[CASO DE ESTUDIO]

LIFESTORE

06 DE SEPTIEMBRE, 2020

ELABORADO POR:

Evelyn Elena Corrales Acosta

EMTECH DATA SCIENCE GRUPO 4

INDICE

.

1 INTRODUCCIÓN

Se realiza un análisis de datos creando un algoritmo en Python que permita a partir de la información proporcionada por LIfestore crear una estrategia para las siguientes consignas:

1) Productos más vendidos y productos rezagados:

- Generar un listado de los 50 productos con mayores ventas y uno con los 100 productos con mayor búsquedas.
- Por categoría, generar un listado con los 50 productos con menores ventas y uno con los 100 productos con menores búsquedas.

2) Productos por reseña en el servicio

- Mostrar dos listados de 20 productos cada una, un listado para productos con las mejores reseñas y otro para las peores, considerando los productos con devolución.

3) Total de ingresos y ventas promedio mensuales, total anual y meses con más ventas al año.

2 DEFINICIÓN DEL CÓDIGO

Se importan las librerias de lifetore_file:

```
7  from lifestore_file import lifestore_products
8  from lifestore_file import lifestore_sales
9  from lifestore_file import lifestore_searches
```

Los usuario permitidos en el programa fueron nombrados user1, user2 y user3 y los administradores son admin1,admin2,admin3 y sus respectivas contraseñas.

```
16 users = ["user1", "user2", "user3"] # usuarios permitidos
17 users_passwords = ["111","222","333"]
18 admins = ["admin1", "admin2", "admin3"]# administradores permitidos
19 admins_passwords = ["a111","a222","a333"]
```

```
22 #Prueba de acceso
23 access = True
24 admin = False
25 user= False
26 while access:
27     enter_username = input("Enter username: ")
28     enter_password = input("Enter password: ")
29
30     if enter_username in users:
31         if enter_username == users[0] and enter_password == users_passwords[0]:
32             access = False
33             user = True
34             print("LOGIN as user1")
35         elif enter_username == users[1] and enter_password == users_passwords[1]:
```

Después se programa un ciclo while como prueba de acceso a las cuentas que se ingresan, y mediante combinaciones if y elif para cada una de las cuentas cambia el booleano de true a false. De la línea 30 a la 64 se ejecuta esta operación donde valida tanto a los user como a los admin con un else al final en caso de un error.

```
46     elif enter_username in admins:
47         if enter_username == admins[0] and enter_password == admins_passwords[0]:
48             access = False
49             admin = True
50             print("LOGIN as admin1")
51         elif enter_username == admins[1] and enter_password == admins_passwords[1]:
52             access = False
53             admin = True
54             print("LOGIN as admin2")
55         elif enter_username == admins[2] and enter_password == admins_passwords[2]:
56             access = False
57             admin = True
58             print("LOGIN as admin3")
59         else:
60             print("Invalid username or password. Please try again.")
61     else:
62         print("Sorry, something went worng....")
63         print("That account does not exist. Please enter a different account.\n(TIP:
64         access = True
```

```
66  continuar = input("Next:(yes/no)")
67  while continuar == "yes":
68      option_1 = False
69      option_2 = False
```

```
75      option_8 = False
76      option_9 = False
77
78      if admin == True or user == True:
79          print("You access as administrator\n Options as admin:\n 1)TOP  50 searches\n 2)BOTTOM searches\n"
80                " 3)PRODUCTS searches by category \n 4)TOTAL SEARCHES AND SALES by category \n 5)PRODUCTS sales
                  by category\n"
81                " 6)TOP 50 sales\n 7)TOP 20 and BOTTOM 20 Reviews\n 8)ANUAL REVENUE-SALES and AVERAGE REVENUE BY
                  MONTH \n"
82                " 9)TOTAL SALES by MONTH and MONTHS TOP sales")
83          select_option =input("Select the option: ")
84          if select_option == "1":
85              print("Selected: TOP 50 searches")
86              option_1 = True
87          elif select_option == "2":
88              print("Selected: BOTTOM searches")
89              option_2 = True
```

```
108             elif select_option == "9":
109                 print("Selected: TOTAL SALES by MONTH and MONTHS TOP sales ")
110                 option_9 = True
111
112             else:
113                 print(("Sorry, something went worng...."))
114                 print(("Try with 1 ,2, 3, 4 up to 9"))
```

Posteriormente se tiene otro while para las opciones de selección a mostrar que tiene disponibles la aplicación, se imprime el mensajes de opciones y dependiendo del número ingresado (1 al 9) otro bloque de if , elif y else es ejecutado y así mismo todas las líneas de código se encuentran dentro del while para permitir cambiar de opciones.

Inicialmente se crean listas independientes de las sublistas de cado una de las listas principales para despues usarlas mediante ciclos for.

```python
#--------Separar las sublistas de las listas princ

frequency_lst = list()
for searches in lifestore_searches:
    frequency_lst.append(searches[1])


name_lst = list()
prices = list()
id_lst = list()
category_lst = list()
frequency_category = list()
for products in lifestore_products:
    name_lst.append(products[1])
    id_lst.append(products[0])
    frequency_category.append(products[3])
    prices.append(products[2])

    if products[3] not in category_lst:
        category_lst.append(products[3])
```

```python
dates = list() #todas las fechas
dates2= list() #fechas unicas
frequency_sales = list()
scores_frequency = list()
refund = list()
for sales in lifestore_sales:
    frequency_sales.append(sales[1]) #cuantas veces se vende un producto
    scores_frequency.append(sales[2])
    refund.append(sales[4])
    dates.append(sales[3]) #todos los dias del año que hubo venta
```

Sección: Búsquedas

El código se divide en secciones a continuación empieza la sección de búsquedas donde se hacen cada uno de los procesos buscados, mayores búsquedas por producto, por categoría etc.

El ciclo for va a recorrer todo el rango de la lista en este caso los 96 ids para después hacer una comparación con la lista que tiene la cantidad de búsquedas por id y si coincide cuenta las repeticiones y las asigna a una variable, junto con el nombre y el id correspondiente. Además con la función sort() de ordena la lista final y se delimita a las 50 mayores búsquedas. Después de imprimen los resultados colocando un if para definir que corresponde a la **opción 1**, al activarse se corren las líneas siguientes. Y al final se encuentra el bloque que permite que una vez finalizada la impresión se pueda acceder a otras opciones.

```python
#----------------------SECCION BUSQUEDAS--------------------------------
#1a)PRODUCTOS CON MAYORES BUSQUEDAS
#----------Contar las veces que se busca el id producto -------------
final_lst = list()
reporte_category = list()
for ids in range(len(id_lst)):#lista con los ids que tuvieron busqueda
    count = 0
    for freq in range(len(frequency_lst)): # listas de busquedas
        if(frequency_lst[freq] == id_lst[ids]): # cuantas veces se repite la busqueda por id
            count = count + 1
    report_search = [count,id_lst[ids],name_lst[ids]]
    report_category_search = [count,name_lst[ids],frequency_category[ids]]
    final_lst.append(report_search)
    reporte_category.append(report_category_search) #reporte con el total de busquedas
#print(reporte_category)
final_lst.sort(reverse=True)#ordenar de mayor a menor las busquedas
final_lst2 = final_lst[0:50]# rango de mayores busquedas
final_lst3 = final_lst2#lista con resultados finales de mayores busquedas
reporte_category.sort()
#print(final_lst)
```

```python
#RESULTADOS DE LAS MAYORES BUSQUEDAS--VALIDADO
#Resultados 1a):
if option_1 == 1:
    print("Loading...")
    print("Results:")
    print("PRODUCTS SEARCHES TOP 50")
    for searches_top in final_lst2:
        print("|Searches:",searches_top[0],"|"" \n| Product:",searches_top[2], "|id: "
        print("_____________________________________________________________________")
    print("Completed")

    continuar = input("new selection (yes/no): ")
    if continuar == "yes":
        print("loading...")
    elif continuar == "no":
        print("logggin out..")

    else: print("type: yes or no")
    continue
```

La **opción 2** funciona de la misma manera, el único cambio es que muestran las menores búsquedas, esto se hace cambiando los índices.

```python
if option_2 == 1:
    print("PRODUCTS SEARCHES BOTTOM 50")
    print("Loading...")
    print("Results:")
    for searches_bottom in final_lst[-51:-1]:
        print("|Searches: ",searches_bottom[0],"|\n""| Product:",searches_bottom[2], "|id: ",
        searches_bottom[1],"|")
        print("_____________________________________________________________________")
    print("Completed")
```

Ahora se crean nuevas listas para almacenar los valores que se obtenga por categoría.

```
#2a)BUSQUEDAS POR CATEGORIA:
#--------------..Calcular los productos con menores busquedas-----------
# 1)Hacer lista clasificando las categorias

    procesadores_search = list()
    procesadores_search2 = list()
    tarjetas_video_search = list()
    tarjetas_video_search2 = list()
for category_search in reporte_category:
    if category_search[2] == "procesadores":
        procesadores_search.append(category_search[0]) #Lista de solo cantidad de busquedas x categoria
        procesadores_search2.append(category_search)#Lista de cantidad de busquedas,producto x categoria
    elif category_search[2] == "tarjetas de video":
        tarjetas_video_search.append(category_search[0])
        tarjetas_video_search2.append(category_search)
```

Se indentan en un for para que almacene la sublista que se creo en el for (reporte_category) anterior en una nueva lista clasificándose por categoría y se ordenan con sort() de menor a mayor.

```
procesadores_search2.sort()
tarjetas_video_search2.sort()
tarjetas_madre_search2.sort()
```

opción 3: Se imprimen los resultados de cada categoría con las búsquedas que les corresponden y al finalizar se inserta de nuevo el bloque para cambiar de opción.

```
#RESULTADOS DE BUSQUEDAS MENOS A MAS POR PRODUCTO EN CADA CATEGORIA
#Resultados 2a)
if option_3 == 1:
    print('SEARCHES IN: "PROCESADORES"\n'
          'Loading...\n'
          'Results:\n')
    for column in procesadores_search2:
        print("|Category: ",column[2],"| Product:",column[1], "|Total SEARCHES: ",column[0],"|\n")
        print("_____________________________________________________________________________")
    print("Completed")

    print('SEARCHES IN: "TARJETAS DE VIDEO"\n'
          'Loading...\n'
          'Results:\n')
    for column in tarjetas_video_search2:
        print("|Category: ",column[2],"| Product:",column[1], "|Total SEARCHES: ",column[0],"|\n")
        print("_____________________________________________________________________________")
    print("Completed")
```

```
    print('SEARCHES IN: "AUDIFONOS"\n'
          'Loading...\n'
          'Results:\n')
    for column in audifonos_search2:
        print("|Category: ",column[2],"| Product:",column[1], "|Total SEARCHE
        print("_____________________________________________________________
    print("Completed")

    continuar = input("new selection (yes/no): ")
    if continuar == "yes":
        print("loading...")
    elif continuar == "no":
        print("logggin out..")

    else: print("type: yes or no")
    continue
```

Para las búsquedas totales se realiza otra vez un for para cada categoría. A demas se guardan en una variable todas las salidas de los for y se agregan a una lista.

```
# 1c) BUSQUEDAS TOTALES DE LAS CATEGORIAS
# 2. ----------Calcular busquedas totales en cada categoria--------------
total_procesadores = 0
for procesador_search in procesadores_search:
    total_procesadores = total_procesadores + procesador_search

total_tarjetasv = 0
for tarjetasv_search in tarjetas_video_search:
    total_tarjetasv = total_tarjetasv + tarjetasv_search
```

```
    #3. Concatenar una lista: usando las ventas totales por
    #categoria y el nombre de la categoria

    #Crear lista de las busquedas totales por categoria
    categories = ([total_procesadores , total_tarjetasv , total_tarjetasm
                 , total_discod , total_memorias , total_pantallas
                 , total_bocinas  , total_audifonos])

    #Lista final de categoria con su respetivas busquedas totales
    final_category = list()
    for i in range(len(categories)):
        total_category = [categories[i],category_lst[i]]
        final_category.append(total_category)
        #print(total_category)

    final_category.sort()
#print(final_category)
```

Lo que nos permite poder imprimir los resultados de la **opción 4**, que son las búsquedas totales por cada categoría de menor a mayor.

```
#RESULTADOS DE BUSQUEDAS TOTALES DE CADA CATEGORIA -validado
#Resultados 1c):
    if option_4 == 1:
        print("Loading...")
        print("Results:")
        print("TOTAL SEARCHES by CATEGORY:\n")
        for scategory in final_category:
            print("|Searches:",scategory[0],  " Category:",scategory[1])
            print("|-------------------------------------------------|")
        print("Completed")
```

Sección 2: Ventas

Sigue el mismo proceso que en la sección 1 para búsquedas, creo listas para almacenar los valores que obtenga del ciclo for lo que cambia es la lista a comparar ahora se toma la que contiene cuantas veces se vendió un producto mediante el id. Ordena de nuevo la lista de mayor a menor y viceversa teniendo dos listas como resultado. Y Finalmente se imprime la **opción 6 de 50 mayores ventas** siguiendo el mismo método.

```
#--------------------------SECION2 VENTAS------------------------------->
# Contar las ventas de cada producto y sacar las mayores
# 2b) MAYORES VENTAS POR PRODUCTO
    sales_lst = list()
    sales_lst2 = list()
    sales_lst3 = list() # para sacar ventas por mes
    for ids in range(len(id_lst)):
        count = 0
        for freq in range(len(frequency_sales)):
            if (frequency_sales[freq] == id_lst[ids]):
                count = count + 1
        #print(count)
        report_sales = [count,id_lst[ids],name_lst[ids]]
        report_sales2 =[count,name_lst[ids],frequency_category[ids]]
        sales_lst.append(report_sales)
        sales_lst2.append(report_sales2)

    #print(sales_lst2)
    sales_lst.sort(reverse=True)#ordenado de mayor a menor
    sales_max = sales_lst[0:50] #mayores 50 ventas por producto
    sales_lst2.sort()
    sales_min = sales_lst2[0:50]#menores 50 ventas por producto
    #print(sales_max)
```

```
    #Resultados 1b): validaddo

    if option_6 == 1:
        print("Loading...")
        print("Results:")
        print(" PRODUCTS SALES TOP 50:\n")
        for sales_top in sales_max:
            print("|Total SALES: ",sales_top[0],"|id: ",sales_top[1],"
            print("-----------------------------------------------------
        print("Completed")

        continuar = input("new selection (yes/no): ")
        if continuar == "yes":
            print("loading...")
        elif continuar == "no":
            print("logggin out..")

        else: print("type: yes or no")
        continue
```

Se crean listas y se calcula otra vez por categoría las ventas en lugar de búsquedas como se hizo anteriormente, crenado un for que contiene comparaciones if y elif para cada categoría usando la lista que contiene el conteo de ventas por id.

```
    procesadores_sales= list()
    procesadores_sales2= list()
    tarjetas_video_sales = list()
```

```
    for col in sales_lst2:
        if col[2] == "procesadores":
            procesadores_sales.append(col)
            procesadores_sales2.append(col[0])
        elif col[2] == "tarjetas de video":
            tarjetas_video_sales.append(col)
            tarjetas_video_sales2.append(col[0])
```

Se imprimen resultados de ventas por categoría de cada producto de menor a mayor que corresponde a la **opción 5** del programa:

```
#RESULTADOS DE VENTAS MENOS A MAS POR PRODUCTO EN CADA CATEGORIA
#Resultados 2b):
    if option_5 == 1:
        print('SEARCHES IN: "PROCESADORES"\n'
              'Loading...\n'
              'Results:\n')
        for column in procesadores_sales:
            print("|Category: ",column[2],"| Product:",column[1], "|Total SALES: ",column[0],"|\n")
            print
            ("-----------------------------------------------------------------------------------------
            -\n")
        print("Completed")
```

Para obtener las ventas totales por categoría se realizan un for para cada una de ellas como en la **opción 4** de búsquedas.

```
580
581 # 1c) VENTAS TOTALES DE LAS CATEGORIAS
582 # 2. Calcular ventas totales en cada categoria
583     ventas_procesadores = 0
584     for procesador_sales in procesadores_sales2:
585         ventas_procesadores = ventas_procesadores + procesador_sales
586
```

```
619 #3. Concatenar una lista: usando las ventas totales por
620 #categoria y el nombre de la categoria
621
622 #Crear lista de las busquedas totales por categoria
623     categories_ventas = ([ventas_procesadores , ventas_tarjetasv , ventas_tarjetasm
624                           ,ventas_discod , ventas_memorias , ventas_pantallas
625                           , ventas_bocinas  , ventas_audifonos])
626
627 #Lista final de categoria con su respetivas busquedas totales
628     final_ventas = list()
629     for i in range(len(categories_ventas)):
630         total_ventas = [categories_ventas[i],category_lst[i]]
631         final_ventas.append(total_ventas)
632         #print(total_ventas)
```

```
634 #RESULTADOS DE VENTAS TOTALES POR CATEGORIA
635 #Resultados 2c):
636
637 #print(final_ventas)
638     final_ventas.sort()
639 #RESULTADOS DE VENTAS TOTALES DE CADA CATEGORIA -validado
640     if option_4 == 1:
641         print("Loading...")
642         print("Results:")
643         print("TOTAL SALES by CATEGORY:")
644         for sales_cat in final_ventas:
645             print("|SALES:",sales_cat[0],  " Category:",sales_cat[1])
646             print("|-------------------------------------------------|")
647         print("Completed")
```

Y con un for final se guardan los totales de cada cateogria con el nombre de la categoria. Se imprimen resutados igual en la **opcion 4**.

Sección 3: Reseñas

Se realiza otro for que contiene la lista de los ids y la lista que contiene las veces que un id se vende y que también indica las veces que se ha calificado un id. Por lo que se crea una lista agregando los nombres correspondientes a los ids repetidos si la comparación es verdadera.

```
660 #---------------------------SECION 3 RESEÑAS--------------------------------->
661 # Calcular -->
662 # 1)Hacer lista clasificando las
663
664     name_reseñas = list()
665     for ids_reseñas in range(len(id_lst)):
666         for tscores in range(len(frequency_sales)):
667             if id_lst[ids_reseñas] == frequency_sales[tscores]:
668                 name_reseñas.append(name_lst[ids_reseñas])
669
670     reseñas = list()
671     for i in range(len(scores_frequency)):
672         total_scores = [name_reseñas[i],scores_frequency[i], refund[i]]
673         reseñas.append(total_scores)
674     #print(reseñas)
```

Mediante otro for de la lista que contiene únicamente las calificaciones, creo una lista de ese tamaño conteniendo nombres, scores y si tiene devoluciones.

con un for final vuelvo a comparar los ids para generar el conteo que me diga cuantas veces el producto ha sido calificado y además el promedio de dichas calificaciones y devoluciones con una serie de if, elif, else. Obteniendo al final la lista y ordenándola usando sort()

```
reporte_reseña = list()
for ids in range(len(id_lst)):#lista con los ids que tuvieron busqueda
    count = 0
    sumas_score = 0
    for freqq in range(len(frequency_sales)): # listas de busquedas
        if(frequency_sales[freqq] == id_lst[ids]): # cuantas veces se repite la busqueda por id
            count = count + 1
            sumas_score = sumas_score + scores_frequency[freqq]
            if sumas_score != 0:
                promedio = sumas_score/count
            else:
                promedio = sumas_score
            #print(sumas_score)
        elif refund[freqq] == 0:
            refund_status = "No refund"

        else:
            refund_status ="With refund"
    report_scores = [promedio, name_lst[ids], refund_status]
    reporte_reseña.append(report_scores)
#print(reporte_reseña)
reporte_reseña.sort()
reporte_reseña2 = sorted(reporte_reseña, reverse = True)
```

Se imprimen resultados de la **opción 7 : 20 mejores y peores reseñas:**

```
#RESULTADOS DE MEJORES Y PEORES RESEÑAS -validado
if option_7 == 1:
    print("Loading...")
    print("Results:")
    print("WORST 20 REVIEWS:")
    for reviews in reporte_reseña[0:20]:
        print("|REVIEWS AVERAGE:",reviews[0]," Status Refund:",reviews[2],  " Product:",reviews[1],"|")
        print("|-------------------------------------------------------------------------------------|")
    print("Completed")

    print("Loading...")
    print("Results:")
    print("\n" "BETTER 20 REVIEWS:")
    for reviews in reporte_reseña2[0:20]:
        print("|REVIEWS AVERAGE:",reviews[0]," Status Refund:",reviews[2],  " Product:",reviews[1],"|")
        print("|-------------------------------------------------------------------------------------|")
    print("Completed")
```

sección 4: Totales finales

Comienza haciendo un ciclo for que vuelve a contar y se almacena ese grupo de valores en una lista.

```
#------------------------------SECCION 4 finales ----------------------

#Calculo de ventas anuales-------------------------------------- SI
reporte_score = list()
for ids in range(len(id_lst)):
    count = 0
    for id_sale in range(len(frequency_sales)):
        if (frequency_sales[id_sale] == id_lst[ids]):
            count = count + 1
    reporte_score.append(count)
#print(reporte_score)
```

Otro ciclo for de la lista de precios permite hacer la multiplicación elemento a elemento lo que se traduce a ganancia por producto.

Para calcular las ventas anuales se usa otro for que va sumando cada elemento de la lista que contiene las ganancias. Y si mismo se calculan las ventas totales.

```
ingresos_anuales = list()
ventas_anuales = list()
total_ingreso = 0
total_ventas = 0
for price in range(len(prices)):#ciclo para todos los precios de la lista
    ingresos_anuales.append(prices[price] * reporte_score[price])#precio del producto por el numero de ventas x producto

for ingreso in ingresos_anuales:
    total_ingreso = total_ingreso + ingreso
#print(total_ingreso)

for ventas in reporte_score:
    total_ventas = total_ventas + ventas
```

Resultados **opción 8: ventas anuales y ingresos anuales**

```
756     if option_8 == 1:
757
758         print("\n""TOTAL ANUAL REVENUE AND  SALES ")
759         print
            ("-------------------------------------------------------------------
            )
760         print("| REVENUE : $",total_ingreso, "|SALES: ",total_ventas,"|\n")
761         print
            ("-------------------------------------------------------------------
```

Esta parte reorganiza el formato de las fechas cambiando los dígitos que corresponden al mes por las posiciones del día, esto para facilitar el ordenamiento.

```
#Organizar en meses las ventas y sacar el promedio de cada mes

new_dates = list()# cambiar posicion de mes de la lista principal de dates
new_dates2 = list()
for date in dates:
    dia = date[0:3]
    mes = date[3:6]
    año = date[6:]
    new_date = mes #+ dia + año
    new_dates.append(new_date)
```

Siguiendo los pasos anteriores, el ciclo se vuelve a realizar haciendo las comparaciones con los meses y si coincide los almacena en una lista que es ordenada de menor a mayor y otra de mayor a menor.

```
784 #Lista de meses del 1 al 12
785     meses = ['01/','02/','03/','04/','05/','06/','07/','08/','09/','10/','11/','12/']
786
787     #print(dates_double2)
788     #Hacer la comparativa de fechas para calcular las ventas por mes
789     reporte_mensual = list()
790     count_lst = list()
791     reporte_mensual2 = list()
792     for mes in range(len(meses)):
793         count = 0
794         for date_new in range(len(new_dates)):
795             if (new_dates[date_new] == meses[mes]):
796                 count = count + 1
797
798         report_mes = [meses[mes],count]
799         report_mes2 = [count,meses[mes]]
800         reporte_mensual2.append(report_mes2)
801         reporte_mensual.append(report_mes)
802
803     reporte_mensual.sort()
804     reporte_mensual2.sort(reverse=True)
```

Se imprimen resultados de la **opción9 : ventas promedio mensuales y meses con mas ventas**

```
807     if option_9 == 1:
808
809         print("MONTH SALES")
810         print("Loading...")
811         print("Results:")
812
813         print("|-----------------------------|")
814         print("| TOTAL SALES JANUARY: ",reporte_mensual[0][1],"     |")
815         print("|-----------------------------|")
816         print("| TOTAL SALES FEBRUARY: ",reporte_mensual[1][1],"    |")
817         print("|-----------------------------|")
818         print("| TOTAL SALES MARCH: ",reporte_mensual[2][1],"      |")
```

```
843         print("TOP SALES by MONTH:")
844
845         for ventas2 in reporte_mensual2[0:7]:
846             print("|-----------------------------|")
847             print("|MES:",ventas2[1],"|Total de ventas:",ventas2[0],)
848             print("|-----------------------------|")
849         print("Completed")
850
```

Para calcular cuanto se vende por mes se sigue con un ciclo for para almacenar las fechas reorganizadas con las ventas. Para clasificar los meses de hacen comparaciones if, elif, y se almacenas en su respectiva lista.

```
859 #-----------Calcular cuanto se vende por mes-----------------------------
860     revenue = list()
861
862     for date_new in range(len(new_dates)):
863         count = 0
864         for mes in range(len(meses)):
865             if (new_dates[date_new] == meses[mes]):
866                 count = count + 1
867         ganancias = [new_dates[date_new] , frequency_sales[date_new]]
868         revenue.append(ganancias)
869     #print(revenue)
```

```
881     noviembre = list()
882     diciembre = list()
883
884     for mes_mes in revenue:
885         if mes_mes[0] == meses[0]:
886             enero.append(mes_mes)
887         elif mes_mes[0] == meses[1]:
888             febrero.append(mes_mes)
```

En cada mes se realiza un for para extraer los ids que tuvieron venta y después multiplicarlos con otra lista que contiene los precios.

```
#ENERO--->
#Tomo los ids de las ventas de enero para compararlos con los ids y su precio
ids_enero = list()
for veces in enero:
    ids_enero.append(veces[1])
#print(ids_enero)
```

El for me sirve para contar los ids que coinciden en enero, lo que significa las ventas por id de enero**.** Para continuar con la multiplicación de la venta del producto con su respectivo precio usando un cilo for para que multiplique uno a uno. Y asi sucesivamente para cada mes hasta diciembre.

```
# Creo un reporte que me dice que ids si tuvieron ventas en enero
reporte_enero = list()
for ids in range(len(id_lst)):
    count = 0
    for id_enero in range(len(ids_enero)):
        if (ids_enero[id_enero] == id_lst[ids]):
            count = count + 1

    reporte_enero.append(count)
#print(reporte_enero)
```

```
#Multiplico cada uno de los ids con ventas por su precio correspondiente
ingresos_enero = list()
for price_enero in range(len(prices)):#ciclo para todos los precios de la lista
    ingresos_enero.append(prices[price_enero] * reporte_enero[price_enero])#precio del producto por el
    numero de ventas x producto
#print(ingresos_enero)

#Calculo de las ventas totales de enero de los ids
total_enero = 0
for ingreso_enero in ingresos_enero:
    total_enero = total_enero + ingreso_enero
#print(total_enero)
```

```
# Creo un reporte que me dice que ids si tuvieron ventas en el mes
reporte_diciembre= list()
for ids in range(len(id_lst)):
    count = 0
    for id_diciembre in range(len(ids_diciembre)):
        if (ids_diciembre[id_diciembre] == id_lst[ids]):
            count = count + 1

    reporte_diciembre.append(count)
#print(reporte_diciembre)
```

```
#Multiplico cada uno de los ids con ventas por su precio correspondiente
ingresos_diciembre = list()
for price_diciembre in range(len(prices)):#ciclo para todos los precios de la lista
    ingresos_diciembre.append(prices[price_diciembre] * reporte_diciembre[price_diciembre])#precio del
    producto por el numero de ventas x producto
#print(ingresos_diciembre)

#Calculo de las ventas totales del mes de los ids
total_diciembre = 0
for ingreso_diciembre in ingresos_diciembre:
    total_diciembre = total_diciembre + ingreso_diciembre
```

Se imprimen los resultados de **la opción 8: ingresos mensuales.**

```
if option_8 == 1:
    print("\n""REVENUE by MONTH")
    print("Loading...")
    print("Results:")
    print("|--------------------------------------|")
    print("| TOTAL REVENUE JANUARY: $",total_enero,)
    print("|--------------------------------------|")
    print("| TOTAL REVENUE FEBRUARY: $",total_febrero,)
    print("|--------------------------------------|")
    print("| TOTAL REVENUE MARCH: $",total_marzo,)
```

Para terminar se cierra tiene el bloque de opciones y el cierre del ciclo while.

```
        continuar = input("new selection (yes/no): ")
        if continuar == "yes":
            print("loading...")
        elif continuar == "no":
            print("logggin out..")

        else: print("type: yes or no")
        continue
else:
    print("Disconecting... you typed:[no] or typed an error\n"
          "See you or try to log in again (TIP: yes/no)")


#---------------------------FINAL DE FINALES------------------------------>
```

3 SOLUCIÓN AL PROBLEMA

- El análisis complete se hizo con la información recolectada en Python.

Reseñas

Reseña	Estado	Producto
1	no refund	Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151
1	no refund	Tarjeta de Video Gigabyte AMD Radeon R7 370 OC
1.83	refund	Tarjeta Madre AORUS micro ATX B450 AORUS M (rev. 1.0)
1.83	refund	Tarjeta Madre ASRock Z390 Phantom Gaming
2	no refund	Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2

Reseña	Estado	Producto
5	no refund	Tarjeta de Video Zotac NVIDIA GeForce GTX 1660 Ti
5	no refund	Tarjeta de Video VisionTek AMD Radeon HD5450
5	no refund	Tarjeta de Video VisionTek AMD Radeon HD 5450
5	no refund	Tarjeta de Video Sapphire AMD Pulse Radeon RX
2	no refund	Tarjeta de Video PNY NVIDIA GeForce RTX 2080

Mayores ventas y búsquedas

Producto	Ventas	Producto	Busquedas
SSD Kingston A400	50	**SSD Kingston A400, 120GB, SATA III, 2.5**	263
Procesador AMD Ryzen 5 2600	42	SSD Adata Ultimate SU800, 256GB,	107
Procesador Intel Core i3-9100F	20	Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING	60
Tarjeta Madre ASRock Micro ATX	18	Procesador AMD Ryzen 5 2600,	55
SSD Adata Ultimate SU800,	15	**Procesador AMD Ryzen 3 3200G**	41

Vibratory soil	14	Logitech Audífonos Gamer G635 7.1, Alámbrico	35
Procesador AMD Ryzen 3 3200G	13	TV Monitor LED 24TL520S-PU 24, HD, Widescreen	32

Categorias

Categoria	Busquedas	Ventas
Discos duros	463	104
Procesadores	222	94
Tarjetas madre	137	49
Tarjetas video	82	26

Ingresos

MESES	VENTAS MENSUALES	INGRESO
Enero	53	$120,237
Febrero	41	$110,139
Marzo	51	$164,729
Abril	75	$193,295
Mayo	36	$96,394
Junio	11	$36,949
Julio	11	$26,949
Agosto	3	$3077
Septiembre	1	$4199
Octubre	0	$0
Noviembre	1	$4209
Diciembre	0	0
TOTAL	**284**	**$760177**

4 CONCLUSIÓN

- Para productos rezagados una sugerencia seria hacer un paquete con dos productos: uno de mayor rotación y otro rezagado . Como el Procesador Intel Core i3-9100F y it Memoria RAM Corsair Vengeance LPX DDR4, 2400MHz, 32GB, Non-ECC, CL16 .De esta forma el atractivo del producto de mayor rotación persuadirá a comprar el paquete.
- Hacer un descuento de los productos con mayor rotación en los meses con menores ventas, esto incrementara el ingreso del mes y la venta de producto.
- Categorías como audífonos, bocinas, pantallas, tienen prácticamente cero ventas y cero búsquedas, a excepción de un producto en audífonos Logitech Audífonos Gamer G635 que tiene considerables búsquedas por lo que darle descuento mínimo podría incrementar su venta, en cuanto a las otras categorías se podría empezar una campaña donde comprando un producto de mayor rotación te lleves gratis el producto de nula rotación así se promociona el producto y a su vez reduces inventario.
- Para las categorías con mayores búsquedas: en la compra de un producto de la categoría participas en un sorteo para otro producto de esa categoría con mayor inventario.
- Para mejorar las reseñas, una sugerencia seria interactuar mas por redes sociales Facebook e Instagram con los clientes y a través de estas aprovechar para informar de promociones y sorteos. La tarjetas madres tienen buenas ventas, pero también malas reseñas, se podría dar cierta garantía para que la venta del producto no decaiga.
- Las tarjetas de video tienen buenas reseñas y esta dentro de las cinco categorías con mas ventas y búsquedas, si se tiene página de internet para incrementar las ventas se puede dar envíos gratis comprando 3 productos de las categorías de audífonos, bocinas y tarjetas madre o pantallas, 2 producto con pocas ventas y un producto más atractivo.