



**INSTITUTO POLITECNICO NACIONAL**  
**CENTRO DE INVESTIGACION EN COMPUTACIÓN**



## **PROGRAMACIÓN PYTHON EN EL ÁMBITO CIENTÍFICO**

### **PRÁCTICA DE EVALUACIÓN**

**Uso de JOIN con DatFrme**

**ALUMNO:**

**Félix Saucedo Garnica**

**INSTRUCTOR:**

**Alan Badillo Salas**

Ciudad de México. Julio 2023

# ÍNDICE

<b>1. Planteamiento</b> . . . . .	<b>3</b>
<b>2. Objetivos</b> . . . . .	<b>4</b>
2.1 JOIN. . . . .	4
2.2 Agrupar y sumar ventas . . . . .	4
2.3 Agrupar productos y sums sus ventas. . . . .	4
2.4 Reportar los valores de los montos obtenidos en la impresión estándar. . .	5
<b>3. Derarrollo</b> . . . . .	<b>6</b>
3.1 Creación de un DataFrame para los productos. . . . .	6
3.2 Creación de un DataFrame para las ventas . . . . .	7
3.3 Combinación entre los DataFrame ventas y productos	8
3.4 Agrupación de ventas y su total	9
3.5 Agrupación de productos y su total	10
<b>4. Resultados</b> . . . . .	<b>11</b>
<b>5. Conclusiones</b> . . . . .	<b>14</b>

## 1. PLANTEAMIENTO

Unir dos DataFrames de productos y ventas relacionados mediante un JOIN. Construir dos DataFrames, uno de productos y uno de ventas, donde el primer DataFrame tendrá las columnas necesarias (PRODUCTO\_ID, NOMBRE Y PRECIO) para guardar cada producto y el segundo tendrá columnas (VENTA\_ID, PRODUCTO\_ID, PRECIO ) para guardar cada venta.

DataFrame de Productos.

PRODUCTO_ID	NOMBRE	PRECIO
1	Coca Cola	17.5
2	Galletas Marías	18.9
3	Gansito	21.3
4	Pepsi	16.0
5	Chetos	11.5

Aquí observamos las columnas que contienen la información de cada producto, un PRODUCTO\_ID que lo identifica, NOMBRE que indica de qué producto se trata y el PRECIO de cada uno de ellos.

DataFrame de Ventas.

VENTA_ID	PRODUCTO_ID	FECHA
101	1	2023-07-05 11:55:00
101	3	2023-07-05 11:55:00
101	1	2023-07-05 11:55:00
102	2	2023-07-05 11:57:00
102	2	2023-07-05 11:57:00
102	3	2023-07-05 11:57:00
103	5	2023-07-05 12:31:00
103	4	2023-07-05 12:31:00

Para las ventas se debe tener un VENTA\_ID para cada venta, este ID se puede repetir si es que se tiene más de un producto en una venta. Se tiene también un PRODUCTO\_ID para los productos que se vendieron y la fecha y hora de la venta.

## 2. OBJETIVOS

### 2.1 JOIN

Crea un DataFrame que extienda el DataFrame de Ventas con el DataFrame de los Productos usando el operado JOIN. De tal forma que se obtenga un nuevo DataFrame que contenga la información contenida en las ventas y que además contenga la información relacionada con los productos, es decir se debe obtener por cada venta: el ID de la venta, el ID del producto, la fecha de la venta, el nombre del producto vendido y el precio del producto. Así el nuevo DataFrame debe tener las siguientes columnas:

VENTA_ID	PRODUCTO_ID	FECHA	NOMBRE	PRECIO
----------	-------------	-------	--------	--------

### 2.2 AGRUPAR Y SUMAR VENTAS

Agrupar el DataFrame extendido que se obtuvo del JOIN para calcular el monto total por cada venta (suma la columna precio). Se Debe obtener dos columnas, una donde indique el ID de la venta y otra columna donde se observe el monto total de esa venta. De modo que se tenga un solo registro por cada venta Esta agrupación debe presentar la información como se indica a continuación:

VENTA_ID	TOTAL
----------	-------

### 2.3 AGRUPAR PRODUCTOS Y SUMAR SUS VENTAS

Agrupar el DataFrame extendido que se obtuvo del JOIN para calcular el monto total por la venta de cada producto (suma la columna precio). Se Debe obtener dos columnas, una donde indique el ID del producto y otra columna donde se observe el monto total por las ventas de ese producto. Esta agrupación debe presentar la información como se indica a continuación:

PRODUCTO_ID	TOTAL
-------------	-------

## **2.4 REPORTA LOS VALORES DE LOS MONTOS OBTENIDOS EN LA IMPRESIÓN ESTÁNDAR**

Mostrar en capturas de pantalla los resultados obtenidos en esta práctica. Se debe observar los DataFrame de los productos, el DataFrame de las Ventas, el DataFrame extendido, el resultado de la agrupación de las ventas con su monto total y el resultado de la agrupación de los productos y el monto total resultado de sus ventas.

### 3. DESARROLLO

Para el desarrollo de esta práctica se usará principalmente los objetos DataFrame de la librería **Pandas** de Python aprovechando algunas de sus características y funciones, así que lo primero que hacemos es importar dicha librería.

```
import pandas as pd
```

#### 3.1 CREACIÓN DE UN DataFrame PARA LOS PRODUCTOS

Una vez que se importó la librería Pandas procedemos a crear el primer DataFrame para los productos. Recordemos que este DataFrame debe contener principalmente 3 columnas: PRODUCTO\_ID, NOMBRE Y PRECIO. Existen diversas formas de crear un DataFrame y en esta práctica se usará la siguiente forma:

```
producto = pd.DataFrame(  
{  
    "PRODUCTO_ID" : [1, 2, 3, 4, 5],  
    "NOMBRE"       : ["Coca Cola", "Galletas Marías", "Gansito",  
                     "Pepsi", "Chetos"],  
    "PRECIO"       : [17.5, 18.9, 21.3, 16.0, 11.5]  
})
```

Esto nos permite crear el DataFrame llamado **producto** y al mismo tiempo agregar cinco productos. Observamos que se tienen las 3 columnas solicitadas y a continuación se muestra el resultado de la ejecución.

	producto		
	PRODUCTO_ID	NOMBRE	PRECIO
0	1	Coca Cola	17.5
1	2	Galletas Marías	18.9
2	3	Gansito	21.3
3	4	Pepsi	16.0
4	5	Chetos	11.5

### 3.2 CREACIÓN DE UN DataFrame PARA LAS VENTAS

Así como hicimos para los productos procedemos ahora para las ventas creando el DataFrame y al mismo tiempo poniendo en su contenido 3 operaciones de venta. La primera venta con 3 productos, la segunda con 3 productos y la tercera con dos productos. Tal como se observa en las indicaciones de la práctica.

```
venta = pd.DataFrame(
({
"VENTA_ID"      : [101,101,101,102,102,102,103,103],
"PRODUCTO_ID"  : [1,3,1,1,2,2,5,4],
"FECHA"         : ["05/07/2023 11:55","05/07/2023 11:55",
                  "05/07/2023 11:55","05/07/2023 11:57",
                  "05/07/2023 11:57","05/07/2023 11:57",
                  "05/07/2023 12:31","05/07/2023 12:31"]
}))
```

Podemos observar que a la vez que se crea el DataFrame **venta**, agregamos los 8 registros indicados. El resultado de la ejecución se muestra enseguida.

venta			
	VENTA_ID	PRODUCTO_ID	FECHA
0	101	1	05/07/2023 11:55
1	101	3	05/07/2023 11:55
2	101	1	05/07/2023 11:55
3	102	1	05/07/2023 11:57
4	102	2	05/07/2023 11:57
5	102	2	05/07/2023 11:57
6	103	5	05/07/2023 12:31
7	103	4	05/07/2023 12:31

### 3.3 COMBINACIÓN ENTRE LOS DataFrame VENTAS Y PRODUCTOS

Usaremos el método `merge()` para crear un DataFrame que combine el DataFrame de Ventas con el DataFrame de los Productos. El método `merge()` de Pandas nos permite realizar "joins" entre tablas y ofrece muchas posibilidades para combinar los elementos de dos DataFrame. Una de esas posibilidades de combinación es **"inner"**, la cual solamente devuelve los registros de ambos objetos DataFrame que cumplan la condición de unión, esta es la función optima para nuestra práctica, así aseguramos que el resultado de la combinación incluya únicamente las ventas y aquellos productos que han sido vendidos. Queremos solamente la información de las ventas combinada con la información de los productos de dichas ventas y para lograrlo usamos `merge()` como se muestra a continuación:

```
ventas_combin = pd.merge(  
    venta, producto,  
    how="inner",  
    on=["PRODUCTO_ID", "PRODUCTO_ID"]  
)
```

Observamos que al método `merge()` le pasamos los DataFrame de **venta** y **producto**, le indicamos que la combinación será de tipo **"inner"** y finalmente indicamos la condición de la combinación que será a través de las columnas `PRODUCTO_ID` de cada DataFrame. El resultado de esto es un nuevo DataFrame llamado **ventas\_combin** que se muestra a continuación.

ventas_combin						
	VENTA_ID	PRODUCTO_ID	FECHA	NOMBRE	PRECIO	
0	101	1	05/07/2023 11:55	Coca Cola	17.5	
1	101	1	05/07/2023 11:55	Coca Cola	17.5	
2	102	1	05/07/2023 11:57	Coca Cola	17.5	
3	101	3	05/07/2023 11:55	Gansito	21.3	
4	102	2	05/07/2023 11:57	Galletas Marías	18.9	
5	102	2	05/07/2023 11:57	Galletas Marías	18.9	
6	103	5	05/07/2023 12:31	Chetos	11.5	
7	103	4	05/07/2023 12:31	Pepsi	16.0	



### 3.4 AGRUPACIÓN DE VENTAS Y SU TOTAL

Inicialmente pensé hacer ciclos for y while para resolver esta parte, sin embargo encontré una forma más fácil y directa para hacerlo. En pandas existen las funciones **groupby** y **agg** que nos servirán para obtener la agrupación de las ventas y el total de estas. Usamos **groupby** para agrupar las ventas y **agg** para obtener la suma. El criterio de agrupación es, para este caso, la columna VENTA\_ID y la suma se hará sobre la columna PRECIO dada la agrupación anterior. Mostramos el código correspondiente:

```
grupo_suma =  
    ventas_combin.groupby(["VENTA_ID"])["PRECIO"].agg('sum')
```

El resultado de esta agrupación y suma es el siguiente:

```
grupo_suma  
VENTA_ID  
101      56.3  
102      55.3  
103      27.5  
Name: PRECIO, dtype: float64
```

Visualizamos los totales por cada venta. Sin embargo, están mostrados de una forma simple, así que para darle un mejor formato le pasamos el resultado anterior a un DataFrame de la siguiente manera:

```
grupo_suma = pd.DataFrame(  
    ventas_combin.groupby(["VENTA_ID"])["PRECIO"].agg('sum')  
)
```




Con esto el resultado se ve más presentable

```
grupo_suma  
PRECIO  
VENTA_ID  
101      56.3  
102      55.3  
103      27.5
```

Esta es una mejor presentación, no obstante el nombre de la columna del total de la venta es PRECIO. Para corregir esto cambiamos el nombre de esa columna a TOTAL

```
grupo_suma.rename(columns={'PRECIO': 'TOTAL_VENTA'})
```

En el resultado final tenemos las ventas agrupadas y su total ya con algunas correcciones:





TOTAL_VENTA	
VENTA_ID	
101	56.3
102	55.3
103	27.5

### 3.5 AGRUPACIÓN DE PRODUCTOS Y SU TOTAL

En este punto deberíamos obtener algo similar al punto anterior con la diferencia de que ahora la agrupación es sobre los productos y ahora el total representará el monto total por la venta de cada producto. La lógica es la misma que con las ventas por lo que se muestra el código y resultado final.

```
grupo_producto = pd.DataFrame(  
    ventas_combin.groupby(["PRODUCTO_ID"])["PRECIO"].agg('sum')  
)  
grupo_producto.rename(columns={'PRECIO': 'TOTAL'})
```



TOTAL	
PRODUCTO_ID	
1	52.5
2	37.8
3	21.3
4	16.0
5	11.5

## 4. RESULTADOS

Para poder observar los resultados obtenidos se muestran los DataFrame de Ventas y Productos:

PRODUCTOS				VENTAS			
				VENTA_ID	PRODUCTO_ID	FECHA	
PRODUCTO_ID	NOMBRE	PRECIO					
0	1	Coca Cola	17.5	0	101	1	05/07/2023 11:55
1	2	Galletas Marías	18.9	1	101	3	05/07/2023 11:55
2	3	Gansito	21.3	2	101	1	05/07/2023 11:55
3	4	Pepsi	16.0	3	102	1	05/07/2023 11:57
4	5	Chetos	11.5	4	102	2	05/07/2023 11:57
				5	102	2	05/07/2023 11:57
				6	103	5	05/07/2023 12:31
				7	103	4	05/07/2023 12:31

A continuación tenemos la combinación de las Ventas y Productos

COMBINACIÓN DE VENTAS Y PRODUCTOS						
VENTA_ID	PRODUCTO_ID	FECHA	NOMBRE	PRECIO		
0	101	1	05/07/2023 11:55	Coca Cola	17.5	
1	101	1	05/07/2023 11:55	Coca Cola	17.5	
2	102	1	05/07/2023 11:57	Coca Cola	17.5	
3	101	3	05/07/2023 11:55	Gansito	21.3	
4	102	2	05/07/2023 11:57	Galletas Marías	18.9	
5	102	2	05/07/2023 11:57	Galletas Marías	18.9	
6	103	5	05/07/2023 12:31	Chetos	11.5	
7	103	4	05/07/2023 12:31	Pepsi	16.0	

Ahora para verificar la agrupación de ventas y su total nos apoyamos de la siguiente imagen donde se indica con colores las cantidades que se deben sumar y su resultado respecto a las ventas.

	VENTA_ID	PRODUCTO_ID	FECHA	NOMBRE	PRECIO	
0	101	1	05/07/2023 11:55	Coca Cola	17.5	17.5
1	101	1	05/07/2023 11:55	Coca Cola	17.5	+ 17.5
2	102	1	05/07/2023 11:57	Coca Cola	17.5	21.3
3	101	3	05/07/2023 11:55	Gansito	21.3	56.3
4	102	2	05/07/2023 11:57	Galletas Marías	18.9	18.9
5	102	2	05/07/2023 11:57	Galletas Marías	18.9	+ 18.9
6	103	5	05/07/2023 12:31	Chetos	11.5	17.5
7	103	4	05/07/2023 12:31	Pepsi	16.0	55.3
						+ 11.5
						16.0
						27.5

Observamos los montos totales para cada una de las ventas marcadas con color rojo para la venta 101, amarillo para la venta 102 y verde para la venta 103. Esto nos sirve para comprobar el resultado de la agrupación de las ventas y su total.

TOTAL_VENTA	
VENTA_ID	
101	56.3
102	55.3
103	27.5

De forma similar a la agrupación de ventas usamos la siguiente imagen para verificar la agrupación por productos. Vemos con colores las cantidades que se deben sumar y su resultado respecto a los productos.

	VENTA_ID	PRODUCTO_ID	FECHA	NOMBRE	PRECIO	
0	101	1	05/07/2023 11:55	Coca Cola	17.5	17.5 x 3
1	101	1	05/07/2023 11:55	Coca Cola	17.5	52.5
2	102	1	05/07/2023 11:57	Coca Cola	17.5	18.9 x 2
3	101	3	05/07/2023 11:55	Gansito	21.3	37.8
4	102	2	05/07/2023 11:57	Galletas Marías	18.9	21.3
5	102	2	05/07/2023 11:57	Galletas Marías	18.9	16.0
6	103	5	05/07/2023 12:31	Chetos	11.5	11.5
7	103	4	05/07/2023 12:31	Pepsi	16.0	

Así obtenemos el monto total de cada producto vendido. En color rojo para el producto 1, amarillo para el producto 2, verde para el producto 3, azul para el producto 4 y gris para el producto 5. Con esto podemos comporbar el resultado de la agrupación por producto y su monto total.

PRODUCTO_ID	
1	52.5
2	37.8
3	21.3
4	16.0
5	11.5

Estos son resultados finales obtenidos en esta práctica.

## 5. CONCLUSIONES

En el desarrollo de esta práctica se usaron elementos vistos en clase pero también fue necesaria la investigación de algunas cosas para poder terminarla exitosamente, ya que se contaba los elementos básicos para empezar a desarrollarla y en este proceso se presentaron tareas desconocidas que requerían el conocimiento de otras funciones y elementos de Python. Esta situación muestra que se cuenta con los fundamentos necesarios para ser autodidacta y poder resolver problemas posteriormente, con ayuda de libros, videos, tutoriales, blogs, etc.

Durante la práctica intenté resolver el problema de la agrupación de forma manual usando ciclos for y while, pero gracias a la indagación y búsqueda, además de leer detenidamente las instrucciones de la práctica, pude descubrir una manera más sencilla de resolverlo. Comprobando, así que ya somos capaces de aplicar lo aprendido combinado con la búsqueda en diferentes medios.

El enfoque del curso de aplicar el lenguaje Python en el ámbito científico es por demás importante dado los avances tecnológicos y su aplicación en la vida diaria. Así también considero notable la interpretación que se dieron a los resultados obtenidos, no solo en esta práctica, sino también a todos los ejercicios y ejemplos vistos en clase. Vimos resultados que son aplicables en el ámbito empresarial como la venta y compra de productos, o en el ámbito científico como el análisis de los datos del Titanic, y los del área de la educación cuando analizamos las edades y situación financiera de los alumnos, etc. No son ejercicios ni resultados aislados, tienen una razón de ser y una interpretación importante. Por esta razón en esta práctica es destacable no solo la implementación del código que resuelve los planteamientos, sino también la interpretación de los resultados en la sección 4.6, donde además de la combinación de información para una mejor visualización de los datos, podemos hacer otro tipo de operaciones como la agrupación y la suma o promedio. Con esta agrupación y suma podemos ver qué producto está dejando más ganancias o qué producto es el que menos se vende y así poder tomar decisiones importantes que ayuden en el inventario y a surtir el almacén o poner productos en oferta para que se vendan rápido.

Esto es solo una pequeña muestra de la enorme cantidad de aplicaciones que tiene la programación python, no solo en el ámbito científico, sino también en múltiples disciplinas.