TELEDUC
Educación a Distancia

# PROCESAMIENTO AVANZADO DE UN Data Frame

>>> Parte 2.

# Recordemos

Funciones para caracterizar un Data Frame

Cómo hacer filtros para extraer información

# ¿Qué aprenderemos?

Herramientas de Pandas que son muy útiles para extraer y visualizar información de un Data Frame

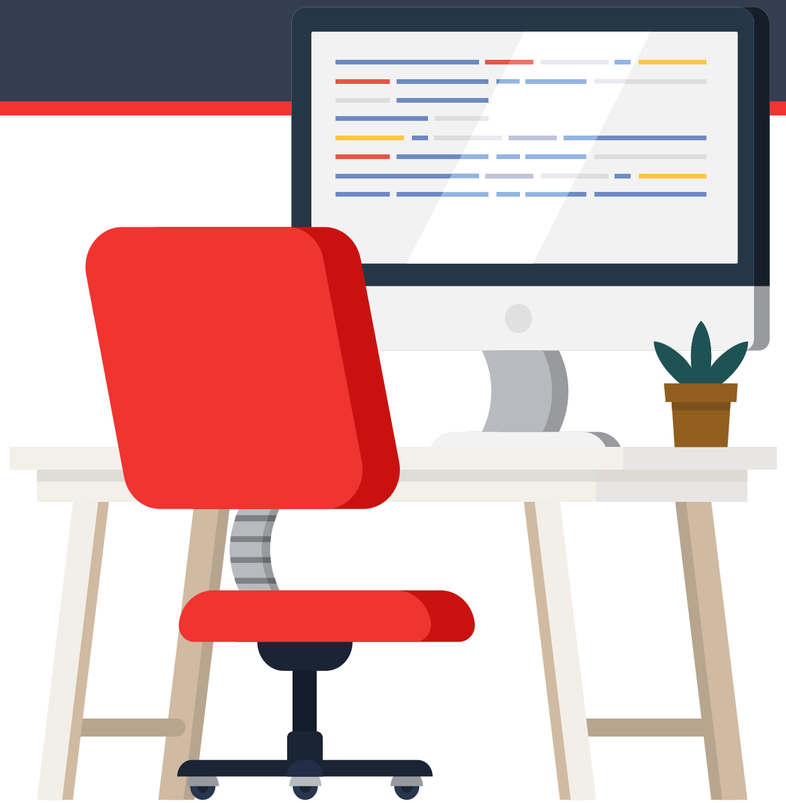
Son funciones particularmente útiles como una base para funciones más avanzadas de Pandas

# >>> Extracción básica de datos de un Data Frame

# Funciones head y tail

| FUNCIÓN | FUNCIÓN |
| --- | --- |
| `head` | `tail` |

Se ocupan para imprimir el contenido de un Data Frame, por sobre el uso del comando `print` directo en el Data Frame.



Recuerda revisar el archivo **m2-ej7.py →**

# Función head

El comando general para un Data Frame de nombre `df` es:

```
df.head()
```

**CÓDIGO**

```
print(df_clientes.head())
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 0 | 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 1 | 11.269.366-8 | Cecilia Paula López Valenzuela | 28-05-62 | A | 8153651 | 2.37 |
| 2 | 9.655.791-3 | Vicente Felipe Robles Muñoz | 02-02-72 | E | 9509104 | 9.91 |
| 3 | 16.644.711-4 | Daniela María Robles Ruiz | 07-07-82 | B | 6065538 | 2.86 |
| 4 | 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |

La función head, esta imprime por defecto las 5 primeras filas del Data Frame.

# Función tail

Recuerda revisar el archivo **m2-ej8.py** →

El comando general para un Data Frame de nombre `df` es:

```
df.tail()
```

**CÓDIGO**

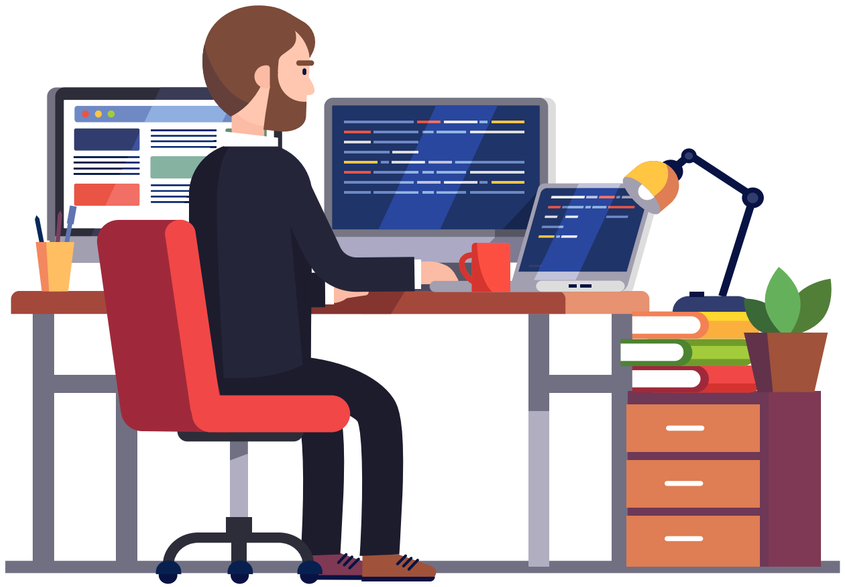
```
print(df_clientes.tail())
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 994 | 21.307.599-10 | Andrea Constanza Castro Robles | 10-07-92 | D | 9277469 | 2.31 |
| 995 | 11.942.539-4 | Isidora Javiera López Rodríguez | 05-01-77 | E | 2949763 | 1.17 |
| 996 | 12.568.934-5 | Paula Daniela Muñoz Quiroga | 06-09-78 | B | 5052388 | 6.01 |
| 997 | 14.407.999-3 | María Valeria Marín Robles | 1995-12-0 | B | 765834 | 8.69 |
| 998 | 20.166.403-3 | Ignacio Bernardo López Valenzuela | 1962-1-0 | E | 7822257 | 7.32 |

La función `tail` imprime por defecto las 5 últimas filas.

# ¿Qué sabemos sobre extracción de filas?



Revisamos la función `loc` para extraer ciertas filas de un Data Frame. Esta extracción podía ser mediante un rango específico, o bien con algún filtro básico.

Además, cómo renombrar el *index*. Todo está relacionado. La función `loc` en realidad trabaja con los *index* de un Data Frame.

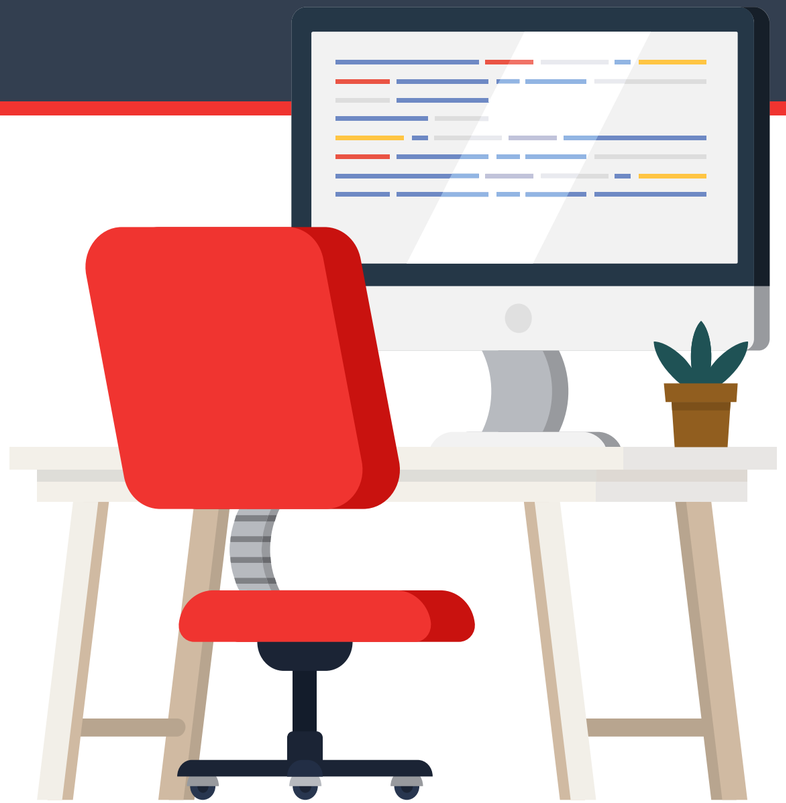
Por lo tanto, veremos en profundidad las funciones `iloc` y `loc`, que permiten extraer una cantidad de filas en base a ciertos criterios.

# Función loc

FUNCIÓN
`loc`

Sirve para extraer un grupo de filas en base a los *index* de un Data Frame. También puede ser en base a una operación lógica.

Los ejemplos que vimos de `loc` se basaban en un Data Frame que tenía como *index* números enteros positivos.



# Función loc

El comando general para un Data Frame de nombre `df` es:

```
df.loc[index, rango de index, o filtro]
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 0 | 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 1 | 11.269.366-8 | Cecilia Paula López Valenzuela | 28-05-62 | A | 8153651 | 2.37 |
| 2 | 9.655.791-3 | Vicente Felipe Robles Muñoz | 02-02-72 | E | 9509104 | 9.91 |
| 3 | 16.644.711-4 | Daniela María Robles Ruiz | 07-07-82 | B | 6065538 | 2.86 |
| 4 | 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |
| 5 | 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6 | 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 7 | 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |
| 8 | 20.749.832-5 | Felipe Andrés Rodríguez Valenzuela | 09-02-73 | E | 4441737 | 8.18 |

Observamos que cada fila tiene asociado un *index* que es un número entero positivo.

Recuerda revisar el archivo **m2-ej9.py** →

# Función loc

El comando general para un Data Frame de nombre `df` es:

```
df.loc[index, rango de index, o filtro]
```

**CÓDIGO**

```
print(df_clientes.loc[5])
```

**RESULTADO**

```
RUT                                  11.170.160-6
NOMBRE               Vicente Vicente Marín Vergara
FECHA_NAC                                11-07-71
TIPO_CLIENTE                                    C
MONTO                                     4056141
PUNTAJE_CREDITICIO                           5.98
Name: 5, dtype: object
```

En este caso imprimimos en consola la fila con *index* 5.

# Función loc

Recuerda revisar el archivo **m2-ej10.py** →

El comando general para un Data Frame de nombre `df` es:

```
df.loc[index, rango de index, o filtro]
```

**CÓDIGO**

```
print(df_clientes.loc[5:9])
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 5 | 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6 | 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 7 | 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |
| 8 | 20.749.832-5 | Felipe Andrés Rodríguez Valenzuela | 09-02-73 | E | 4441737 | 8.18 |
| 9 | 15.910.648-9 | Bernardo Ignacio Quiroga Muñoz | 03-02-68 | B | 8613487 | 3.44 |

En este caso imprimimos en consola las filas con *index* del 5 al 9.

# Función loc

Si cambiamos el *index* a la columna RUT mediante `set_index`, el Data Frame `df_clientes` se ve la siguiente manera:

**RESULTADO**

| RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|
| 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 11.269.366-8 | Cecilia Paula López Valenzuela | 28-05-62 | A | 8153651 | 2.37 |
| 9.655.791-3 | Vicente Felipe Robles Muñoz | 02-02-72 | E | 9509104 | 9.91 |
| 16.644.711-4 | Daniela María Robles Ruiz | 07-07-82 | B | 6065538 | 2.86 |
| 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |

Podemos observar que cada fila tiene asociado un *index* que es un RUT.

# Función loc

Recuerda revisar el archivo **m2-ej11.py →**

El comando general para un Data Frame de nombre `df` es:

```
df.loc[index, rango de index, o filtro]
```

**CÓDIGO**

```
print(df_clientes.loc["11.170.160-6"])
```

**RESULTADO**

```
NOMBRE              Vicente Vicente Marín Vergara
FECHA_NAC                                11-07-71
TIPO_CLIENTE                                    C
MONTO                                     4056141
PUNTAJE_CREDITICIO                           5.98
Name: 11.170.160-6, dtype: object
```

Imprimimos en consola la fila con *index* “11.170.160-6”.

# Función loc

Recuerda revisar el archivo **m2-ej12.py →**

El comando general para un Data Frame de nombre `df` es:

```
df.loc[index, rango de index, o filtro]
```

**CÓDIGO**

```
print(df_clientes.loc["11.170.160-6":"15.910.648-9"])
```

**RESULTADO**

| RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|
| 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |
| 20.749.832-5 | Felipe Andrés Rodríguez Valenzuela | 09-02-73 | E | 4441737 | 8.18 |
| 15.910.648-9 | Bernardo Ignacio Quiroga Muñoz | 03-02-68 | B | 8613487 | 3.44 |

Imprimimos en consola las filas con *index* del “11.170.160-6” al “15.910.648-9”.

# Función loc

Recuerda revisar el archivo **m2-ej12.py** →

El comando general para un Data Frame de nombre `df` es:

```
df.loc[index, rango de index, o filtro]
```

**CÓDIGO**

```
print(df_clientes.loc["11.170.160-6":"15.910.648-9"])
```

**RESULTADO**

| RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|
| 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |
| 20.749.832-5 | Felipe Andrés Rodríguez Valenzuela | 09-02-73 | E | 4441737 | 8.18 |
| 15.910.648-9 | Bernardo Ignacio Quiroga Muñoz | 03-02-68 | B | 8613487 | 3.44 |

Es importante notar que los *index* no necesariamente son únicos.

Entonces si es que se aplica la función `loc` sobre *index* que están en más de una fila, Python arrojará error.

# Función loc

Recuerda revisar el archivo **m2-ej13.py →**

El comando general para `df` es:

```
df.loc[index, rango de index, o filtro]
```

**CÓDIGO**

```
print(df_clientes.loc[df_clientes["TIPO_CLIENTE"]=="C"])
```

**RESULTADO**

| RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|
| 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |
| 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |

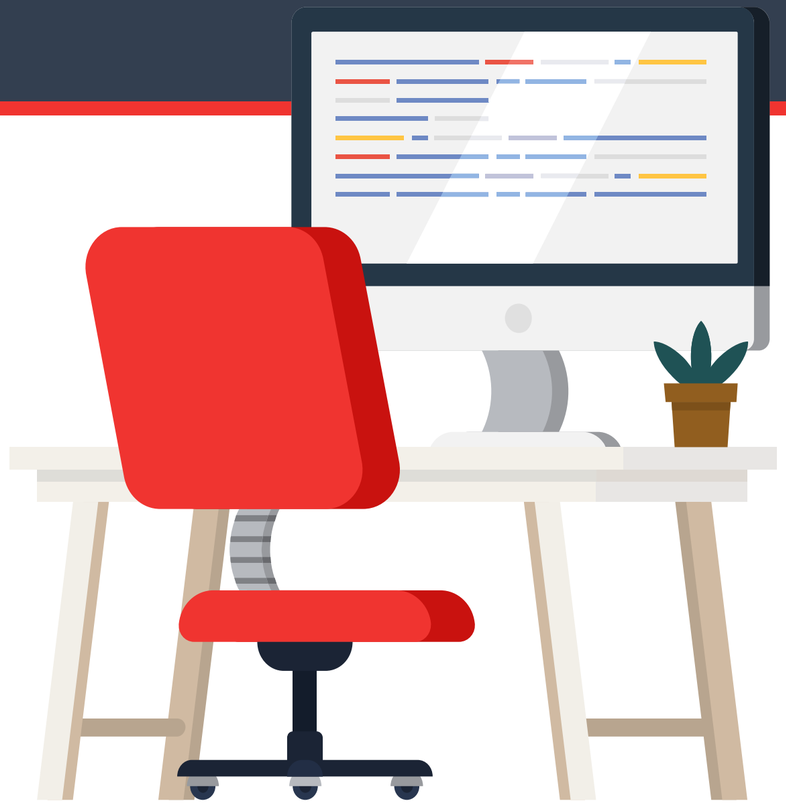
En este ejemplo filtramos los clientes que tienen tipo de cliente “C”.

# Función iloc

FUNCIÓN
`iloc`

¿Cómo podríamos filtrar en base al índice de cada fila (la posición del Data Frame), si es que cambiamos el *index* a un texto como “RUT” por ejemplo?.

Con esta función podemos filtrar por los índices de las filas, aunque el *index* no sea un número entero positivo.



# Función iloc

Si tenemos el Data Frame `df_clientes` con RUT como *index*:

**RESULTADO**

| RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|
| 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 11.269.366-8 | Cecilia Paula López Valenzuela | 28-05-62 | A | 8153651 | 2.37 |
| 9.655.791-3 | Vicente Felipe Robles Muñoz | 02-02-72 | E | 9509104 | 9.91 |
| 16.644.711-4 | Daniela María Robles Ruiz | 07-07-82 | B | 6065538 | 2.86 |
| 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |
| 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |

# Función iloc

Recuerda revisar el archivo **m2-ej14.py →**

El comando general para `df` es:

```
df.iloc[índice de fila o rango de índices de fila]
```

**CÓDIGO**

```
print(df_clientes.iloc[5])
```

**RESULTADO**

```
RUT                                 11.170.160-6
NOMBRE          Vicente Vicente Marín Vergara
FECHA_NAC                               11-07-71
TIPO_CLIENTE                                   C
MONTO                                    4056141
PUNTAJE_CREDITICIO                          5.98
Name: 5, dtype: object
```

En este caso imprimimos en consola la fila de índice 5.

# Función iloc

Recuerda revisar el archivo **m2-ej15.py** →

El comando general para `df` es:

```
df.iloc[índice de fila o rango de índices de fila]
```

**CÓDIGO**

```
print(df_clientes.iloc[5:9])
```

**RESULTADO**

```
          RUT                                NOMBRE FECHA_NAC TIPO_CLIENTE    MONTO  PUNTAJE_CREDITICIO
11.170.160-6       Vicente Vicente Marín Vergara  11-07-71             C  4056141                5.98
 6.172.108-0        Pamela Isabel Castro Vergara  11-05-76             C  3061858                3.09
15.844.106-2     Daniela Pamela Saavedra Vergara  14-06-79             C  5197540                2.26
20.749.832-5  Felipe Andrés Rodríguez Valenzuela  09-02-73             E  4441737                8.18
```

Imprimimos en consola las filas con índice del 5 al 8.

# Función iloc

Recuerda revisar el archivo **m2-ej15.py** →

El comando general para `df` es:

```
df.iloc[índice de fila o rango de índices de fila]
```

**CÓDIGO**

```
print(df_clientes.iloc[5:9])
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 5 | 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6 | 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 7 | 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |
| 8 | 20.749.832-5 | Felipe Andrés Rodríguez Valenzuela | 09-02-73 | E | 4441737 | 8.18 |

Este caso es bastante interesante ya que a diferencia de `loc`, tenemos una fila menos. Esto se debe a que `iloc` sigue una lógica similar al extraer un conjunto de caracteres de un `string`. Es decir, si tenemos el rango `n:m`, en realidad extrae los valores de `n` a `m-1`.

# >>> Manejo de datos faltantes

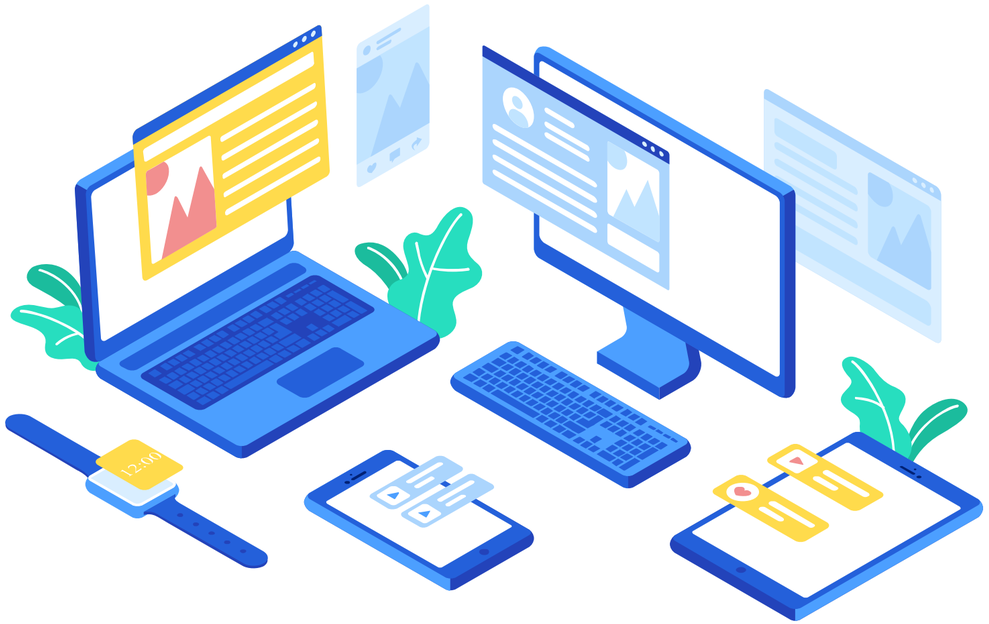
# ¿Cómo manejar los datos faltantes?

1

Muchas veces, la información que manejamos contiene datos faltantes.

2

Por ejemplo, si la información fue cargada por un ser humano en algún sistema de información, es altamente probable que algunos datos falten.



Es vital conocer las herramientas que posee Pandas para manejar estos casos.

# Datos faltantes

Veamos un ejemplo alternativo con `df_clientes` donde hay muchos datos faltantes:

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 0 | 20.137.631-2 | NaN | 1962/3/24 | C | 5463504.0 | 5.95 |
| 1 | 21.276.271-7 | Ignacio Gabriel Quiroga Quiroga | NaN | D | 5767794.0 | 0.31 |
| 2 | 10.175.351-8 | Francisco Alejandro Castro Valenzuela | 1992/6/2 | D | 8360849.0 | NaN |
| 3 | 11.107.732-2 | Paula Javiera Campos Rojas | 1999/8/5 | D | 3541238.0 | 7.35 |
| 4 | 14.564.298-9 | Andrés Bernardo Campos Quiroga | 1961/9/15 | E | 1709146.0 | 7.99 |
| 5 | 11.878.049-8 | Pedro Javier Vergara Castro | NaN | A | 6336266.0 | 8.50 |
| 6 | 6.570.081-4 | Gabriel Ramón Muñoz Robles | 1984/3/2 | D | 1718269.0 | 5.21 |
| 7 | 19.713.565-3 | Ramón Juan Robles Vergara | 1964/7/14 | D | 4552090.0 | 8.15 |
| 8 | 18.222.859-3 | Victoria Ignacia Muñoz Rodríguez | 1971/4/24 | D | 8166157.0 | 9.04 |
| 9 | 4.018.889-8 | Felipe Alejandro Valenzuela Vergara | 1992/12/13 | A | 6801606.0 | 0.65 |
| 10 | 18.580.449-4 | Ramón Andrés González González | 1997/4/1 | NaN | 3133436.0 | 7.46 |

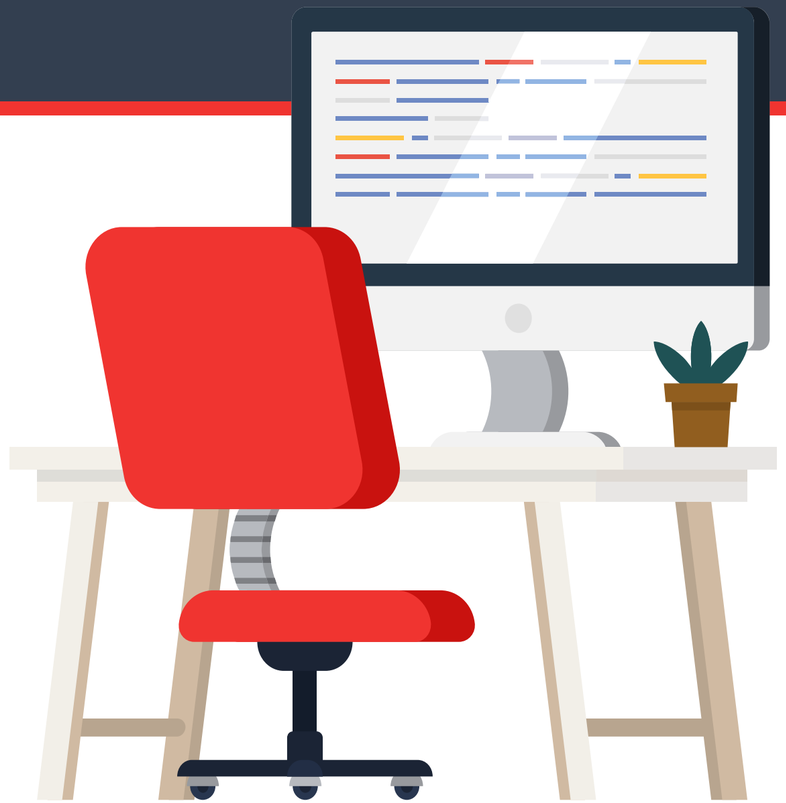
Al cargar este archivo CSV en Pandas, automáticamente los datos faltantes se almacenan como `NaN`.

# Función fillna

FUNCIÓN
`fillna`

Sirve para rellenar los valores `NaN` podemos ocupar la función `fillna`.

Esta función reemplaza los `NaN` por el valor que se entregue como parámetro.



# Función fillna

Recuerda revisar el archivo **m2-ej16.py** →

El comando general para `df` es:

```
df = df.fillna(valor con el que se quiere reemplazar los valores NaN)
```

**CÓDIGO**

```
df_clientes = df_clientes.fillna("SIN_INFO")
print(df_clientes)
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 0 | 20.137.631-2 | SIN_INFO | 1962/3/24 | C | 5.4635e+06 | 5.95 |
| 1 | 21.276.271-7 | Ignacio Gabriel Quiroga Quiroga | SIN_INFO | D | 5.76779e+06 | 0.31 |
| 2 | 10.175.351-8 | Francisco Alejandro Castro Valenzuela | 1992/6/2 | D | 8.36085e+06 | SIN_INFO |
| 3 | 11.107.732-2 | Paula Javiera Campos Rojas | 1999/8/5 | D | 3.54124e+06 | 7.35 |
| 4 | 14.564.298-9 | Andrés Bernardo Campos Quiroga | 1961/9/15 | E | 1.70915e+06 | 7.99 |
| 5 | 11.878.049-8 | Pedro Javier Vergara Castro | SIN_INFO | A | 6.33627e+06 | 8.5 |

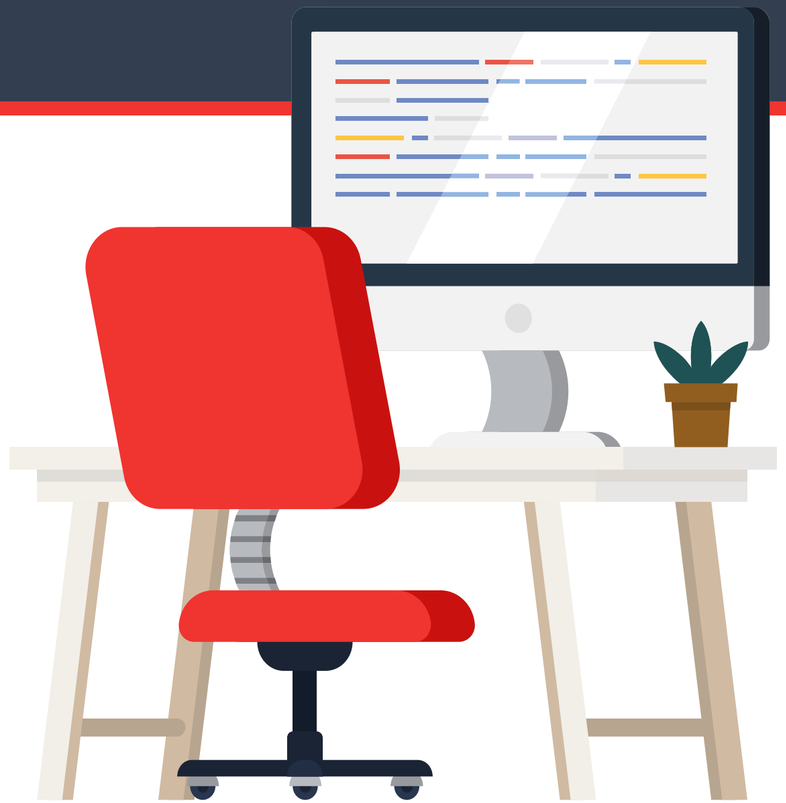
En este caso rellenamos los valores `NaN` por `SIN_INFO`.

# Función append

FUNCIÓN
append

Otro tipo de información faltante en un Data Frame pueden ser filas. Para agregar más filas a un Data Frame, lo podemos hacer mediante la función append.

Es necesario que al agregar estas filas tengan exactamente las mismas columnas que el Data Frame original.



# Función append

Si tenemos el Data Frame `df` y el Data Frame `df2` con las mismas columnas, podemos agregar las filas de `df2` a `df` de la siguiente manera:

```
df = df.append(df2)
```

**RESULTADO**

```
     983   10.667.321-1        Pedro Ramón Valenzuela Rodríguez   11-02-66          C  3912129          2.53
     984   12.395.690-6          María Isidora Robles Rodríguez   01-03-88          B  3672301          7.71
     985   13.478.062-6          María Valeria González Campos   20-10-65          D  1498431          6.84
     986   13.073.200-2          Felipe Javier Rodríguez Rojas   22-05-61          D  6366020          3.24
     987   14.129.210-7             Vicente Juan Castro Castro   10-10-72          D  7110127          5.85
     988  16.986.836-10       Javiera María Vergara Valenzuela   12-01-86          A  3352147          8.92
     989   15.116.681-1         Vicente Pedro Rodríguez Campos   17-04-66          E  1472637          1.39
     990   15.269.626-4                Paula Valeria Ruiz Marín   08-04-99          D  4423588          8.33
     991  16.148.625-10         Blanca Isabel Vergara González   20-07-73          A   389740          9.98
     992    7.184.479-2           Isidora Paula Vergara Robles   08-10-84          E  3174100          1.92
     993   10.302.602-4    Javiera Cecilia Saavedra Valenzuela   07-06-73          E  2068637          6.30
     994  21.307.599-10         Andrea Constanza Castro Robles   10-07-92          D  9277469          2.31
     995   11.942.539-4        Isidora Javiera López Rodríguez   05-01-77          E  2949763          1.17
     996   12.568.934-5           Paula Daniela Muñoz Quiroga   06-09-78          B  5052388          6.01
     997   14.407.999-3           María Valeria Marín Robles  1995-12-0          B   765834          8.69
     998   20.166.403-3     Ignacio Bernardo López Valenzuela   1962-1-0          E  7822257          7.32

[999 rows x 6 columns]
```

En este caso, tenemos el archivo **clientes.csv** y el archivo **clientes2.csv**. Ambos tienen las mismas columnas (las mismas que `df_clientes`), por lo que podemos unirlos.

Inicialmente, el Data Frame que representa a **clientes.csv** se ve de esta manera.

# Función append

Si tenemos el Data Frame `df` y el Data Frame `df2` con las mismas columnas, podemos agregar las filas de `df2` a `df` de la siguiente manera:

```
df = df.append(df2)
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 0 | 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 1 | 11.269.366-8 | Cecilia Paula López Valenzuela | 28-05-62 | A | 8153651 | 2.37 |
| 2 | 9.655.791-3 | Vicente Felipe Robles Muñoz | 02-02-72 | E | 9509104 | 9.91 |
| 3 | 16.644.711-4 | Daniela María Robles Ruiz | 07-07-82 | B | 6065538 | 2.86 |
| 4 | 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |
| 5 | 11.170.160-6 | Vicente Vicente Marín Vergara | 11-07-71 | C | 4056141 | 5.98 |
| 6 | 6.172.108-0 | Pamela Isabel Castro Vergara | 11-05-76 | C | 3061858 | 3.09 |
| 7 | 15.844.106-2 | Daniela Pamela Saavedra Vergara | 14-06-79 | C | 5197540 | 2.26 |
| 8 | 20.749.832-5 | Felipe Andrés Rodríguez Valenzuela | 09-02-73 | E | 4441737 | 8.18 |
| 9 | 15.910.648-9 | Bernardo Ignacio Quiroga Muñoz | 03-02-68 | B | 8613487 | 3.44 |
| 10 | 9.487.844-9 | Javier Juan Castro Campos | 1985/3/0 | B | 8030691 | 2.70 |
| 11 | 7.271.618-0 | Isidora Andrea Rodríguez Valenzuela | 29-09-76 | B | 5832527 | 8.08 |
| 12 | 13.674.785-2 | Javier Juan Robles Robles | 23-07-73 | A | 469341 | 2.81 |
| 13 | 17.452.036-6 | Ramón Vicente Vergara Robles | 16-07-97 | E | 8845556 | 5.90 |
| 14 | 20.481.183-10 | Andrés Francisco López López | 27-07-77 | E | 6285039 | 4.68 |
| 15 | 6.355.671-0 | Rodrigo Andrés Valenzuela Valenzuela | 01-10-70 | B | 668186 | 9.84 |
| 16 | 5.814.759-5 | Andrés Felipe Quiroga Robles | 17-07-90 | E | 7790975 | 0.91 |
| 17 | 15.452.563-10 | Gabriel Vicente Castro Robles | 11-06-85 | B | 3156647 | 7.26 |
| 18 | 8.471.223-1 | Felipe Andrés Robles Rodríguez | 06-05-70 | E | 7976926 | 9.06 |

Al cargar el archivo **clientes2.csv** en un Data Frame de nombre `df_clientes2`, se ve de esta manera (tiene solo 18 filas).

# Función append

Recuerda revisar el archivo **m2-ej17.py →**

**CÓDIGO**

```
df_clientes = df_clientes.append(df_clientes2)
print(df_clientes)
```

**RESULTADO**

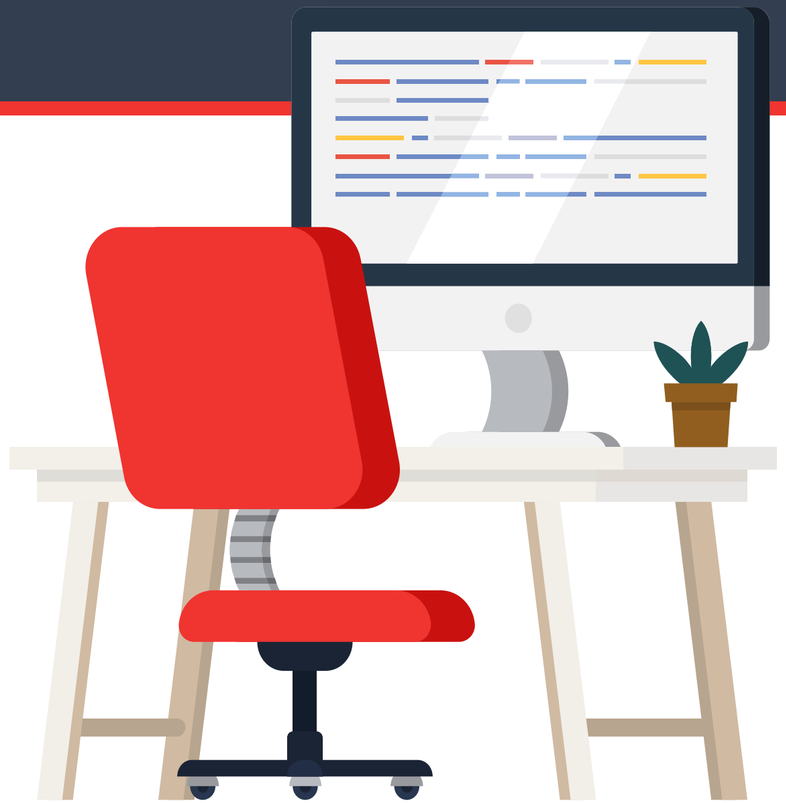
```
     994  21.307.599-10        Andrea Constanza Castro Robles   10-07-92          D  9277469               2.31
     995   11.942.539-4       Isidora Javiera López Rodríguez   05-01-77          E  2949763               1.17
     996   12.568.934-5           Paula Daniela Muñoz Quiroga   06-09-78          B  5052388               6.01
     997   14.407.999-3            María Valeria Marín Robles  1995-12-0          B   765834               8.69
     998   20.166.403-3     Ignacio Bernardo López Valenzuela    1962-1-0          E  7822257               7.32
     0     21.930.631-4              Isabel Blanca Marín Díaz   13-04-97          C  5407949               1.17
     1     11.269.366-8        Cecilia Paula López Valenzuela   28-05-62          A  8153651               2.37
     2      9.655.791-3           Vicente Felipe Robles Muñoz   02-02-72          E  9509104               9.91
     3     16.644.711-4             Daniela María Robles Ruiz   07-07-82          B  6065538               2.86
     4     17.054.286-6    Isabel Javiera Valenzuela Saavedra   05-07-71          C  8024077               0.56
     5     11.170.160-6         Vicente Vicente Marín Vergara   11-07-71          C  4056141               5.98
     6      6.172.108-0          Pamela Isabel Castro Vergara   11-05-76          C  3061858               3.09
     7     15.844.106-2       Daniela Pamela Saavedra Vergara   14-06-79          C  5197540               2.26
     8     20.749.832-5    Felipe Andrés Rodríguez Valenzuela   09-02-73          E  4441737               8.18
     9     15.910.648-9        Bernardo Ignacio Quiroga Muñoz   03-02-68          B  8613487               3.44
     10     9.487.844-9             Javier Juan Castro Campos  1985/3/0          B  8030691               2.70
     11     7.271.618-0   Isidora Andrea Rodríguez Valenzuela   29-09-76          B  5832527               8.08
     12    13.674.785-2             Javier Juan Robles Robles   23-07-73          A   469341               2.81
     13    17.452.036-6          Ramón Vicente Vergara Robles   16-07-97          E  8845556               5.90
     14   20.481.183-10          Andrés Francisco López López   27-07-77          E  6285039               4.68
     15     6.355.671-0  Rodrigo Andrés Valenzuela Valenzuela   01-10-70          B   668186               9.84
     16     5.814.759-5          Andrés Felipe Quiroga Robles   17-07-90          E  7790975               0.91
     17   15.452.563-10         Gabriel Vicente Castro Robles   11-06-85          B  3156647               7.26
     18     8.471.223-1        Felipe Andrés Robles Rodríguez   06-05-70          E  7976926               9.06

    [1018 rows x 6 columns]
```

Notemos que si hacemos append podría ocurrir tener *index* repetidos.

# Función drop

FUNCIÓN
`drop`

Sirve para eliminar filas.



# Eliminar filas de un Data Frame en Python

Recuerda revisar el archivo **m2-ej18.py** →

De forma general, y para un Data Frame de nombre `df`:

```
df = df.drop(df.iloc[filas o rango de filas].index)
```

**CÓDIGO**

```
df_clientes = df_clientes.drop(df_clientes.iloc[0:10].index)
print(df_clientes)
```

**RESULTADO**

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 0 | 21.930.631-4 | Isabel Blanca Marín Díaz | 13-04-97 | C | 5407949 | 1.17 |
| 1 | 11.269.366-8 | Cecilia Paula López Valenzuela | 28-05-62 | A | 8153651 | 2.37 |
| 2 | 9.655.791-3 | Vicente Felipe Robles Muñoz | 02-02-72 | E | 9509104 | 9.91 |
| 3 | 16.644.711-4 | Daniela María Robles Ruiz | 07-07-82 | B | 6065538 | 2.86 |
| 4 | 17.054.286-6 | Isabel Javiera Valenzuela Saavedra | 05-07-71 | C | 8024077 | 0.56 |

| | RUT | NOMBRE | FECHA_NAC | TIPO_CLIENTE | MONTO | PUNTAJE_CREDITICIO |
|---|---|---|---|---|---|---|
| 10 | 9.487.844-9 | Javier Juan Castro Campos | 1985-3-0 | B | 8030691 | 2.70 |
| 11 | 7.271.618-0 | Isidora Andrea Rodríguez Valenzuela | 29-09-76 | B | 5832527 | 8.08 |
| 12 | 13.674.785-2 | Javier Juan Robles Robles | 23-07-73 | A | 469341 | 2.81 |
| 13 | 17.452.036-6 | Ramón Vicente Vergara Robles | 16-07-97 | E | 8845556 | 5.90 |
| 14 | 20.481.183-10 | Andrés Francisco López López | 27-07-77 | E | 6285039 | 4.68 |

Notemos que para eliminar filas, especificamos el intervalo de índices de las filas que queremos eliminar dentro del `iloc`. Todo lo otro se debe mantenter igual.

# Conclusiones

- En esta clase aprendimos cómo usar correctamente la función loc e iloc, y la diferencia entre usar *index* e índices de un Data Frame.
- Además, vimos cómo poder agregar y eliminar filas de un Data Frame, lo que es muy útil en contextos donde la información con la que se trabaja es dinámica.
- Finalmente, vimos también qué hacer con datos faltantes, lo que en ciertos contextos puede ser sumamente relevante.

# >>> Cierre

Has finalizado la revisión de los contenidos de esta clase.

A continuación, te invitamos a realizar las actividades y a revisar los recursos del módulo que encontrarás en plataforma.