TELEDUC
Educación a Distancia

PROCESAMIENTO AVANZADO DE UN DATA FRAME

>>> Parte 1.

>>> Introducción a la caracterización de un Data Frame

Recordemos

Las operaciones básicas de un Data Frame:

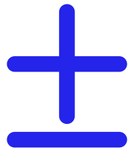
- Crear Data Frames
- Ver, crear, editar y eliminar columnas
- Filtro básico de las filas de un Data Frame
- Guardar un Data Frame en un archivo CSV

¿Qué no sabemos sobre operaciones básicas de un Data Frame?

1

Reconocer las características básicas de un Data Frame o de una serie en particular dentro del Data Frame.

2



Cómo agregar o eliminar filas a un Data Frame.

3

Cómo puedo manejar datos faltantes en un Data Frame.

¿Cómo aprenderemos?

Practicando a través del archivo **clientes.csv**, como la base de todos los ejemplos

Al cargarlo en un Data Frame, se ve de la siguiente manera:

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

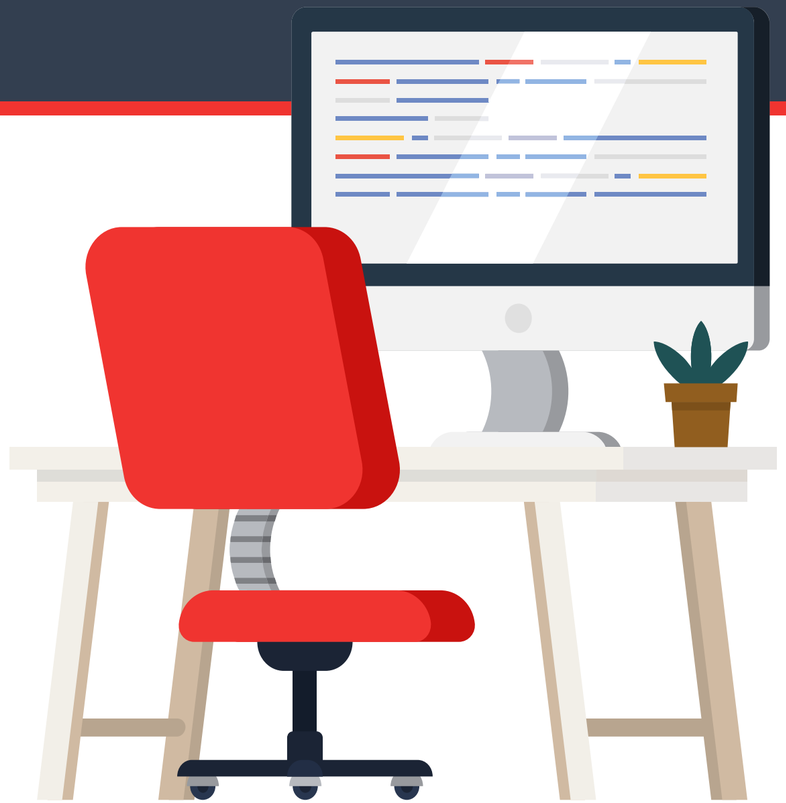
>>> Función dtypes

¿Cómo caracterizar a un Data Frame de mejor manera?

FUNCIÓN
`dtypes`

Lista las columnas de un Data Frame.

Puntualiza el tipo de dato de cada una de ellas.



Función dtypes

Recuerda revisar el archivo **m2-ej1.py** →

```
df.dtypes
```

CÓDIGO

```
print(df_clientes.dtypes)
```

RESULTADO

```
RUT                   object
NOMBRE                object
FECHA_NAC             object
TIPO_CLIENTE          object
MONTO                  int64
PUNTAJE_CREDITICIO   float64
dtype: object
```

Así se visualizan los nombres de todas las columnas y sus tipos de datos.

Ahora surgen dos interrogantes

¿Cómo podríamos cambiar el nombre a una columna?

¿Cómo podríamos cambiar el tipo de dato de columna?

>>> Renombrar una columna: Función rename

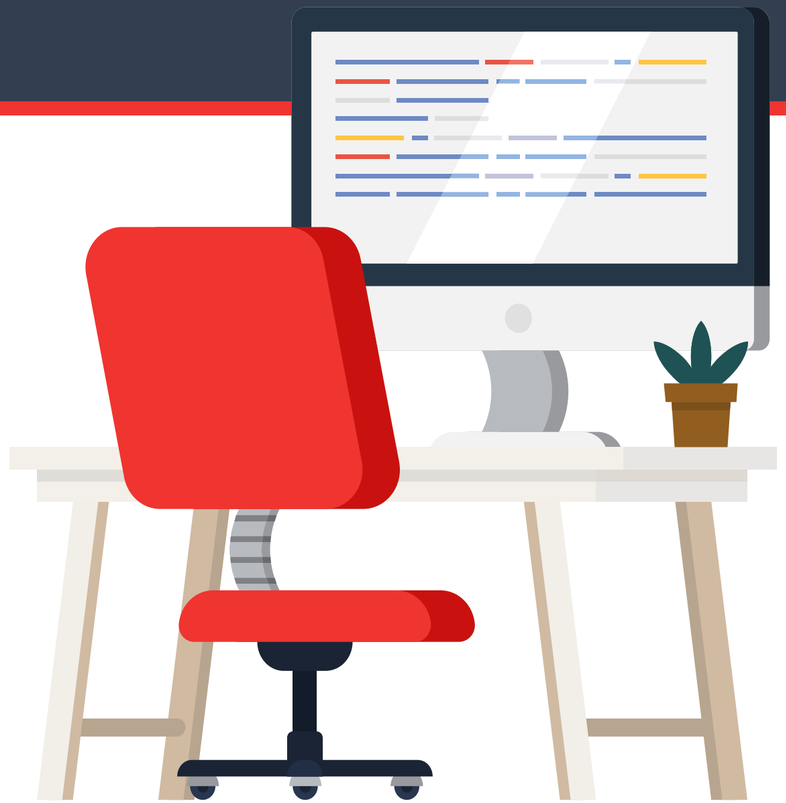
¿Cómo renombrar una columna?

FUNCIÓN

`rename`

Permite cambiar el nombre a una columna.

Recibe como parámetro el nombre de la columna antigua y el nombre al que se quiere cambiar.



Función rename

Recuerda revisar el archivo **m2-ej2.py →**

Si tenemos un Data Frame de nombre `df`, podemos cambiar el nombre de las columnas:

```
df.rename(columns={"nombre_antiguo_columna":"nombre_nuevo_columna"})
```

CÓDIGO

```
df_clientes = df_clientes.rename(columns={"FECHA_NAC":"FECHA_NACIMIENTO"})

print(df_clientes.dtypes)
```

RESULTADO

```
RUT                  object
NOMBRE               object
FECHA_NACIMIENTO     object
TIPO_CLIENTE         object
MONTO                 int64
PUNTAJE_CREDITICIO  float64
dtype: object
```

Es importante mantener los caracteres que aparecen en el parámetro “`columns`” de la función `rename`.

Siempre deben escribir el `{"nombre_antiguo_columna": "nombre_nuevo_columna"}`

Recuerda revisar el archivo **m2-ej2.py →**

Función rename

Si tenemos un Data Frame de nombre df, podemos cambiar el nombre de las columnas:

```
df.rename(columns={"nombre_antiguo_columna":"nombre_nuevo_columna"})
```

CÓDIGO

```
df_clientes = df_clientes.rename(columns={"FECHA_NAC":"FECHA_NACIMIENTO"})

print(df_clientes.dtypes)
```

RESULTADO

```
RUT                  object
NOMBRE               object
FECHA_NACIMIENTO     object
TIPO_CLIENTE         object
MONTO                 int64
PUNTAJE_CREDITICIO  float64
dtype: object
```

Al ejecutar la función rename, el resultado se asigna nuevamente a df clientes, porque es necesario “guardarlo” en el Data Frame original o en alguna columna en particular.

>>> Columnas y tipos de datos

Columnas y tipos de datos

En el caso `df_clientes` se observa que cada columna tiene un tipo de dato:

`object`	RUT, NOMBRE, FECHA_NAC y TIPO_CLIENTE
`int64`	MONTO
`float64`	PUNTAJE_CREDITICIO

RESULTADO

```
RUT                  object
NOMBRE               object
FECHA_NAC            object
TIPO_CLIENTE         object
MONTO                 int64
PUNTAJE_CREDITICIO  float64
dtype: object
```

¿Qué tipos de datos puede tomar cada columna?

Tipo de dato	Resultado	Descripción
`object`	Es un string	Un texto
`int64`	Es un int	Un entero
`float64`	Es un float	Un decimal
`bool`	True o False	Valor binario, base de operaciones lógicas
`datetime64`	Valores fecha y tiempo	Día, hora, mes, etc.
`timedelta[ns]`	Diferencia entre valores de fecha y tiempo	En segundos
`category`	Lista finita	Valores de texto

Los tipos de datos más comúnmente usados son los 5 primeros.

>>> Cambiar tipo a una columna: Función astype

¿Cómo cambiar el tipo de dato de una columna?

En la siguiente situación:

Al cargar un archivo CSV, puede que los datos no se carguen correctamente. Por ejemplo, podría ocurrir que una columna que tiene decimales se cargue como texto. Es decir, la columna es de tipo `object`, y nos gustaría que pasará a ser de tipo `float64`.

`object`

`float64`

¿Cómo cambiar el tipo de dato de una columna?

Cargamos el Data Frame `df_clientes`, y la columna **PUNTAJE_CREDITICIO** se cargó como object, al aplicar la función `dtypes`.

CÓDIGO

```
print(df_clientes.dtypes)
```

RESULTADO

```
RUT                  object
NOMBRE               object
FECHA_NACIMIENTO     object
TIPO_CLIENTE         object
MONTO                 int64
PUNTAJE_CREDITICIO   object
dtype: object
```

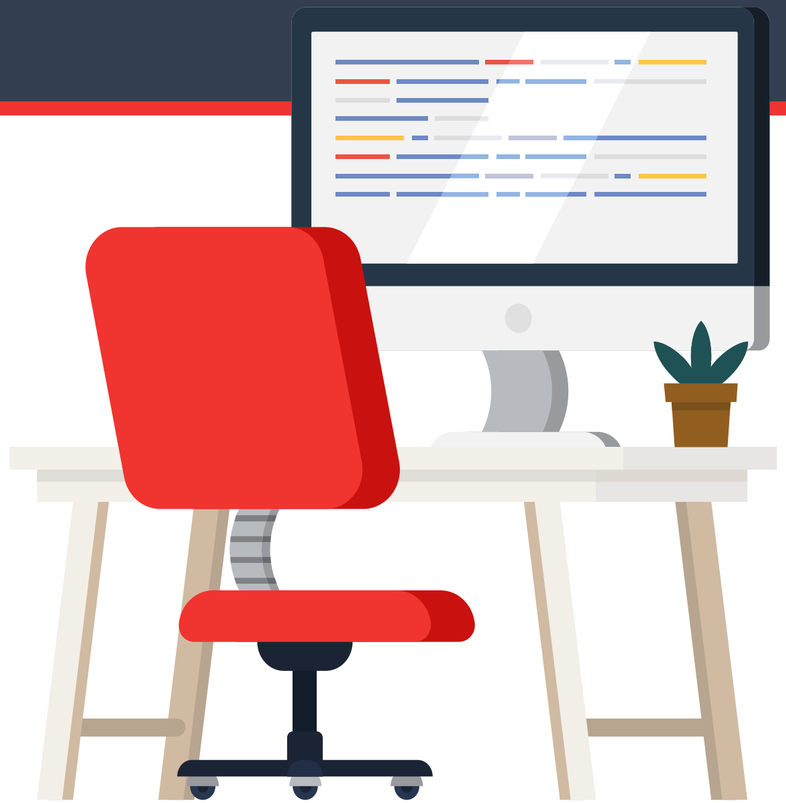
¿Pero cómo podríamos cambiar “PUNTAJE_CREDITICIO” a `float64`?

¿Para qué nos sirve la función astype?

FUNCIÓN
`astype`

Permite cambiar el tipo de dato de una columna.

No solo puede convertir el tipo de dato de una columna a *float64*, sino que a cualquier tipo de dato.



Función astype

Recuerda revisar el archivo **m2-ej3.py →**

Si tenemos un Data Frame de nombre `df` creamos el comando general de la siguiente manera:

```
df= df[nombre columna].astype(tipo de dato al que se quiere cambiar el tipo de la columna)
```

CÓDIGO

```
print(df_clientes.dtypes)
df_clientes["PUNTAJE_CREDITICIO"] = df_clientes["PUNTAJE_CREDITICIO"].astype("float64")

print(df_clientes.dtypes)
```

RESULTADO

```
RUT                 object
NOMBRE              object
FECHA_NAC           object
TIPO_CLIENTE        object
MONTO                int64
PUNTAJE_CREDITICIO  object
dtype: object

RUT                 object
NOMBRE              object
FECHA_NAC           object
TIPO_CLIENTE        object
MONTO                int64
PUNTAJE_CREDITICIO  float64
dtype: object
```

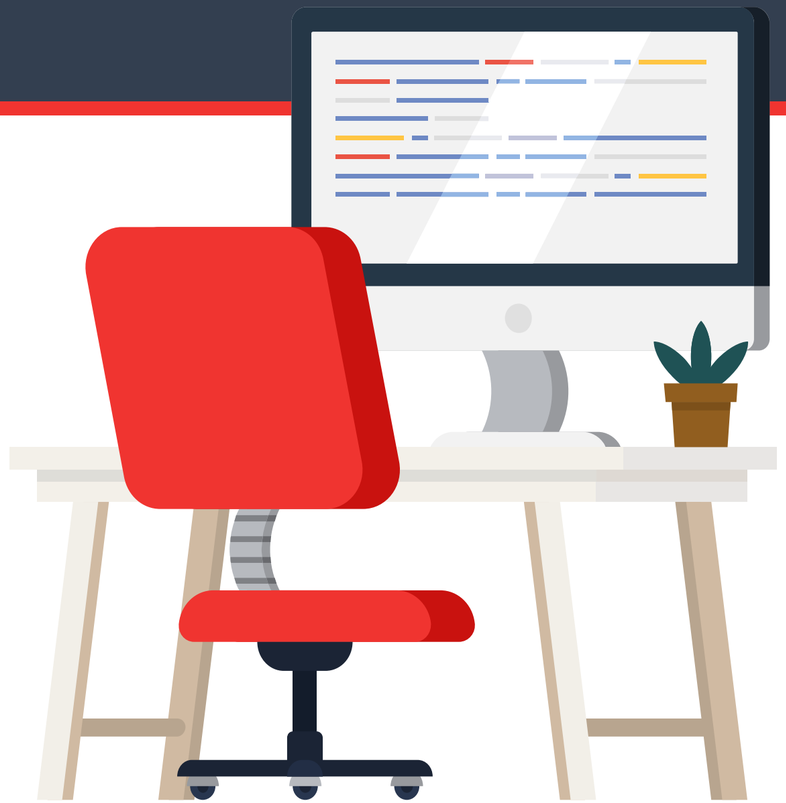
Al aplicar la función `astype` en la columna "**PUNTAJE CREDITICIO**", podemos cambiar el tipo de datos de "object" a "float64". No obstante, debemos volver asignarlo en la misma columna del Data Frame original. Si no lo hiciéramos, y solo ejecutáramos lo que está a la derecha del "=" en la primera línea del código anterior, entonces esto no se "guarda" en ningún lado y se pierde.

>>> Función shape

¿Para qué se usa la función shape?

FUNCIÓN
shape

Esta función sirve para conocer la cantidad de filas y columnas de un Data Frame.



Recuerda revisar el archivo **m2-ej4.py** →

Función shape

De forma general, para un Data Frame df:

```
df.shape
```

CÓDIGO

```
print(df_clientes.shape)
```

RESULTADO

```
(999, 6)
```

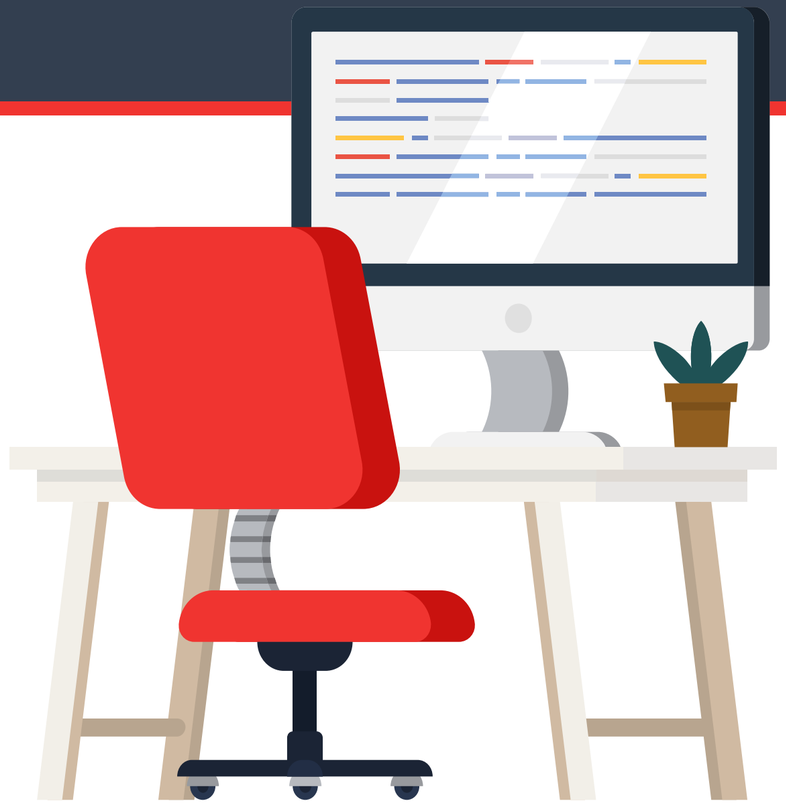
La función `shape` indica que el Data Frame `df_clientes` tiene 999 filas y 6 columnas.

>>> Función size

¿Para qué nos sirve la función size?

FUNCIÓN
`size`

Sirve para conocer la cantidad total de datos del Data Frame.



Recuerda revisar el archivo **m2-ej5.py** →

Función size

De forma general, para un Data Frame df:

```
df.size
```

CÓDIGO

```
print(df_clientes.size)
```

RESULTADO

```
(5994)
```

La función `size` arroja que `df_clientes` tiene un total de 5994 datos, que es la multiplicación de filas y columnas.

>>> *Index* de un Data Frame

¿Qué es *Index*?

Cuando cargamos un Data Frame y lo imprimimos en consola, podemos observar lo siguiente:

RESULTADO

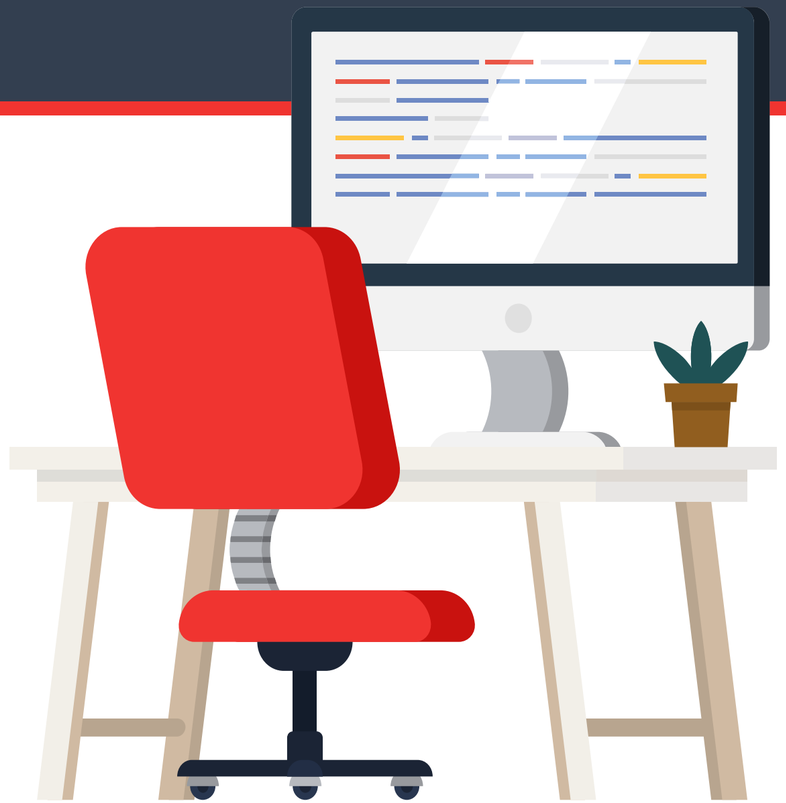
	RUT	NOMBRE	FECHA_NAC	TIPO_CLIENTE	MONTO	PUNTAJE_CREDITICIO
0	21.930.631-4	Isabel Blanca Marín Díaz	13-04-97	C	5407949	1.17
1	11.269.366-8	Cecilia Paula López Valenzuela	28-05-62	A	8153651	2.37
2	9.655.791-3	Vicente Felipe Robles Muñoz	02-02-72	E	9509104	9.91
3	16.644.711-4	Daniela María Robles Ruiz	07-07-82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

Cada fila tiene un número asignado. **Este número se denomina *index*, y es un identificador para cada fila.**

¿Para qué se utiliza la función set_index?

FUNCIÓN
`set_index`

El index es un identificador de cada fila. Para definir una columna del Data Frame como el *index*, podemos ocupar la función `set_index`.



Función set_index

Recuerda revisar el archivo **m2-ej6.py** →

De forma general, para setear una columna como el *index* de un Data Frame de nombre `df`:

```
df = df.set_index(nombre columna)
```

CÓDIGO

```
df_clientes = df_clientes.set_index("RUT")
print(df_clientes)
```

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

La columna con números enteros positivos desapareció, y en su lugar está la columna RUT.
Esta última también desapareció de la lista de columnas original.

Como aclaración, el cambio de la columna RUT al *index* del Data Frame es solo válido para este ejemplo.

Conclusiones

- En esta clase aprendimos características básicas de un Data Frame. En particular, cómo caracterizar una columna o serie, y cómo editar estar características, es decir, cambiar su nombre y tipo.
- Además, aprendimos cómo caracterizar a un Data Frame, mediante el número de filas, columnas y la cantidad total de datos.

Conclusiones

- Finalmente, aprendimos que el *index* de un Data Frame es un identificador por fila, y cómo poder asignarlo de alguna de las columnas originales de un Data Frame.
- Todo esto será fundamental para el trabajo posterior. Especialmente, cuando veamos interacciones y operaciones entre Data Frames. Aquí, es muy importante saber reconocer características básicas de un Data Frame para poder hacer estas acciones.

>>> Cierre

Has finalizado la revisión de los contenidos que corresponden a esta clase.

A continuación, te invitamos a estudiar la siguiente clase del módulo.