

```
In [1]:
```

```
In [2]:
```

```
v.reader(fichero_csv)

# Leer las primeras dos filas (pueden ser encabezados previos o datos inicia
for _ in range(2):
    next(lector)

# Leer la tercera fila
```

```
In [3]:
```

un DataFrame de pandas

```
df = pd.read_csv(archivo_csv, encoding='latin1', skiprows=2)
```

```
# Muestra el DataFrame
print(df)
```

	Date	Food_Price_Index	Meat	Dairy	Cereals	Oils	Sugar	\
0	1990-01	64.1	73.4	53.5	64.1	44.59	87.9	
1	1990-02	64.5	76.0	52.2	62.2	44.50	90.7	
2	1990-03	63.8	77.8	41.4	61.3	45.75	95.1	
3	1990-04	65.8	80.4	48.4	62.8	44.02	94.3	
4	1990-05	64.4	81.0	39.2	62.0	45.50	90.4	
..	
398	2023-07	127.0	114.7	126.8	138.6	131.80	127.0	
399	2023-04	127.7	116.8	122.6	136.1	130.00	149.4	
400	2023-05	124.2	118.1	117.8	129.3	118.70	157.2	
401	2023-06	122.4	118.1	116.7	126.6	115.80	152.2	
402	2023-07	123.9	117.8	116.3	125.9	129.80	146.3	

	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 56	Unnamed: 57	\
0	NaN	NaN	NaN	...	NaN	NaN	
1	NaN	NaN	NaN	...	NaN	NaN	
2	NaN	NaN	NaN	...	NaN	NaN	
3	NaN	NaN	NaN	...	NaN	NaN	
4	NaN	NaN	NaN	...	NaN	NaN	
..	
398	NaN	NaN	NaN	...	NaN	NaN	
399	NaN	NaN	NaN	...	NaN	NaN	
400	NaN	NaN	NaN	...	NaN	NaN	
401	NaN	NaN	NaN	...	NaN	NaN	
402	NaN	NaN	NaN	...	NaN	NaN	

	Unnamed: 58	Unnamed: 59	Unnamed: 60	Unnamed: 61	Unnamed: 62	\
0	NaN	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	NaN	
..	
398	NaN	NaN	NaN	NaN	NaN	
399	NaN	NaN	NaN	NaN	NaN	
400	NaN	NaN	NaN	NaN	NaN	
401	NaN	NaN	NaN	NaN	NaN	
402	NaN	NaN	NaN	NaN	NaN	

	Unnamed: 63	Unnamed: 64	Unnamed: 65	
0	NaN	NaN	NaN	
1	NaN	NaN	NaN	
2	NaN	NaN	NaN	
3	NaN	NaN	NaN	
4	NaN	NaN	NaN	
..	
398	NaN	NaN	NaN	
399	NaN	NaN	NaN	
400	NaN	NaN	NaN	
401	NaN	NaN	NaN	
402	NaN	NaN	NaN	

[403 rows x 66 columns]

```
In [4]:
```

```
In [5]: Muestra el DataFrame original
```

```
print("DataFrame original:")
print(df.head())
```

DataFrame original:

	Date	Food_Price_Index	Meat	Dairy	Cereals	Oils	Sugar
0	1990-01	64.1	73.4	53.5	64.1	44.59	87.9
1	1990-02	64.5	76.0	52.2	62.2	44.50	90.7
2	1990-03	63.8	77.8	41.4	61.3	45.75	95.1
3	1990-04	65.8	80.4	48.4	62.8	44.02	94.3
4	1990-05	64.4	81.0	39.2	62.0	45.50	90.4

```
In [6]: # Asegúrate de que la columna 'Date' sea de tipo datetime
```

```
df['Date'] = pd.to_datetime(df['Date'])
```

```
# Crea una nueva columna 'ID_YYYY' para el año de cada producto
```

```
df['ID_YYYY'] = df['Date'].dt.year.astype(str) + '_' + df.index.astype(str)
```

```
In [7]: # Muestra el DataFrame con la nueva columna 'ID_YYYY'
```

```
print("\nDataFrame con columna 'ID_YYYY':")
print(df.head())
```

DataFrame con columna 'ID_YYYY':

	Date	Food_Price_Index	Meat	Dairy	Cereals	Oils	Sugar	ID_YYYY
0	1990-01-01	64.1	73.4	53.5	64.1	44.59	87.9	1990_0
1	1990-02-01	64.5	76.0	52.2	62.2	44.50	90.7	1990_1
2	1990-03-01	63.8	77.8	41.4	61.3	45.75	95.1	1990_2
3	1990-04-01	65.8	80.4	48.4	62.8	44.02	94.3	1990_3
4	1990-05-01	64.4	81.0	39.2	62.0	45.50	90.4	1990_4

```
In [8]: # Calcula la media anual del precio de cada producto
```

```
df_media_anual = df.groupby(['ID_YYYY']).mean()
```

```
In [9]: # MEDIA ANUAL RESULTADOS
```

```
print("\nDataFrame con la media anual del precio de cada producto:")
print(df_media_anual.head())
```

DataFrame con la media anual del precio de cada producto:

	Date	Food_Price_Index	Meat	Dairy	Cereals	Oils	Sugar
ID_YYYY							
1990_0	1990-01-01	64.1	73.4	53.5	64.1	44.59	87.9
1990_1	1990-02-01	64.5	76.0	52.2	62.2	44.50	90.7
1990_10	1990-11-01	61.9	86.2	39.4	52.4	48.30	62.3
1990_11	1990-12-01	61.9	83.7	45.1	52.6	49.60	60.3
1990_2	1990-03-01	63.8	77.8	41.4	61.3	45.75	95.1

```
In [10]: # Calcula la media anual del precio de cada producto
```

```
df_media_anual = df.groupby(['ID_YYYY']).mean()
```

```
# Muestra el DataFrame con la media anual
print("\nDataFrame con la media anual del precio de cada producto:")
print(df_media_anual.head())
```

DataFrame con la media anual del precio de cada producto:

	Date	Food_Price_Index	Meat	Dairy	Cereals	Oils	Sugar	
ID_YYYY								
1990_0	1990-01-01		64.1	73.4	53.5	64.1	44.59	87.9
1990_1	1990-02-01		64.5	76.0	52.2	62.2	44.50	90.7
1990_10	1990-11-01		61.9	86.2	39.4	52.4	48.30	52.3
1990_11	1990-12-01		61.9	83.7	45.1	52.6	49.60	60.3
1990_2	1990-03-01		63.8	77.8	41.4	61.3	45.75	95.1

```
# Calcula la correlación entre las columnas del DataFrame
correlacion = df.corr()

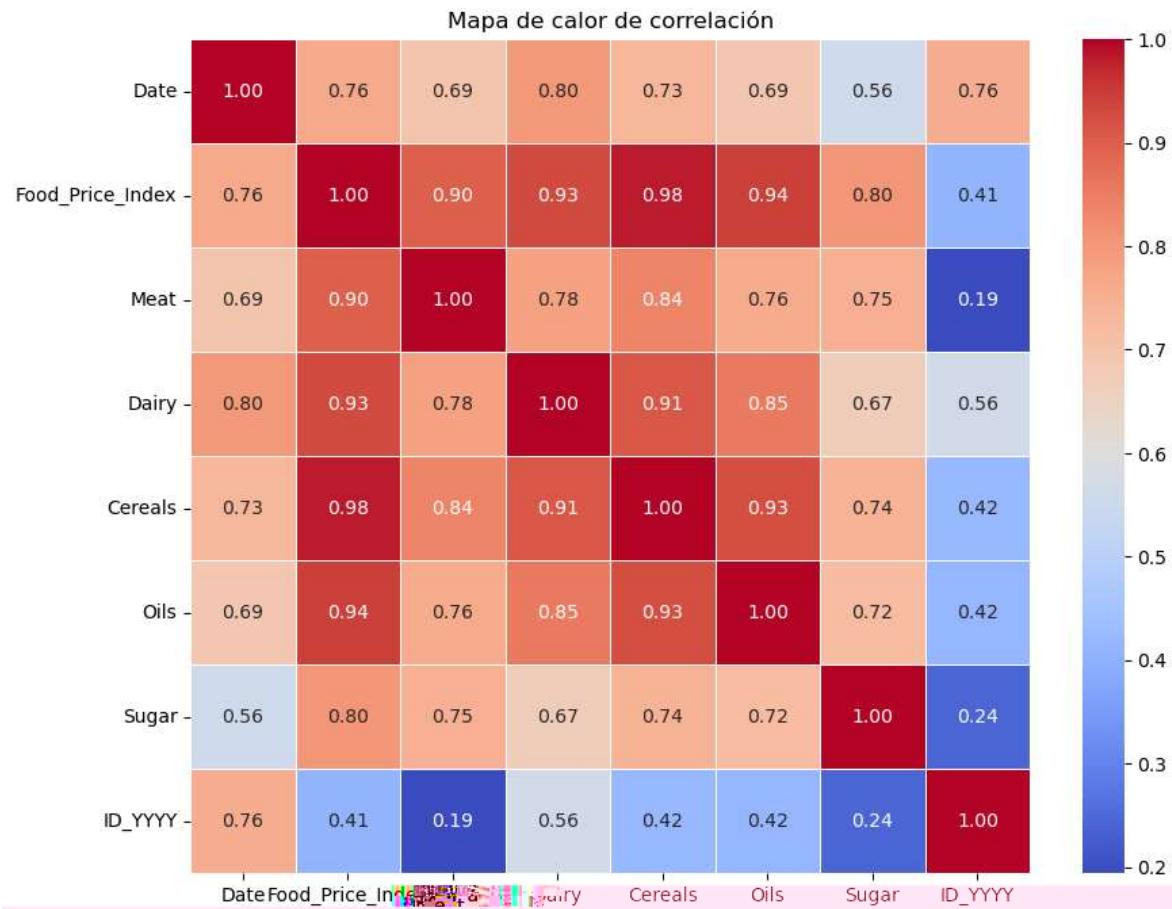
# Muestra la matriz de correlación
print("Matriz de correlación:")
print(correlacion)

# Si solo deseas ver la correlación con una columna específica (por ejemplo
```

```

sns.heatmap(correlacion, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Mapa de calor de correlación')
plt.show()

```



La correlación con otras variables, como 'Meat', 'Dairy', 'Cereals', 'Oils', y 'Sugar', es bastante alta, especialmente con 'Cereals' y 'Oils'. Esto sugiere una fuerte relación lineal positiva entre el índice de precios de los alimentos y estas categorías específicas de alimentos.

La correlación con otras variables, como 'Meat', 'Dairy', 'Cereals', 'Oils', y 'Sugar', es bastante alta, especialmente con 'Cereals' y 'Oils'. Esto sugiere una fuerte relación lineal positiva entre el índice de precios de los alimentos y estas categorías específicas de alimentos.

AHORA CON XLS

instalar pip install xlrd para abrir XLS

In [13]: `import pandas as pd`

```

# Especifica la ruta al archivo Excel
archivo_excel = 'padinpresafao.xlsx'

```

pip install openpyxl

In [14]: `# Lee las columnas del archivo Excel sin cargar los datos`
`columnas_excel = pd.read_excel(archivo_excel, sheet_name='Indices_Monthly_Real',`

```
In [15]: # Muestra las columnas
print("Columnas del archivo Excel:")
print(columnas_excel)
```

```
Columnas del archivo Excel:
Index([1990-01-01 00:00:00, 77.6276304143571, 88.79702716282455,
       64.74874473966906, 77.62221601837265, 53.9594811195994,
       106.34872983101246],  
      dtype='object')
```

```
In [16]: # Crea un DataFrame de pandas con datos de la hoja 'Indices_Monthly_Real'
df = pd.read_excel(archivo_excel, sheet_name='Indices_Monthly_Real')
```

```
In [17]: .read_excel(archivo_excel, sheet_name='Indices_Monthly_Real', skiprows=0)

# Muestra el DataFrame
print(df)
```

```

      DATE Food_Price_Index Meat_Price_Index Dairy_Price_Index \
0      NaT      NaN      NaN      NaN
1 1990-01-01      77.627630      88.797027      64.748745
2 1990-02-01      78.000947      91.921703      63.194383
3 ...
4 1990-01-01      79.590160      97.350032      58.605874
...
399 2023-02-01      122.142562      112.300202      121.025100
400 2023-04-01      122.834058      112.364211      117.873093
401 2023-05-01      119.400700      113.593066      113.265805
402 2023-06-01      117.715018      113.609584      112.256166
403 2023-07-01      119.193331      113.262411      111.808133

      Cereals_Price_Index Oils_Price_Index Sugar_Price_Index
0                  NaN      NaN      NaN
1      77.622216      53.959481      106.348730
2      75.300796      53.854003      109.718936
3      74.135998      55.360591      115.036373
4      76.024917      53.268974      114.137651
...
399     133.252084      126.743387      122.135135
400     130.930790      125.053977      143.681568
401     124.324199      114.140627      151.196154
402     121.757567      111.358605      146.334654
403     121.097298      124.838550      140.690499

```

[404 rows x 7 columns]

Eliminamos la linea vacia con index 0 , la que esta debajo de titulo columna

In [18]: `import pandas as pd`

```
# Carga Los datos del archivo Excel en un DataFrame, omitiendo La primera fila
df = pd.read_excel(archivo_excel, sheet_name='Indices_Monthly_Real', header=None)
```

Agregamos nombres a las columnas de la hoja 3 de archivo_excel

In [19]: `column_names = ['Date', 'Food_Price_Index', 'Meat_Price_Index', 'Dairy_Price_Index']
df.columns = column_names`

verificamos que esta bien montado el dataframe

In [20]: `# Muestra el DataFrame
print(df)`

	Date	Food_Price_Index	Meat_Price_Index	Dairy_Price_Index	\
0	1990-01-01	77.627630	88.797027	64.748745	
1	1990-02-01	78.000947	91.921703	63.194383	
2	1990-03-01	77.194362	94.159745	50.062011	
3	1990-04-01	77.910563	98.018607	47.426859	
4	1990-05-01	77.910563	98.018607	47.426859	
..
398	2023-03-01	122.143563	110.300392	121.935100	
399	2023-04-01	122.834058	112.364211	117.873093	
400	2023-05-01	119.400700	113.593066	113.265805	
401	2023-06-01	117.715018	113.609584	112.256166	
402	2023-07-01	119.193331	113.262411	111.808133	
		Cereals_Price_Index	Oils_Price_Index	Sugar_Price_Index	
0		77.622216	53.959481	106.348730	
1		75.300796	53.854003	109.718936	
2		74.135998	55.360591	115.036373	
3		76.024917	53.268974	114.137651	
4		75.010894	55.063124	109.344469	
..	
398		133.252084	126.743387	122.135135	
399		130.930790	125.053977	143.681568	
400		124.324199	114.140627	151.196154	
401		121.757567	111.358605	146.334654	
402		121.097298	124.838550	140.690499	

[403 rows x 7 columns]

```
In [21]: df['Date'] = pd.to_datetime(df['Date'])
df.iloc[:, 1:] = df.iloc[:, 1: ].round(2)
```

```
In [22]: # Muestra el DataFrame
print(df)
```

	Date	Food_Price_Index	Meat_Price_Index	Dairy_Price_Index	\
0	1990-01-01	77.63	88.80	64.75	
1	1990-02-01	78.00	91.92	62.10	
2	1990-03-01	77.19	94.16	50.96	
3	1990-04-01	79.59	97.35	58.61	
4	1990-05-01	77.91	98.02	47.43	
..	
398	2023-03-01	122.14	110.30	121.94	
399	2023-04-01	122.83	112.36	117.87	
400	2023-05-01	119.40	113.59	113.27	
401	2023-06-01	117.72	113.61	112.26	
402	2023-07-01	119.19	113.26	111.81	
		Cereals_Price_Index	Oils_Price_Index	Sugar_Price_Index	
0		77.62	53.96	106.35	
1		75.30	53.85	109.72	
2		74.14	55.36	115.04	
3		76.02	53.27	114.14	
4		75.01	55.06	109.34	
..		
398		133.25	126.74	122.14	
399		130.93	125.05	143.68	
400		124.32	114.14	151.20	
401		121.76	111.36	146.33	
402		121.10	124.84	140.69	

[403 rows x 7 columns]

```
In [23]: import pandas as pd

# Especifica la ruta al archivo Excel
archivo_excel = 'padinpresafao.xlsx' # Reemplaza con la ruta real de tu archivo

# Carga los datos del archivo Excel en un DataFrame, omitiendo la primera fila
df = pd.read_excel(archivo_excel, sheet_name='Indices_Monthly_Real', header=None)

# Agrega nombres a las columnas
column_names = ['Date', 'Food_Price_Index', 'Meat_Price_Index', 'Dairy_Price_Index']
df.columns = column_names

# Muestra el DataFrame
print(df)
```

	Date	Food_Price_Index	Meat_Price_Index	Dairy_Price_Index	\
0	1990-01-01	77.627630	88.797027	64.748745	
1	1990-02-01	78.000947	91.921703	63.194383	
2	1990-03-01	77.194362	94.159745	50.062011	
3	1990-04-01	77.910563	98.018607	47.426859	
4	1990-05-01	77.910563	98.018607	47.426859	
..
398	2023-03-01	122.143563	110.300392	121.935100	
399	2023-04-01	122.834058	112.364211	117.873093	
400	2023-05-01	119.400700	113.593066	113.265805	
401	2023-06-01	117.715018	113.609584	112.256166	
402	2023-07-01	119.193331	113.262411	111.808133	
	Cereals_Price_Index	Oils_Price_Index	Sugar_Price_Index		
0	77.622216	53.959481	106.348730		
1	75.300796	53.854003	109.718936		
2	74.135998	55.360591	115.036373		
3	76.024917	53.268974	114.137651		
4	75.010894	55.063124	109.344469		
..
398	133.252084	126.743387	122.135135		
399	130.930790	125.053977	143.681568		
400	124.324199	114.140627	151.196154		
401	121.757567	111.358605	146.334654		
402	121.097298	124.838550	140.690499		

[403 rows x 7 columns]

In [24]: `import pandas as pd`

```
# Supongamos que tu DataFrame se llama df
# Convierte la columna 'Date' a tipo datetime
df['Date'] = pd.to_datetime(df['Date'])

# Convierte las demás columnas a tipo numérico
numeric_columns = ['Food_Price_Index', 'Meat_Price_Index', 'Dairy_Price_Index',
df[numeric_columns] = df[numeric_columns].apply(pd.to_numeric, errors='coerce')

# Muestra la información actualizada sobre el DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 403 entries, 0 to 402
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Date             403 non-null    datetime64[ns]
 1   Food_Price_Index 403 non-null    float64 
 2   Meat_Price_Index 403 non-null    float64 
 3   Dairy_Price_Index 403 non-null    float64 
 4   Cereals_Price_Index 403 non-null  float64 
 5   Oils_Price_Index  403 non-null    float64 
 6   Sugar_Price_Index 403 non-null    float64 
dtypes: datetime64[ns](1), float64(6)
memory usage: 22.2 KB
```

In [25]: `# Muestra el DataFrame`
`print(df)`

	Date	Food_Price_Index	Meat_Price_Index	Dairy_Price_Index	\
0	1990-01-01	77.627630	88.797027	64.748745	
1	1990-02-01	78.000947	91.921703	63.194383	
2	1990-03-01	77.194362	94.159745	50.062011	
3	1990-04-01	79.590160	97.350032	58.605874	
4	1990-05-01	77.910563	98.018607	47.426859	
..
398	2023-03-01	122.143563	110.300392	121.935100	
399	2023-04-01	122.834058	112.364211	117.873093	
400	2023-05-01	119.400700	113.593066	113.265805	
401	2023-06-01	117.715018	113.609584	112.256166	
402	2023-07-01	119.193331	113.262411	111.808133	
	Cereals_Price_Index	Oils_Price_Index	Sugar_Price_Index		
0	77.622216	53.959481	106.348730		
1	75.300796	53.854003	109.718936		
2	74.135998	55.360591	115.036373		
3	76.024917	53.268974	114.137651		
4	75.010894	55.063124	109.344469		
..
398	133.252084	126.743387	122.135135		
399	130.930790	125.053977	143.681568		
400	124.324199	114.140627	151.196154		
401	121.757567	111.358605	146.334654		
402	121.097298	124.838550	140.690499		

[403 rows x 7 columns]

```
# Crea una nueva columna 'ID_YYYY' para el año de cada producto
df['ID_
```

ID_YYYY	Date	Cloud	Price	Index	Month	Day	Time	Year
1990	1990-06-16	12:00:00	0.00000000	76.258216	98.680344			
1991	1991-06-16	12:00:00	0.00000000	75.805676	98.482114			
1992	1992-06-16	08:00:00	0.00000000	76.658137	92.939991			
1993	1993-06-16	12:00:00	0.00000000	71.795337	85.849373			
1994	1994-06-16	12:00:00	0.00000000	70.385917	90.015506			
1995	1995-06-16	12:00:00	0.00000000	83.401894	92.124173			
1996	1996-06-16	08:00:00	0.00000000	86.142104	92.969130			
1997	1997-06-16	12:00:00	0.00000000	82.214188	91.829277			
1998	1998-06-16	12:00:00	0.00000000	78.728758	80.093022			
1999	1999-06-16	12:00:00	0.00000000	68.489556	76.699664			
2000	2000-06-16	08:00:00	0.00000000	67.059820	75.828585			
2001	2001-06-16	12:00:00	0.00000000	71.811902	80.660335			
2002	2002-06-16	12:00:00	0.00000000	70.225184	72.023100			
2003	2003-06-16	12:00:00	0.00000000	72.623180	73.320792			
2004	2004-06-16	08:00:00	0.00000000	77.136082	79.502874			
2005	2005-06-16	12:00:00	0.00000000	76.860779	81.871738			
2006	2006-06-16	12:00:00	0.00000000	80.721572	79.462067			
2007	2007-06-16	12:00:00	0.00000000	98.816024	80.587139			
2008	2008-06-16	08:00:00	0.00000000	114.328409	87.720772			
2009	2009-06-16	12:00:00	0.00000000	95.069817	84.243285			
2010	2010-06-16	12:00:00	0.00000000	106.796273	91.047427			
2011	2011-06-16	12:00:00	0.00000000	118.832658	94.922408			
2012	2012-06-16	08:00:00	0.00000000	111.497389	95.273985			
2013	2013-06-16	12:00:00	0.00000000	109.510425	96.779063			
2014	2014-06-16	12:00:00	0.00000000	106.217020	102.715965			
2015	2015-06-16	12:00:00	0.00000000	95.120644	98.887943			
2016	2016-06-16	08:00:00	0.00000000	97.806350	96.945203			
2017	2017-06-16	14:00:00	0.00000000	100.807798	100.519022			
2018	2018-06-16	22:00:00	0.00000000	94.213040	93.207948			
2019	2019-06-16	16:00:00	0.00000000	95.649945	100.567634			
2020	2020-06-16	08:00:00	0.00000000	99.180189	96.563352			
2021	2021-06-16	12:00:00	0.00000000	125.075926	107.161635			
2022	2022-06-16	12:00:00	0.00000000	140.553208	116.213194			
2023	2023-04-01	06:51:25	714285824	121.621711	111.282832			

ID_YYYY	Rainy	Cloud	Index	Price	Month	Day	Time	Year
1990		51.579629		70.287439		55.026992		
1991		55.526158		70.676036		59.859303		
1992		65.707631		73.088803		63.537907		
1993		56.347229		68.403377		62.547913		
1994		56.662541		74.008495		85.874993		
1995		68.272294		77.112969		87.029628		
1996		68.371701		92.822781		77.598555		
1997		69.221382		77.446682		82.306452		
1998		67.911589		71.719753		102.136657		
1999		59.976201		65.946190		72.285125		
2000		68.491322		64.657498		53.939579		
2001		79.546993		67.736993		55.502830		
2002		60.940517		73.468108		72.898223		
2003		68.465501		74.621951		78.703021		
2004		82.190309		75.309863		81.911139		
2005		88.077745		69.319780		73.463156		
2006		81.276967		79.246540		78.486366		
2007		128.324380		105.792715		112.509529		
2008		128.752892		133.914425		137.252551		
2009		94.804720		100.804013		97.938711		
2010		111.953936		107.552120		122.010090		

2011	11	936249	128.088998	140.994074
2012	10	346051	124.710627	125.535018
2013	12	455810	117.702115	108.940382
2014	12	345458	107.046903	102.238845
2015	8	987348	98.010146	91.945606
2016	8	936718	93.958842	105.805648
2017	11	979992	93.569227	104.804837
2018	10	366355	99.031550	86.202287
2019	10	406045	97.158464	83.717151
2020	10	922211	104.220505	100.507993
2021	11	495544	130.473056	163.996004
2022	13	290816	151.327007	183.726982
2023	11	703383	130.615862	123.977620

ID_YYYY	Sugar_Pri	Index
1990	9	016271
1991	6	765144
1992	6	95786
1993	7	393875
1994	8	50451
1995	8	69274
1996	8	97267
1997	8	58381
1998	6	37794
1999	4	66360
2000	6	47412
2001	6	60854
2002	5	58629
2003	5	08414
2004	5	81387
2005	6	09343
2006	10	81251
2007	6	88209
2008	7	50729
2009	11	98671
2010	13	88768
2011	14	77272
2012	12	07640
2013	9	81607
2014	9	28517
2015	8	35039
2016	11	65837
2017	10	28092
2018	7	27257
2019	7	66342
2020	8	80004
2021	10	67494
2022	11	85530
2023	13	19460

```
In [27]: import pandas as pd

# Asegúrate de que la columna 'Date' sea de tipo datetime
df_media_anual['Date'] = pd.to_datetime(df_media_anual['Date'])

# Crea un ID correlativo a la columna 'ID_YYYY'
df_media_anual['ID'] = df_media_anual.groupby(df_media_anual['ID_YYYY']).dt.yea
```

```
# Combina 'ID_YYYY' y 'Date' para formar el nuevo formato
df_media_anual['Date'] = 'ID_' + df_media_anual['ID_YYYY'].astype(str) + '_' + df_media_anual['Date'].dt.strftime('%m-%d-%Y')

# Elimina la columna temporal 'ID_YYYY' si no la necesitas
df_media_anual.drop('ID_YYYY', axis=1, inplace=True)

# Muestra el DataFrame resultante
print(df_media_anual)
```

ID_YYYY	Date	Food_Price_Index	Meat_Price_Index	Dairy_Price_Index	\
1990	ID_1_1990	76.258216	98.680344	51.579629	
1991	ID_1_1991	75.805676	98.482114	55.526158	
1992	ID_1_1992	76.658137	92.939991	65.707631	
1993	ID_1_1993	71.795337	85.849373	56.347229	
1994	ID_1_1994	79.985917	90.015506	56.662541	
1995	ID_1_1995	83.401894	92.124173	68.272294	
1996	ID_1_1996	86.112101	92.060120	69.271701	
1997	ID_1_1997	82.214188	91.829277	69.221382	
1998	ID_1_1998	78.728758	80.093022	67.911589	
1999	ID_1_1999	68.489556	76.699664	59.976201	
2000	ID_1_2000	67.059820	75.828585	68.491322	
2001	ID_1_2001	71.811902	80.660335	79.546993	
2002	ID_1_2002	70.225184	72.933199	60.940517	
2003	ID_1_2003	72.623180	73.320792	68.465501	
2004	ID_1_2004	77.136082	79.502874	82.190309	
2005	ID_1_2005	76.860779	81.871738	88.077745	
2006	ID_1_2006	80.731573	78.462067	81.276967	
2007	ID_1_2007	76.98.816021	80.587120	128.321380	
2008	ID_1_2008	114.328409	87.720772	128.752892	
2009	ID_1_2009	95.069817	84.243285	94.804720	
2010	ID_1_2010	106.796273	91.047427	111.953936	
2011	ID_1_2011	118.832658	94.922408	117.036249	
2012	ID_1_2012	111.497380	95.273085	101.346051	
2013	ID_1_2013	109.510425	96.779063	128.455810	
2014	ID_1_2014	106.317090	103.705965	100.345158	
2015	ID_1_2015	95.120644	98.887943	89.087348	
2016	ID_1_2016	97.806359	96.845293	87.936718	
2017	ID_1_2017	100.807798	100.519022	111.079992	
2018	ID_1_2018	94.213040	93.207948	105.366355	
2019	ID_1_2019	95.649945	100.567634	103.406045	
2020	ID_1_2020	99.180189	96.563352	102.922211	
2021	ID_1_2021	125.075926	107.161635	118.495544	
2022	ID_1_2022	140.553208	116.213194	139.290816	
2023	ID_1_2023	121.621711	111.282832	118.703383	

ID_YYYY	Cereals_Price_Index	Oils_Price_Index	Sugar_Price_Index
1990	70.287439	55.026992	94.016271
1991	70.676036	59.859303	67.765144
1992	73.088803	63.537907	67.195786
1993	68.402277	62.547913	71.893875
1994	74.008495	85.874993	89.550451
1995	77.112969	87.029628	89.469274
1996	92.822781	77.598555	82.097267
1997	77.446682	82.306452	81.958381
1998	71.719753	102.136657	67.237794
1999	65.946190	72.285125	48.166360
2000	64.657498	53.939579	63.647412
2001	67.726002	55.502920	69.860054
2002	73.468108	72.882232	56.358629
2003	74.621951	78.703021	55.108414
2004	75.309863	81.911139	52.181387
2005	69.319780	73.463156	69.809343
2006	79.246540	78.486366	101.681251
2007	105.792715	112.509529	65.388209
2008	133.914425	137.252551	77.050729
2009	100.804013	97.938711	116.398671
2010	107.552120	122.010090	131.788768

2011	128.088998	140.994074	144.977272
2012	124.710627	125.535018	121.007640
2013	117.702115	108.940382	99.781607
2014	107.046903	102.238845	97.228517
2015	98.010146	91.945606	85.035039
2016	93.958842	105.805648	118.765837
2017	93.569227	104.804837	101.928092
2018	99.031550	86.202287	76.027257
2019	97.158464	83.717151	79.066342
2020	104.220505	100.507993	80.380004
2021	130.473056	163.996004	108.767494
2022	151.327007	183.726982	111.985530
2023	130.615862	123.977620	133.819460