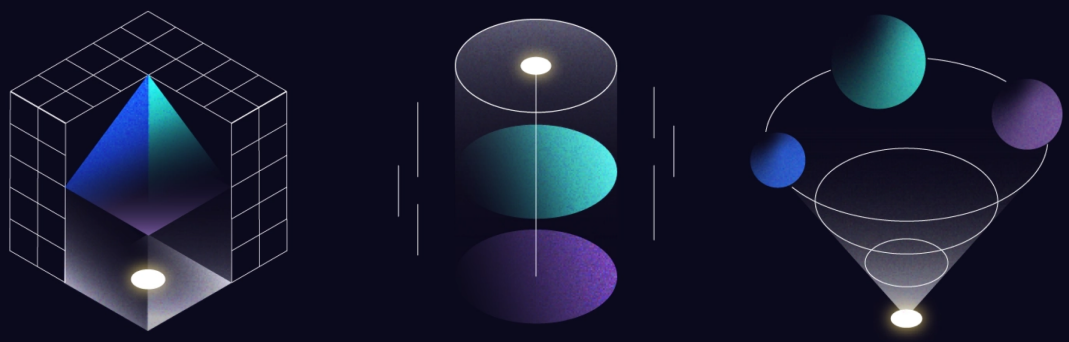
Técnicas de Filtrado de Datos con Pandas



Por Antonio Richaud

Técnicas de Filtrado de Datos con Pandas

Autor: *Antonio Richaud*

Introducción

En esta pequeña guía, vamos a checar y explicar algunas técnicas de filtrado de datos utilizando la librería Pandas en Python. Este tutorial es bastante útil para quienes buscan aprender a manejar y limpiar datasets de manera eficiente. A lo largo de la guía, trabajaremos con un dataset de ejemplo que contiene información sobre jugadores de la NBA, como sus nombres, equipos, posiciones, edades, alturas, pesos, universidades y salarios.

Nuestro objetivo será aprender a identificar y manejar valores nulos, filtrar filas y columnas, y aplicar técnicas avanzadas para limpiar y preparar nuestros datos para un análisis más profundo. :)

Dataset Propuesto

Para esta guía, utilizaremos un dataset que contiene información detallada sobre jugadores de la NBA. El dataset incluye columnas como `player_name`, `team_abbreviation`, `age`, `player_height`, `player_weight`, entre otras. Este tipo de datos es ideal para practicar técnicas de filtrado, ya que combina valores numéricos y categóricos, además de incluir algunos valores nulos que tendremos que gestionar.

Puedes descargar el dataset desde este enlace.

Cargando y Explorando el Dataset

Comencemos por cargar el dataset y visualizar las primeras filas para tener una idea general de su estructura.

In [3]:

```
# Cargamos Pandas :)
import pandas as pd
```

In [4]:

```
# Paso 1: Descargar el dataset
# Puedes descargar el dataset desde el siguiente enlace de Kaggle:
# https://www.kaggle.com/datasets/justinas/nba-players-data

# Paso 2: Cargar el dataset en un DataFrame
# Asumiendo que has descargado el archivo CSV y lo has guardado como "nba.csv"
df = pd.read_csv("nba.csv")

# Paso 3: Visualizar las primeras filas del DataFrame
# Esto nos permite echarle un ojito a los primeros registros para entender la estructura del dataset
df.head()
```

Out[4]:

	Unnamed: 0	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	...	
0	0	Randy Livingston	HOU	22.0	193.04	94.800728	Louisiana State	USA	1996	2	...	
1	1	Gaylon Nickerson	WAS	28.0	190.50	86.182480	Northwestern Oklahoma	USA	1994	2	...	
2	2	George Lynch	VAN	26.0	203.20	103.418976	North Carolina	USA	1993	1	...	
3	3	George McCloud	LAL	30.0	203.20	102.058200	Florida State	USA	1989	1	...	1
4	4	George Zidek	DEN	23.0	213.36	119.748288	UCLA	USA	1995	1	...	

5 rows × 22 columns

1. Filtrado de Datos con [ ]

La técnica de filtrado utilizando corchetes [ ] es la más básica y directa en Pandas. Con [ ], puedes seleccionar columnas específicas de un DataFrame o filtrar filas en base a una condición lógica. Esta técnica es ideal para acceder rápidamente a los datos que necesitas sin complicaciones.

Selección de Columnas

Puedes seleccionar una o más columnas del DataFrame utilizando el nombre de la columna entre corchetes. Esta es una manera sencilla de trabajar solo con las columnas que te interesan.

In [6]:

```
# Seleccionando una sola columna (la columna 'player_name')
df['player_name']

# Seleccionando múltiples columnas ('player_name', 'team_abbreviation', y 'age')
df[['player_name', 'team_abbreviation', 'age']]
```

Out[6]:

	player_name	team_abbreviation	age
0	Randy Livingston	HOU	22.0
1	Gaylon Nickerson	WAS	28.0
2	George Lynch	VAN	26.0
3	George McCloud	LAL	30.0
4	George Zidek	DEN	23.0
...	...	...	...
12839	Joel Embiid	PHI	29.0
12840	John Butler Jr.	POR	20.0
12841	John Collins	ATL	25.0
12842	Jericho Sims	NYK	24.0
12843	JaMychal Green	GSW	33.0

12844 rows × 3 columns

Filtrado de Filas por Condición

Además de seleccionar columnas, también puedes usar [ ] para filtrar filas en función de una condición lógica. Esto es útil para aislar subconjuntos de datos que cumplen con ciertos criterios.

In [7]:

```
# Filtrando filas donde la columna 'team_abbreviation' es igual a 'HOU'
df_hou = df[df['team_abbreviation'] == 'HOU']

# Visualizando las primeras filas del DataFrame filtrado
df_hou.head()
```

Out[7]:

	Unnamed: 0	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	...	pts
0	0	Randy Livingston	HOU	22.0	193.04	94.800728	Louisiana State	USA	1996	2	...	3.9
18	18	Hakeem Olajuwon	HOU	34.0	213.36	115.665960	Houston	Nigeria	1984	1	...	23.2
29	29	Emanual Davis	HOU	28.0	195.58	87.996848	Delaware State	USA	Undrafted	Undrafted	...	5.0
61	61	Joe Stephens	HOU	24.0	200.66	95.254320	Arkansas-Little Rock	USA	Undrafted	Undrafted	...	1.5
97	97	Eddie Johnson	HOU	38.0	200.66	97.522280	Illinois	USA	1981	2	...	8.2

5 rows × 22 columns

En este ejemplo, hemos filtrado el DataFrame para mostrar solo los jugadores que pertenecen al equipo con la abreviatura "HOU". Esta técnica te permite aplicar filtros sencillos pero potentes, basados en cualquier columna del DataFrame.

2. Filtrado de Datos con loc e iloc

Las funciones loc e iloc en Pandas son métodos avanzados para seleccionar datos en un DataFrame. Mientras que [ ] es útil para selecciones básicas, loc e iloc te permiten un control más preciso sobre las filas y columnas que seleccionas, ya sea por etiquetas o por posiciones.

Selección de Filas y Columnas con loc

El método loc se utiliza para seleccionar datos en función de las etiquetas de las filas y columnas. Es especialmente útil cuando deseas acceder a un subconjunto de datos basado en nombres de columnas o índices.

In [10]:

```
# Seleccionar una fila específica usando el índice y todas las columnas
fila_3 = df.loc[3]
print("Fila completa (índice 3):")
print(fila_3)

# Seleccionar una fila específica usando el índice y columnas específicas
fila_3_especifica = df.loc[3, ['player_name', 'team_abbreviation', 'age']]
print("\nFila específica (índice 3) con columnas 'player_name', 'team_abbreviation' y 'age':")
print(fila_3_especifica)

# Filtrar todas las filas donde el equipo es 'HOU' y seleccionar las columnas 'player_name' y 'age'
hou_players = df.loc[df['team_abbreviation'] == 'HOU', ['player_name', 'age']]
print("\nFilas donde el equipo es 'HOU' con columnas 'player_name' y 'age':")
print(hou_players)
```

```
Fila completa (índice 3):
Unnamed: 0                      3
player_name          George McCloud
team_abbreviation               LAL
age                            30.0
player_height                 203.2
player_weight              102.0582
college                Florida State
country                         USA
draft_year                     1989
draft_round                       1
draft_number                      7
gp                               64
pts                            10.2
reb                             2.8
ast                             1.7
net_rating                     -2.7
oreb_pct                      0.027
dreb_pct                      0.111
usg_pct                       0.206
ts_pct                        0.527
ast_pct                       0.125
season                      1996-97
Name: 3, dtype: object

Fila específica (índice 3) con columnas 'player_name', 'team_abbreviation' y 'age':
player_name          George McCloud
team_abbreviation               LAL
age                            30.0
Name: 3, dtype: object

Filas donde el equipo es 'HOU' con columnas 'player_name' y 'age':
            player_name   age
0      Randy Livingston  22.0
18      Hakeem Olajuwon  34.0
29        Emanual Davis  28.0
61         Joe Stephens  24.0
97        Eddie Johnson  38.0
...                 ...   ...
12667  Kevin Porter Jr.  23.0
12772       Jalen Green  21.0
12783   Jabari Smith Jr.  20.0
```

```
12765  Jabari Smith Jr.  20.0
12792     Jae'Sean Tate  27.0
12810  Josh Christopher  21.0

[433 rows x 2 columns]
```

En estos ejemplos:

• Seleccionamos una fila completa usando su índice (por ejemplo, la fila con índice 3).

• Seleccionamos una fila con columnas específicas usando tanto el índice como una lista de nombres de columnas.

• Filtramos el DataFrame para obtener solo las filas donde el equipo es 'HOU' y luego seleccionamos las columnas 'player_name' y 'age'.

Selección de Filas y Columnas con iloc

El método iloc es similar a loc, pero se basa en la posición numérica (índices) en lugar de etiquetas. Es útil cuando conoces las posiciones exactas de las filas y columnas que quieres seleccionar.

In [11]:

```
# Seleccionar la primera fila (índice 0) y todas las columnas
primera_fila = df.iloc[0]
print("Primera fila completa (índice 0):")
print(primera_fila)

# Seleccionar la primera fila (índice 0) y las primeras tres columnas
primera_fila_columnas = df.iloc[0, :3]
print("\nPrimera fila (índice 0) con las primeras tres columnas:")
print(primera_fila_columnas)

# Seleccionar las primeras cinco filas y las columnas de la 1 a la 3
primeras_filas_columnas = df.iloc[:5, 1:4]
print("\nPrimeras cinco filas y columnas 1 a la 3:")
print(primeras_filas_columnas)
```

```
Primera fila completa (índice 0):
Unnamed: 0                        0
player_name          Randy Livingston
team_abbreviation                HOU
age                             22.0
player_height                 193.04
player_weight              94.800728
college              Louisiana State
country                          USA
draft_year                      1996
draft_round                        2
draft_number                      42
gp                                64
pts                              3.9
reb                              1.5
ast                              2.4
net_rating                       0.3
oreb_pct                       0.042
dreb_pct                       0.071
usg_pct                        0.169
ts_pct                         0.487
ast_pct                        0.248
season                       1996-97
Name: 0, dtype: object

Primera fila (índice 0) con las primeras tres columnas:
Unnamed: 0                        0
player_name          Randy Livingston
team_abbreviation                HOU
Name: 0, dtype: object

Primeras cinco filas y columnas 1 a la 3:
        player_name team_abbreviation   age
0  Randy Livingston               HOU  22.0
1  Gaylon Nickerson               WAS  28.0
2      George Lynch               VAN  26.0
3    George McCloud               LAL  30.0
4      George Zidek               DEN  23.0
```

En estos ejemplos:

• Seleccionamos la primera fila del DataFrame (index 0).

• Seleccionamos la primera fila y las tres primeras columnas.

• Seleccionamos las primeras cinco filas y las columnas en las posiciones 1 a 3.

Diferencias Clave Entre loc e iloc

• loc: Utiliza las etiquetas de filas y columnas para la selección. Es útil cuando trabajas con nombres específicos de columnas o índices.

• iloc: Utiliza las posiciones numéricas de filas y columnas. Es útil cuando deseas seleccionar datos basándote en sus posiciones dentro del DataFrame.

Con esto, hemos visto cómo usar loc e iloc para seleccionar y filtrar datos en un DataFrame. Estas herramientas son súper importantes cuando necesitas un control preciso sobre la selección de tus datos.

3. Filtrado de Datos con isin

El método isin en Pandas es una herramienta poderosa para filtrar datos en un DataFrame basado en si los valores de una columna pertenecen a una lista de valores. Esto es particularmente útil cuando deseas seleccionar filas que coinciden con múltiples valores en una columna.

Uso Básico de isin

Puedes usar isin para filtrar filas donde los valores de una columna coinciden con alguno de los valores en una lista específica.

In [12]:

```
# Filtrar filas donde la columna 'team_abbreviation' es igual a 'HOU', 'LAL' o 'BOS'
teams = ['HOU', 'LAL', 'BOS']
filtered_df = df[df['team_abbreviation'].isin(teams)]
print("Filas donde 'team_abbreviation' es 'HOU', 'LAL' o 'BOS':")
print(filtered_df)
```

```
Filas donde 'team_abbreviation' es 'HOU', 'LAL' o 'BOS':
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0  Randy Livingston               HOU  22.0         193.04
3               3    George McCloud               LAL  30.0         203.20
16             16        Greg Minor               BOS  25.0         198.12
18             18   Hakeem Olajuwon               HOU  34.0         213.36
29             29     Emanual Davis               HOU  28.0         195.58
...           ...               ...               ...   ...            ...
12783       12783  Jabari Smith Jr.               HOU  20.0         210.82
12792       12792     Jae'Sean Tate               HOU  27.0         195.58
12810       12810  Josh Christopher               HOU  21.0         193.04
12822       12822      Jaylen Brown               BOS  26.0         198.12
12825       12825      Jayson Tatum               BOS  25.0         203.20

       player_weight          college  country draft_year draft_round  ...  \
0          94.800728  Louisiana State      USA       1996           2  ...
3         102.058200    Florida State      USA       1989           1  ...
16         95.254320       Louisville      USA       1994           1  ...
18        115.665960          Houston  Nigeria       1984           1  ...
29         87.996848   Delaware State      USA  Undrafted   Undrafted  ...
...              ...              ...      ...        ...         ...  ...
12783      99.790240           Auburn      USA       2022           1  ...
12792     104.326160       Ohio State      USA  Undrafted   Undrafted  ...
12810      97.522280    Arizona State      USA       2021           1  ...
12822     101.151016       California      USA       2016           1  ...
12825      95.254320             Duke      USA       2017           1  ...

        pts  reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0       3.9  1.5  2.4         0.3     0.042     0.071    0.169   0.487
3      10.2  2.8  1.7        -2.7     0.027     0.111    0.206   0.527
16      9.6  3.5  1.5        -2.1     0.058     0.110    0.189   0.519
18     23.2  9.2  3.0         6.5     0.075     0.206    0.308   0.558
29      5.0  1.7  2.0         6.6     0.011     0.098    0.144   0.565
...     ...  ...  ...         ...       ...       ...      ...     ...
```

```
12783  12.8  7.2  1.3       -11.3     0.047     0.180    0.183   0.514
12792   9.1  3.8  2.7        -7.3     0.054     0.114    0.178   0.546
12810   5.8  1.1  1.1       -13.9     0.024     0.068    0.215   0.523
12822  26.6  6.9  3.5         5.4     0.032     0.152    0.307   0.581
12825  30.1  8.8  4.6         8.5     0.029     0.200    0.319   0.607

       ast_pct   season
0        0.248  1996-97
3        0.125  1996-97
16       0.103  1996-97
18       0.158  1996-97
29       0.191  1996-97
...        ...      ...
12783    0.059  2022-23
12792    0.184  2022-23
12810    0.149  2022-23
12822    0.163  2022-23
12825    0.205  2022-23

[1287 rows x 22 columns]
```

En este ejemplo:

• Hemos definido una lista de equipos (teams = ['HOU', 'LAL', 'BOS']).

• Luego, utilizamos isin para filtrar el DataFrame, seleccionando solo las filas donde la columna team_abbreviation coincide con alguno de los valores en la lista.

• Finalmente, se imprimen las filas que cumplen con el criterio.

Aplicaciones Comunes de isin

• Filtrado de múltiples categorías: isin es ideal cuando necesitas filtrar datos basados en múltiples categorías. Por ejemplo, si quieres analizar solo los jugadores de ciertos equipos, isin te permitirá hacer esto de manera eficiente.

• Combinación con otras condiciones: Puedes combinar isin con otras condiciones lógicas para crear filtros más complejos, utilizando operadores como & (and) y | (or).

Este método es muy útil cuando tienes listas de valores específicas y quieres encontrar todas las filas que coinciden con esos valores. Es una técnica flexible y potente para realizar filtros múltiples de manera simple y rápida.

4. Filtrado de Datos con str.contains

El método str.contains en Pandas se utiliza para filtrar filas en un DataFrame basándose en si una columna de texto contiene una subcadena específica. Esto es especialmente útil cuando necesitas buscar patrones o palabras clave dentro de cadenas de texto en tus datos.

Uso Básico de str.contains

Puedes usar str.contains para filtrar filas en las que los valores de una columna de texto contengan una subcadena específica.

In [13]:

```
# Filtrar filas donde la columna 'player_name' contiene la subcadena 'George'
filtered_df = df[df['player_name'].str.contains('George')]
print("Filas donde 'player_name' contiene 'George':")
print(filtered_df)
```

```
Filas donde 'player_name' contiene 'George':
       Unnamed: 0     player_name team_abbreviation   age  player_height  \
2               2    George Lynch              VAN  26.0         203.20
3               3  George McCloud              LAL  30.0         203.20
4               4    George Zidek              DEN  23.0         213.36
859           859  George McCloud              PHX  31.0         203.20
860           860    George Lynch              VAN  27.0         203.20
...           ...             ...              ...   ...            ...
11769       11769   Georges Niang              PHI  29.0         200.66
12080       12080     Paul George              LAC  32.0         203.20
12315       12315     Paul George              LAC  33.0         203.20
12518       12518     George Hill              IND  37.0         193.04
12519       12519   Georges Niang              PHI  30.0         200.66
```

```
12519      12519   Georges Niang              PHI  30.0         200.66

       player_weight                     college country draft_year  \
2         103.418976              North Carolina     USA       1993
3         102.058200               Florida State     USA       1989
4         119.748288                        UCLA     USA       1995
859       102.058200               Florida State     USA       1989
860       103.418976              North Carolina     USA       1993
...              ...                         ...     ...        ...
11769     104.326160                  Iowa State     USA       2016
12080      99.790240                Fresno State     USA       2010
12315      99.790240                Fresno State     USA       2010
12518      85.275296  Indiana-Purdue Indianapolis     USA       2008
12519     104.326160                  Iowa State     USA       2016

      draft_round  ...   pts  reb  ast  net_rating  oreb_pct  dreb_pct  \
2               1  ...   8.3  6.4  1.9        -8.2     0.106     0.185
3               1  ...  10.2  2.8  1.7        -2.7     0.027     0.111
4               1  ...   2.8  1.7  0.3       -14.1     0.102     0.169
859             1  ...   7.2  3.5  1.3         3.5     0.042     0.167
860             1  ...   7.5  4.4  1.5        -3.3     0.115     0.170
...           ...  ...   ...  ...  ...         ...       ...       ...
11769           2  ...   9.2  2.7  1.3         1.9     0.014     0.097
12080           1  ...  24.3  6.9  5.7         4.2     0.011     0.171
12315           1  ...  23.8  6.1  5.1         4.0     0.025     0.153
12518           1  ...   5.0  1.8  2.4         1.9     0.017     0.077
12519           2  ...   8.2  2.4  1.0         1.3     0.013     0.109

       usg_pct  ts_pct  ast_pct   season
2        0.175   0.512    0.125  1996-97
3        0.206   0.527    0.125  1996-97
4        0.195   0.500    0.064  1996-97
859      0.189   0.507    0.105  1997-98
860      0.207   0.526    0.146  1997-98
...        ...     ...      ...      ...
11769    0.160   0.594    0.090  2021-22
12080    0.329   0.538    0.286  2021-22
12315    0.294   0.588    0.237  2022-23
12518    0.110   0.610    0.178  2022-23
12519    0.162   0.610    0.074  2022-23

[67 rows x 22 columns]
```

En este ejemplo:

• Utilizamos str.contains('George') para encontrar todas las filas donde la columna player_name contiene la subcadena "George".

• str.contains es muy útil para buscar patrones de texto dentro de columnas de tipo string.

Opciones Avanzadas de str.contains

• Distinción entre mayúsculas y minúsculas: Por defecto, str.contains distingue entre mayúsculas y minúsculas. Si quieres que sea insensible a las mayúsculas, puedes añadir el parámetro case=False.

In [14]:

```
# Filtrar filas donde 'player_name' contiene 'george' sin importar mayúsculas o minúsculas
filtered_df = df[df['player_name'].str.contains('george', case=False)]
print("Filas donde 'player_name' contiene 'george' (sin importar mayúsculas o minúsculas):")
print(filtered_df)
```

```
Filas donde 'player_name' contiene 'george' (sin importar mayúsculas o minúsculas):
       Unnamed: 0     player_name team_abbreviation   age  player_height  \
2               2    George Lynch              VAN  26.0         203.20
3               3  George McCloud              LAL  30.0         203.20
4               4    George Zidek              DEN  23.0         213.36
859           859  George McCloud              PHX  31.0         203.20
860           860    George Lynch              VAN  27.0         203.20
...           ...             ...              ...   ...            ...
11769       11769   Georges Niang              PHI  29.0         200.66
12080       12080     Paul George              LAC  32.0         203.20
12315       12315     Paul George              LAC  33.0         203.20
12518       12518     George Hill              IND  37.0         193.04
12519       12519   Georges Niang              PHI  30.0         200.66

       player_weight                     college country draft_year  \
```

```
2         103.418976               North Carolina     USA       1993
3         102.058200                Florida State     USA       1989
4         119.748288                         UCLA     USA       1995
859       102.058200                Florida State     USA       1989
860       103.418976               North Carolina     USA       1993
...              ...                          ...     ...        ...
11769     104.326160                   Iowa State     USA       2016
12080      99.790240                 Fresno State     USA       2010
12315      99.790240                 Fresno State     USA       2010
12518      85.275296  Indiana-Purdue Indianapolis     USA       2008
12519     104.326160                   Iowa State     USA       2016

      draft_round  ...   pts  reb  ast  net_rating  oreb_pct  dreb_pct  \
2               1  ...   8.3  6.4  1.9        -8.2     0.106     0.185
3               1  ...  10.2  2.8  1.7        -2.7     0.027     0.111
4               1  ...   2.8  1.7  0.3       -14.1     0.102     0.169
859             1  ...   7.2  3.5  1.3         3.5     0.042     0.167
860             1  ...   7.5  4.4  1.5        -3.3     0.115     0.170
...           ...  ...   ...  ...  ...         ...       ...       ...
11769           2  ...   9.2  2.7  1.3         1.9     0.014     0.097
12080           1  ...  24.3  6.9  5.7         4.2     0.011     0.171
12315           1  ...  23.8  6.1  5.1         4.0     0.025     0.153
12518           1  ...   5.0  1.8  2.4         1.9     0.017     0.077
12519           2  ...   8.2  2.4  1.0         1.3     0.013     0.109

       usg_pct  ts_pct  ast_pct   season
2        0.175   0.512    0.125  1996-97
3        0.206   0.527    0.125  1996-97
4        0.195   0.500    0.064  1996-97
859      0.189   0.507    0.105  1997-98
860      0.207   0.526    0.146  1997-98
...        ...     ...      ...      ...
11769    0.160   0.594    0.090  2021-22
12080    0.329   0.538    0.286  2021-22
12315    0.294   0.588    0.237  2022-23
12518    0.110   0.610    0.178  2022-23
12519    0.162   0.610    0.074  2022-23

[67 rows x 22 columns]
```

• Expresiones Regulares: str.contains también soporta expresiones regulares, permitiéndote hacer búsquedas más complejas. Puedes activar el uso de expresiones regulares con el parámetro regex=True, que está activado por defecto.

In [15]:

```
# Filtrar filas donde 'player_name' contiene una cadena que empieza con 'Geo' o termina con 'son'
filtered_df = df[df['player_name'].str.contains(r'^Geo|son$', regex=True)]
print("Filas donde 'player_name' empieza con 'Geo' o termina con 'son':")
print(filtered_df)
```

```
Filas donde 'player_name' empieza con 'Geo' o termina con 'son':
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
4               4      George Zidek               DEN  23.0         213.36
8               8    Glenn Robinson               MIL  24.0         200.66
...           ...               ...               ...   ...           ...
12771       12771     Jalen Johnson               ATL  21.0         203.20
12774       12774     Jalen Brunson               NYK  26.0         187.96
12779       12779        JD Davison               BOS  20.0         185.42
12802       12802   Jordan Clarkson               UTA  31.0         195.58
12816       12816   Josh Richardson               NOP  29.0         198.12

       player_weight                college country draft_year draft_round  \
1          86.182480  Northwestern Oklahoma     USA       1994           2
2         103.418976         North Carolina     USA       1993           1
3         102.058200          Florida State     USA       1989           1
4         119.748288                   UCLA     USA       1995           1
8         106.594120                 Purdue     USA       1994           1
...              ...                    ...     ...        ...         ...
12771      99.336648                   Duke     USA       2021           1
12774      86.182480              Villanova     USA       2018           2
12779      88.450440                Alabama     USA       2022           2
12802      87.996848               Missouri     USA       2014           2
12816      90.718400              Tennessee     USA       2015           2

       ...   pts  reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
1      ...   3.8  1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3  6.4  1.9        -8.2     0.106     0.185    0.175   0.512
```

```
3       ...  10.2  2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4       ...   2.8  1.7  0.3       -14.1     0.102     0.169    0.195   0.500
8       ...  21.1  6.3  3.1        -2.9     0.051     0.144    0.278   0.528
...     ...   ...  ...  ...         ...       ...       ...      ...     ...
12771   ...   5.6  4.0  1.2         0.3     0.047     0.214    0.156   0.554
12774   ...  24.0  3.5  6.2         2.1     0.016     0.085    0.266   0.597
12779   ...   1.6  0.8  0.9        -5.8     0.033     0.093    0.157   0.478
12802   ...  20.8  4.0  4.4         0.8     0.036     0.083    0.273   0.558
12816   ...  10.1  2.7  2.7        -5.2     0.026     0.087    0.180   0.555

       ast_pct   season
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.064  1996-97
8        0.146  1996-97
...        ...      ...
12771    0.106  2022-23
12774    0.280  2022-23
12779    0.234  2022-23
12802    0.209  2022-23
12816    0.160  2022-23

[1090 rows x 22 columns]
```

Este método es una herramienta barbara para trabajar con texto dentro de un DataFrame. Te permite realizar búsquedas específicas y patrones complejos, haciendo que sea muy útil para analizar y filtrar datos textuales.

5. Filtrado de Datos con where

La función where en Pandas se utiliza para aplicar condiciones lógicas a un DataFrame y devolver un nuevo DataFrame donde se mantienen los valores que cumplen la condición, mientras que los valores que no la cumplen se reemplazan por NaN. Esta función es muy útil cuando quieres filtrar datos manteniendo la estructura del DataFrame original.

Uso Básico de where

Puedes usar where para aplicar una condición a todo un DataFrame o a una serie y obtener un nuevo DataFrame donde solo los valores que cumplen la condición son retenidos.

In [16]:

```python
# Mantener filas donde la columna 'age' es mayor que 25, el resto se convierte en NaN
filtered_df = df.where(df['age'] > 25)
print("DataFrame donde 'age' es mayor que 25:")
print(filtered_df)
```

```
DataFrame donde 'age' es mayor que 25:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0             NaN               NaN               NaN   NaN            NaN
1             1.0  Gaylon Nickerson               WAS  28.0         190.50
2             2.0      George Lynch               VAN  26.0         203.20
3             3.0    George McCloud               LAL  30.0         203.20
4             NaN               NaN               NaN   NaN            NaN
...           ...               ...               ...   ...            ...
12839     12839.0       Joel Embiid               PHI  29.0         213.36
12840         NaN               NaN               NaN   NaN            NaN
12841         NaN               NaN               NaN   NaN            NaN
12842         NaN               NaN               NaN   NaN            NaN
12843     12843.0    JaMychal Green               GSW  33.0         205.74

       player_weight                college   country draft_year draft_round  \
0                NaN                    NaN       NaN        NaN         NaN
1          86.182480  Northwestern Oklahoma       USA       1994           2
2         103.418976         North Carolina       USA       1993           1
3         102.058200          Florida State       USA       1989           1
4                NaN                    NaN       NaN        NaN         NaN
...              ...                    ...       ...        ...         ...
12839     127.005760                 Kansas  Cameroon       2014           1
12840            NaN                    NaN       NaN        NaN         NaN
12841            NaN                    NaN       NaN        NaN         NaN
12842            NaN                    NaN       NaN        NaN         NaN
12843     102.965384                Alabama       USA  Undrafted   Undrafted

       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0      ...   NaN   NaN  NaN         NaN       NaN       NaN      NaN     NaN
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
```

```
3       ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4       ...   NaN   NaN  NaN         NaN       NaN       NaN      NaN     NaN
...     ...   ...   ...  ...         ...       ...       ...      ...     ...
12839   ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
12840   ...   NaN   NaN  NaN         NaN       NaN       NaN      NaN     NaN
12841   ...   NaN   NaN  NaN         NaN       NaN       NaN      NaN     NaN
12842   ...   NaN   NaN  NaN         NaN       NaN       NaN      NaN     NaN
12843   ...   6.4   3.6  0.9        -8.2     0.087     0.164    0.169   0.650

       ast_pct   season
0          NaN      NaN
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4          NaN      NaN
...        ...      ...
12839    0.233  2022-23
12840      NaN      NaN
12841      NaN      NaN
12842      NaN      NaN
12843    0.094  2022-23

[12844 rows x 22 columns]
```

En este ejemplo:

• La condición df ['age'] > 25 se aplica a todo el DataFrame.

• Las filas donde age es mayor que 25 se mantienen intactas, mientras que en las filas donde age no cumple la condición, los valores se convierten en NaN.

Uso Avanzado de where

• Reemplazo de valores: Puedes reemplazar los valores que no cumplen la condición con un valor específico en lugar de NaN.

In [17]:

```
# Reemplazar los valores donde 'age' no es mayor que 25 por 0
filtered_df = df.where(df['age'] > 25, 0)
print("DataFrame donde 'age' es mayor que 25, el resto se convierte en 0:")
print(filtered_df)
```

```
DataFrame donde 'age' es mayor que 25, el resto se convierte en 0:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0                 0                 0   0.0           0.00
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
4               0                 0                 0   0.0           0.00
...           ...               ...               ...   ...            ...
12839       12839       Joel Embiid               PHI  29.0         213.36
12840           0                 0                 0   0.0           0.00
12841           0                 0                 0   0.0           0.00
12842           0                 0                 0   0.0           0.00
12843       12843    JaMychal Green               GSW  33.0         205.74

       player_weight                college   country draft_year draft_round  \
0           0.000000                      0         0          0           0
1          86.182480  Northwestern Oklahoma       USA       1994           2
2         103.418976         North Carolina       USA       1993           1
3         102.058200          Florida State       USA       1989           1
4           0.000000                      0         0          0           0
...              ...                    ...       ...        ...         ...
12839     127.005760                Kansas  Cameroon       2014           1
12840       0.000000                      0         0          0           0
12841       0.000000                      0         0          0           0
12842       0.000000                      0         0          0           0
12843     102.965384                Alabama       USA  Undrafted   Undrafted

       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0      ...   0.0   0.0  0.0         0.0     0.000     0.000    0.000   0.000
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4      ...   0.0   0.0  0.0         0.0     0.000     0.000    0.000   0.000
...    ...   ...   ...  ...         ...       ...       ...      ...     ...
12839  ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
```

```
12840  ...    0.0   0.0  0.0          0.0      0.000      0.000     0.000    0.000
12841  ...    0.0   0.0  0.0          0.0      0.000      0.000     0.000    0.000
12842  ...    0.0   0.0  0.0          0.0      0.000      0.000     0.000    0.000
12843  ...    6.4   3.6  0.9         -8.2      0.087      0.164     0.169    0.650

       ast_pct   season
0        0.000        0
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.000        0
...        ...      ...
12839    0.233  2022-23
12840    0.000        0
12841    0.000        0
12842    0.000        0
12843    0.094  2022-23

[12844 rows x 22 columns]
```

• Combinación de condiciones: Puedes combinar múltiples condiciones usando operadores lógicos como & (and) o | (or).

In [18]:

```
# Mantener filas donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU'
filtered_df = df.where((df['age'] > 25) & (df['team_abbreviation'] == 'HOU'))
print("DataFrame donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU':")
print(filtered_df)
```

```
DataFrame donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU':
       Unnamed: 0 player_name team_abbreviation  age  player_height  \
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
...           ...         ...               ...  ...            ...
12839         NaN         NaN               NaN  NaN            NaN
12840         NaN         NaN               NaN  NaN            NaN
12841         NaN         NaN               NaN  NaN            NaN
12842         NaN         NaN               NaN  NaN            NaN
12843         NaN         NaN               NaN  NaN            NaN

       player_weight college country draft_year draft_round  ... pts  reb  \
0                NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
1                NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
2                NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
3                NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
4                NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
...              ...     ...     ...        ...         ...  ...  ..  ...
12839            NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
12840            NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
12841            NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
12842            NaN     NaN     NaN        NaN         NaN  ... NaN  NaN
12843            NaN     NaN     NaN        NaN         NaN  ... NaN  NaN

       ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  ast_pct  season
0      NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
1      NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
2      NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
3      NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
4      NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
...    ...         ...       ...       ...      ...     ...      ...     ...
12839  NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
12840  NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
12841  NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
12842  NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN
12843  NaN         NaN       NaN       NaN      NaN     NaN      NaN     NaN

[12844 rows x 22 columns]
```

En este último ejemplo, solo se mantendrán las filas donde ambas condiciones (age > 25 y team_abbreviation == 'HOU') son verdaderas; el resto de los valores se convertirá en NaN.

La función where es muy versátil y útil cuando necesitas aplicar filtros sin alterar la estructura original de tu DataFrame. Es ideal para situaciones donde quieres mantener la forma del DataFrame pero solo mostrar o trabajar con los datos que cumplen ciertas condiciones.

6. Filtrado de Datos con query

La función query en Pandas permite filtrar filas de un DataFrame utilizando una expresión de consulta similar a SQL. Es una manera muy conveniente y legible de aplicar condiciones complejas para seleccionar datos, especialmente cuando se combinan múltiples criterios.

Uso Básico de query

Puedes usar query para filtrar filas que cumplen con una condición específica. La expresión de consulta se escribe como una cadena de texto y se pasa como argumento a la función query.

In [19]:

```
# Filtrar filas donde 'age' es mayor que 25
filtered_df = df.query('age > 25')
print("Filas donde 'age' es mayor que 25:")
print(filtered_df)
```

```
Filas donde 'age' es mayor que 25:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
5               5    Gerald Wilkins               ORL  33.0         198.12
6               6  Gheorghe Muresan               WAS  26.0         231.14
...           ...               ...               ...   ...            ...
12835       12835      Jock Landale               PHX  27.0         210.82
12836       12836        Joe Harris               BKN  31.0         198.12
12837       12837        Joe Ingles               MIL  35.0         205.74
12839       12839       Joel Embiid               PHI  29.0         213.36
12843       12843    JaMychal Green               GSW  33.0         205.74

       player_weight                college    country draft_year draft_round  \
1          86.182480  Northwestern Oklahoma        USA       1994           2
2         103.418976         North Carolina        USA       1993           1
3         102.058200          Florida State        USA       1989           1
5         102.058200  Tennessee-Chattanooga        USA       1985           2
6         137.438376                    NaN        USA       1993           2
...              ...                    ...        ...        ...         ...
12835     115.665960            St. Mary's  Australia  Undrafted   Undrafted
12836      99.790240               Virginia        USA       2014           2
12837      99.790240                    NaN  Australia  Undrafted   Undrafted
12839     127.005760                 Kansas   Cameroon       2014           1
12843     102.965384                Alabama        USA  Undrafted   Undrafted

       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
5      ...  10.6   2.2  2.2        -5.8     0.031     0.064    0.203   0.503
6      ...  10.6   6.6  0.4         6.9     0.098     0.217    0.185   0.618
...    ...   ...   ...  ...         ...       ...       ...      ...     ...
12835  ...   6.6   4.1  1.0         7.9     0.114     0.164    0.182   0.595
12836  ...   7.6   2.2  1.4        -1.0     0.016     0.089    0.141   0.621
12837  ...   6.9   2.8  3.3         2.5     0.012     0.102    0.122   0.616
12839  ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
12843  ...   6.4   3.6  0.9        -8.2     0.087     0.164    0.169   0.650

       ast_pct   season
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
5        0.143  1996-97
6        0.024  1996-97
...        ...      ...
12835    0.101  2022-23
12836    0.091  2022-23
12837    0.181  2022-23
12839    0.233  2022-23
12843    0.094  2022-23

[7350 rows x 22 columns]
```

En este ejemplo:

• La consulta age > 25 selecciona todas las filas donde la columna age tiene un valor mayor que 25.

- **El resultado es un DataFrame filtrado que contiene solo las filas que cumplen con esta condición.**

Uso Avanzado de query

- **Combinar múltiples condiciones:** **Puedes combinar varias condiciones usando & (and) y | (or) en la expresión de consulta.**

In [20]:

```
# Filtrar filas donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU'
filtered_df = df.query('age > 25 & team_abbreviation == "HOU"')
print("Filas donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU':")
print(filtered_df)
```

```
Filas donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU':
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
18             18   Hakeem Olajuwon               HOU  34.0         213.36
29             29     Emanual Davis               HOU  28.0         195.58
97             97     Eddie Johnson               HOU  38.0         200.66
111           111     Clyde Drexler               HOU  35.0         200.66
138           138   Charles Barkley               HOU  34.0         198.12
...           ...               ...               ...   ...            ...
11937       11937    Christian Wood               HOU  26.0         205.74
11951       11951       David Nwaba               HOU  29.0         195.58
12553       12553     Frank Kaminsky               HOU  30.0         213.36
12576       12576  Boban Marjanovic               HOU  34.0         223.52
12792       12792     Jae'Sean Tate               HOU  27.0         195.58

       player_weight         college  country draft_year draft_round  ...  \
18        115.665960         Houston  Nigeria       1984           1  ...
29         87.996848  Delaware State      USA  Undrafted   Undrafted  ...
97         97.522280        Illinois      USA       1981           2  ...
111       100.697424         Houston      USA       1983           1  ...
138       114.305184          Auburn      USA       1984           1  ...
...              ...             ...      ...        ...         ...  ...
11937      97.068688            UNLV      USA  Undrafted   Undrafted  ...
11951      99.336648        Cal Poly      USA  Undrafted   Undrafted  ...
12553     108.862080       Wisconsin      USA       2015           1  ...
12576     131.541680             NaN   Serbia  Undrafted   Undrafted  ...
12792     104.326160      Ohio State      USA  Undrafted   Undrafted  ...

        pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
18     23.2   9.2  3.0         6.5     0.075     0.206    0.308   0.558
29      5.0   1.7  2.0         6.6     0.011     0.098    0.144   0.565
97      8.2   2.7  1.0         4.1     0.034     0.126    0.220   0.541
111    18.0   6.0  5.7         7.5     0.062     0.122    0.231   0.548
138    19.2  13.5  4.7         7.5     0.127     0.280    0.230   0.581
...     ...   ...  ...         ...       ...       ...      ...     ...
11937  17.9  10.1  2.3        -9.1     0.051     0.265    0.233   0.595
11951   5.1   3.3  0.8        -9.0     0.061     0.179    0.150   0.567
12553   2.5   1.4  0.9        -1.9     0.029     0.207    0.146   0.635
12576   3.3   1.9  0.3        18.5     0.121     0.226    0.206   0.710
12792   9.1   3.8  2.7        -7.3     0.054     0.114    0.178   0.546

       ast_pct   season
18       0.158  1996-97
29       0.191  1996-97
97       0.102  1996-97
111      0.261  1996-97
138      0.208  1996-97
...        ...      ...
11937    0.124  2021-22
11951    0.085  2021-22
12553    0.194  2022-23
12576    0.079  2022-23
12792    0.184  2022-23

[251 rows x 22 columns]
```

En este caso, el DataFrame resultante contendrá solo las filas donde age es mayor que 25 y team_abbreviation es 'HOU'.

- **Uso de variables en query:** **Puedes usar variables externas dentro de una expresión de query utilizando el símbolo @ antes del nombre de la variable.**

In [21]:

```
age_threshold = 25
team = 'LAL'
```

```
filtered_df = df.query('age > @age_threshold & team_abbreviation == @team')
print(f'Filas donde "age" es mayor que {age_threshold} y "team_abbreviation" es "{team}":')
print(filtered_df)
```

```
Filas donde "age" es mayor que 25 y "team_abbreviation" es "LAL":
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
3               3    George McCloud               LAL  30.0         203.20
47             47    Elden Campbell               LAL  28.0         213.36
59             59     Jerome Kersey               LAL  35.0         200.66
112           112      Corie Blount               LAL  28.0         208.28
132           132       Byron Scott               LAL  36.0         193.04
...           ...               ...               ...   ...            ...
12505       12505        Davon Reed               LAL  28.0         195.58
12585       12585     Anthony Davis               LAL  30.0         208.28
12626       12626  D'Angelo Russell               LAL  27.0         193.04
12725       12725     Malik Beasley               LAL  26.0         193.04
12733       12733      LeBron James               LAL  38.0         205.74

       player_weight        college country draft_year draft_round  ...   pts  \
3         102.058200  Florida State     USA       1989           1  ...  10.2
47        113.398000        Clemson     USA       1990           1  ...  14.9
59        108.862080       Longwood     USA       1984           2  ...   6.8
112       109.769264     Cincinnati     USA       1993           1  ...   4.2
132        92.986360  Arizona State     USA       1983           1  ...   6.7
...              ...            ...     ...        ...         ...  ...   ...
12505      93.439952          Miami     USA       2017           2  ...   2.1
12585     114.758776       Kentucky     USA       2012           1  ...  25.9
12626      87.543256     Ohio State     USA       2015           1  ...  17.8
12725      84.821704  Florida State     USA       2016           1  ...  12.7
12733     113.398000            NaN     USA       2003           1  ...  28.9

        reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  ast_pct  \
3       2.8  1.7        -2.7     0.027     0.111    0.206   0.527    0.125
47      8.0  1.6         3.3     0.095     0.183    0.222   0.520    0.087
59      5.2  1.3         4.1     0.072     0.164    0.145   0.475    0.077
112     4.8  0.6         2.0     0.130     0.182    0.119   0.559    0.056
132     1.5  1.3         6.7     0.017     0.075    0.157   0.590    0.110
...     ...  ...         ...       ...       ...      ...     ...      ...
12505   1.4  0.5       -14.1     0.031     0.146    0.142   0.471    0.099
12585  12.5  2.6         4.8     0.102     0.248    0.277   0.627    0.128
12626   3.0  6.2         1.5     0.016     0.075    0.225   0.605    0.274
12725   3.5  1.5        -2.2     0.016     0.116    0.207   0.531    0.086
12733   8.3  6.8         4.6     0.033     0.191    0.322   0.583    0.308

        season
3      1996-97
47     1996-97
59     1996-97
112    1996-97
132    1996-97
...        ...
12505  2022-23
12585  2022-23
12626  2022-23
12725  2022-23
12733  2022-23

[253 rows x 22 columns]
```

Aquí, age_threshold y team son variables definidas en el código, y se utilizan dentro de la consulta query.

• Filtrar por valores de texto: También puedes filtrar utilizando condiciones basadas en texto.

In [22]:

```
# Filtrar filas donde 'player_name' contiene la palabra 'George'
filtered_df = df.query('player_name.str.contains("George")', engine='python')
print('Filas donde "player_name" contiene "George":')
print(filtered_df)
```

```
Filas donde "player_name" contiene "George":
       Unnamed: 0     player_name team_abbreviation   age  player_height  \
2               2    George Lynch               VAN  26.0         203.20
3               3   George McCloud               LAL  30.0         203.20
4               4     George Zidek               DEN  23.0         213.36
859           859   George McCloud               PHX  31.0         203.20
860           860     George Lynch               VAN  27.0         203.20
...           ...               ...               ...   ...            ...
11769       11769    Georges Niang               PHI  29.0         200.66
12080       12080     Paul George               LAC  32.0         203.20
```

```
12060       12060     Paul George               LAC  32.0         203.20
12315       12315     Paul George               LAC  33.0         203.20
12518       12518     George Hill               IND  37.0         193.04
12519       12519   Georges Niang               PHI  30.0         200.66

       player_weight                     college country draft_year  \
2         103.418976               North Carolina     USA       1993
3         102.058200                Florida State     USA       1989
4         119.748288                         UCLA     USA       1995
859       102.058200                Florida State     USA       1989
860       103.418976               North Carolina     USA       1993
...              ...                          ...     ...        ...
11769     104.326160                   Iowa State     USA       2016
12080      99.790240                 Fresno State     USA       2010
12315      99.790240                 Fresno State     USA       2010
12518      85.275296  Indiana-Purdue Indianapolis     USA       2008
12519     104.326160                   Iowa State     USA       2016

      draft_round  ...   pts  reb  ast  net_rating  oreb_pct  dreb_pct  \
2               1  ...   8.3  6.4  1.9        -8.2     0.106     0.185
3               1  ...  10.2  2.8  1.7        -2.7     0.027     0.111
4               1  ...   2.8  1.7  0.3       -14.1     0.102     0.169
859             1  ...   7.2  3.5  1.3         3.5     0.042     0.167
860             1  ...   7.5  4.4  1.5        -3.3     0.115     0.170
...           ...  ...   ...  ...  ...         ...       ...       ...
11769           2  ...   9.2  2.7  1.3         1.9     0.014     0.097
12080           1  ...  24.3  6.9  5.7         4.2     0.011     0.171
12315           1  ...  23.8  6.1  5.1         4.0     0.025     0.153
12518           1  ...   5.0  1.8  2.4         1.9     0.017     0.077
12519           2  ...   8.2  2.4  1.0         1.3     0.013     0.109

       usg_pct  ts_pct  ast_pct   season
2        0.175   0.512    0.125  1996-97
3        0.206   0.527    0.125  1996-97
4        0.195   0.500    0.064  1996-97
859      0.189   0.507    0.105  1997-98
860      0.207   0.526    0.146  1997-98
...        ...     ...      ...      ...
11769    0.160   0.594    0.090  2021-22
12080    0.329   0.538    0.286  2021-22
12315    0.294   0.588    0.237  2022-23
12518    0.110   0.610    0.178  2022-23
12519    0.162   0.610    0.074  2022-23

[67 rows x 22 columns]
```

Nota que para usar expresiones con str.contains, necesitas establecer el parámetro engine='python'.

La función query es una herramienta súper buenas que hace que las consultas y filtros complejos sean más legibles y fáciles de escribir, especialmente cuando trabajas con varias condiciones al mismo tiempo.

7. Filtrado de Datos con mask

La función mask en Pandas es similar a where, pero con una diferencia clave: en lugar de mantener los valores que cumplen con la condición, mask reemplaza los valores que cumplen con la condición y deja intactos los que no la cumplen. Es una técnica útil cuando quieres transformar selectivamente un DataFrame basándote en condiciones lógicas.

Uso Básico de mask

Puedes usar mask para aplicar una condición a un DataFrame y reemplazar los valores que cumplen la condición por NaN u otro valor específico.

In [23]:

```
# Reemplazar con NaN donde 'age' es mayor que 25
masked_df = df.mask(df['age'] > 25)
print("DataFrame con NaN donde 'age' es mayor que 25:")
print(masked_df)
```

```
DataFrame con NaN donde 'age' es mayor que 25:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0             0.0  Randy Livingston               HOU  22.0         193.04
1             NaN               NaN               NaN   NaN            NaN
2             NaN               NaN               NaN   NaN            NaN
3             NaN               NaN               NaN   NaN            NaN
4             4.0      George Zidek               DEN  23.0         213.36
```

```
...           ...                ...                ...   ...            ...
12839         NaN                NaN                NaN   NaN            NaN
12840     12840.0   John Butler Jr.                POR  20.0         213.36
12841     12841.0      John Collins                ATL  25.0         205.74
12842     12842.0      Jericho Sims                NYK  24.0         208.28
12843         NaN                NaN                NaN   NaN            NaN

       player_weight          college country draft_year draft_round  ...  \
0          94.800728  Louisiana State     USA       1996           2  ...
1                NaN              NaN     NaN        NaN         NaN  ...
2                NaN              NaN     NaN        NaN         NaN  ...
3                NaN              NaN     NaN        NaN         NaN  ...
4         119.748288             UCLA     USA       1995           1  ...
...              ...              ...     ...        ...         ...  ...
12839            NaN              NaN     NaN        NaN         NaN  ...
12840      86.182480    Florida State     USA  Undrafted   Undrafted  ...
12841     102.511792      Wake Forest     USA       2017           1  ...
12842     113.398000            Texas     USA       2021           2  ...
12843            NaN              NaN     NaN        NaN         NaN  ...

        pts  reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0       3.9  1.5  2.4         0.3     0.042     0.071    0.169   0.487
1       NaN  NaN  NaN         NaN       NaN       NaN      NaN     NaN
2       NaN  NaN  NaN         NaN       NaN       NaN      NaN     NaN
3       NaN  NaN  NaN         NaN       NaN       NaN      NaN     NaN
4       2.8  1.7  0.3       -14.1     0.102     0.169    0.195   0.500
...     ...  ...  ...         ...       ...       ...      ...     ...
12839   NaN  NaN  NaN         NaN       NaN       NaN      NaN     NaN
12840   2.4  0.9  0.6       -16.1     0.012     0.065    0.102   0.411
12841  13.1  6.5  1.2        -0.2     0.035     0.180    0.168   0.593
12842   3.4  4.7  0.5        -6.7     0.117     0.175    0.074   0.780
12843   NaN  NaN  NaN         NaN       NaN       NaN      NaN     NaN

       ast_pct   season
0        0.248  1996-97
1          NaN      NaN
2          NaN      NaN
3          NaN      NaN
4        0.064  1996-97
...        ...      ...
12839      NaN      NaN
12840    0.066  2022-23
12841    0.052  2022-23
12842    0.044  2022-23
12843      NaN      NaN

[12844 rows x 22 columns]
```

En este ejemplo:

• La condición df['age'] > 25 se aplica al DataFrame.

• Las filas donde age es mayor que 25 se reemplazan con NaN, mientras que los demás valores permanecen intactos.

Uso Avanzado de mask

• Reemplazo con un valor específico: En lugar de reemplazar con NaN, puedes especificar un valor con el cual reemplazar los valores que cumplen la condición.

In [24]:

```
# Reemplazar con 0 donde 'age' es mayor que 25
masked_df = df.mask(df['age'] > 25, 0)
print("DataFrame con 0 donde 'age' es mayor que 25:")
print(masked_df)
```

```
DataFrame con 0 donde 'age' es mayor que 25:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0  Randy Livingston               HOU  22.0         193.04
1               0                 0                 0   0.0           0.00
2               0                 0                 0   0.0           0.00
3               0                 0                 0   0.0           0.00
4               4      George Zidek               DEN  23.0         213.36
...           ...               ...               ...   ...            ...
12839           0                 0                 0   0.0           0.00
12840       12840   John Butler Jr.               POR  20.0         213.36
12841       12841      John Collins               ATL  25.0         205.74
```

```
12842       12842      Jericho Sims              NYK  24.0         208.28
12843           0                 0                0   0.0           0.00

       player_weight          college country draft_year draft_round  ...  \
0          94.800728  Louisiana State     USA       1996           2  ...
1           0.000000                0       0          0           0  ...
2           0.000000                0       0          0           0  ...
3           0.000000                0       0          0           0  ...
4         119.748288             UCLA     USA       1995           1  ...
...              ...              ...     ...        ...         ...  ...
12839       0.000000                0       0          0           0  ...
12840      86.182480    Florida State     USA  Undrafted   Undrafted  ...
12841     102.511792      Wake Forest     USA       2017           1  ...
12842     113.398000            Texas     USA       2021           2  ...
12843       0.000000                0       0          0           0  ...

        pts  reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0       3.9  1.5  2.4         0.3     0.042     0.071    0.169   0.487
1       0.0  0.0  0.0         0.0     0.000     0.000    0.000   0.000
2       0.0  0.0  0.0         0.0     0.000     0.000    0.000   0.000
3       0.0  0.0  0.0         0.0     0.000     0.000    0.000   0.000
4       2.8  1.7  0.3       -14.1     0.102     0.169    0.195   0.500
...     ...  ...  ...         ...       ...       ...      ...     ...
12839   0.0  0.0  0.0         0.0     0.000     0.000    0.000   0.000
12840   2.4  0.9  0.6       -16.1     0.012     0.065    0.102   0.411
12841  13.1  6.5  1.2        -0.2     0.035     0.180    0.168   0.593
12842   3.4  4.7  0.5        -6.7     0.117     0.175    0.074   0.780
12843   0.0  0.0  0.0         0.0     0.000     0.000    0.000   0.000

       ast_pct   season
0        0.248  1996-97
1        0.000        0
2        0.000        0
3        0.000        0
4        0.064  1996-97
...        ...      ...
12839    0.000        0
12840    0.066  2022-23
12841    0.052  2022-23
12842    0.044  2022-23
12843    0.000        0

[12844 rows x 22 columns]
```

• Combinación de condiciones: Al igual que con otras técnicas, puedes combinar condiciones usando operadores lógicos.

In [25]:

```
# Reemplazar con NaN donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU'
masked_df = df.mask((df['age'] > 25) & (df['team_abbreviation'] == 'HOU'))
print("DataFrame con NaN donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU':")
print(masked_df)
```

```
DataFrame con NaN donde 'age' es mayor que 25 y 'team_abbreviation' es 'HOU':
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0             0.0  Randy Livingston               HOU  22.0         193.04
1             1.0  Gaylon Nickerson               WAS  28.0         190.50
2             2.0      George Lynch               VAN  26.0         203.20
3             3.0    George McCloud               LAL  30.0         203.20
4             4.0      George Zidek               DEN  23.0         213.36
...           ...               ...               ...   ...           ...
12839     12839.0       Joel Embiid               PHI  29.0         213.36
12840     12840.0   John Butler Jr.               POR  20.0         213.36
12841     12841.0      John Collins               ATL  25.0         205.74
12842     12842.0      Jericho Sims               NYK  24.0         208.28
12843     12843.0    JaMychal Green               GSW  33.0         205.74

       player_weight               college   country draft_year draft_round  \
0          94.800728       Louisiana State       USA       1996           2
1          86.182480  Northwestern Oklahoma       USA       1994           2
2         103.418976        North Carolina       USA       1993           1
3         102.058200         Florida State       USA       1989           1
4         119.748288                  UCLA       USA       1995           1
...              ...                   ...       ...        ...         ...
12839     127.005760               Kansas  Cameroon       2014           1
12840      86.182480         Florida State       USA  Undrafted   Undrafted
12841     102.511792           Wake Forest       USA       2017           1
12842     113.398000                 Texas       USA       2021           2
12843     102.965384               Alabama       USA  Undrafted   Undrafted
```

```
       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0      ...   3.9   1.5  2.4         0.3     0.042     0.071    0.169   0.487
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4      ...   2.8   1.7  0.3       -14.1     0.102     0.169    0.195   0.500
...    ...   ...   ...  ...         ...       ...       ...      ...     ...
12839  ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
12840  ...   2.4   0.9  0.6       -16.1     0.012     0.065    0.102   0.411
12841  ...  13.1   6.5  1.2        -0.2     0.035     0.180    0.168   0.593
12842  ...   3.4   4.7  0.5        -6.7     0.117     0.175    0.074   0.780
12843  ...   6.4   3.6  0.9        -8.2     0.087     0.164    0.169   0.650

       ast_pct   season
0        0.248  1996-97
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.064  1996-97
...        ...      ...
12839    0.233  2022-23
12840    0.066  2022-23
12841    0.052  2022-23
12842    0.044  2022-23
12843    0.094  2022-23

[12844 rows x 22 columns]
```

En este ejemplo, solo se reemplazarán los valores que cumplen ambas condiciones (age > 25 y team_abbreviation == 'HOU'), y el resto del DataFrame permanecerá sin cambios.

La función mask es muy útil cuando necesitas modificar un DataFrame basándote en condiciones, manteniendo o transformando solo ciertas partes de los datos. Es una herramienta poderosa para la manipulación selectiva de datos.

8. Filtrado de Datos con dropna

La función dropna en Pandas se utiliza para eliminar filas o columnas que contienen valores nulos (NaN). Es una herramienta esencial cuando deseas limpiar tu DataFrame y trabajar solo con datos completos, eliminando cualquier fila o columna que tenga valores faltantes.

Uso Básico de dropna

Puedes usar dropna para eliminar todas las filas que contienen al menos un valor nulo.

In [26]:

```
# Eliminar filas con cualquier valor nulo
cleaned_df = df.dropna()
print("DataFrame después de eliminar filas con valores nulos:")
print(cleaned_df)
```

```
DataFrame después de eliminar filas con valores nulos:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0  Randy Livingston              HOU  22.0         193.04
1               1  Gaylon Nickerson              WAS  28.0         190.50
2               2      George Lynch              VAN  26.0         203.20
3               3    George McCloud              LAL  30.0         203.20
4               4      George Zidek              DEN  23.0         213.36
...           ...               ...              ...   ...            ...
12839       12839       Joel Embiid              PHI  29.0         213.36
12840       12840   John Butler Jr.              POR  20.0         213.36
12841       12841      John Collins              ATL  25.0         205.74
12842       12842      Jericho Sims              NYK  24.0         208.28
12843       12843    JaMychal Green              GSW  33.0         205.74

       player_weight               college   country draft_year draft_round  \
0          94.800728       Louisiana State       USA       1996           2
1          86.182480  Northwestern Oklahoma       USA       1994           2
2         103.418976        North Carolina       USA       1993           1
3         102.058200         Florida State       USA       1989           1
4         119.748288                  UCLA       USA       1995           1
...              ...                   ...       ...        ...         ...
12839     127.005760               Kansas  Cameroon       2014           1
12840      86.182480         Florida State       USA  Undrafted   Undrafted
12841     102.511792           Wake Forest       USA       2017           1
12842     113.398000                 Texas       USA       2021           2
12843     102.965384               Alabama       USA  Undrafted   Undrafted
```

```
       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0      ...   3.9   1.5  2.4         0.3     0.042     0.071    0.169   0.487
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4      ...   2.8   1.7  0.3       -14.1     0.102     0.169    0.195   0.500
...    ...   ...   ...  ...         ...       ...       ...      ...     ...
12839  ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
12840  ...   2.4   0.9  0.6       -16.1     0.012     0.065    0.102   0.411
12841  ...  13.1   6.5  1.2        -0.2     0.035     0.180    0.168   0.593
12842  ...   3.4   4.7  0.5        -6.7     0.117     0.175    0.074   0.780
12843  ...   6.4   3.6  0.9        -8.2     0.087     0.164    0.169   0.650

       ast_pct   season
0        0.248  1996-97
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.064  1996-97
...        ...      ...
12839    0.233  2022-23
12840    0.066  2022-23
12841    0.052  2022-23
12842    0.044  2022-23
12843    0.094  2022-23

[10990 rows x 22 columns]
```

En este ejemplo:

• dropna() elimina todas las filas que tienen al menos un valor nulo en cualquier columna, resultando en un DataFrame que solo contiene datos completos.

Uso Avanzado de dropna

• Eliminar columnas con valores nulos: Si prefieres eliminar columnas en lugar de filas, puedes utilizar el parámetro axis=1.

In [27]:

```
# Eliminar columnas con cualquier valor nulo
cleaned_df = df.dropna(axis=1)
print("DataFrame después de eliminar columnas con valores nulos:")
print(cleaned_df)
```

```
DataFrame después de eliminar columnas con valores nulos:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0  Randy Livingston               HOU  22.0         193.04
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
4               4      George Zidek               DEN  23.0         213.36
...           ...               ...               ...   ...            ...
12839       12839       Joel Embiid               PHI  29.0         213.36
12840       12840   John Butler Jr.               POR  20.0         213.36
12841       12841      John Collins               ATL  25.0         205.74
12842       12842      Jericho Sims               NYK  24.0         208.28
12843       12843    JaMychal Green               GSW  33.0         205.74

       player_weight   country draft_year draft_round draft_number  ...   pts  \
0          94.800728       USA       1996           2           42  ...   3.9
1          86.182480       USA       1994           2           34  ...   3.8
2         103.418976       USA       1993           1           12  ...   8.3
3         102.058200       USA       1989           1            7  ...  10.2
4         119.748288       USA       1995           1           22  ...   2.8
...              ...       ...        ...         ...          ...  ...   ...
12839     127.005760  Cameroon       2014           1            3  ...  33.1
12840      86.182480       USA  Undrafted   Undrafted    Undrafted  ...   2.4
12841     102.511792       USA       2017           1           19  ...  13.1
12842     113.398000       USA       2021           2           58  ...   3.4
12843     102.965384       USA  Undrafted   Undrafted    Undrafted  ...   6.4

        reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  ast_pct  \
0       1.5  2.4         0.3     0.042     0.071    0.169   0.487    0.248
1       1.3  0.3         8.9     0.030     0.111    0.174   0.497    0.043
2       6.4  1.9        -8.2     0.106     0.185    0.175   0.512    0.125
3       2.8  1.7        -2.7     0.027     0.111    0.206   0.527    0.125
4       1.7  0.3       -14.1     0.102     0.169    0.195   0.500    0.064
```

```
...     ...  ...         ...       ...       ...      ...     ...      ...
12839  10.2  4.2         8.8     0.057     0.243    0.370   0.655    0.233
12840   0.9  0.6       -16.1     0.012     0.065    0.102   0.411    0.066
12841   6.5  1.2        -0.2     0.035     0.180    0.168   0.593    0.052
12842   4.7  0.5        -6.7     0.117     0.175    0.074   0.780    0.044
12843   3.6  0.9        -8.2     0.087     0.164    0.169   0.650    0.094

        season
0      1996-97
1      1996-97
2      1996-97
3      1996-97
4      1996-97
...        ...
12839  2022-23
12840  2022-23
12841  2022-23
12842  2022-23
12843  2022-23

[12844 rows x 21 columns]
```

• Eliminar filas o columnas si todos los valores son nulos: **Puedes ajustar el comportamiento de dropna para eliminar solo aquellas filas o columnas donde todos los valores son nulos.**

In [28]:

```
# Eliminar filas donde todos los valores son nulos
cleaned_df = df.dropna(how='all')
print("DataFrame después de eliminar filas donde todos los valores son nulos:")
print(cleaned_df)
```

```
DataFrame después de eliminar filas donde todos los valores son nulos:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0  Randy Livingston               HOU  22.0         193.04
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
4               4      George Zidek               DEN  23.0         213.36
...           ...               ...               ...   ...            ...
12839       12839       Joel Embiid               PHI  29.0         213.36
12840       12840   John Butler Jr.               POR  20.0         213.36
12841       12841      John Collins               ATL  25.0         205.74
12842       12842      Jericho Sims               NYK  24.0         208.28
12843       12843    JaMychal Green               GSW  33.0         205.74

       player_weight                college   country draft_year draft_round  \
0          94.800728        Louisiana State       USA       1996           2
1          86.182480  Northwestern Oklahoma       USA       1994           2
2         103.418976         North Carolina       USA       1993           1
3         102.058200          Florida State       USA       1989           1
4         119.748288                   UCLA       USA       1995           1
...              ...                    ...       ...        ...         ...
12839     127.005760                 Kansas  Cameroon       2014           1
12840      86.182480          Florida State       USA  Undrafted   Undrafted
12841     102.511792            Wake Forest       USA       2017           1
12842     113.398000                  Texas       USA       2021           2
12843     102.965384                Alabama       USA  Undrafted   Undrafted

       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0      ...   3.9   1.5  2.4         0.3     0.042     0.071    0.169   0.487
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4      ...   2.8   1.7  0.3       -14.1     0.102     0.169    0.195   0.500
...    ...   ...   ...  ...         ...       ...       ...      ...     ...
12839  ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
12840  ...   2.4   0.9  0.6       -16.1     0.012     0.065    0.102   0.411
12841  ...  13.1   6.5  1.2        -0.2     0.035     0.180    0.168   0.593
12842  ...   3.4   4.7  0.5        -6.7     0.117     0.175    0.074   0.780
12843  ...   6.4   3.6  0.9        -8.2     0.087     0.164    0.169   0.650

       ast_pct   season
0        0.248  1996-97
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.064  1996-97
...        ...      ...
12839    0.233  2022-23
12840    0.066  2022-23
```

```
12841    0.052  2022-23
12842    0.044  2022-23
12843    0.094  2022-23

[12844 rows x 22 columns]
```

• Eliminar filas o columnas con un umbral de valores no nulos: Puedes especificar cuántos valores no nulos debe tener una fila o columna para no ser eliminada, utilizando el parámetro thresh.

In [29]:

```
# Mantener filas que tienen al menos 3 valores no nulos
cleaned_df = df.dropna(thresh=3)
print("DataFrame después de eliminar filas con menos de 3 valores no nulos:")
print(cleaned_df)
```

```
DataFrame después de eliminar filas con menos de 3 valores no nulos:
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
0               0  Randy Livingston               HOU  22.0         193.04
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
4               4      George Zidek               DEN  23.0         213.36
...           ...               ...               ...   ...            ...
12839       12839       Joel Embiid               PHI  29.0         213.36
12840       12840   John Butler Jr.               POR  20.0         213.36
12841       12841      John Collins               ATL  25.0         205.74
12842       12842      Jericho Sims               NYK  24.0         208.28
12843       12843    JaMychal Green               GSW  33.0         205.74

       player_weight                college   country draft_year draft_round  \
0          94.800728        Louisiana State       USA       1996           2
1          86.182480  Northwestern Oklahoma       USA       1994           2
2         103.418976         North Carolina       USA       1993           1
3         102.058200          Florida State       USA       1989           1
4         119.748288                   UCLA       USA       1995           1
...              ...                    ...       ...        ...         ...
12839     127.005760                Kansas  Cameroon       2014           1
12840      86.182480          Florida State       USA  Undrafted   Undrafted
12841     102.511792            Wake Forest       USA       2017           1
12842     113.398000                  Texas       USA       2021           2
12843     102.965384                Alabama       USA  Undrafted   Undrafted

       ...   pts   reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
0      ...   3.9   1.5  2.4         0.3     0.042     0.071    0.169   0.487
1      ...   3.8   1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3   6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2   2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4      ...   2.8   1.7  0.3       -14.1     0.102     0.169    0.195   0.500
...    ...   ...   ...  ...         ...       ...       ...      ...     ...
12839  ...  33.1  10.2  4.2         8.8     0.057     0.243    0.370   0.655
12840  ...   2.4   0.9  0.6       -16.1     0.012     0.065    0.102   0.411
12841  ...  13.1   6.5  1.2        -0.2     0.035     0.180    0.168   0.593
12842  ...   3.4   4.7  0.5        -6.7     0.117     0.175    0.074   0.780
12843  ...   6.4   3.6  0.9        -8.2     0.087     0.164    0.169   0.650

       ast_pct   season
0        0.248  1996-97
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.064  1996-97
...        ...      ...
12839    0.233  2022-23
12840    0.066  2022-23
12841    0.052  2022-23
12842    0.044  2022-23
12843    0.094  2022-23

[12844 rows x 22 columns]
```

La función dropna es crucial para el manejo de datos faltantes, permitiéndote limpiar rápidamente tu DataFrame de filas o columnas incompletas. Es una técnica básica pero poderosa en la preparación de datos para análisis posteriores.

9. Filtrado de Datos con filter

La función filter en Pandas se utiliza para seleccionar subconjuntos específicos de columnas o filas en un DataFrame

basándose en nombres o patrones de nombres. Es una herramienta útil cuando trabajas con DataFrames que tienen muchas columnas y solo quieres trabajar con aquellas que coinciden con ciertos criterios.

Uso Básico de filter para Seleccionar Columnas

Puedes usar filter para seleccionar columnas que coincidan con un patrón específico o que contengan ciertos caracteres en sus nombres.

In [30]:

```
# Seleccionar columnas cuyos nombres contienen 'name'
filtered_columns_df = df.filter(like='name')
print("Columnas que contienen 'name' en su nombre:")
print(filtered_columns_df)
```

```
Columnas que contienen 'name' en su nombre:
       Unnamed: 0       player_name
0               0  Randy Livingston
1               1  Gaylon Nickerson
2               2      George Lynch
3               3    George McCloud
4               4      George Zidek
...           ...               ...
12839       12839       Joel Embiid
12840       12840   John Butler Jr.
12841       12841      John Collins
12842       12842      Jericho Sims
12843       12843    JaMychal Green

[12844 rows x 2 columns]
```

En este ejemplo:

- El parámetro like='name' indica que queremos seleccionar todas las columnas cuyos nombres contienen la palabra “name”.

- El DataFrame resultante solo incluirá las columnas que cumplen con este criterio.

Otras Opciones de filter

- **Filtrar columnas que comienzan o terminan con un patrón específico:** Puedes usar los parámetros regex para patrones más complejos.

In [31]:

```
# Seleccionar columnas que empiezan con 'player'
filtered_columns_df = df.filter(regex='^player')
print("Columnas que comienzan con 'player':")
print(filtered_columns_df)
```

```
Columnas que comienzan con 'player':
            player_name  player_height  player_weight
0      Randy Livingston         193.04      94.800728
1      Gaylon Nickerson         190.50      86.182480
2          George Lynch         203.20     103.418976
3        George McCloud         203.20     102.058200
4          George Zidek         213.36     119.748288
...                 ...            ...            ...
12839       Joel Embiid         213.36     127.005760
12840   John Butler Jr.         213.36      86.182480
12841      John Collins         205.74     102.511792
12842      Jericho Sims         208.28     113.398000
12843    JaMychal Green         205.74     102.965384

[12844 rows x 3 columns]
```

- **Seleccionar columnas específicas:** Puedes proporcionar una lista de nombres de columnas que deseas seleccionar.

In [32]:

```
# Seleccionar columnas específicas por nombre
filtered_columns_df = df.filter(items=['player_name', 'team_abbreviation', 'age'])
print("Columnas 'player_name', 'team_abbreviation' y 'age':")
print(filtered_columns_df)
```

```
Columnas 'player_name', 'team_abbreviation' y 'age':
            player_name team_abbreviation   age
0      Randy Livingston               HOU  22.0
```

```
1      Gaylon Nickerson               WAS  28.0
2          George Lynch               VAN  26.0
3        George McCloud               LAL  30.0
4          George Zidek               DEN  23.0
...                 ...               ...   ...
12839       Joel Embiid               PHI  29.0
12840   John Butler Jr.               POR  20.0
12841      John Collins               ATL  25.0
12842      Jericho Sims               NYK  24.0
12843    JaMychal Green               GSW  33.0

[12844 rows x 3 columns]
```

• Filtrar Filas (con el parámetro axis): Aunque es menos común, también puedes filtrar filas utilizando filter estableciendo axis=0.

In [33]:

```
# Seleccionar filas que coincidan con ciertos índices
filtered_rows_df = df.filter(items=[0, 1, 2], axis=0)
print("Filas con índices 0, 1, 2:")
print(filtered_rows_df)
```

```
Filas con índices 0, 1, 2:
   Unnamed: 0       player_name team_abbreviation   age  player_height  \
0           0  Randy Livingston               HOU  22.0         193.04
1           1  Gaylon Nickerson               WAS  28.0         190.50
2           2      George Lynch               VAN  26.0         203.20

   player_weight               college country draft_year draft_round  ...  \
0      94.800728        Louisiana State     USA       1996           2  ...
1      86.182480  Northwestern Oklahoma     USA       1994           2  ...
2     103.418976         North Carolina     USA       1993           1  ...

   pts  reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  ast_pct  \
0  3.9  1.5  2.4         0.3     0.042     0.071    0.169   0.487    0.248
1  3.8  1.3  0.3         8.9     0.030     0.111    0.174   0.497    0.043
2  8.3  6.4  1.9        -8.2     0.106     0.185    0.175   0.512    0.125

    season
0  1996-97
1  1996-97
2  1996-97

[3 rows x 22 columns]
```

La función filter es una herramienta poderosa y flexible para seleccionar columnas o filas en un DataFrame basado en patrones o nombres específicos. Es especialmente útil cuando necesitas trabajar con subconjuntos de datos sin tener que especificar manualmente cada columna o fila que te interesa.

10. Filtrado de Datos con Expresiones Regulares (str.match)

El método str.match en Pandas se utiliza para filtrar filas en un DataFrame basado en si los valores de una columna de texto coinciden con un patrón específico al principio de la cadena. A diferencia de str.contains, que busca en cualquier parte de la cadena, str.match solo hace coincidir los patrones desde el inicio de la cadena.

Uso Básico de str.match

Puedes usar str.match para filtrar filas donde los valores de una columna comienzan con un patrón específico.

In [34]:

```
# Filtrar filas donde 'player_name' empieza con 'Geo'
filtered_df = df[df['player_name'].str.match('^Geo')]
print("Filas donde 'player_name' comienza con 'Geo':")
print(filtered_df)
```

```
Filas donde 'player_name' comienza con 'Geo':
       Unnamed: 0          player_name team_abbreviation   age  \
2               2         George Lynch               VAN  26.0
3               3       George McCloud               LAL  30.0
4               4         George Zidek               DEN  23.0
859           859       George McCloud               PHX  31.0
860           860         George Lynch               VAN  27.0
866           866         George Zidek               SEA  24.0
974           974         George Lynch               PHI  28.0
```

975	975	George McCloud	PHX	32.0	
1661	1661	George McCloud	DEN	33.0	
1662	1662	George Lynch	PHI	29.0	
1804	1804	George McCloud	DEN	34.0	
1812	1812	George Lynch	PHI	30.0	
2369	2369	George McCloud	DEN	35.0	
2377	2377	George Lynch	CHH	31.0	
3045	3045	George Lynch	NOH	32.0	
3269	3269	George Lynch	NOH	33.0	
3954	3954	George Lynch	NOH	34.0	
5506	5506	George Hill	SAS	23.0	
5964	5964	George Hill	SAS	24.0	
6255	6255	George Hill	SAS	25.0	
6817	6817	George Hill	IND	26.0	
7442	7442	George Hill	IND	27.0	
7652	7652	George Hill	IND	28.0	
8395	8395	George Hill	IND	29.0	
8604	8604	George Hill	IND	30.0	
9549	9549	Georgios Papagiannis	SAC	19.0	
9558	9558	Georges Niang	IND	24.0	
9559	9559	George Hill	UTA	31.0	
10067	10067	George Hill	CLE	32.0	
10096	10096	Georgios Papagiannis	POR	20.0	
10097	10097	Georges Niang	UTA	25.0	
10111	10111	George Hill	MIL	33.0	
10112	10112	George King	PHX	25.0	
10113	10113	Georges Niang	UTA	26.0	
10675	10675	Georges Niang	UTA	27.0	
10676	10676	George Hill	MIL	34.0	
11169	11169	George Hill	PHI	35.0	
11173	11173	Georges Niang	UTA	28.0	
11762	11762	Georgios Kalaitzakis	OKC	23.0	
11763	11763	George King	DAL	28.0	
11764	11764	George Hill	MIL	36.0	
11769	11769	Georges Niang	PHI	29.0	
12518	12518	George Hill	IND	37.0	
12519	12519	Georges Niang	PHI	30.0	

	player_height	player_weight	college	country	\
2	203.20	103.418976	North Carolina	USA	
3	203.20	102.058200	Florida State	USA	
4	213.36	119.748288	UCLA	USA	
859	203.20	102.058200	Florida State	USA	
860	203.20	103.418976	North Carolina	USA	
866	213.36	113.398000	UCLA	USA	
974	203.20	103.418976	North Carolina	USA	
975	203.20	102.058200	Florida State	USA	
1661	203.20	102.058200	Florida State	USA	
1662	203.20	103.418976	North Carolina	USA	
1804	203.20	102.058200	Florida State	USA	
1812	203.20	103.418976	North Carolina	USA	
2369	203.20	102.058200	Florida State	USA	
2377	203.20	103.418976	North Carolina	USA	
3045	203.20	106.594120	North Carolina	USA	
3269	203.20	106.594120	North Carolina	USA	
3954	203.20	106.594120	North Carolina	USA	
5506	187.96	81.646560	Indiana Purdue-Indianapolis	USA	
5964	187.96	81.646560	Indiana Purdue-Indianapolis	USA	
6255	187.96	81.646560	Indiana Purdue-Indianapolis	USA	
6817	187.96	81.646560	Indiana Purdue-Indianapolis	USA	
7442	187.96	86.182480	Indiana Purdue-Indianapolis	USA	
7652	190.50	85.275296	Indiana Purdue-Indianapolis	USA	
8395	190.50	85.275296	Indiana Purdue-Indianapolis	USA	
8604	190.50	85.275296	Indiana Purdue-Indianapolis	USA	
9549	215.90	108.862080	NaN	Greece	
9558	203.20	104.326160	Iowa State	USA	
9559	190.50	85.275296	Indiana Purdue-Indianapolis	USA	
10067	190.50	85.275296	NaN	USA	
10096	215.90	108.862080	NaN	Greece	
10097	198.12	104.779752	Iowa State	USA	
10111	190.50	85.275296	NaN	USA	
10112	198.12	99.790240	University of Colorado Boulder	USA	
10113	203.20	104.326160	Iowa State	USA	
10675	200.66	104.326160	Iowa State	USA	
10676	190.50	85.275296	Indiana-Purdue Indianapolis	USA	
11169	193.04	85.275296	Indiana-Purdue Indianapolis	USA	
11173	200.66	104.326160	Iowa State	USA	
11762	200.66	87.089664	NaN	Greece	
11763	198.12	99.790240	Colorado	USA	
11764	193.04	85.275296	Indiana-Purdue Indianapolis	USA	
11769	200.66	104.326160	Iowa State	USA	
12518	193.04	85.275296	Indiana-Purdue Indianapolis	USA	

```
12518         193.04      85.275290     Indiana-Purdue Indianapolis     USA
12519         200.66     104.326160                     Iowa State     USA

      draft_year draft_round  ...   pts  reb  ast  net_rating  oreb_pct  \
2           1993           1  ...   8.3  6.4  1.9        -8.2     0.106
3           1989           1  ...  10.2  2.8  1.7        -2.7     0.027
4           1995           1  ...   2.8  1.7  0.3       -14.1     0.102
859         1989           1  ...   7.2  3.5  1.3         3.5     0.042
860         1993           1  ...   7.5  4.4  1.5        -3.3     0.115
866         1995           1  ...   2.4  1.4  0.2       -12.3     0.069
974         1993           1  ...   8.3  6.5  1.8         5.1     0.095
975         1989           1  ...   8.9  3.4  1.6         1.4     0.032
1661        1989           1  ...  10.1  3.7  3.2        -2.4     0.038
1662        1993           1  ...   9.6  7.8  1.8         2.6     0.097
1804        1989           1  ...   9.6  2.9  3.7        -5.1     0.031
1812        1993           1  ...   8.4  7.2  1.7         4.1     0.087
2369        1989           1  ...   8.8  3.6  3.0       -10.4     0.040
2377        1993           1  ...   3.8  4.1  1.2         0.2     0.085
3045        1993           1  ...   4.5  4.4  1.3         4.8     0.105
3269        1993           1  ...   4.8  4.0  1.5        -2.9     0.062
3954        1993           1  ...   3.7  4.0  2.0        -6.6     0.068
5506        2008           1  ...   5.7  2.1  1.8         2.2     0.028
5964        2008           1  ...  12.4  2.6  2.9         4.6     0.022
6255        2008           1  ...  11.6  2.6  2.5         7.3     0.014
6817        2008           1  ...   9.6  3.0  2.9         3.0     0.028
7442        2008           1  ...  14.2  3.7  4.7         8.7     0.020
7652        2008           1  ...  10.3  3.7  3.5         6.2     0.026
8395        2008           1  ...  16.1  4.2  5.1         7.1     0.022
8604        2008           1  ...  12.1  4.0  3.5         2.5     0.026
9549        2016           1  ...   5.6  3.9  0.9        -6.2     0.090
9558        2016           2  ...   0.9  0.7  0.2        -5.3     0.021
9559        2008           1  ...  16.9  3.4  4.2         8.6     0.017
10067       2008           1  ...  10.0  2.7  2.8        -3.3     0.023
10096       2016           1  ...   2.1  2.2  0.5        -3.7     0.100
10097       2016           2  ...   1.0  1.0  0.3       -18.1     0.103
10111       2008           1  ...   7.6  2.5  2.3         5.6     0.028
10112       2018           2  ...   0.0  1.0  0.0       -75.0     0.000
10113       2016           2  ...   4.0  1.5  0.6        -7.7     0.022
10675       2016           2  ...   5.9  1.9  0.7        -1.1     0.015
10676       2008           1  ...   9.4  3.0  3.1        10.6     0.038
11169       2008           1  ...   8.7  2.0  2.4        -0.4     0.023
11173       2016           2  ...   6.9  2.4  0.8        14.6     0.024
11762       2021           2  ...   6.6  1.6  0.9       -29.3     0.026
11763       2018           2  ...   0.3  1.3  0.0       -62.2     0.000
11764       2008           1  ...   6.2  2.9  2.2        11.7     0.029
11769       2016           2  ...   9.2  2.7  1.3         1.9     0.014
12518       2008           1  ...   5.0  1.8  2.4         1.9     0.017
12519       2016           2  ...   8.2  2.4  1.0         1.3     0.013

       dreb_pct  usg_pct  ts_pct  ast_pct   season
2         0.185    0.175   0.512    0.125  1996-97
3         0.111    0.206   0.527    0.125  1996-97
4         0.169    0.195   0.500    0.064  1996-97
859       0.167    0.189   0.507    0.105  1997-98
860       0.170    0.207   0.526    0.146  1997-98
866       0.186    0.266   0.402    0.061  1997-98
974       0.150    0.160   0.461    0.100  1998-99
975       0.125    0.153   0.591    0.101  1998-99
1661      0.112    0.176   0.548    0.182  1999-00
1662      0.169    0.154   0.498    0.089  1999-00
1804      0.096    0.190   0.496    0.223  2000-01
1812      0.163    0.138   0.496    0.084  2000-01
2369      0.123    0.192   0.466    0.187  2001-02
2377      0.146    0.131   0.399    0.100  2001-02
3045      0.166    0.133   0.464    0.109  2002-03
3269      0.148    0.126   0.472    0.114  2003-04
3954      0.162    0.131   0.412    0.160  2004-05
5506      0.120    0.195   0.502    0.183  2008-09
5964      0.085    0.190   0.572    0.154  2009-10
6255      0.093    0.185   0.588    0.142  2010-11
6817      0.103    0.173   0.557    0.194  2011-12
7442      0.099    0.188   0.558    0.220  2012-13
7652      0.101    0.148   0.563    0.170  2013-14
8395      0.134    0.243   0.579    0.304  2014-15
8604      0.105    0.160   0.555    0.157  2015-16
9549      0.183    0.169   0.573    0.085  2016-17
9558      0.169    0.189   0.285    0.074  2016-17
9559      0.105    0.236   0.599    0.227  2016-17
10067     0.081    0.161   0.580    0.145  2017-18
10096     0.216    0.170   0.429    0.097  2017-18
10097     0.182    0.190   0.379    0.167  2017-18
10111     0.075    0.147   0.554    0.146  2018-19
```

```
10112     0.250    0.000   0.000    0.000  2018-19
10113     0.135    0.170   0.613    0.107  2018-19
10675     0.117    0.163   0.589    0.069  2019-20
10676     0.086    0.152   0.659    0.196  2019-20
11169     0.063    0.161   0.596    0.156  2020-21
11173     0.117    0.169   0.602    0.071  2020-21
11762     0.074    0.200   0.536    0.090  2021-22
11763     0.250    0.159   0.085    0.000  2021-22
11764     0.084    0.115   0.552    0.124  2021-22
11769     0.097    0.160   0.594    0.090  2021-22
12518     0.077    0.110   0.610    0.178  2022-23
12519     0.109    0.162   0.610    0.074  2022-23

[44 rows x 22 columns]
```

En este ejemplo:

• str.match('^Geo') busca todas las filas donde la columna player_name comienza con “Geo”.

• El símbolo ^ en la expresión regular indica que la coincidencia debe ocurrir al principio de la cadena.

Uso Avanzado de str.match

• Coincidencia con múltiples patrones: Puedes buscar coincidencias que empiecen con diferentes patrones utilizando el operador | en la expresión regular.

In [35]:

```
# Filtrar filas donde 'player_name' empieza con 'Geo' o 'Rand'
filtered_df = df[df['player_name'].str.match('^Geo|^Rand')]
print("Filas donde 'player_name' comienza con 'Geo' o 'Rand':")
print(filtered_df)
```

```
Filas donde 'player_name' comienza con 'Geo' o 'Rand':
       Unnamed: 0         player_name team_abbreviation   age  player_height  \
0               0    Randy Livingston               HOU  22.0         193.04
2               2        George Lynch               VAN  26.0         203.20
3               3      George McCloud               LAL  30.0         203.20
4               4        George Zidek               DEN  23.0         213.36
359           359  Randolph Childress               DET  24.0         187.96
...           ...                 ...               ...   ...            ...
11763       11763         George King               DAL  28.0         198.12
11764       11764         George Hill               MIL  36.0         193.04
11769       11769       Georges Niang               PHI  29.0         200.66
12518       12518         George Hill               IND  37.0         193.04
12519       12519       Georges Niang               PHI  30.0         200.66

       player_weight                      college country draft_year  \
0          94.800728              Louisiana State     USA       1996
2         103.418976               North Carolina     USA       1993
3         102.058200                Florida State     USA       1989
4         119.748288                         UCLA     USA       1995
359        85.275296                 Wake Forest     USA       1995
...              ...                          ...     ...        ...
11763      99.790240                     Colorado     USA       2018
11764      85.275296  Indiana-Purdue Indianapolis     USA       2008
11769     104.326160                   Iowa State     USA       2016
12518      85.275296  Indiana-Purdue Indianapolis     USA       2008
12519     104.326160                   Iowa State     USA       2016

      draft_round  ...   pts  reb  ast  net_rating  oreb_pct  dreb_pct  \
0               2  ...   3.9  1.5  2.4         0.3     0.042     0.071
2               1  ...   8.3  6.4  1.9        -8.2     0.106     0.185
3               1  ...  10.2  2.8  1.7        -2.7     0.027     0.111
4               1  ...   2.8  1.7  0.3       -14.1     0.102     0.169
359             1  ...   1.7  0.3  0.7        -0.9     0.008     0.037
...           ...  ...   ...  ...  ...         ...       ...       ...
11763           2  ...   0.3  1.3  0.0       -62.2     0.000     0.250
11764           1  ...   6.2  2.9  2.2        11.7     0.029     0.084
11769           2  ...   9.2  2.7  1.3         1.9     0.014     0.097
12518           1  ...   5.0  1.8  2.4         1.9     0.017     0.077
12519           2  ...   8.2  2.4  1.0         1.3     0.013     0.109

       usg_pct  ts_pct  ast_pct   season
0        0.169   0.487    0.248  1996-97
2        0.175   0.512    0.125  1996-97
3        0.206   0.527    0.125  1996-97
4        0.195   0.500    0.064  1996-97
```

```
359      0.176   0.448    0.205  1996-97
...        ...     ...      ...      ...
11763    0.159   0.085    0.000  2021-22
11764    0.115   0.552    0.124  2021-22
11769    0.160   0.594    0.090  2021-22
12518    0.110   0.610    0.178  2022-23
12519    0.162   0.610    0.074  2022-23

[81 rows x 22 columns]
```

• Combinación con otras funciones de Pandas: Puedes combinar str.match con otras funciones de Pandas para crear filtros más complejos.

In [37]:

```
# Filtrar filas donde 'player_name' empieza con 'Geo' y 'team_abbreviation' es 'PHI'
filtered_df = df[df['player_name'].str.match('^Geo') & (df['team_abbreviation'] == 'PHI')]
print("Filas donde 'player_name' comienza con 'Geo' y 'team_abbreviation' es 'PHI':")
print(filtered_df)
```

```
Filas donde 'player_name' comienza con 'Geo' y 'team_abbreviation' es 'PHI':
       Unnamed: 0    player_name team_abbreviation   age  player_height  \
974           974   George Lynch               PHI  28.0         203.20
1662         1662   George Lynch               PHI  29.0         203.20
1812         1812   George Lynch               PHI  30.0         203.20
11169       11169    George Hill               PHI  35.0         193.04
11769       11769  Georges Niang               PHI  29.0         200.66
12519       12519  Georges Niang               PHI  30.0         200.66

       player_weight                     college country draft_year  \
974       103.418976               North Carolina     USA       1993
1662      103.418976               North Carolina     USA       1993
1812      103.418976               North Carolina     USA       1993
11169      85.275296  Indiana-Purdue Indianapolis     USA       2008
11769     104.326160                   Iowa State     USA       2016
12519     104.326160                   Iowa State     USA       2016

      draft_round  ...  pts  reb  ast  net_rating  oreb_pct  dreb_pct  \
974             1  ...  8.3  6.5  1.8         5.1     0.095     0.150
1662            1  ...  9.6  7.8  1.8         2.6     0.097     0.169
1812            1  ...  8.4  7.2  1.7         4.1     0.087     0.163
11169           1  ...  8.7  2.0  2.4        -0.4     0.023     0.063
11769           2  ...  9.2  2.7  1.3         1.9     0.014     0.097
12519           2  ...  8.2  2.4  1.0         1.3     0.013     0.109

       usg_pct  ts_pct  ast_pct   season
974      0.160   0.461    0.100  1998-99
1662     0.154   0.498    0.089  1999-00
1812     0.138   0.496    0.084  2000-01
11169    0.161   0.596    0.156  2020-21
11769    0.160   0.594    0.090  2021-22
12519    0.162   0.610    0.074  2022-23

[6 rows x 22 columns]
```

En este ejemplo, la combinación de str.match con una condición adicional (team_abbreviation == 'PHI') filtra aún más los datos para encontrar filas que coincidan con ambos criterios.

Expresiones Regulares en str.match

Las expresiones regulares son poderosas para encontrar patrones complejos en texto. Con str.match, puedes buscar patrones como:

- **Empieza con: ^**
- **Termina con: $**
- **Cualquier carácter: . (punto)**
- **Conjunto de caracteres: [abc]**
- **Rango de caracteres: [a-z]**

Por ejemplo:

In [38]:

```
# Filtrar filas donde 'player_name' empieza con una letra entre 'A' y 'G'
```

```
filtered_df = df[df['player_name'].str.match('^[A-G]')]
print("Filas donde 'player_name' comienza con una letra entre 'A' y 'G':")
print(filtered_df)
```

```
Filas donde 'player_name' comienza con una letra entre 'A' y 'G':
       Unnamed: 0       player_name team_abbreviation   age  player_height  \
1               1  Gaylon Nickerson               WAS  28.0         190.50
2               2      George Lynch               VAN  26.0         203.20
3               3    George McCloud               LAL  30.0         203.20
4               4      George Zidek               DEN  23.0         213.36
5               5    Gerald Wilkins               ORL  33.0         198.12
...           ...               ...               ...   ...           ...
12651       12651       Cedi Osman               CLE  28.0         200.66
12652       12652   Chance Comanche               POR  27.0         208.28
12653       12653    Charles Bassey               SAS  22.0         205.74
12654       12654      Chima Moneke               SAC  27.0         195.58
12655       12655       Cam Reddish               POR  23.0         200.66

       player_weight                college  country draft_year draft_round  \
1          86.182480  Northwestern Oklahoma      USA       1994           2
2         103.418976         North Carolina      USA       1993           1
3         102.058200          Florida State      USA       1989           1
4         119.748288                   UCLA      USA       1995           1
5         102.058200  Tennessee-Chattanooga      USA       1985           2
...              ...                    ...      ...        ...         ...
12651     104.326160                    NaN   Turkey       2015           2
12652      95.254320                Arizona      USA  Undrafted   Undrafted
12653     104.326160       Western Kentucky  Nigeria       2021           2
12654     101.151016              Cal-Davis  Nigeria  Undrafted   Undrafted
12655      98.429464                   Duke      USA       2019           1

       ...   pts  reb  ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  \
1      ...   3.8  1.3  0.3         8.9     0.030     0.111    0.174   0.497
2      ...   8.3  6.4  1.9        -8.2     0.106     0.185    0.175   0.512
3      ...  10.2  2.8  1.7        -2.7     0.027     0.111    0.206   0.527
4      ...   2.8  1.7  0.3       -14.1     0.102     0.169    0.195   0.500
5      ...  10.6  2.2  2.2        -5.8     0.031     0.064    0.203   0.503
...    ...   ...  ...  ...         ...       ...       ...      ...     ...
12651  ...   8.7  2.3  1.5         7.5     0.017     0.100    0.178   0.579
12652  ...   7.0  3.0  0.0       -68.1     0.080     0.067    0.140   0.518
12653  ...   5.7  5.5  1.4        -2.4     0.140     0.232    0.152   0.658
12654  ...   1.0  1.0  0.5       -57.2     0.250     0.000    0.250   0.347
12655  ...   9.7  2.2  1.4        -9.0     0.017     0.070    0.166   0.561

       ast_pct   season
1        0.043  1996-97
2        0.125  1996-97
3        0.125  1996-97
4        0.064  1996-97
5        0.143  1996-97
...        ...      ...
12651    0.107  2022-23
12652    0.000  2022-23
12653    0.131  2022-23
12654    0.250  2022-23
12655    0.085  2022-23

[4753 rows x 22 columns]
```

El método str.match es extremadamente útil cuando necesitas filtrar datos textuales en función de patrones que deben coincidir al inicio de una cadena. Las expresiones regulares permiten una flexibilidad y potencia considerable en el manejo y análisis de datos textuales.