtener los 5 productos dentro de la categoría con mejores ventas, con peores ventas, con mayores búsquedas y menores búsquedas. Figura 35 y 36.

```
#filtro de las categorias
ventas_categoria = []
for venta in ventas_ordenadas:
    if str(venta[3]) == str(categorias_unicas[(int
        (opcion_categorias)-1)]):
        ventas_categoria.append(venta)
```

Figura 35

```
#filtro de las categorias
busquedas_categoria = []
for venta in busquedas_ordenadas:
    if str(venta[3]) == str(categorias_unicas[(int
        (opcion_categorias)-1)]):
        busquedas_categoria.append(venta)
```

Figura 36

Todo lo demás es exactamente igual a los cálculos en secciones anteriores por lo que no es necesario explicarlo de nuevo.



Solución al problema

Para la solución de la consigna de identificar las categorías con mayor y menor venta dependiendo de su categoría se utiliza la extracción de la lista generada en ventas y mediante Tableau es posible presentar visualmente el contenido. Podemos ver en la Figura 37 de manera muy clara que existe una división entre 4 categorías con ventas mayores a 20 productos (procesadores, discos duros, tarjetas madre y tarjetas de video) y las otras 4 se encuentran por debajo inclusive de los 10 productos vendidos (audífonos, pantallas, bocinas y memorias usb).

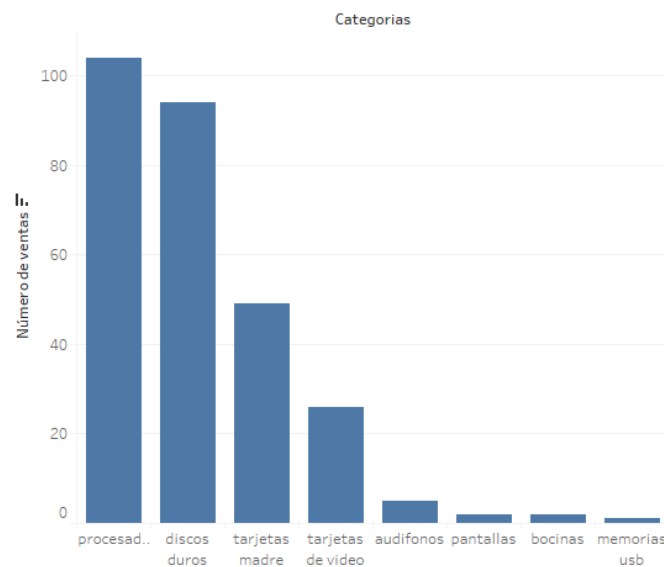


Figura 37

Con la extracción de búsquedas es posible hacer un análisis a nivel categoría de las búsquedas vs el stock por producto.

Un ejemplo es que al graficar para procesadores tenemos que en general el stock va sobrado respecto a las búsquedas. Hay un producto que tiene casi 60 búsquedas y casi 1000 en stock. Figura 38. Esto puede hacerse para cada categoría para ver casos particulares.

Debe considerarse una relación búsqueda/stock para tener un inventario que vaya acorde al interés del producto. Esta relación puede pintarse fácilmente como una recta cuya pendiente depende de la proporción establecida y divide a los productos en tres. Los que tengan más stock que búsquedas según la proporción. Los que tengan más búsquedas que stock y los que estén sobre la recta tienen la proporción óptima.

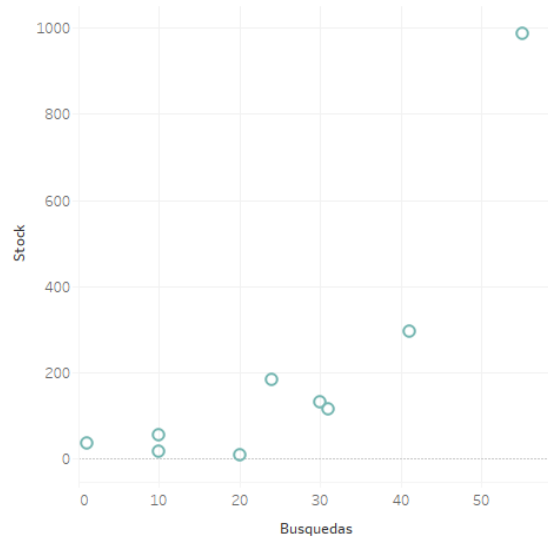


Figura 38

También es posible hacer un análisis general como en la Figura 39 para detectar tendencias generales como la concentración de la mayoría de los productos en un rango de 50 búsquedas y 200 de stock. O detectar puntos particulares como el producto con 1000 piezas en stock y solo 50 búsquedas, el producto con más de 100 búsquedas y prácticamente sin stock y el producto con muchas búsquedas, pero suficiente stock.

El color del círculo corresponde a la categoría por lo que es fácil identificar si hay tendencias por categorías.

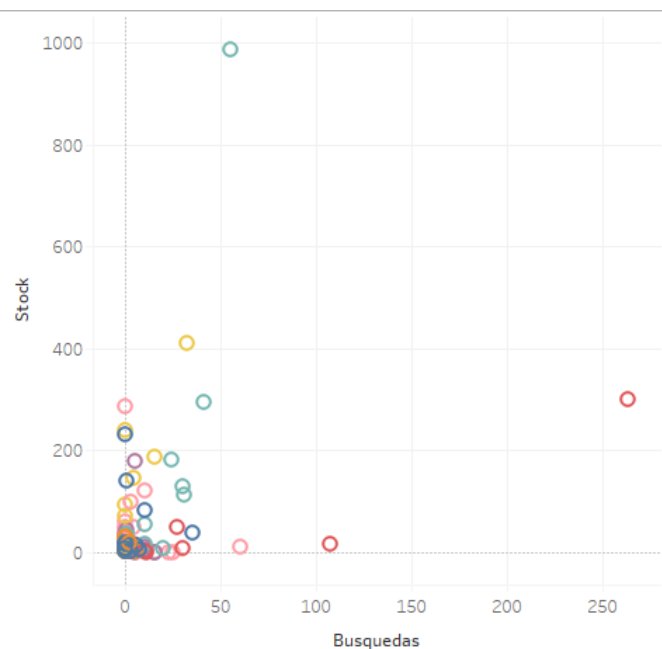


Figura 39



Otro análisis interesante resulta de la extracción de ventas que contiene la información de ventas y stock. A nivel global por categoría se puede observar que existe una tendencia general a tener mayor stock que ventas, sin embargo, hay categorías muy desproporcionadas en ese sentido como los procesadores que pese a ser la categoría con mayor venta presentan un inventario excesivo que representa gastos innecesarios para la empresa. Figura 40.



Figura 40

La calificación promedio también puede graficarse mediante la extracción de ventas. En este caso para tener una idea general se agrupa por categoría y se obtiene un promedio de calificación. Un punto clave aquí es que existen productos que al no tener ventas tampoco tienen reseña por lo que su calificación es 0 y hace que el promedio por categoría descienda considerablemente. Teniendo eso claro es fácil identificar que nuestra categoría con mayores ventas también es la que tiene el promedio de calificación más alto. Y que las memorias usb pese a ser la categoría con menos ventas se encuentra en tercer lugar de mejores calificaciones. Figura 41.

Este análisis se puede hacer a nivel de producto para detectar casos particulares y tomar decisiones en pro del negocio.

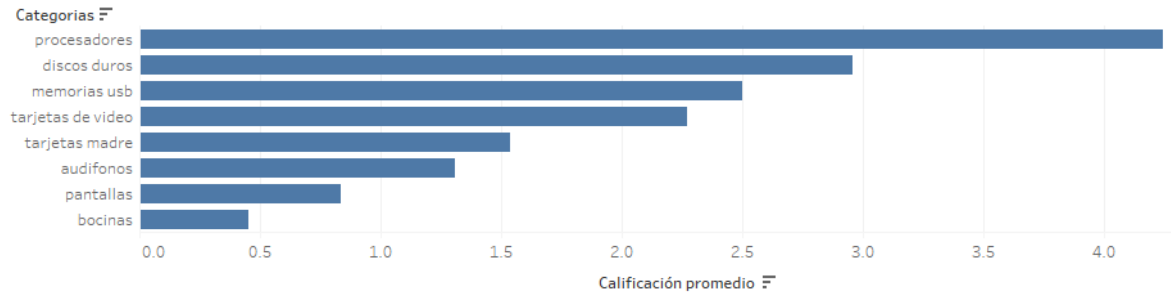


Figura 41

En cuanto a las finanzas gracias a la extracción de esta lista es posible detectar una tendencia de venta muy marcada ya que la mayoría de las ventas se presentan en los primeros meses del año y en el segundo semestre del año estas caen drásticamente. También es posible identificar fácilmente a abril como el mes con mayores ventas, seguido de marzo.

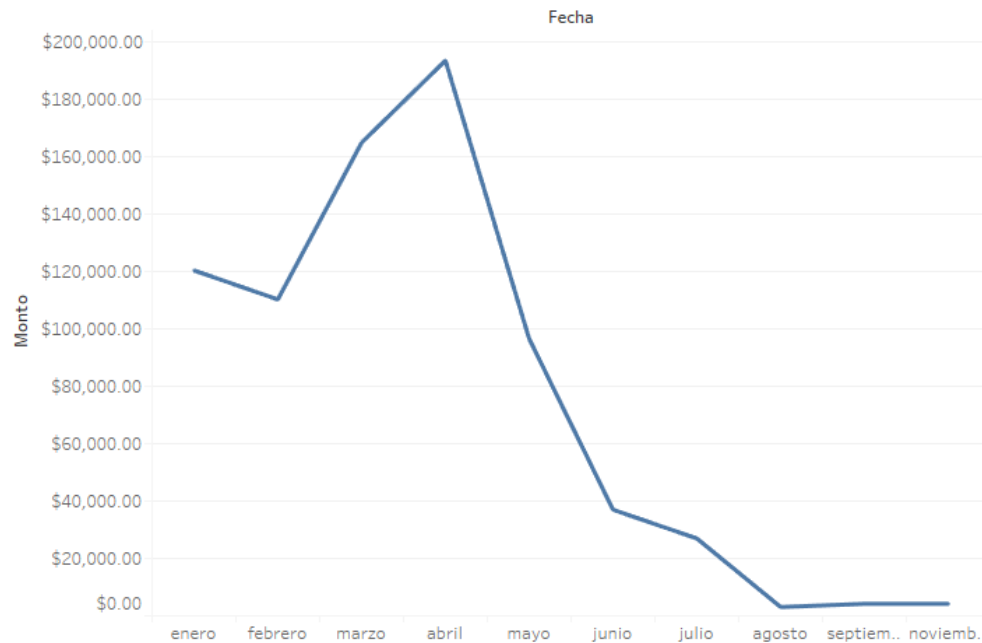


Figura 42

Finalmente, la propuesta de productos a eliminar se hace con base a los productos que no se han vendido debido a que se tiene un primer registro de ventas en 2002 hasta 2020 lo que implica un periodo demasiado largo para no haber tenido ventas. Teniendo en cuenta esto los id de los productos a eliminar son:

- 9, 14, 15, 16, 19, 20, 23, 24, 26, 27, 30, 32, 34, 35, 36, 37, 38, 39, 41, 43, 53, 55, 56, 58, 59, 61, 62, 63, 64, 65, 68, 69, 70, 71, 72, 73, 75, 76, 77, 78, 79, 80, 81, 82, 83, 86, 87, 88, 90, 91, 92, 93, 95, 96.



Conclusiones

El proyecto se presentó como un desafío global tanto a nivel técnico como a nivel de negocio. El estar limitado a instrucciones básicas de programación hace que la solución sea más compleja, sin embargo, presenta una ventaja muy clara que es el entendimiento de los procesos a detalle, ya que al estar limitados fue necesario hacer de manera detallada procesos y/o tareas que con alguna instrucción o librería pudieran resultar triviales. Esto en este punto de nuestra formación resulta si bien desafiante, también esencial para formar un criterio más profundo de solución y un entendimiento más abstracto de las necesidades y soluciones específicas de tareas complejas.

El trabajar con un proyecto muy próximo a un caso real de negocio también generó aprendizajes multidisciplinarios. Algo que se puede entender tras realizar este estudio de caso es que para poder generar análisis y conclusiones asertivas acerca de un proyecto es necesario tener bien claro la lógica de negocio; entender su funcionamiento, sus puntos clave, su estructura, operación, métricas, leads, etc. El conocer el negocio permite generar análisis ajustados a los requerimientos específicos del caso y obtener de esta forma conclusiones claves que sean aplicables y presenten un retorno de inversión que justifique el proyecto.

Finalmente, como conclusión personal considero este proyecto como un desafío que será pilar en mi desarrollo. Considero el caso como una primera aproximación realizada con las funciones más elementales de la programación estructurada al ramo del análisis de datos y que tras entender desde el nivel más bajo el funcionamiento seré capaz de desarrollar un conocimiento más complejo en el área.