

PRESENTACION DE CASO SANOFI.

ALAN RODRIGO AGUILAR BUSTILLOS.

CANDIDATO AL PUESTO

“Star Data Management & Data Analytics”

sanofi

CASE 1: PYTHON

Código:

```
1  item_prices = {
2      "apple": 1.25,
3      "banana": 0.75,
4      "orange": 1.50,
5      "grapes": 2.00,
6      "chocolate": 3.50
7  }
8
9  shopping_cart = []
10
11 while item != "done":
12     item = input("Enter an item (or 'done' to finish): ").lower()
13     if item == "done":
14         break
15     elif item in item_prices:
16         shopping_cart.append(item)
17     else:
18         print("Item not found. Please choose a valid item.")
19
20 total_cost = sum(item_prices[item] for item in shopping_cart)
21
22 print("\nShopping Cart:")
23 for item in shopping_cart:
24     print(f"- {item} (${item_prices[item]})")
25 print(f"\nTotal Cost: ${total_cost}")
```

Explica que hace el código:

El código en cuestión es básicamente, un carrito de compras, cuya finalidad es llevar el control y registro de los productos que se encuentran en el diccionario de datos. Su principal objetivo es realizar la suma de los productos seleccionados para al finalizar obtener el total de la compra del carrito (principal similitud con un ejemplo cotidiano seria las cajas de autocobro que se emplean en los super mercados).

CODIGO COMENTADO:

```
Ejercicio sanofi.py •
Ejercicio sanofi.py > ...
1  #PRESENTACION DE CASO SANOFI.
2  #AGUILAR BUSTILLOS ALAN RODRIGO.
3
4  #Creación de un diccionario de articulos con la asignacion de sus precios respectivamente:
5  item_prices = {
6      "apple": 1.25,
7      "banana": 0.75,
8      "orange": 1.50,
9      "grapes": 2.00,
10     "chocolate": 3.50
11 }
12
13 #creación del carrito de compras empleando una lista que se encuentra vacia para poder ingresar los articulos al carrito de compras:
14 shopping_cart = []
15
16 # creación de un bucle para ingresar los items al shoppint cart, el cual se encuentra condicionado a que se
17 # ingrese un articulo que ya se encuentre dentro del diccionario creado, o de lo contrario se le pide al usuario el ingresar un item valido
18 # al ingresar la palabra 'done' el bucle termina, causando el cierre del carrito de compras:
19 while True:
20     item = input("Enter an item (or 'done' to finish): ").lower()
21     if item == "done":
22         break
23     elif item in item_prices:
24         shopping_cart.append(item)
25     else:
26         print("Item not found. Please choose a valid item.")
27
28 # se realiza el calculo del costo total, realizando la suma de los articulos que se seleccionaron con anterioridad:
29 total_cost = sum(item_prices[item] for item in shopping_cart)
30
31 print("\nShopping cart:")
32 for item in shopping_cart:
33     print(f" - {item} (${item_prices[item]})")
34
35 #impresion del total del carrito:
36 print(f"\nTotal Cost: ${total_cost}")
```

Se emplea VS CODE para la compilación del código python

¿El código tiene algún error?

Se realizan pruebas de escritorio y no se encuentran errores, se realizan pruebas seleccionando varios artículos repetidas veces, una unidad por cada artículo y en ambos casos la sumatoria es la correcta. También se comprueba el caso en el cual no se elige algún artículo válido y no se encuentra error alguno:

```
PS C:\Users\alan_\OneDrive\SANOFI\EJERCICIO INICIAL>
sanofi.py"
Enter an item (or 'done' to finish): apple
Enter an item (or 'done' to finish): apple
Enter an item (or 'done' to finish): apple
Enter an item (or 'done' to finish): banana
Enter an item (or 'done' to finish): orange
Enter an item (or 'done' to finish): grapes
Enter an item (or 'done' to finish): grapes
Enter an item (or 'done' to finish): chocolate
Enter an item (or 'done' to finish): done

Shopping Cart:
- apple ($1.25)
- apple ($1.25)
- apple ($1.25)
- banana ($0.75)
- orange ($1.5)
- grapes ($2.0)
- grapes ($2.0)
- chocolate ($3.5)

Total Cost: $13.5
```

Caso de varios artículos en repetidas ocasiones, no se detecta error.

```
PS C:\Users\alan_\OneDrive\SANOFI\EJERCICIO INICIAL>
sanofi.py"
Enter an item (or 'done' to finish): apple
Enter an item (or 'done' to finish): banana
Enter an item (or 'done' to finish): orange
Enter an item (or 'done' to finish): grapes
Enter an item (or 'done' to finish): chocolate
Enter an item (or 'done' to finish): done

Shopping Cart:
- apple ($1.25)
- banana ($0.75)
- orange ($1.5)
- grapes ($2.0)
- chocolate ($3.5)

Total Cost: $9.0
```

Caso de selección unaria de cada ítem, no se detecta error.

```
PS C:\Users\alan_\OneDrive\SANOFI\EJERCICIO INICIAL>
sanofi.py"
Enter an item (or 'done' to finish): chocolate
Enter an item (or 'done' to finish): grapes
Enter an item (or 'done' to finish): lime
Item not found. Please choose a valid item.
Enter an item (or 'done' to finish): orange
Enter an item (or 'done' to finish): done

Shopping Cart:
- chocolate ($3.5)
- grapes ($2.0)
- orange ($1.5)

Total Cost: $7.0
```

Caso de selección de ítem no existente, no se detectan errores

¿Mejorarías algo?

Como tal, el código cumple con su finalidad, llevar el control de los ítems que están dentro del carrito de compras, lo único que yo consideraría sería el caso en el cual el usuario lleva varios artículos repetidas veces, y debe escribir el nombre varias veces, agregaría la opción de indicar si solo se lleva una o varias unidades del artículo para hacer más eficiente el proceso.

IMPLEMENTACION:

```
Ejercicio modificado sanofi.py > ...
1  item_prices = {
2      "apple": 1.25,
3      "banana": 0.75,
4      "orange": 1.50,
5      "grapes": 2.00,
6      "chocolate": 3.50
7  }
8
9  shopping_cart = {}
10 while True:
11     item = input("Enter an item (or 'done' to finish): ").lower()
12     if item == "done":
13         break
14     elif item in item_prices:
15         # modificacion para indicar cuantas unidades de un mismo producto se aÑadiran al carrito de compras:
16         quantity = int(input(f"How many {item}s would you like to add? "))
17         if item in shopping_cart:
18             shopping_cart[item] += quantity
19         else:
20             shopping_cart[item] = quantity
21     else:
22         print("Item not found. Please choose a valid item.")
23
24 total_cost = sum(item_prices[item] * quantity for item, quantity in shopping_cart.items())
25
26 print("\nShopping Cart:")
27 for item, quantity in shopping_cart.items():
28     print(f" - {item} x{quantity} (${item_prices[item] * quantity:.2f})")
29 print(f"\nTotal cost: ${total_cost:.2f}")
```

```
PS C:\Users\alan_OneDrive\SANOFI\EJERCICIO INICIAL> &
OFI/EJERCICIO INICIAL/Ejercicio modificado sanofi.py"
Enter an item (or 'done' to finish): banana
How many bananas would you like to add? 10
Enter an item (or 'done' to finish): chocolate
How many chocolates would you like to add? 8
Enter an item (or 'done' to finish): grapes
How many grapess would you like to add? 15
Enter an item (or 'done' to finish): done

Shopping Cart:
- banana x10 ($7.50)
- chocolate x8 ($28.00)
- grapes x15 ($30.00)

Total cost: $65.50
PS C:\Users\alan_OneDrive\SANOFI\EJERCICIO INICIAL>
```

Implementación del código, junto con la compilación del mismo, sin errores detectados.

CASE 2. SQL

- **¿Sabes qué es una relación uno a uno?**

Las relaciones uno a uno en el área de bases de datos, nos indican la forma en como se relacionan los elementos de una tabla A con los elementos de una tabla B. el el caso de este tipo de relaciones, como su nombre lo indica, es que un elemento de la tabla A solo se relaciona con un elemento de la tabla B.

Las relaciones uno a uno se pueden encontrar principalmente en bases de datos que ya se encuentran normalizadas.

- **¿Qué hace un inner join?**

El inner join lo que nos brinda es los elementos que se encuentran tanto en un conjunto A y un conjunto B, en bases de datos seria la intersección de elementos resultantes entre una tabla A y una tabla B.

Considerando las siguientes dos tablas:

1. CATALOGO: PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA
2. REPORTE: PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES

Queremos obtener una sola tabla con los siguientes campos ¿Cómo realizarías una consulta?:

PRODUCTO_ID, PRODUCTO_DESC, FAMILIA, VENTA, UNIDADES

SOLO CONSIDERANDO PRODUCTOS DE STATUS ACTIVO

Creación de las tablas

Creación de la tabla catalogo:

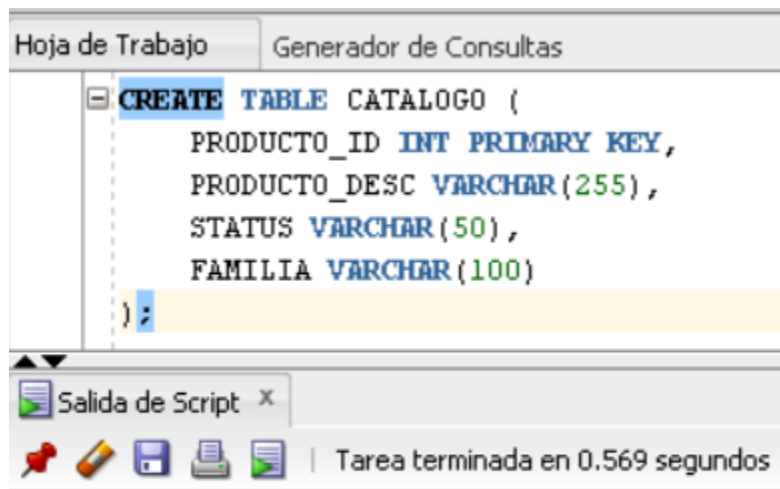


Table CATALOGO creado.

Creación de la tabla reporte:

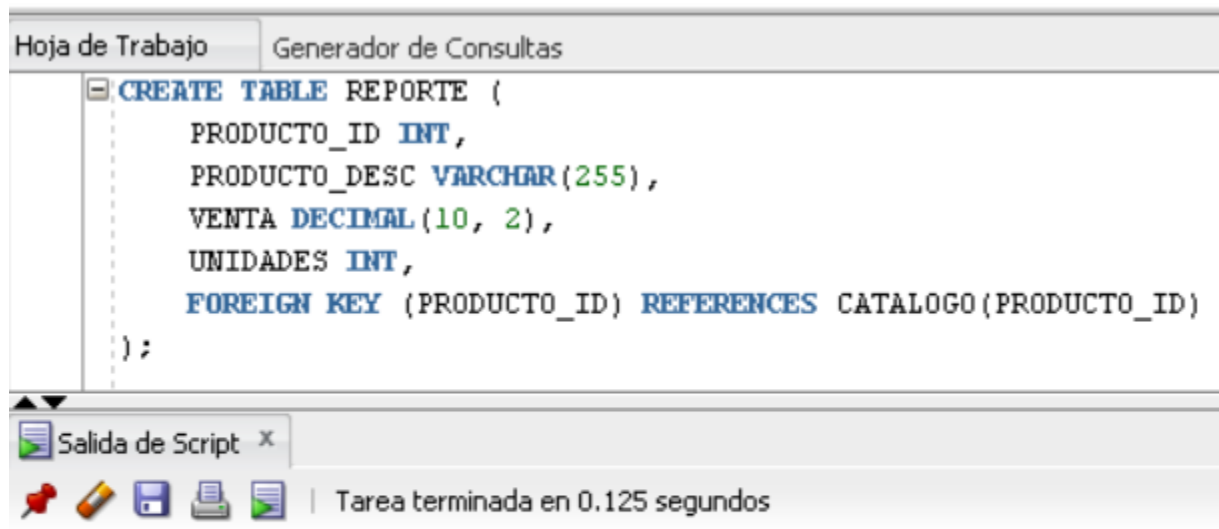


Table CATALOGO creado.

Table REPORTE creado.

Inserción de valores en las tablas:

```
-- Datos para la tabla CATALOGO
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(1, 'Producto A', 'ACTIVO', 'Familia 1');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(2, 'Producto B', 'INACTIVO', 'Familia 2');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(3, 'Producto C', 'ACTIVO', 'Familia 1');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(4, 'Producto D', 'ACTIVO', 'Familia 3');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(5, 'Producto E', 'ACTIVO', 'Familia 2');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(6, 'Producto F', 'INACTIVO', 'Familia 3');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(7, 'Producto G', 'ACTIVO', 'Familia 1');
INSERT INTO CATALOGO (PRODUCTO_ID, PRODUCTO_DESC, STATUS, FAMILIA) VALUES
(8, 'Producto H', 'ACTIVO', 'Familia 2');
```

Salida de Script x

Tarea terminada en 0.287 segundos

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

```
-- Datos para la tabla REPORTE
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(1, 'Producto A', 1500.00, 100);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(2, 'Producto B', 2000.00, 150);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(3, 'Producto C', 1800.00, 120);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(4, 'Producto D', 1700.00, 130);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(5, 'Producto E', 1900.00, 110);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(6, 'Producto F', 1600.00, 140);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(7, 'Producto G', 1550.00, 105);
INSERT INTO REPORTE (PRODUCTO_ID, PRODUCTO_DESC, VENTA, UNIDADES) VALUES
(8, 'Producto H', 1750.00, 125);
```

Salida de Script x

Tarea terminada en 0.097 segundos

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

1 fila insertadas.

Consulta solicitada:

Se crean las tablas en SQL Developer de Oracle, con valores ficticios. La consulta nos pide los elementos existentes con el estatus de activo y se nos solicitan la impresión de los siguientes datos:





PRODUCTO_ID, PRODUCTO_DESC, FAMILIA, VENTA, UNIDADES

La consulta se realiza de la siguiente forma:

```
-- select nos ayuda a seleccionar las columnas que queremos que aparezcan en la consulta
SELECT
    C.PRODUCTO_ID,
    C.PRODUCTO_DESC,
    C.FAMILIA,
    R.VENTA,
    R.UNIDADES
--from nos indica de donde provienen los datos, C es para la tabla Catalogo y R para la tabla reporte
FROM
    CATALOGO C
-- realiza la union entre las tablas en base a Producto ID
JOIN
    REPORTE R ON C.PRODUCTO_ID = R.PRODUCTO_ID
--filtro de busqueda para que solo nos aparezcan los elementos con el status de activo
WHERE
    C.STATUS = 'ACTIVO';
```

Resultado de la consulta:

Salida de Script x Resultado de la Consulta x

    SQL | Todas las Filas Recuperadas: 6 en 0.159 segundos

	PRODUCTO_ID	PRODUCTO_DESC	FAMILIA	VENTA	UNIDADES
1	1	Producto A	Familia 1	1500	100
2	3	Producto C	Familia 1	1800	120
3	4	Producto D	Familia 3	1700	130
4	5	Producto E	Familia 2	1900	110
5	7	Producto G	Familia 1	1550	105
6	8	Producto H	Familia 2	1750	125