

# Estadística Univariada y control de flujo \_



# Manejo de Archivos

# Pandas: Lectura de Archivos

- Pandas es una librería orientada a la manipulación de estructuras de datos, ingesta de datos en múltiples formatos.
- La convención sugiere importar pandas de la siguiente manera:

```
import pandas as pd
```

- Existen dos estructuras de datos básicas de pandas: DataFrame y Series

# Pandas II: DataFrame

- La representación de una matriz de datos, ordenados por filas y columnas.
- Presenta una serie de métodos que operan a nivel de matriz.
- Cada fila y columna se considera una serie

Altura: Columna

Índice	Altura	Peso
1	1.67	67
2	1.78	80
3	1.54	56

1: Características Asociadas al registro

# Pandas III: Series

- Una serie se compone de valores e índices
- Una serie también presentará métodos disponibles gracias a Pandas.

índice	Altura
0	1.67
1	1.78
2	1.54

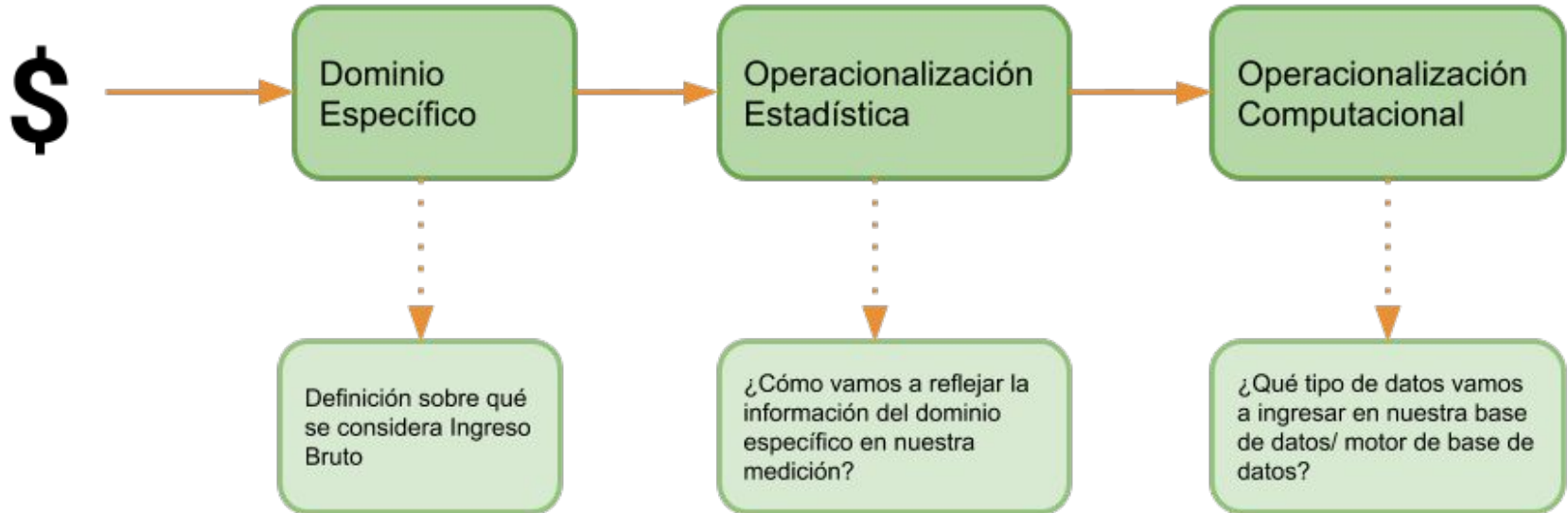
# Estadística Univariada

# Conceptualización y Operacionalización

Existen tres dimensiones a considerar en los **datos**:

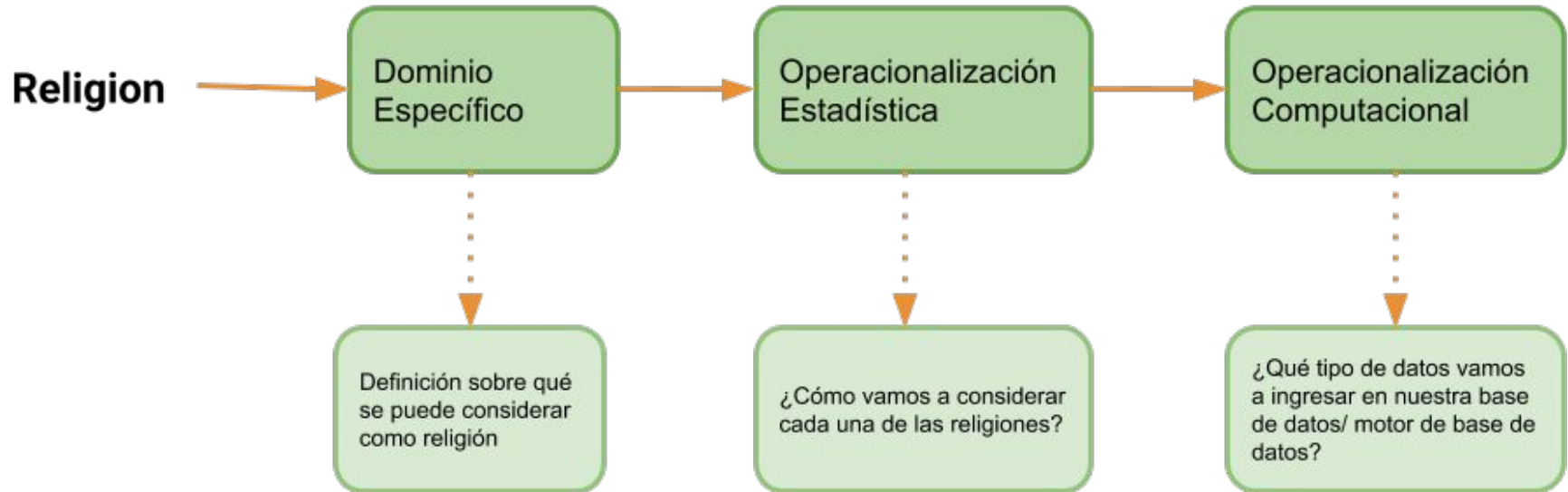
- Estructura específica de los datos ingresados.
- Desde el conocimiento específico de la industria.
- Desde la estadística.

# Ejemplo de Operacionalización I





# Ejemplo de Operacionalización II



# Conceptualización y Operacionalización

Desde la estadística existen dos grandes tipos de **variables**:

- Variables Cualitativas
- Variables

Cuantitativas

# Momentos Estadísticos

Cuatro dimensiones a tomar en cuenta en el **análisis descriptivo de los datos**:

- Medidas de Tendencia Central
- Medidas de Dispersión
- Medidas de Sesgo
- Medidas

de

Kurtosis

# Medida de Tendencia Central: Media

- **Objetivo:** Generar una cifra que represente de mejor manera la muestra que estudiamos.

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$$

# Medida de Tendencia Central: Media

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$$

```
# Python nativo  
sum(altura) / len(altura)  
Resultado: 1.71
```

```
# Pandas  
df['altura'].mean()  
Resultado: 1.71
```

```
# Numpy  
import numpy as np  
np.mean(df['altura'])  
Resultado: 1.71
```

# Medida de Tendencia Central: Moda

- **Objetivo:** Identificar cuál es el valor observado con una mayor frecuencia en la muestra.

La implementación de la media en Python Nativo es más compleja.

Haremos uso de scipy:

```
import scipy.stats as stats
stats.mode(altura)
ModeResult(mode=array[1.58]), count=array([3]))
```

# Medida de Tendencia Central: Mediana

- **Objetivo:** Identificar el punto equidistante en una variable.

La implementación es fácil utilizando Numpy:

```
np.median(df['altura'])
```

# Medida de Dispersión: Varianza

- **Objetivo:** reportar qué tan dispersos están las observaciones respecto al punto de origen. La implementación en Python nativo es un poco más compleja.

Afortunadamente, tanto Pandas como Numpy presentan implementaciones más fáciles.

$$\sigma_2 = \frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N - 1}$$

```
# pandas
df['altura'].var()

# numpy
np.var(df['altura'])
```



# Medida de Dispersión: Desviación Estándar

- La desviación estándar está fuertemente asociada con la varianza.
- Para obtenerla, es necesario obtener la raíz cuadrada de ésta última.

$$\text{Desv.Est} = \sqrt{\frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N - 1}}$$

```
# pandas
df['altura'].std()
# numpy
np.std(df['altura'])
```

# Control de Flujo en Pandas

# A nivel de serie

df['Altura']

índice	Altura
0	1.64
1	1.23
2	1.87
3	1.90
4	2.01
5	1.45
6	1.67
7	1.93
8	1.72
9	1.64
10	1.67

El iterador entrega el elemento,  
no su posición

Podemos aplicar funciones a  
nivel de **serie**

```
for i in df['Altura']:
    print(i, type(i))
```

```
# 1.64, float
# ...
```

```
for i in df['Altura'].index:
    print(i)
```

```
# 0
# ...
```

```
for i in df['Altura'] * 2:
    print(i)
```

```
# 3.28
# ...
```

# A nivel de DataFrame: iteritems

Dirección del recorrido `df.iteritems()`



índice	Altura	Peso	Nombre
0	1.64	68	Javiera
1	1.23	43	José
2	1.87	90	Tomás
3	1.90	95	María
4	2.01	100	José
5	1.45	50	Magdalena
6	1.67	67	Trinidad
7	1.93	102	Gonzalo
8	1.72	76	David
9	1.64	68	Javier
10	1.67	70	Alicia

Inspección de colname


```
for colname, serie in df.iteritems():  
    print(colname)  
  
# Altura  
# Peso  
# Nombre
```

Inspección de serie

```
for colname, serie in df.iteritems():  
    print(serie.dtype)  
  
# float  
# int  
# str
```

# A nivel de DataFrame: iterrows

Dirección del recorrido df.iterrows()



índice	Altura	Peso	Nombre
0	1.64	68	Javiera
1	1.23	43	José
2	1.87	90	Tomás
3	1.90	95	María
4	2.01	100	José
5	1.45	50	Magdalena
6	1.67	67	Trinidad
7	1.93	102	Gonzalo
8	1.72	76	David
9	1.64	68	Javier
10	1.67	70	Alicia

## Inspección de rowname

```
for rowname, serie in df.iteritems():  
    print(rowname)  
  
# 0  
# 1 ...  
# 10
```

## Inspección de serie

```
for colname, serie in df.iteritems():  
    print(serie)  
  
# Altura: 1.64  
# Peso: 68  
# Nombre: Javiera  
# Name: 0, dtype: object
```

# A nivel de DataFrame: subset booleano

índice	Altura	Peso	Nombre
0	1.64	68	Javiera
1	1.23	43	José
2	1.87	90	Tomás
3	1.90	95	María
4	2.01	100	José
5	1.45	50	Magdalena
6	1.67	67	Trinidad
7	1.93	102	Gonzalo
8	1.72	76	David
9	1.64	68	Javier
10	1.67	70	Alicia

Selección en base a atributos

```
df[df['Peso'] < 70]
```

índice	Altura	Peso	Nombre
0	1.64	68	Javiera
1	1.23	43	José
5	1.45	50	Magdalena
6	1.67	67	Trinidad
9	1.64	68	Javier

# A nivel de DataFrame: iterrows condicionales

índice	Altura	Peso	Nombre
0	1.64	68	Javiera
1	1.23	43	José
2	1.87	90	Tomás
3	1.90	95	María
4	2.01	100	José
5	1.45	50	Magdalena
6	1.67	67	Trinidad
7	1.93	102	Gonzalo
8	1.72	76	David
9	1.64	68	Javier
10	1.67	70	Alicia

Selección en base a atributos

```
mean_male = 0
for index, rowserie in df.iterrows():
    if rowserie['Sexo'] == 'Hombre':
        mean_male += rowserie['Peso']

print(mean_male / 6)
```

# A nivel de DataFrame: iterrows condicionales

índice	Altura	Peso	Nombre
0	1.64	68	Javiera
5	1.45	50	Magdalena
6	1.67	67	Trinidad
10	1.67	70	Alicia

Selección en base a atributos

```
mean_female_below_height = 0
for index, serie in df.iterrows():
    if (serie['Sexo'] == 'Mujer') and (serie['Altura'] < df['Altura'].mean()):
        mean_female_below_height += serie['Peso']
print(mean_female_below_height / 6)
```



**{desafío}**  
**latam\_**

*Academia de  
talentos digitales*

[www.desafiolatam.com](http://www.desafiolatam.com)