



INSTITUTO POLITECNICO NACIONAL



CENTRO DE INVESTIGACION EN COMPUTACION

Práctica Final del Curso de Programación Python en el Ámbito Científico.

Instructor del Curso: Alan Badillo Salas.

Elaboró: Enrique Aguirre.

Julio, 2023.

JUSTIFICACIÓN.

El motivo de la elección de la Práctica P109 para realizar es con la finalidad de obtener una noción de la forma de establecer conexión a una Base de Datos con Python.

INTRODUCCIÓN.

Para la Práctica P109 consta en poder hacer conexión a una Base de Datos a través de Python, en la cual deben existir previamente 4 tablas; tabla de Frutas y tabla de Ventas, la cuales serán el origen de la información. Los datos de la tabla de Frutas serán filtrados por los que tengan un precio menor a 20 y la información resultante será guardada en la tabla destino Frutas_20. Los datos de la tabla Ventas serán filtrados por los del día actual y la información resultante será guardada en la tabla destino Ventas_día.

Para ésta práctica el manejador de Base de Datos utilizado será SQL Server.

DESARROLLO.

1- Pasos Previos.

Se debe instalar el manejador de Base de Datos, en este caso SQL Server. El cual se puede descargar de la siguiente liga:

<https://www.microsoft.com/es-mx/sql-server/sql-server-downloads>



Se elige la opción Instalación Básica con las configuraciones predeterminadas hasta finalizar su instalación.

Una vez instalado SQL Server, se deben ejecutar los siguientes scripts para crear las 4 Tablas a ocupar:

```
CREATE TABLE frutas_precios
(
    fruta_id INT,
    nombre VARCHAR(100),
    precio MONEY,
    existencias INT
)
GO

CREATE TABLE frutas20
(
    fruta_id INT,
    nombre VARCHAR(100),
    precio MONEY,
    existencias INT
)
GO

CREATE TABLE frutas_ventas
(
    venta_id INT,
    fruta_id INT,
    fecha DATE
)
GO

CREATE TABLE ventas_dia
(
    venta_id INT,
    fruta_id INT,
    fecha DATE
)
GO
```

Posteriormente se deben agregar datos a las Tablas Origen: Tabla *frutas_precios* y Tabla *frutas_ventas*, para ello se ejecuta el siguiente script:

```
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (1, N'Manzana', 5.0000, 520)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (2, N'Mango', 25.0000, 480)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (3, N'Pera', 8.0000, 960)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (4, N'Guanabana', 30.0000, 650)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (5, N'Kiwi', 45.0000, 250)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (6, N'Papaya', 60.0000, 80)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (7, N'Zarzamora', 42.0000, 625)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (8, N'Naranja', 13.0000, 970)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (9, N'Fresa', 60.0000, 540)
INSERT [dbo].[frutas_precios] ([fruta_id], [nombre], [precio], [existencias]) VALUES (10, N'Melon', 63.0000, 850)
GO

INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (1, 5, CAST(N'2023-07-13' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (2, 7, CAST(N'2023-07-13' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (3, 1, CAST(N'2023-07-14' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (4, 3, CAST(N'2023-07-14' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (5, 6, CAST(N'2023-07-14' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (6, 4, CAST(N'2023-07-15' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (7, 8, CAST(N'2023-07-15' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (8, 2, CAST(N'2023-07-15' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (9, 10, CAST(N'2023-07-15' AS Date))
INSERT [dbo].[frutas_ventas] ([venta_id], [fruta_id], [fecha]) VALUES (10, 9, CAST(N'2023-07-15' AS Date))
GO
```

Los resultados de la inserción de datos a las Tablas *frutas_precios* y *frutas_ventas* se pueden apreciar en la siguiente imagen. Como observación, Se debe considerar que en la Tabla *frutas_precio* haya datos con precio menor a 20 y en la tabla *frutas_ventas* haya ventas del día para que se pueda visualizar el funcionamiento de esta práctica.

```
SELECT *  
FROM [frutas_precios]
```

fruta_id	nombre	precio	existencias
1	Manzana	5,00	520
2	Mango	25,00	480
3	Pera	8,00	960
4	Guanabana	30,00	650
5	Kiwi	45,00	250
6	Papaya	60,00	80
7	Zarzamora	42,00	625
8	Naranja	13,00	970
9	Fresa	60,00	540
10	Melon	63,00	850

```
SELECT *  
FROM [frutas_ventas]
```

venta_id	fruta_id	fecha
1	5	2023-07-13
2	7	2023-07-13
3	1	2023-07-14
4	3	2023-07-14
5	6	2023-07-14
6	4	2023-07-15
7	8	2023-07-15
8	2	2023-07-15
9	10	2023-07-15
10	9	2023-07-15

Como esta práctica se realizó el día 15 de Julio de 2023, ésta fecha será la fecha del día para la Tabla *frutas_ventas*.

Con esto se finaliza los pasos previos requeridos.

2- Construcción de Código Python.

En el primer bloque de código se muestran las bibliotecas que se van a utilizar, resaltando *pyodbc* que será la biblioteca para realizar la conexión a la Base de Datos y *datetime* que se ocupará para las conversiones de datos fecha.

```
+ ✂ 📄 📌 ▶ ■ ↺ ▶▶ Code ▼  
  
[ ]: import pyodbc  
import pandas as pd  
from datetime import date
```

En el segundo bloque de código se muestran las configuraciones para la conexión a la base de datos.

```
+ ✂ 📄 ▶ ■ ↺ ▶▶ Code ▼

[ ]: server = 'Server'
      database = 'DBname'
      username = 'user'
      password = 'password'
      cnxn = pyodbc.connect('DRIVER={SQL Server};SERVER='+server+';DATABASE='+database+';UID='+username+';PWD='+ password)
```

En el siguiente bloque de código se muestra como en la variable *query1* se almacena la consulta a la tabla *frutas_precios*. Después en el DataFrame *df_frutas* se almacena el resultado de la consulta. Posteriormente en un nuevo DataFrame *df_frutas20* se filtran los datos que tengan un precio menor a 20 del primer DataFrame. Finalmente se iteran los datos del DataFrame *df_frutas20* con una instrucción For para insertar cada uno de sus registros a la Tabla Destino *frutas_20*.

```
📄 📄 ▶ ■ ↺ ▶▶ Code ▼

query1 = "SELECT * FROM frutas_precios;"
df_frutas = pd.read_sql(query1, cnxn)
df_frutas20 = df_frutas[(df_frutas["precio"]<20)]

for index, row in df_frutas20.iterrows():
    cursor.execute("INSERT INTO [frutas20] (fruta_id,nombre,precio,existencias) values(?,?,?,?)", row.fruta_id, row.nombre, row.precio,row.existencias)
cnxn.commit()
```

De forma similar, en el siguiente bloque de código se muestra como en la variable *query2* se almacena la consulta a la tabla *frutas_ventas*. Después en el DataFrame *df_ventas* se almacena el resultado de la consulta. Posteriormente en un nuevo DataFrame *df_ventashoy* se filtran los datos que tengan en la columna fecha; la fecha del día actual del primer DataFrame. Finalmente se iteran los datos del DataFrame *df_ventashoy* con una instrucción For para insertar cada uno de sus registros a la Tabla Destino *ventas_dia*.

```
📄 📄 ▶ ■ ↺ ▶▶ Code ▼

query2 = "SELECT * FROM frutas_ventas;"
df_ventas = pd.read_sql(query2, cnxn)
df_ventashoy = df_ventas[(df_ventas["fecha"] == date.today().strftime('%Y-%m-%d'))]

for index, row in df_ventashoy.iterrows():
    cursor.execute("INSERT INTO [ventas_dia] (venta_id ,fruta_id ,fecha) values(?,?,?)", row.venta_id, row.fruta_id, row.fecha)
cnxn.commit()
```

RESULTADOS

Cómo se puede observar en la siguiente imagen se encuentran solamente las frutas con un precio menor a 20 pesos en la tabla destino *frutas20*, siendo lo especificado en el filtro.

```
SELECT *  
FROM [frutas20]
```

%

Results Messages

fruta_id	nombre	precio	existencias
1	Manzana	5,00	520
3	Pera	8,00	960
8	Naranja	13,00	970

De igual forma se puede observar en la siguiente imagen que solamente fueron insertados los registros con la fecha del día en la tabla destino *ventas_dia*. (Cabe señalar que ésta práctica fue realizada el día 15 de Julio de 2023, siendo considerada ésta le fecha del día).

```
SELECT *  
FROM [ventas_dia]
```

results Messages

venta_id	fruta_id	fecha
6	4	2023-07-15
7	8	2023-07-15
8	2	2023-07-15
9	10	2023-07-15
10	9	2023-07-15

CONCLUSIONES Y RESUMEN.

Como se pudo observar se hizo uso de la biblioteca *pyodbc* para poder establecer conexión con la base de datos, y también se usó *DataFrames* para poder aplicar los filtros solicitados a los datos.