

Seaborn Con Python

¿Qué es Seaborn y por qué usarlo?

- Seaborn es una biblioteca de visualización de datos basada en Matplotlib que proporciona una interfaz de alto nivel para crear gráficos estadísticos atractivos y fáciles de interpretar y se integra estrechamente con Pandas.
- Seaborn te ayuda explorar y comprender los datos. Sus funciones de trazado operan en marcos de datos y matrices que contienen conjuntos de datos completos y realizan internamente el mapeo semántico y la agregación estadística necesarios para producir gráficos informativos.
- Su API declarativa orientada a conjuntos de datos le permite centrarse en lo que significan los diferentes elementos de sus gráficos en lugar de en los detalles de cómo dibujarlos.

Ventajas de Seaborn:

- Sintaxis sencilla y concisa.
- Mejora la estética de Matplotlib por defecto.
- Manejo eficiente de DataFrames de Pandas.
- Gráficos avanzados con menos código.
- Funcionalidades estadísticas incorporadas.

Instalación y Configuración.

- Antes de comenzar, hay que asegurarnos de tener Seaborn instalado.
 - `pip install seaborn -->` desde la terminal
- También es recomendable actualizar Matplotlib y Pandas para evitar incompatibilidades.
 - `pip install --upgrade matplotlib pandas -->` desde la terminal.

Importación de Librerías

- Una vez realizada la instalación, importamos junto con las librerías comunes.

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Para asegurarnos que está funcionando correctamente, vamos a cargar un dataset solo como ejemplo de prueba de Seaborn.

```
In [2]: df = sns.load_dataset('penguins')
print(df.head())
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	\
0	Adelie	Torgersen	39.1	18.7	181.0	
1	Adelie	Torgersen	39.5	17.4	186.0	
2	Adelie	Torgersen	40.3	18.0	195.0	
3	Adelie	Torgersen	NaN	NaN	NaN	
4	Adelie	Torgersen	36.7	19.3	193.0	

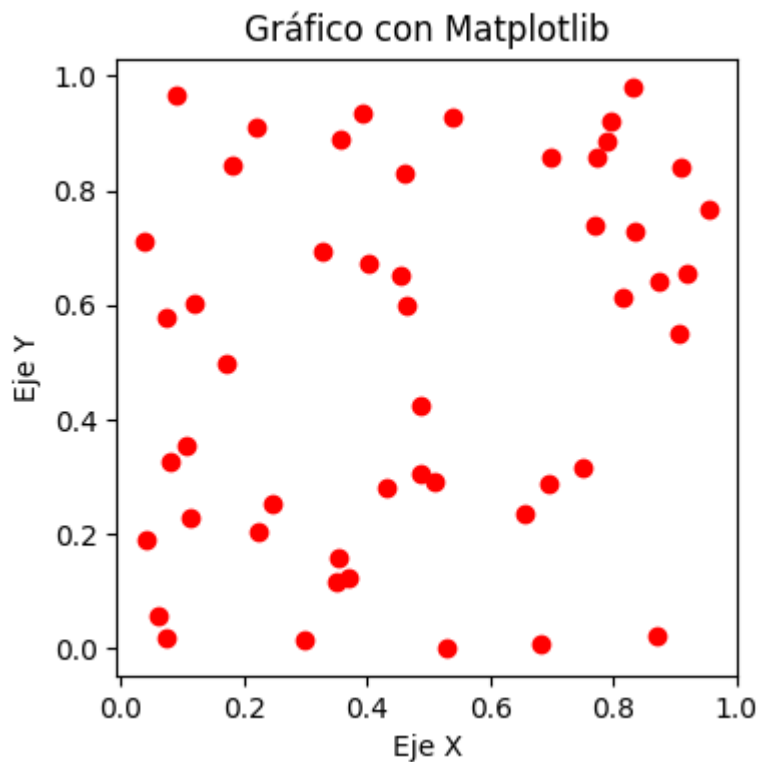
	body_mass_g	sex
0	3750.0	Male
1	3800.0	Female
2	3250.0	Female
3	NaN	NaN
4	3450.0	Female

Comparación con Matplotlib y Pandas

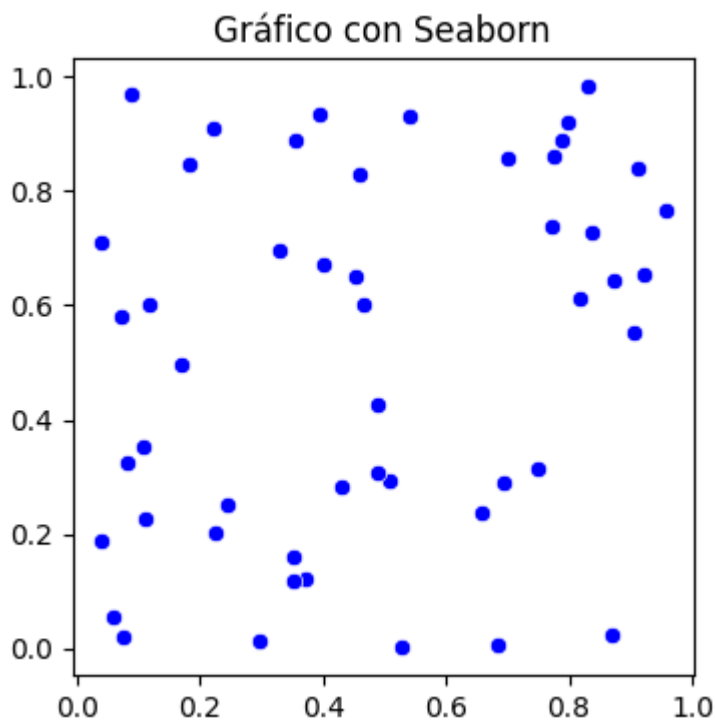
- Seaborn simplifica la creación de gráficos. Vamos a ver un ejemplo de gráfico de dispersión en Matplotlib y el mismo gráfico pero con Seaborn

```
In [ ]: x = np.random.rand(50)
y = np.random.rand(50)

plt.figure(figsize=(4,4))
plt.scatter(x, y, color='red')
plt.title('Gráfico con Matplotlib')
plt.xlabel('Eje X')
plt.ylabel('Eje Y')
plt.show()
```



```
In [ ]: sns.scatterplot(x=x, y=y, color='blue')
plt.title('Gráfico con Seaborn')
plt.show()
```



Carga de datasets predefinidos en Seaborn

- Seaborn incluye varios datasets integrados que facilitan la exploración de datos.
- Para ver una lista de datasets disponibles.

```
In [7]: print(sns.get_dataset_names())
```

```
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds',
'dots', 'dowjones', 'exercise', 'flights', 'fmri', 'geyser', 'glue', 'healthexp',
'iris', 'mpg', 'penguins', 'planets', 'seaice', 'taxis', 'tips', 'titanic']
```

Veamos que contiene algunos de los datasets incluidos en Seaborn

```
In [8]: df = sns.load_dataset("anagrams")
print(df.head(5))
```

	subidr	attnr	num1	num2	num3
0	1	divided	2	4.0	7
1	2	divided	3	4.0	5
2	3	divided	3	5.0	6
3	4	divided	5	7.0	5
4	5	divided	4	5.0	8

```
In [11]: df = sns.load_dataset("car_crashes")
print(df.head(5))
```

	total	speeding	alcohol	not_distracted	no_previous	ins_premium \
0	18.8	7.332	5.640	18.048	15.040	784.55
1	18.1	7.421	4.525	16.290	17.014	1053.48
2	18.6	6.510	5.208	15.624	17.856	899.47
3	22.4	4.032	5.824	21.056	21.280	827.34
4	12.0	4.200	3.360	10.920	10.680	878.41

	ins_losses	abbrev
0	145.08	AL
1	133.93	AK
2	110.35	AZ
3	142.39	AR
4	165.63	CA

```
In [13]: df = sns.load_dataset("planets")
print(df.head(5))
```

	method	number	orbital_period	mass	distance	year
0	Radial Velocity	1	269.300	7.10	77.40	2006
1	Radial Velocity	1	874.774	2.21	56.95	2008
2	Radial Velocity	1	763.000	2.60	19.84	2011
3	Radial Velocity	1	326.030	19.40	110.62	2007
4	Radial Velocity	1	516.220	10.50	119.47	2009

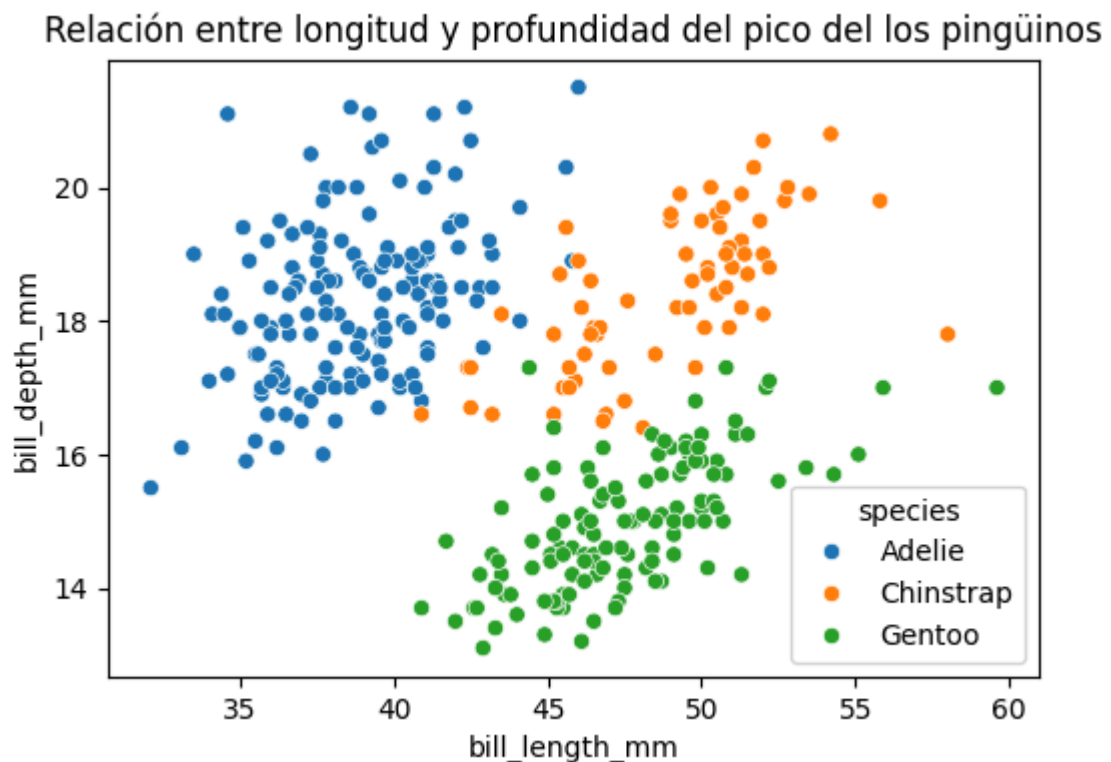
Primer Gráfico con Seaborn

- Crearemos un gráfico de dispersión usando el dataset penguins.

```
In [14]: df = sns.load_dataset("penguins")
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   species                344 non-null   object  
1   island                 344 non-null   object  
2   bill_length_mm         342 non-null   float64 
3   bill_depth_mm          342 non-null   float64 
4   flipper_length_mm      342 non-null   float64 
5   body_mass_g            342 non-null   float64 
6   sex                    333 non-null   object  
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
None
```

```
In [19]: plt.figure(figsize=(6,4))
sns.scatterplot(x='bill_length_mm', y='bill_depth_mm', hue='species', data=df)
plt.title('Relación entre longitud y profundidad del pico del los pingüinos')
plt.show()
```



Fundamentos de Seaborn

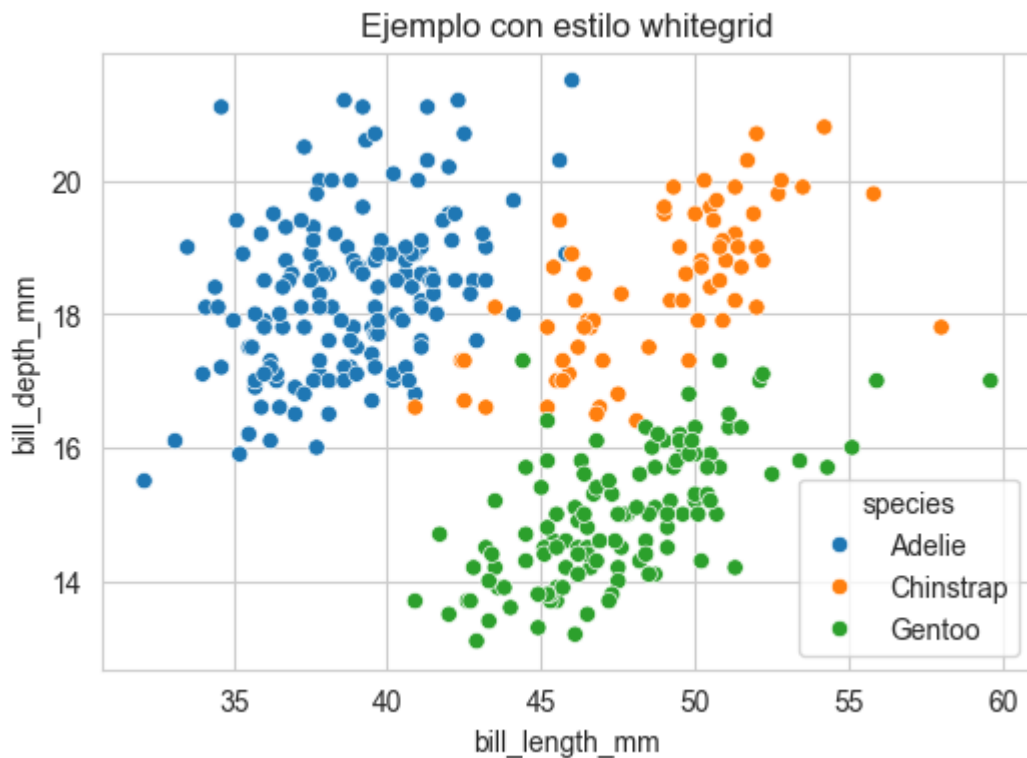
Estilos y temas visuales.

- Seaborn ofrece cinco estilos predefinidos para mejorar la estética de los gráficos.

darkgrid	Por defecto, útil para gráficos con líneas
whitegrid	Ideal para gráficos de barras o boxplots
dark	Fondo oscuro sin cuadrícula
white	Fondo blanco sin cuadrícula
ticks	Similar a "white", pero con marcas en los ejes

```
In [20]: df = sns.load_dataset('penguins')

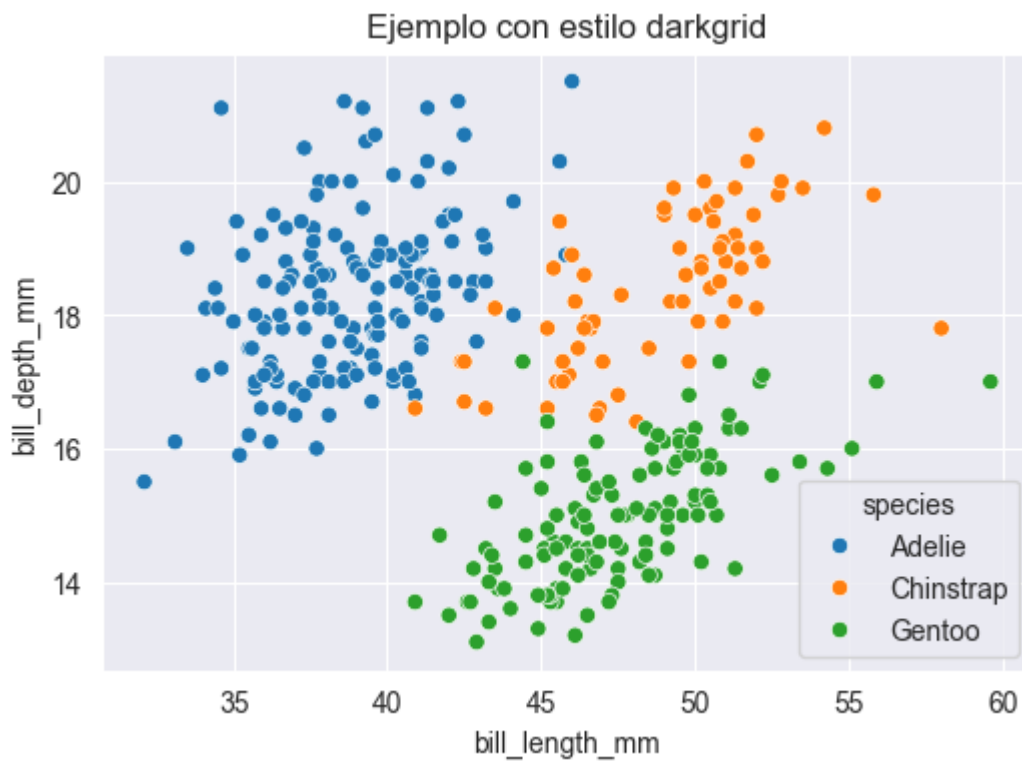
sns.set_style('whitegrid')
plt.figure(figsize=(6,4))
sns.scatterplot(x='bill_length_mm', y='bill_depth_mm', hue='species', data=df)
plt.title('Ejemplo con estilo whitegrid')
plt.show()
```



Vamos a probar el mismo gráfico pero con diferente estilo.

```
In [24]: df = sns.load_dataset('penguins')

sns.set_style('darkgrid')
plt.figure(figsize=(6,4))
sns.scatterplot(x='bill_length_mm', y='bill_depth_mm', hue='species', data=df)
plt.title('Ejemplo con estilo darkgrid')
plt.show()
```



Paletas de colores

- Seaborn incluye muchas paletas de colores predefinidas y también permite crear personalizadas.
- Seaborn facilita el uso de colores que se adaptan bien a las características de sus datos y sus objetivos de visualización.

Paletas disponibles y la forma de verlas.

Paletas de color cualitativas

- Las paletas cualitativas son adecuadas para representar datos categóricos porque la mayor parte de su variación está en el componente del tono

```
In [42]: sns.color_palette()
```

```
Out[42]:
```

```
In [43]: sns.color_palette('tab10')
```

```
Out[43]:
```

Usando sistemas de color circulares

- Cuando se tiene un número arbitrario de categorías, el enfoque más fácil para encontrar tonos únicos es dibujar colores espaciados uniformemente en un espacio de color circular.

```
In [2]: sns.color_palette("hls", 8)
```



```
In [3]: sns.color_palette("husl", 8)
```



Paletas categóricas de Color Brewer

- Otra fuente de paletas categóricas visualmente agradables proviene de la herramienta Color Brewer
- Que también tiene paletas secuenciales y divergentes.

```
In [4]: sns.color_palette("Set2")
```



```
In [5]: sns.color_palette("Paired")
```



Paletas de color secuenciales.

- La segunda clase importante de paletas de colores se llama secuencial. Este tipo de mapeo es apropiado cuando los datos van desde valores relativamente bajos o poco interesantes hasta valores relativamente alto o interesantes.




Paletas perpetuamente uniformes

- Debido a que están destinados a representar valores numéricos, las mejores paletas secuenciales serán perpetuamente uniformes, lo que significa que la discriminabilidad relativa de dos colores es proporcional a la diferencia entre los valores correspondientes.

```
In [6]: sns.color_palette("rocket", as_cmap=True)
```

Out[6]: **rocket**



 under  bad  over

```
In [7]: sns.color_palette("mako", as_cmap=True)
```

Out[7]: **mako**



In [8]: `sns.color_palette("flare", as_cmap=True)`

Out[8]: **flare**



In [9]: `sns.color_palette("crest", as_cmap=True)`

Out[9]: **crest**



In [152...]: `sns.color_palette('deep')`



In [153...]: `sns.color_palette('pastel')`



In [154...]: `sns.color_palette('bright')`



In [10]: `sns.color_palette('dark')`



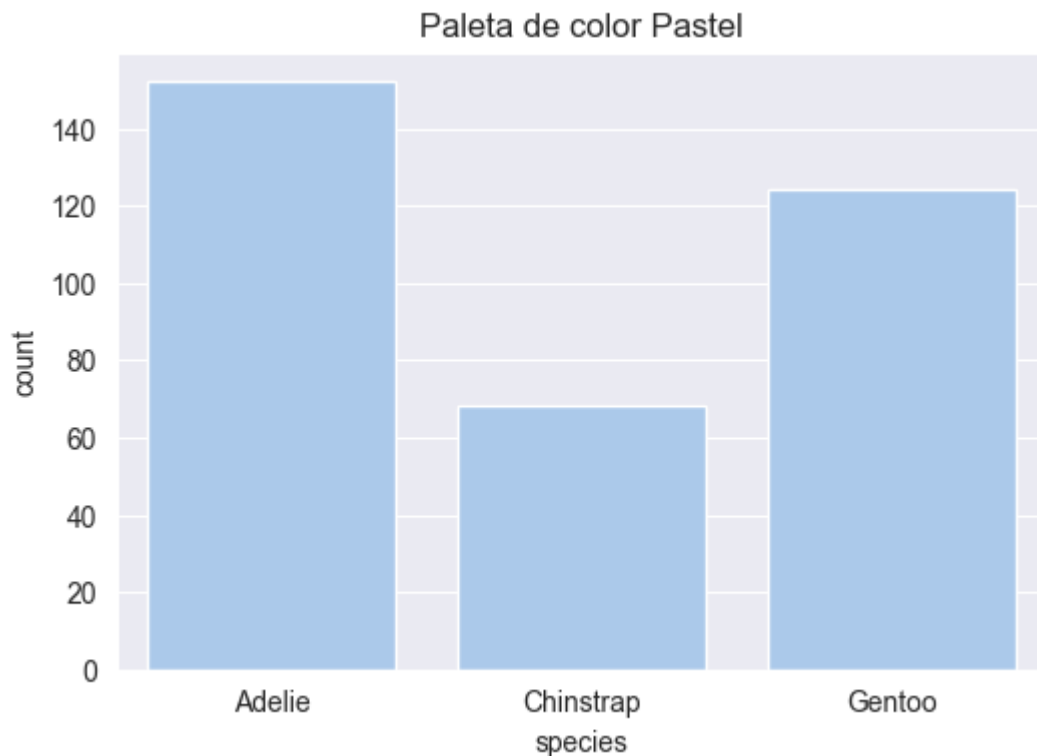
In [11]: `sns.color_palette('colorblind')`




```
In [31]: paletas = sns.palettes.SEABORN_PALETTES
print(list(paletas.keys()))
```

['deep', 'deep6', 'muted', 'muted6', 'pastel', 'pastel6', 'bright', 'bright6', 'dark', 'dark6', 'colorblind', 'colorblind6']

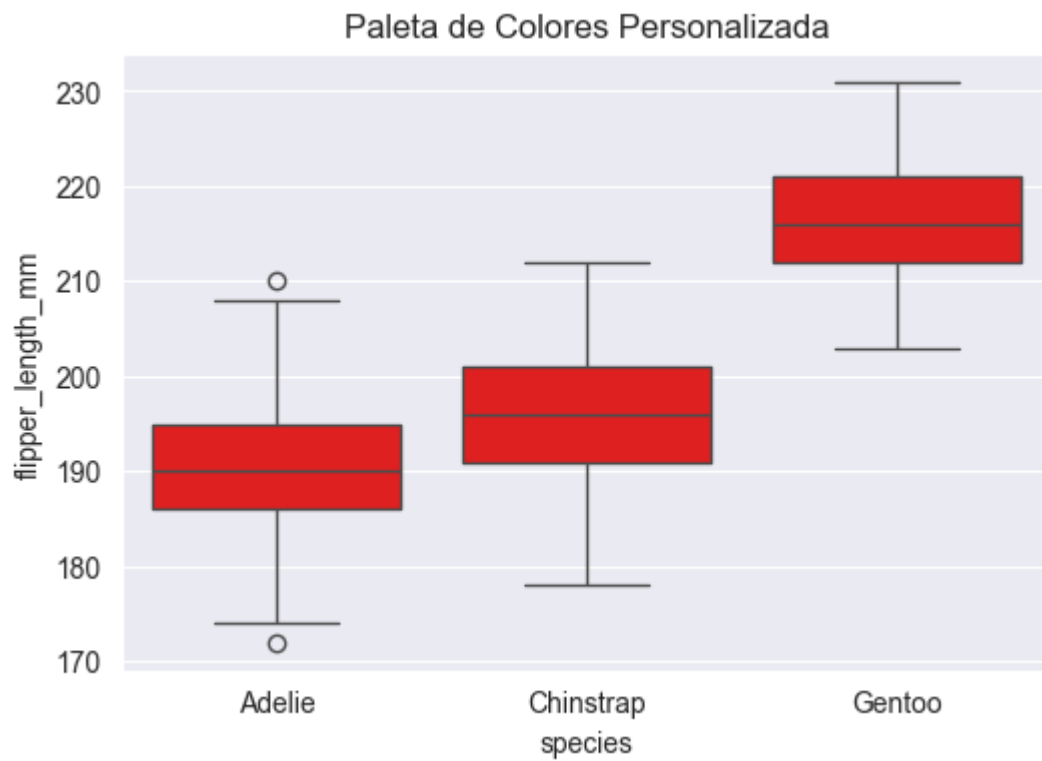
```
In [33]: sns.set_palette('pastel')
plt.figure(figsize=(6,4))
sns.countplot(x='species', data=df)
plt.title('Paleta de color Pastel')
plt.show()
```



Creando una Paleta personalizada.

```
In [ ]: personalizada = sns.color_palette(["#FF0000", "#FF0000", "#0000FF"]) # Código Hexadecimal
plt.figure(figsize=(6,4))
sns.set_palette(personalizada)
sns.boxplot(x='species', y='flipper_length_mm', data=df)
plt.title('Paleta de Colores Personalizada')
plt.show
```

```
Out[ ]: <function matplotlib.pyplot.show(close=None, block=None)>
```



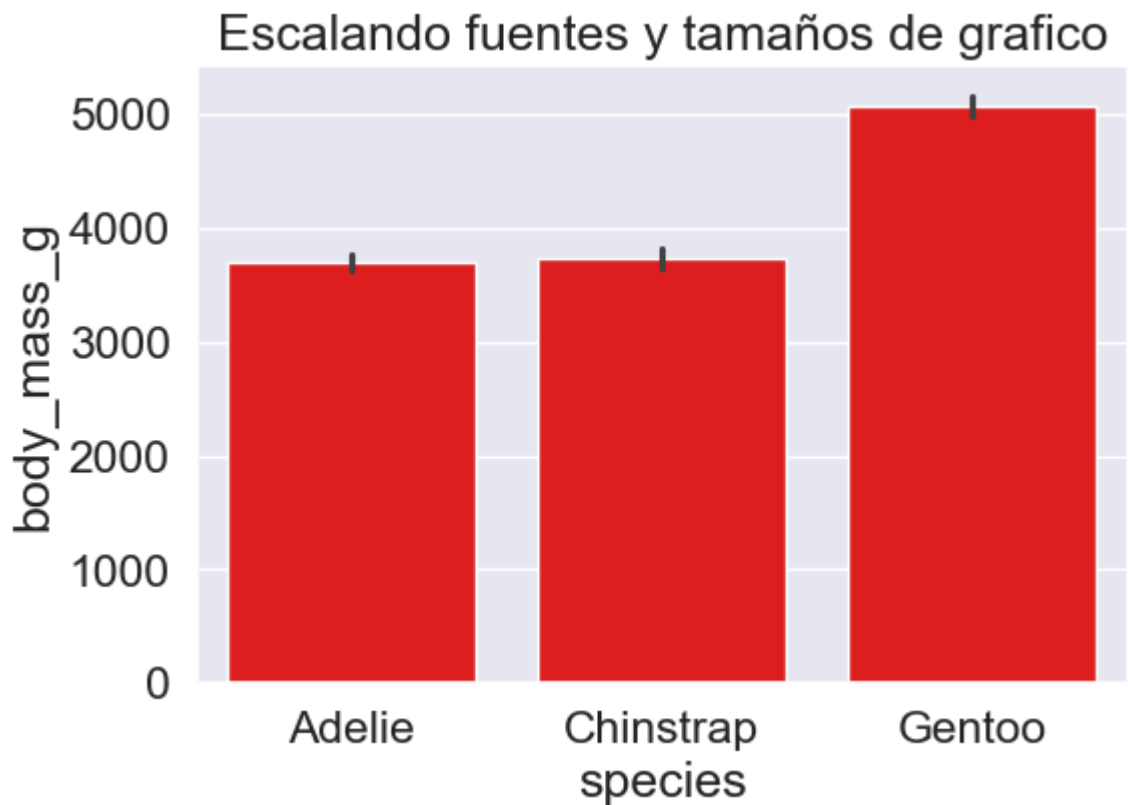
Ajuste de tamaños y fuentes.

- Seaborn permite modificar el tamaño de los gráficos y el tamaño de fuente

paper	Más pequeño
notebook	Por defecto
talk	Más grande, ideal para presentaciones
poster	Mucho más grande

Cambiar el tamaño del gráfico y la escala de fuentes.

```
In [47]: plt.figure(figsize=(6,4))
sns.set_context('notebook', font_scale=1.5)
sns.barplot(x = 'species', y = 'body_mass_g', data = df)
plt.title('Escalando fuentes y tamaños de grafico')
plt.show()
```



Vamos a visualizar como cambia la apariencia de cada estilo con cinco gráficos en un fila.

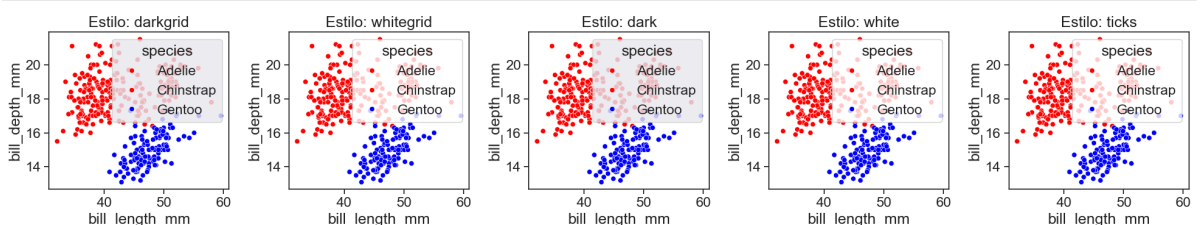
- Se crea un gráfico de dispersión para cada estilo en una fila
- Se utiliza zip(axes,, estilos) para asignar cada gráfico a un estilo distinto
- plt.tight_layout() ajusta automáticamente los gráficos.

```
In [58]: estilos = ['darkgrid', 'whitegrid', 'dark', 'white', 'ticks']

fig, axes = plt.subplots(1, 5, figsize=(20,4))

for ax, estilo in zip(axes, estilos):
    sns.set_style(estilo)
    sns.scatterplot(x = 'bill_length_mm', y = 'bill_depth_mm', hue='species', data=
    ax.set_title(f'Estilo: {estilo}')

plt.tight_layout()
plt.show()
```



Vamos a visualizar como cambia la apariencia en cinco gráficos con diferentes paletas.

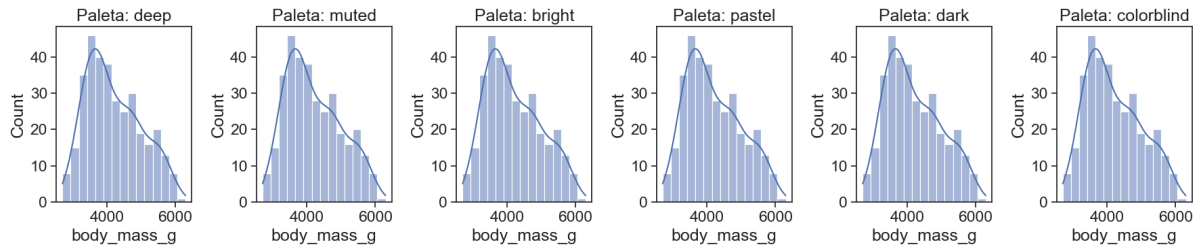
- Se usa un histograma con histplot() para ver el efecto de cada paleta
- kde=True agrega una curva de densidad
- zip(axes, paletas) permite aplicar una paleta diferente en cada gráfico

```
In [60]: paletas = ['deep', 'muted', 'bright', 'pastel', 'dark', 'colorblind']

fig, axes = plt.subplots(1, 6, figsize=(18, 4))
```

```
for ax, paleta in zip(axes, paletas):
    sns.set_palette(paleta)
    sns.histplot(sns.load_dataset('penguins')['body_mass_g'], bins = 15, kde=True,
    ax.set_title(f'Paleta: {paleta}'))

plt.tight_layout()
plt.show()
```



Vamos a visualizar como cambia la apariencia en cinco gráficos con diferentes contextos.

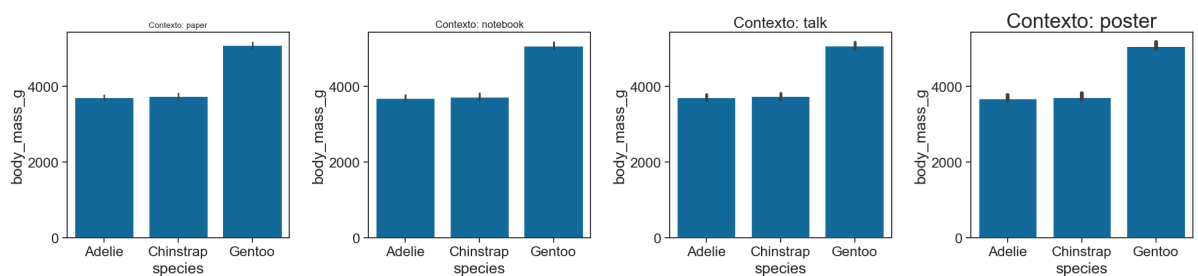
- Se crean cuatro gráficos de barras, cada uno con un texto diferente
- paper es el más pequeño, poster es el más grande
- set_context(contexto) cambia el tamaño de fuente y ejes.

```
In [61]: contextos = ['paper', 'notebook', 'talk', 'poster']

fig, axes = plt.subplots(1, 4, figsize=(20, 5))

for ax, contexto in zip(axes, contextos):
    sns.set_context(contexto)
    sns.barplot(x='species', y='body_mass_g', data=sns.load_dataset('penguins'), ax=ax)
    ax.set_title(f'Contexto: {contexto}')

plt.tight_layout()
plt.show()
```



Gráficos básicos con Seaborn

- Seaborn ofrece una variedad de gráficos para explorar datos de manera sencilla.
- En este apartado veremos:
 1. Gráficos de dispersión (scatterplot)
 2. Gráficos de barras (barplot)
 3. Gráficos de cajas (boxplot)
 4. Histogramas y KDE (histplot, kdeplot)

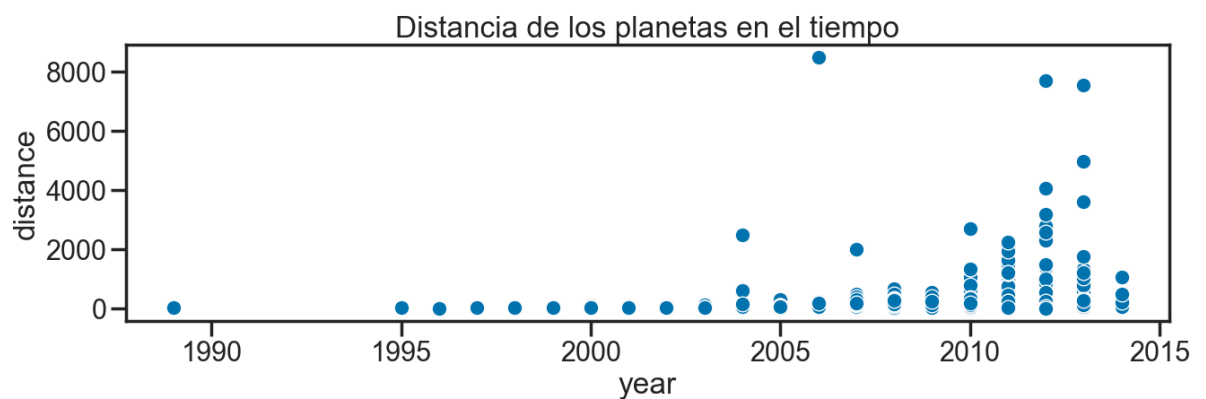
Gráficos de dispersión (scatterplot)

- Muestra relación entre dos variables numéricas
- Puede incluir colores para agrupar datos.

```
In [62]: df = sns.load_dataset("planets")
print(df.info())

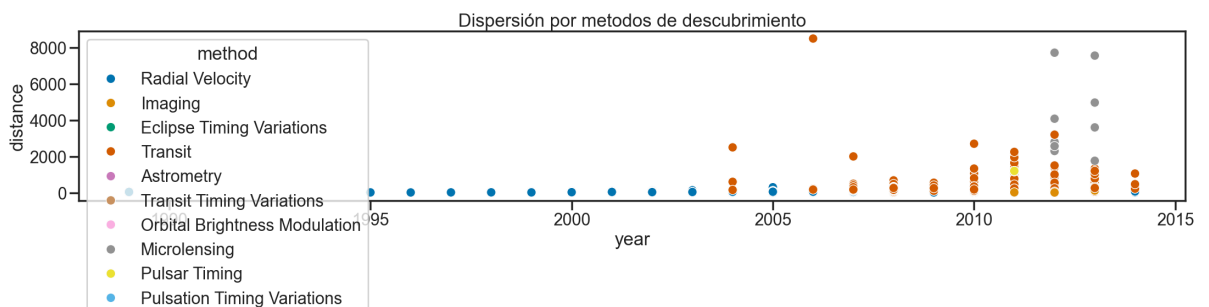
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1035 entries, 0 to 1034
Data columns (total 6 columns):
#   Column          Non-Null Count  Dtype
---  -
0   method          1035 non-null   object
1   number           1035 non-null   int64
2   orbital_period   992 non-null    float64
3   mass             513 non-null    float64
4   distance         808 non-null    float64
5   year            1035 non-null   int64
dtypes: float64(3), int64(2), object(1)
memory usage: 48.6+ KB
None
```

```
In [67]: plt.figure(figsize=(15, 4))
sns.scatterplot(x='year', y = 'distance', data=df)
plt.title('Distancia de los planetas en el tiempo')
plt.show()
```



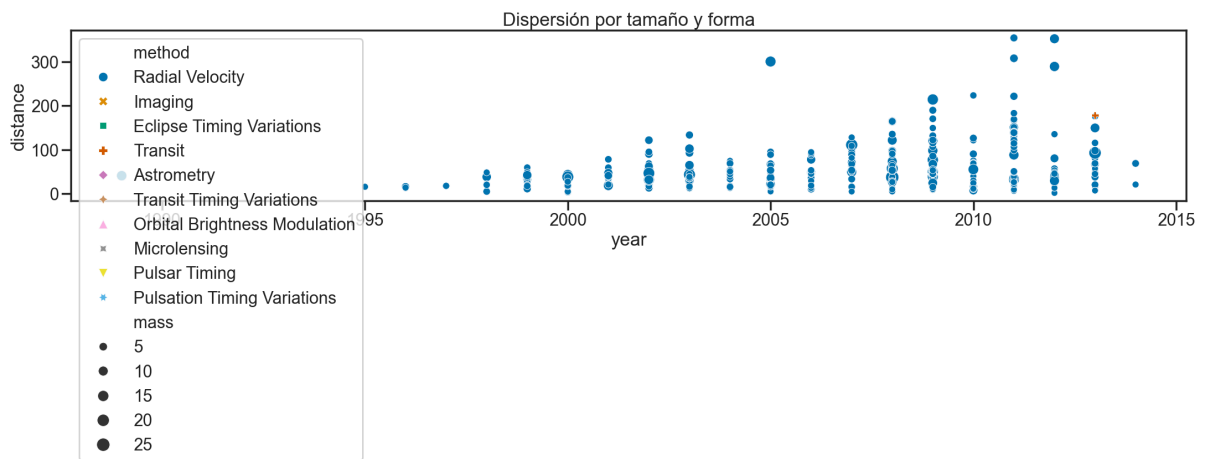
Agregando colores según categorías

```
In [77]: plt.figure(figsize=(26, 4))
sns.scatterplot(x='year', y = 'distance', hue='method', data=df)
plt.title('Dispersión por metodos de descubrimiento')
plt.show()
```



Cambiando tamaño y forma de puntos

```
In [78]: plt.figure(figsize=(26, 4))
sns.scatterplot(x='year', y = 'distance', hue='method', size='mass', style='method')
plt.title('Dispersión por tamaño y forma')
plt.show()
```



Gráficos de líneas (lineplot)

- Ideal para mostrar tendencias a lo largo del tiempo

```
In [79]: df = sns.load_dataset("flights")
print(df.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   year        144 non-null    int64
1   month       144 non-null    category
2   passengers  144 non-null    int64
dtypes: category(1), int64(2)
memory usage: 2.9 KB
None
```

```
In [ ]: plt.figure(figsize=(10, 4))
sns.lineplot(x = 'year', y = 'passengers', data = df)
plt.title('Evolución de pasajeros por año')
plt.show()
```

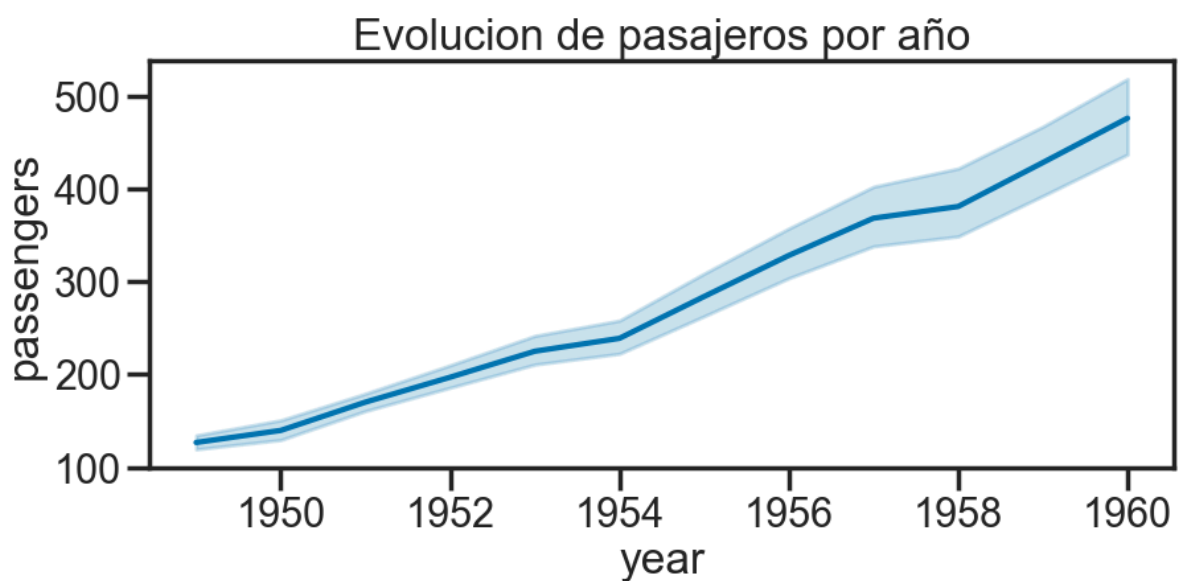
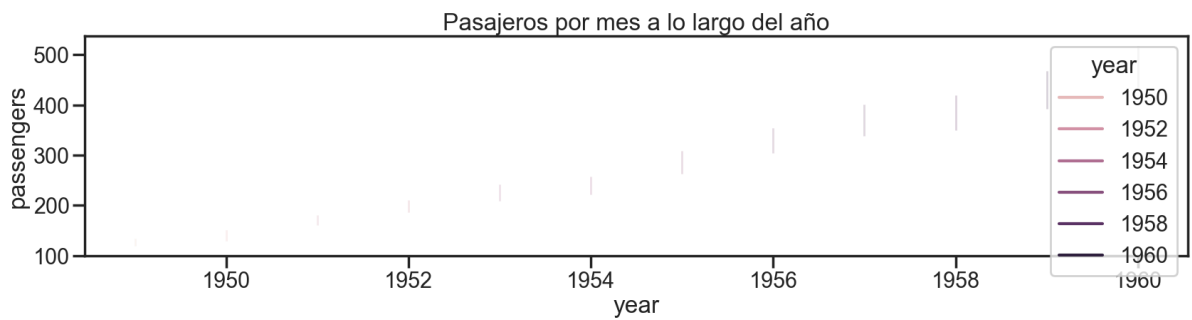


Gráfico com multiples líneas

- hue = year, dibuja una línea para cada año en diferentes colores

```
In [84]: plt.figure(figsize=(20, 4))
sns.lineplot(x = 'year', y = 'passengers', hue='year', data = df)
plt.title('Pasajeros por mes a lo largo del año')
plt.show()
```



Gráficos de barras (barplot)

- Representa valores categóricos con sus respectivas cantidades.

```
In [85]: df = sns.load_dataset("titanic")
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
#   Column      Non-Null Count  Dtype
---  -
0   survived    891 non-null    int64
1   pclass      891 non-null    int64
2   sex         891 non-null    object
3   age        714 non-null    float64
4   sibsp      891 non-null    int64
5   parch      891 non-null    int64
6   fare       891 non-null    float64
7   embarked   889 non-null    object
8   class      891 non-null    category
9   who        891 non-null    object
10  adult_male  891 non-null    bool
11  deck       203 non-null    category
12  embark_town 889 non-null    object
13  alive      891 non-null    object
14  alone      891 non-null    bool
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
None
```

```
In [88]: plt.figure(figsize=(10, 4))
sns.barplot(x = 'class', y = 'fare', data=df)
plt.title('Tarifa promedio por clase en el Titanic')
plt.show()
```

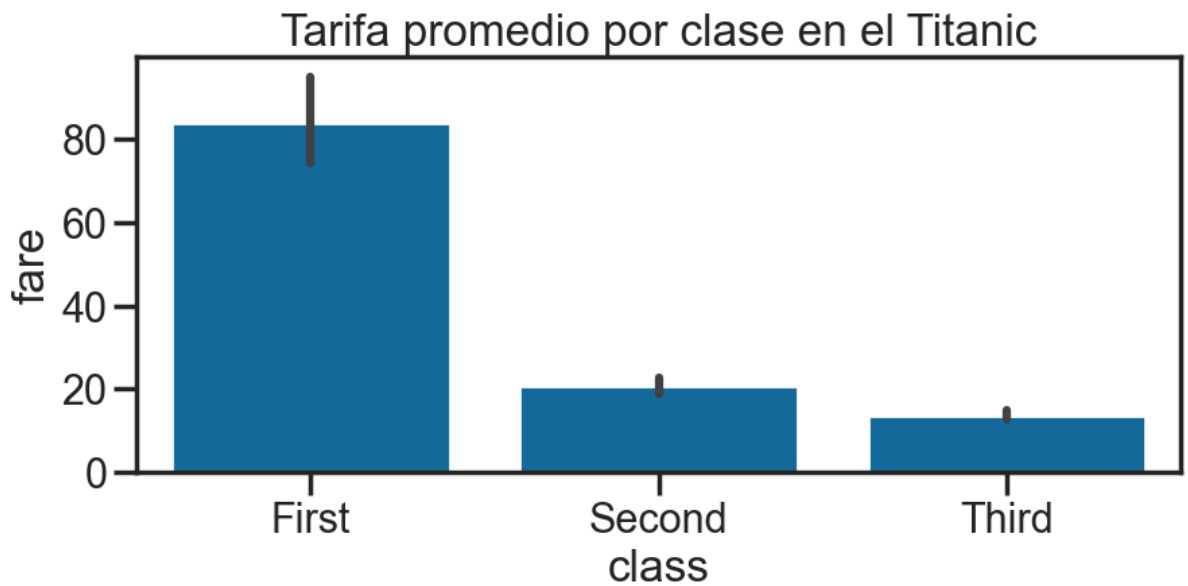


Gráfico con agregaciones y desgloses

```
In [89]: plt.figure(figsize=(10, 4))
sns.barplot(x='class', y='fare', hue='sex', data=df)
plt.title('Tarifa promedio por clase y género')
plt.show()
```

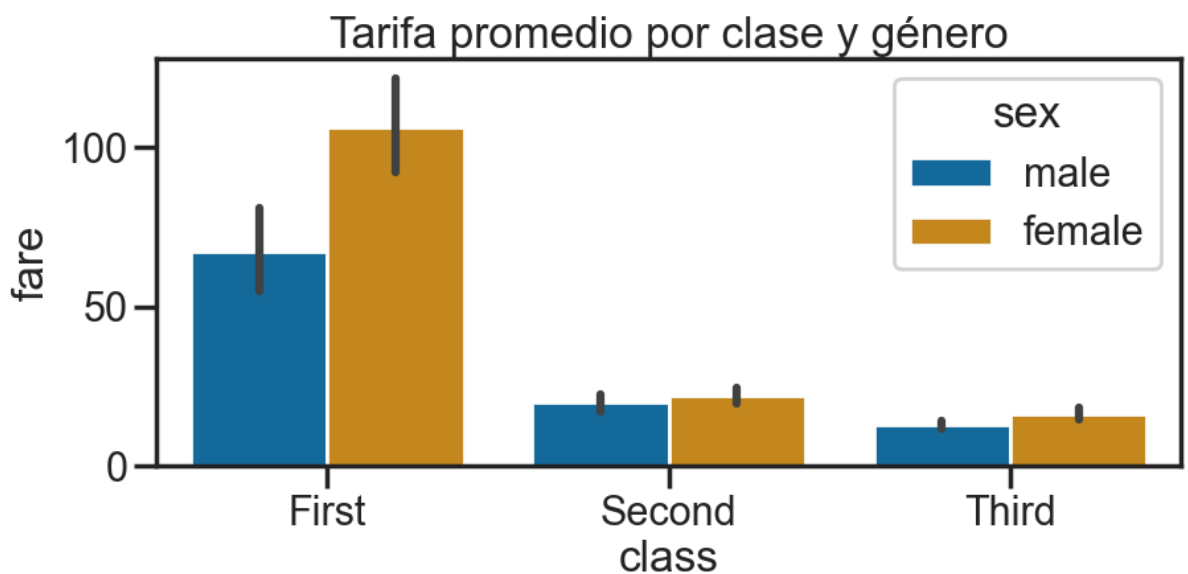


Gráfico de cajas (boxplot)

- Muestra la distribución de los datos y posibles valores atípicos.

```
In [93]: df = sns.load_dataset("penguins")
print(df.info())
```



```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  ---                ---
0   species                344 non-null   object
1   island                 344 non-null   object
2   bill_length_mm         342 non-null   float64
3   bill_depth_mm          342 non-null   float64
4   flipper_length_mm      342 non-null   float64
5   body_mass_g            342 non-null   float64
6   sex                    333 non-null   object
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
None

```

```

In [94]: plt.figure(figsize=(10, 4))
sns.boxplot(x='species', y='bill_length_mm', data=df)
plt.title('Distribución de longitudes por especie')
plt.show()

```

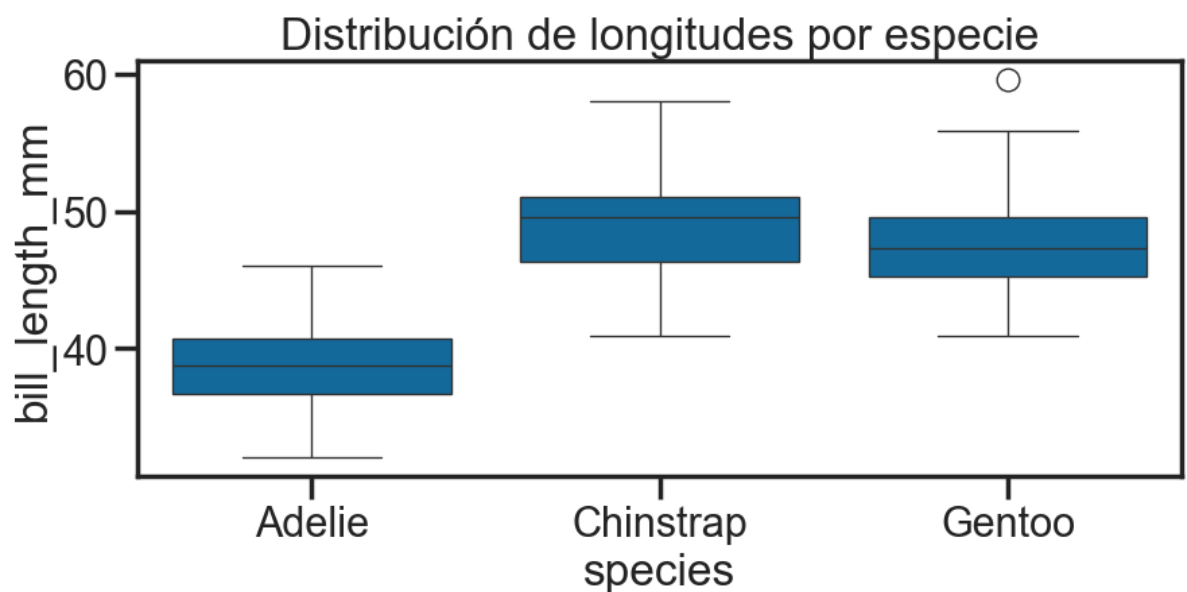
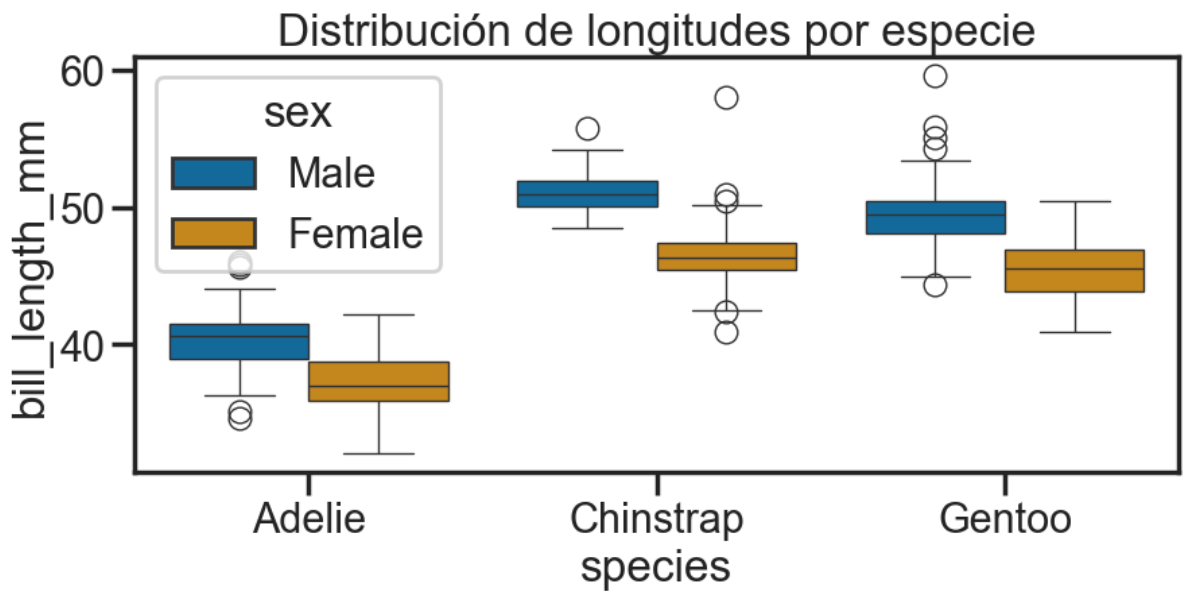


Gráfico con división por categoría

```

In [95]: plt.figure(figsize=(10, 4))
sns.boxplot(x='species', y='bill_length_mm', hue='sex', data=df)
plt.title('Distribución de longitudes por especie')
plt.show()

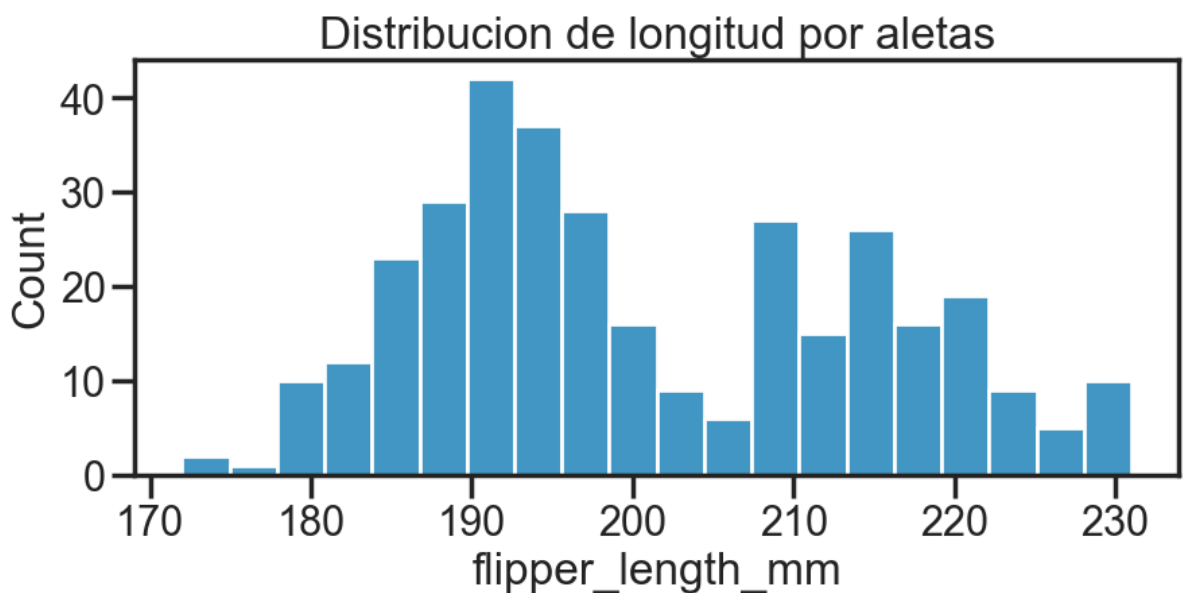
```



Histogramas y KDE (histplot, kdedplot)

- Para ver la distribución de datos numéricos

```
In [96]: plt.figure(figsize=(10, 4))
sns.histplot(df['flipper_length_mm'], bins=20)
plt.title('Distribucion de longitud por aletas')
plt.show()
```



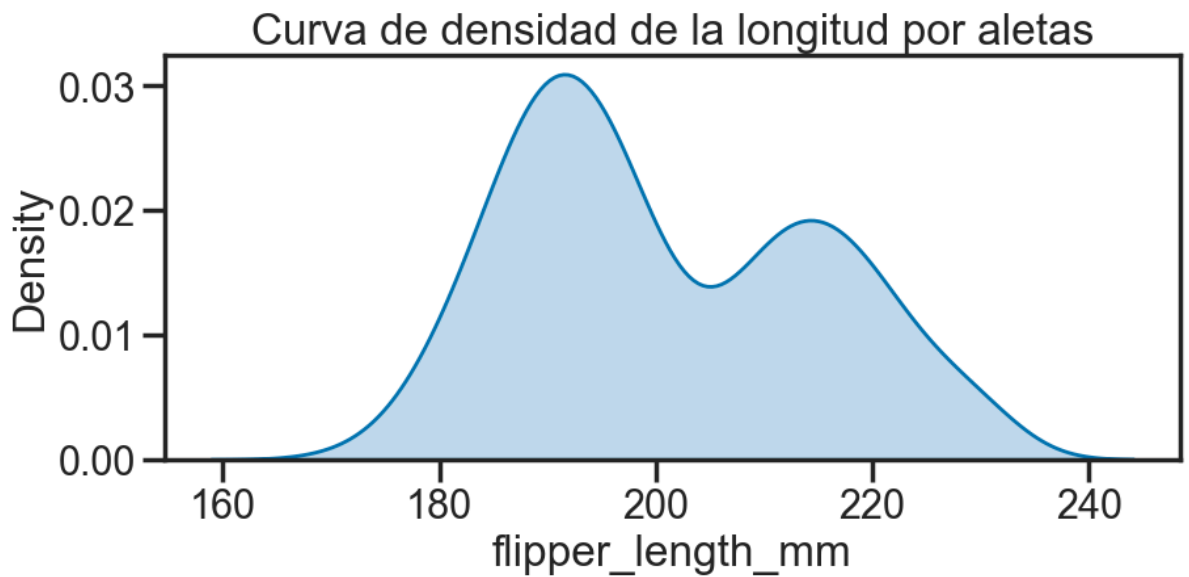
KDE Plot (Curva de densidad)

```
In [99]: plt.figure(figsize=(10, 4))
sns.kdeplot(df['flipper_length_mm'], shade=True) # En la compilación nos dice que u
plt.title('Curva de densidad de la longitud por aletas')
plt.show()
```

C:\Users\Lalo\AppData\Local\Temp\ipykernel_11280\4122259467.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

```
sns.kdeplot(df['flipper_length_mm'], fill=True) # En la compilación nos dice qu
e usamos fill=True en lugar de shade
```



Gráficos avanzados con Seaborn

- En este apartado veremos
 1. Gráficos de violín (violinplot)
 2. Gráficos de enjambre (swarmplot)
 3. Gráficos de caja y enjambre combinados
 4. Mapas de calor (heatmap)
 5. Gráficos de regresión (regplot, lmplo)
 6. Gráficos de pares (pairplot)
 7. Facet Grid: múltiples gráficos en una misma figura

Gráficos de violín (violinplot)

- Muestra la distribución de los datos combinando un boxplot con un KDE (curva de densidad)
- Es útil para ver cómo se distribuyen los valores dentro de una categoría.
- El ancho de cada violín representa la densidad de los datos en esa región.
- Es una versión mejorada del boxplot, ya que muestra más detalles de la distribución.

In [100...

```
plt.figure(figsize=(10, 4))
sns.violinplot(x='species', y='bill_length_mm', data=df)
plt.title('Distribución de longitudes de pico por especie')
plt.show()
```

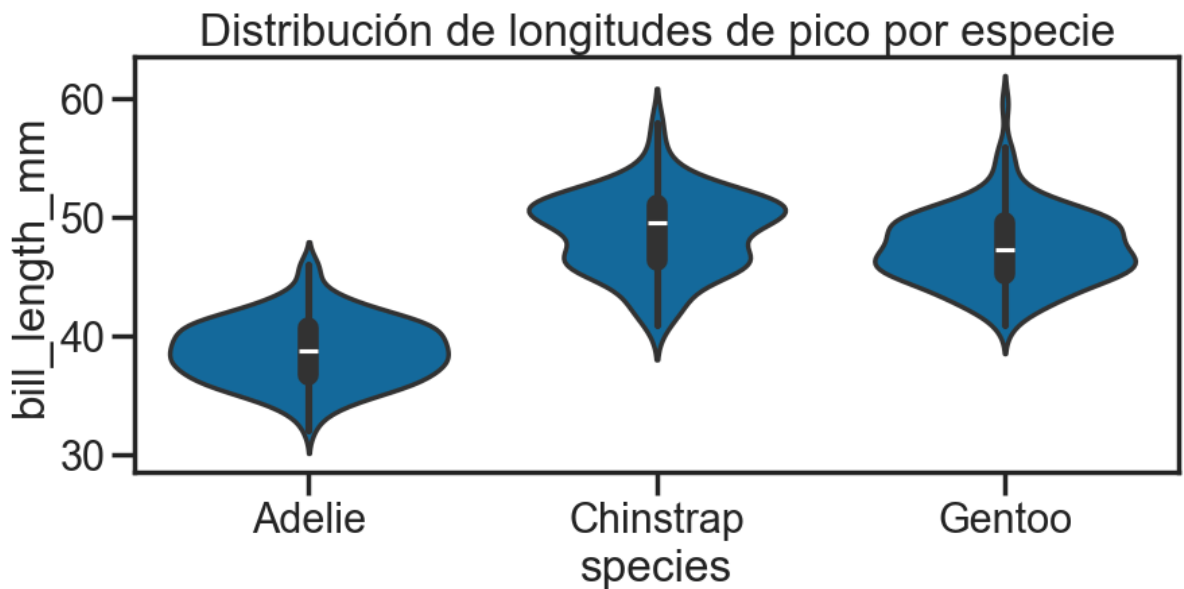
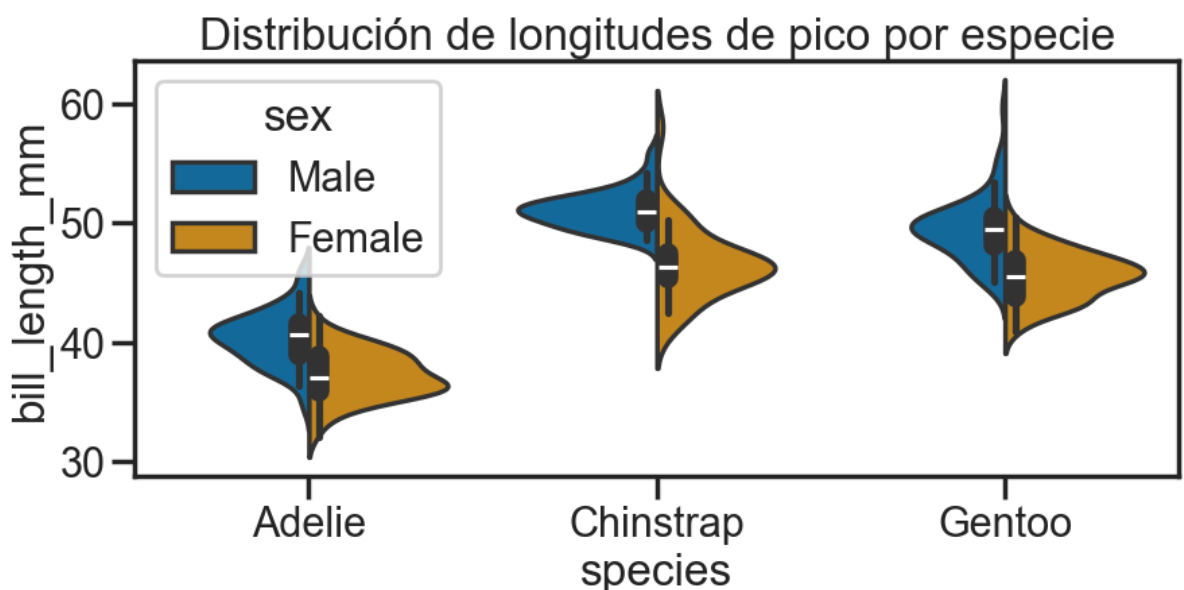


Gráfico de violin con separación por género (hue)

- split=True divide cada violin en dos partes según la variable hue

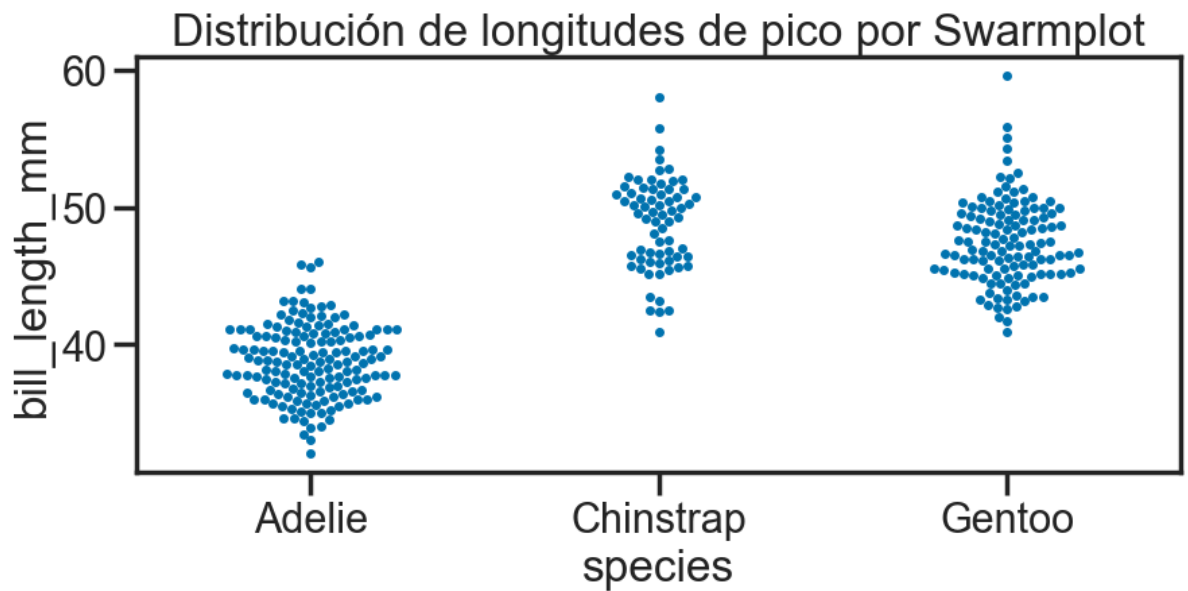
```
In [101...] plt.figure(figsize=(10, 4))
sns.violinplot(x='species', y='bill_length_mm', hue='sex', data=df, split=True)
plt.title('Distribución de longitudes de pico por especie')
plt.show()
```



Graficos de enjambre (swarmplot)

- Muestra cada punto individualmente en una categoría.
- Ideal para ver la dispersión real de los datos.
- Mientras que violinplot muestra una estimación de la densidad, el swarmplot muestra los valores exactos de los datos.

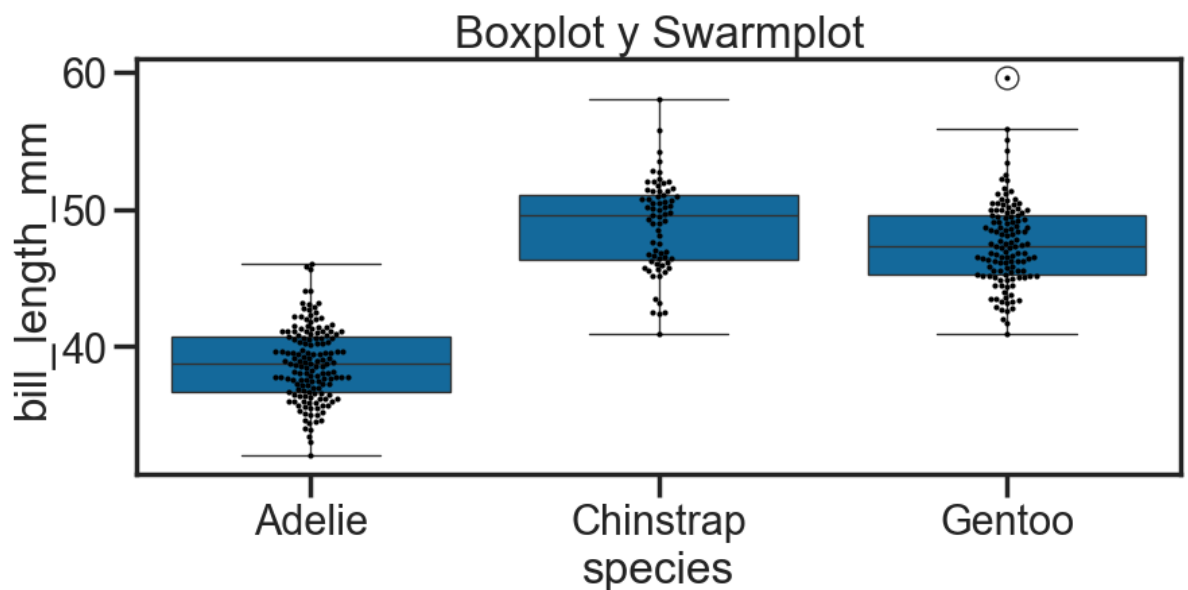
```
In [102...] plt.figure(figsize=(10, 4))
sns.swarmplot(x='species', y='bill_length_mm', data=df)
plt.title('Distribución de longitudes de pico por Swarmplot')
plt.show()
```



Combinación de Boxplot y Swarmplot

- Podemos combinar ambos gráficos para ver la distribución y valores individuales.

```
In [104... plt.figure(figsize=(10, 4))
sns.boxplot(x='species', y='bill_length_mm', data=df)
sns.swarmplot(x='species', y='bill_length_mm', color='black', size=3, data=df)
plt.title('Boxplot y Swarmplot')
plt.show()
```



Mapas de calor (hetamap)

- Muestra matrices de datos numéricos con colores.
- Es muy útil para ver correlaciones entre variables.

```
In [105... df = sns.load_dataset("flights")
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   year        144 non-null    int64
1   month       144 non-null    category
2   passengers  144 non-null    int64
dtypes: category(1), int64(2)
memory usage: 2.9 KB
None
```

- cmap= usa una paleta de colores cálidos y fríos
- annot=True muestra los valores en cada celda
- fmt para dar formato y evita mostrar decimales

```
In [109... plt.figure(figsize=(10, 4))
df_corr = df.pivot(index='month', columns='year', values='passengers')
sns.heatmap(df_corr, cmap='coolwarm', annot=True, fmt='.0f')
plt.title('Numero de pasajeros por mes y año')
plt.show()
```

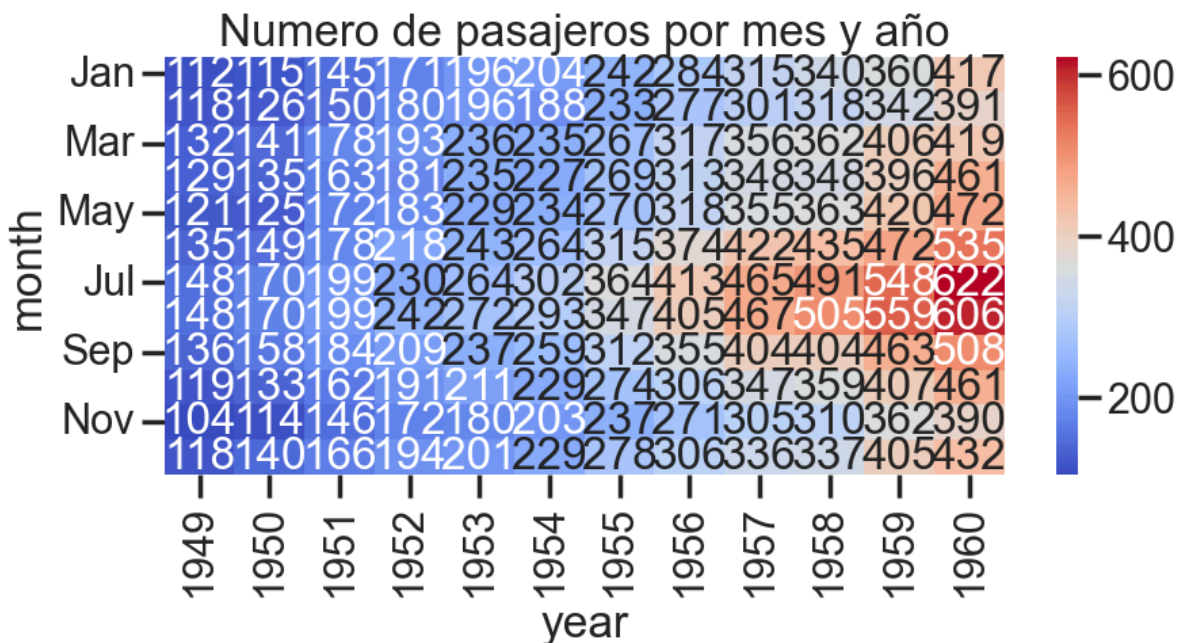
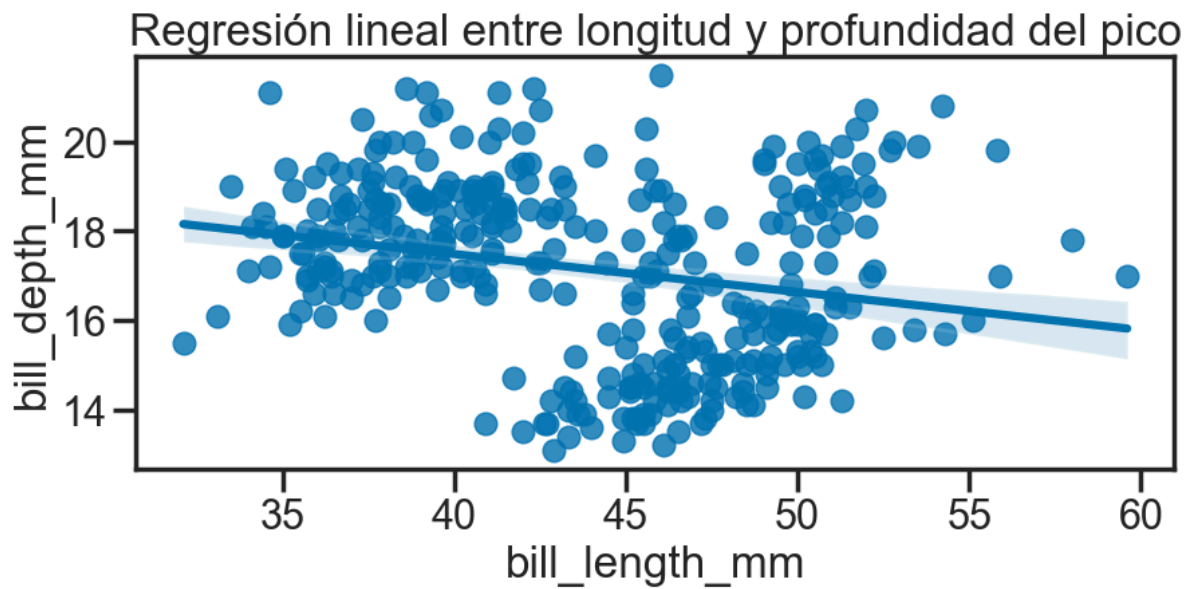


Gráfico de regresión (regplot, Implot)

- Visualiza la relación entre dos variables con una linea de regresión.

```
In [ ]: df = sns.load_dataset("penguins")
plt.figure(figsize=(10, 4))
sns.regplot(x='bill_length_mm', y='bill_depth_mm', data=df)
plt.title('Regresión lineal entre longitud y profundidad del pico')
plt.show()
```

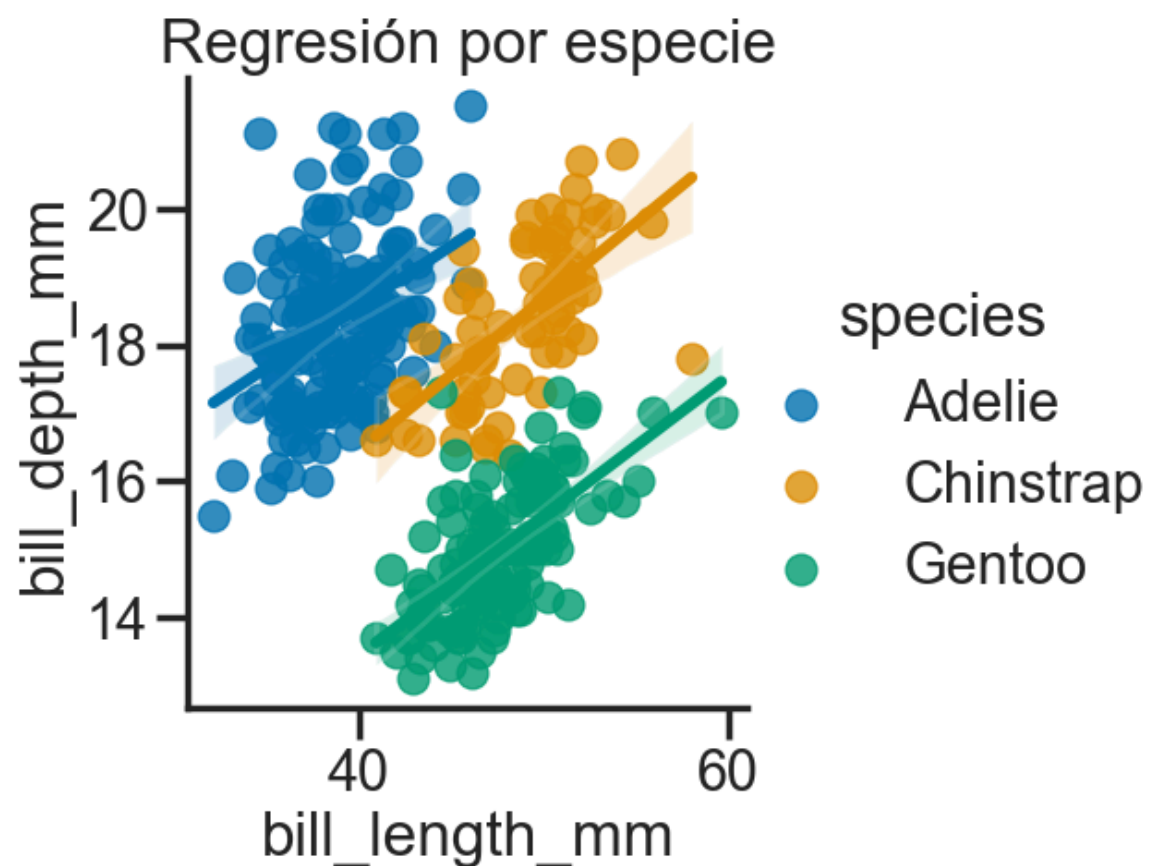


Regplot con diferentes especies Implot

In [120...

```
plt.figure(figsize=(3, 3))
sns.lmplot(x='bill_length_mm', y='bill_depth_mm', hue='species', data=df)
plt.title('Regresión por especie')
plt.show()
```

<Figure size 300x300 with 0 Axes>



Gráficos de pares (pairplot)

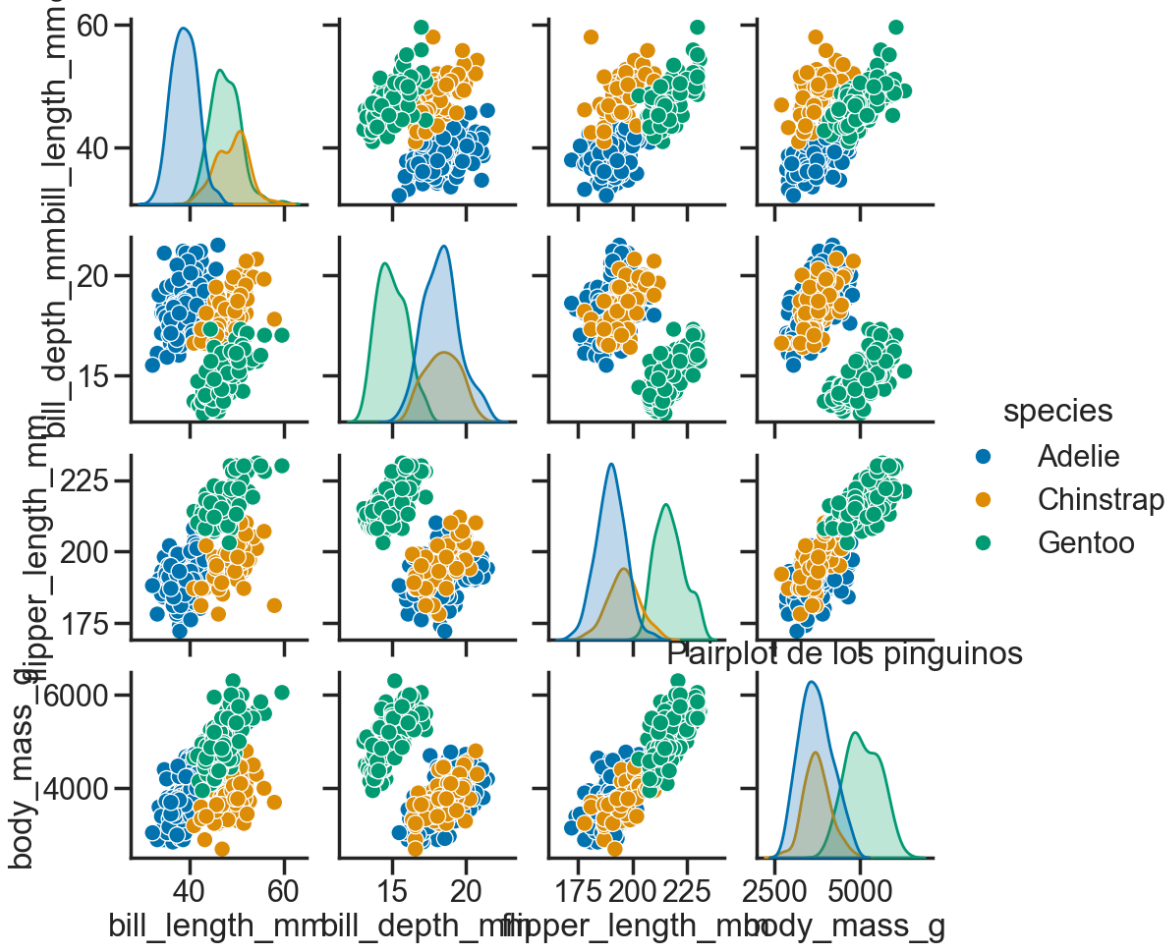
- Muestra relaciones entre múltiples variables numéricas a la vez

In [123...

```
plt.figure(figsize=(5, 5))
sns.pairplot(df, hue='species')
```

```
plt.title('Pairplot de los pingüinos')
plt.show()
```

<Figure size 500x500 with 0 Axes>



Facet Grid: múltiples gráficos en una misma figura.

- Para crear gráficos múltiples según una categoría.

In [124...]

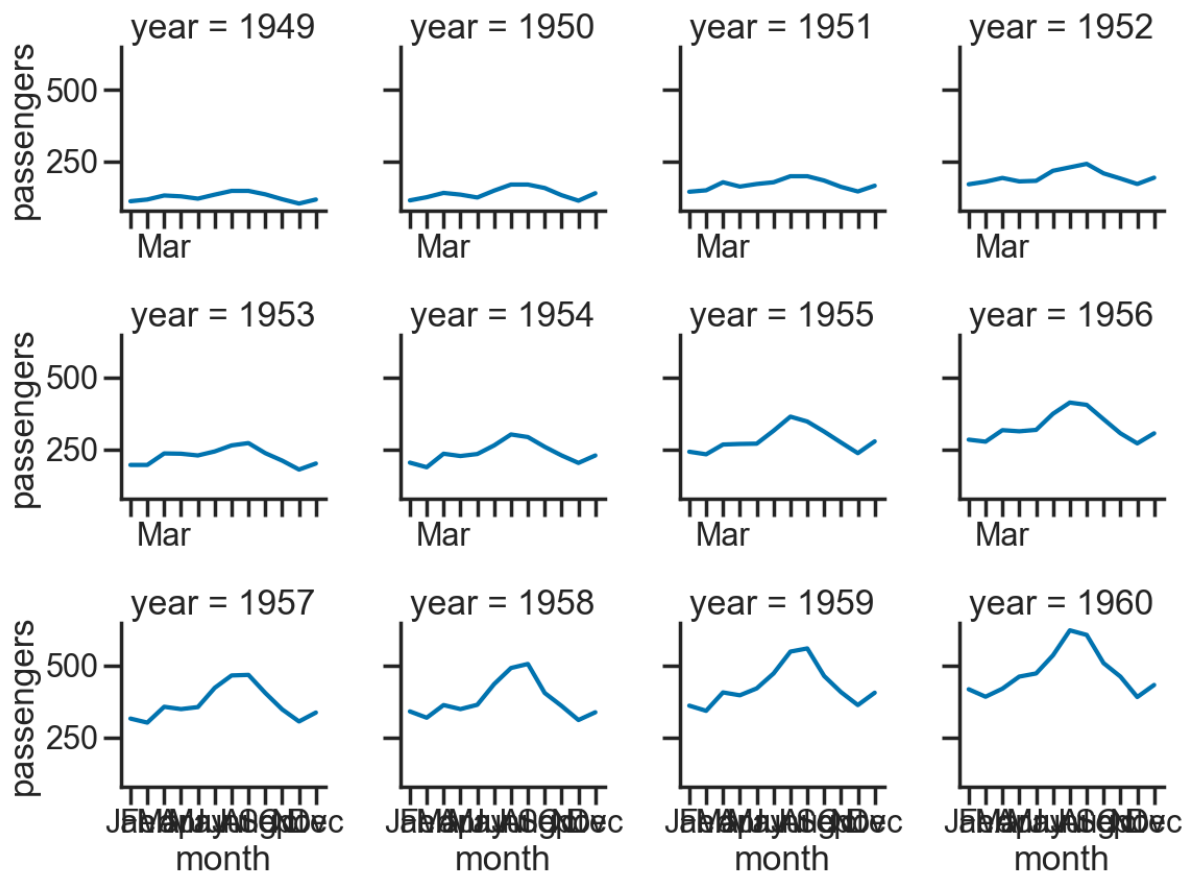
```
df = sns.load_dataset("flights")
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  -
0   year        144 non-null    int64
1   month       144 non-null    category
2   passengers  144 non-null    int64
dtypes: category(1), int64(2)
memory usage: 2.9 KB
None
```

In []:

```
plt.figure(figsize=(5, 5))
graf = sns.FacetGrid(df, col='year', col_wrap=4) # col_wrap es el número de columnas
graf.map(sns.lineplot, 'month', 'passengers')
plt.show()
```

<Figure size 500x500 with 0 Axes>



Transformación y ajustes de Gráficos.

En este apartado veremos

- . Personalización de colores en Seaborn2
- . Ajuste de tamaño y aspecto de gráficos
- . Modificación de etiquetas y títulos
- . Uso de estilos y temas en Seaborn
- . Escalas logarítmicas y normalización de datos
- . Rotación de etiquetas en graficos de barras
- . Ajuste de limites en ejes (xlim, ylim)
- . Agregar anotaciones en gráficos
- . Personalización por tipo de gráfico.

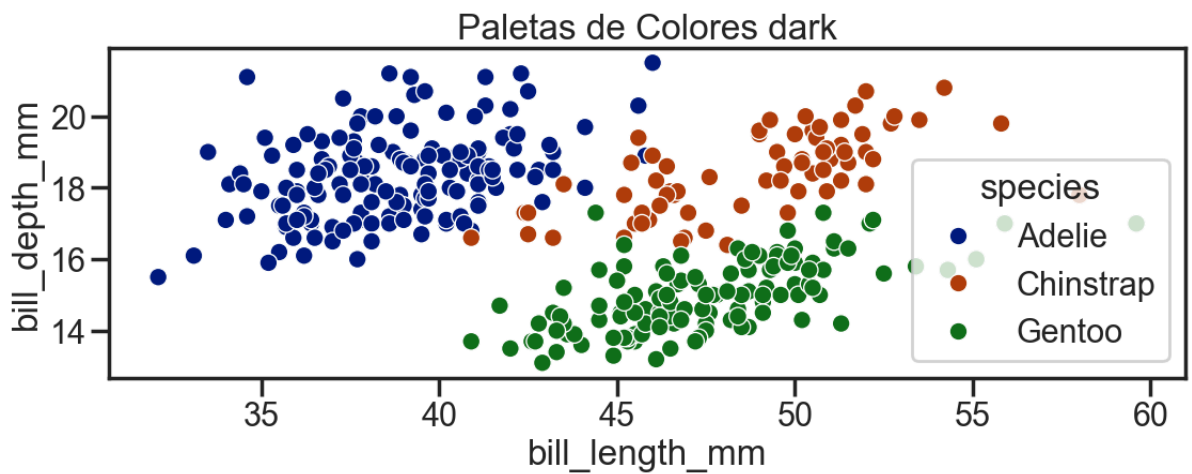
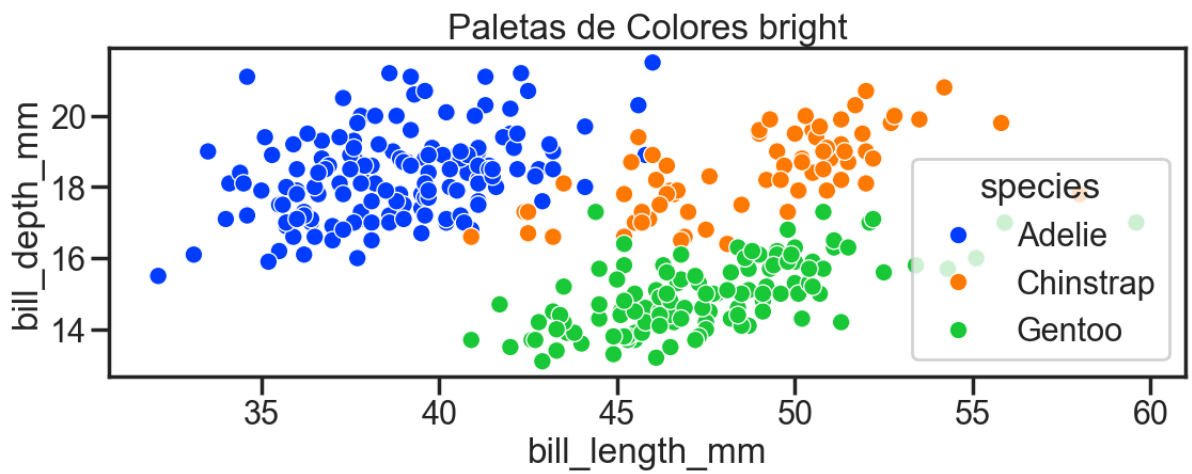
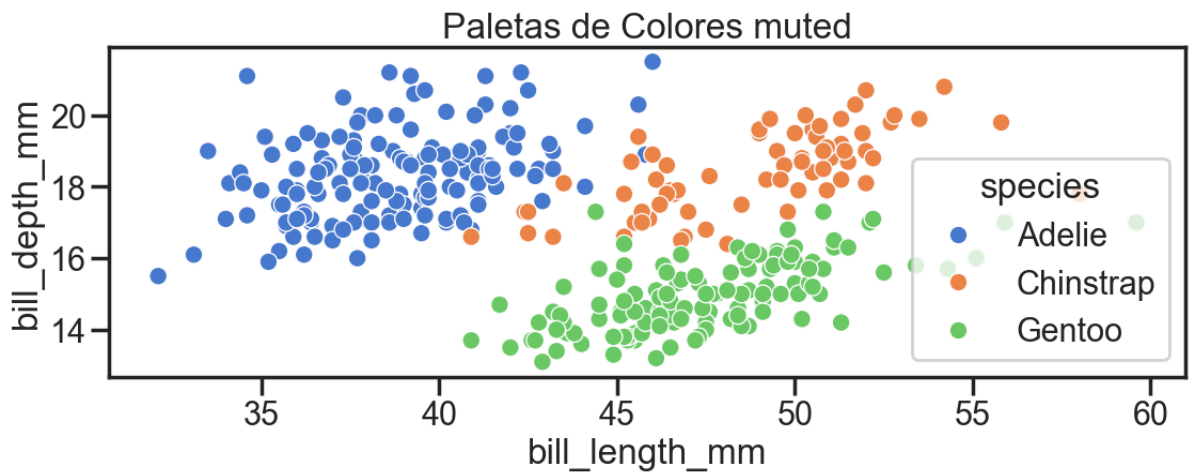
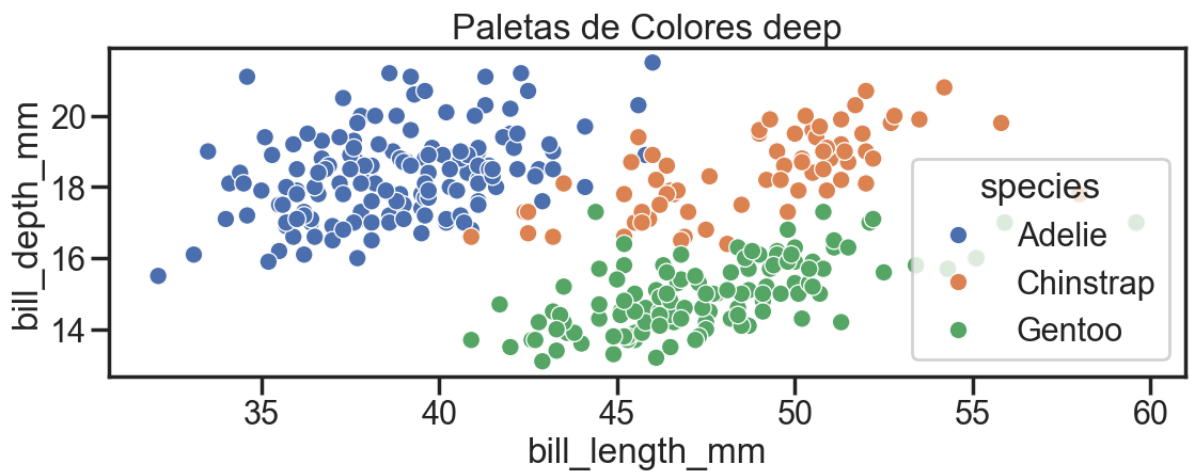
Personalización de colores en Seaborn

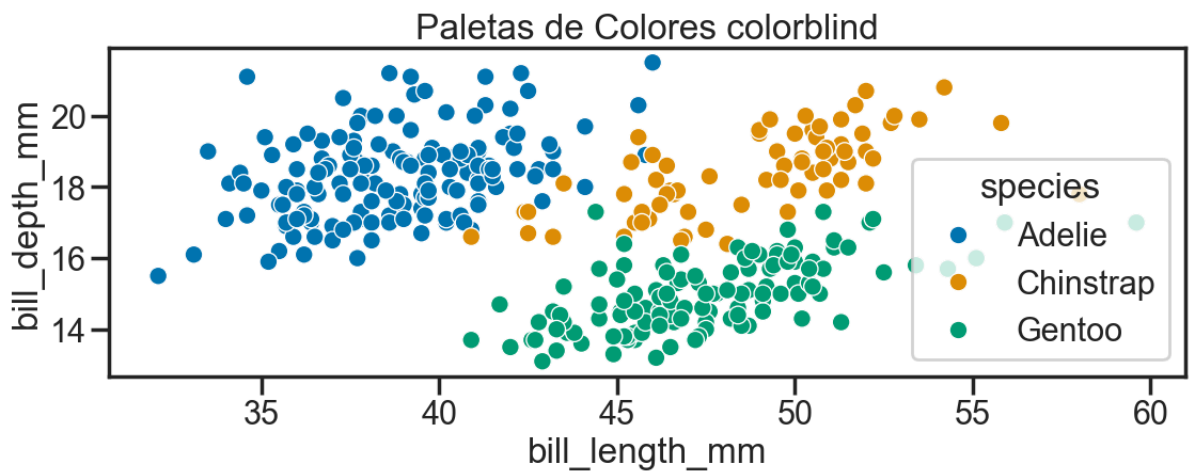
- Para mejorar la legibilidad y presentación de los gráficos.
- Se pueden usar paletas predefinidas, colores manuales o gradientes.

```
In [147...] df = sns.load_dataset('penguins')
```

```
In [134...] paletas = ['deep', 'muted', 'bright', 'dark', 'colorblind']

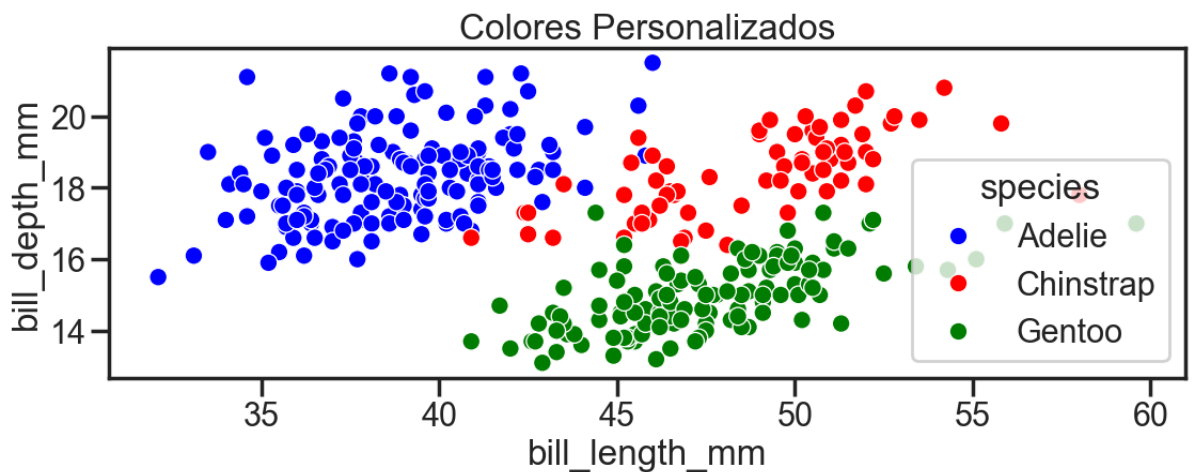
for paletas in paletas:
    plt.figure(figsize=(13, 4))
    sns.scatterplot(x='bill_length_mm', y='bill_depth_mm', hue='species', data=df, palette=paletas)
    plt.title(f'Paletas de Colores {paletas}')
    plt.show()
```





Definir colores manualmente

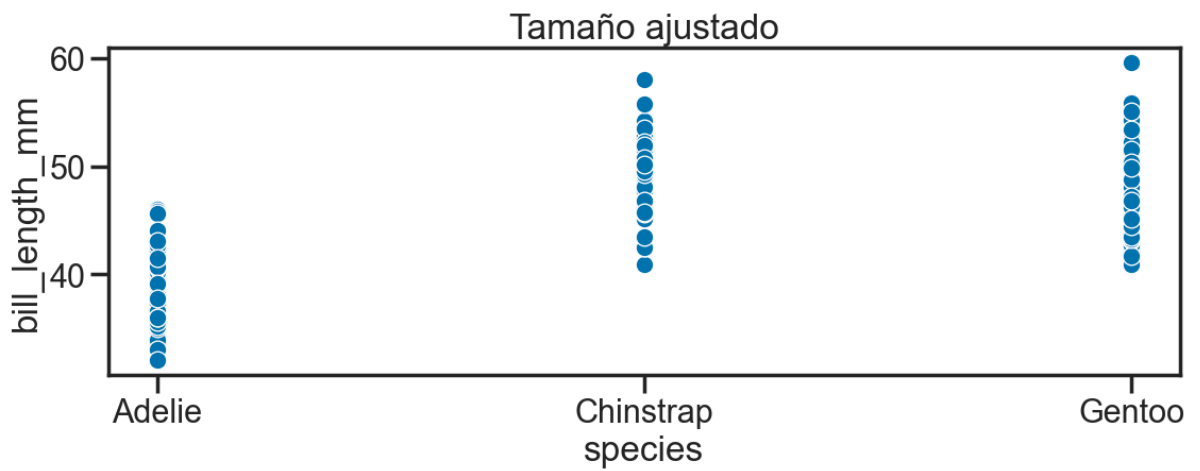
```
In [137... colores = {'Adelie': 'blue', 'Chinstrap': 'red', 'Gentoo': 'green'}
plt.figure(figsize=(13, 4))
sns.scatterplot(x='bill_length_mm', y='bill_depth_mm', hue='species', data=df, palette=
plt.title('Colores Personalizados')
plt.show()
```



Ajuste de tamaño y aspecto de gráficos

- Sirve para mejorar la visualización en diferentes contextos.

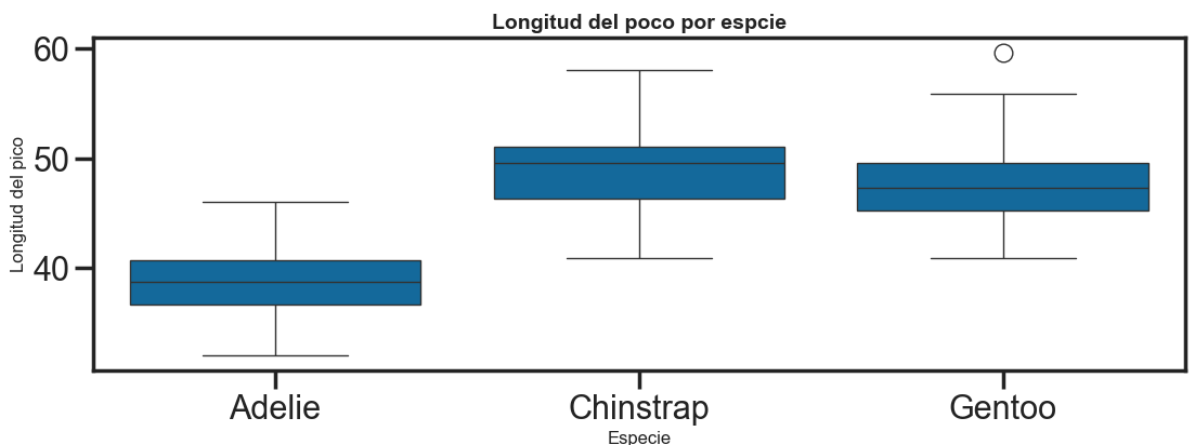
```
In [138... plt.figure(figsize=(13, 4))
sns.scatterplot(x='species', y='bill_length_mm', data=df)
plt.title('Tamaño ajustado')
plt.show()
```



Modificación de etiquetas y títulos

- Sirve para que los gráficos sea más comprensibles

```
In [139... plt.figure(figsize=(13, 4))
sns.boxplot(x='species', y='bill_length_mm', data=df)
plt.title('Longitud del poco por especie', fontsize=14, fontweight='bold')
plt.xlabel('Especie', fontsize=12)
plt.ylabel('Longitud del pico', fontsize=12)
plt.show()
```

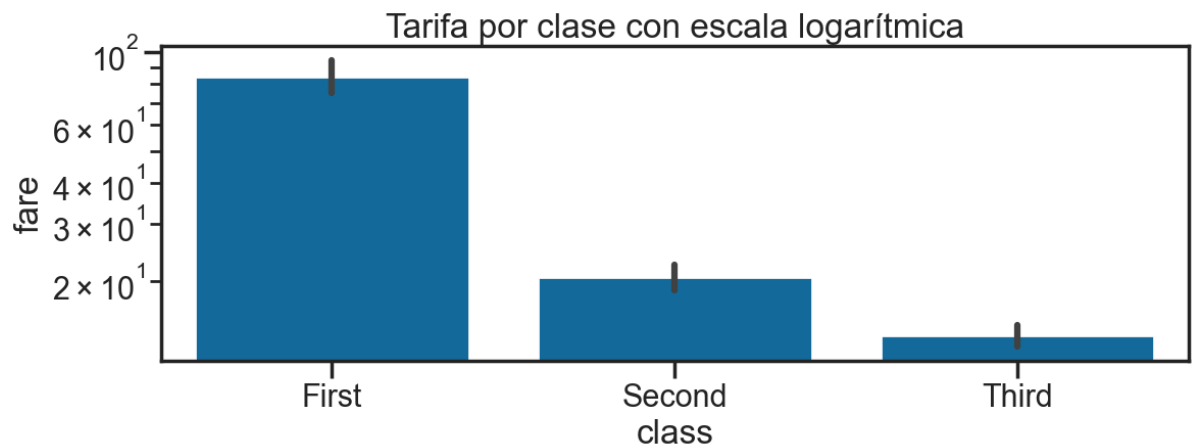


Escalas logarítmicas y normalización de datos

- Para visualizar datos con distribuciones amplias o valores muy grandes.

```
In [143... data=sns.load_dataset("titanic")
```

```
In [141... plt.figure(figsize=(13, 4))
sns.barplot(x='class', y='fare', data=data)
plt.yscale('log')
plt.title('Tarifa por clase con escala logarítmica')
plt.show()
```



Rotación de etiquetas en gráficos de barras

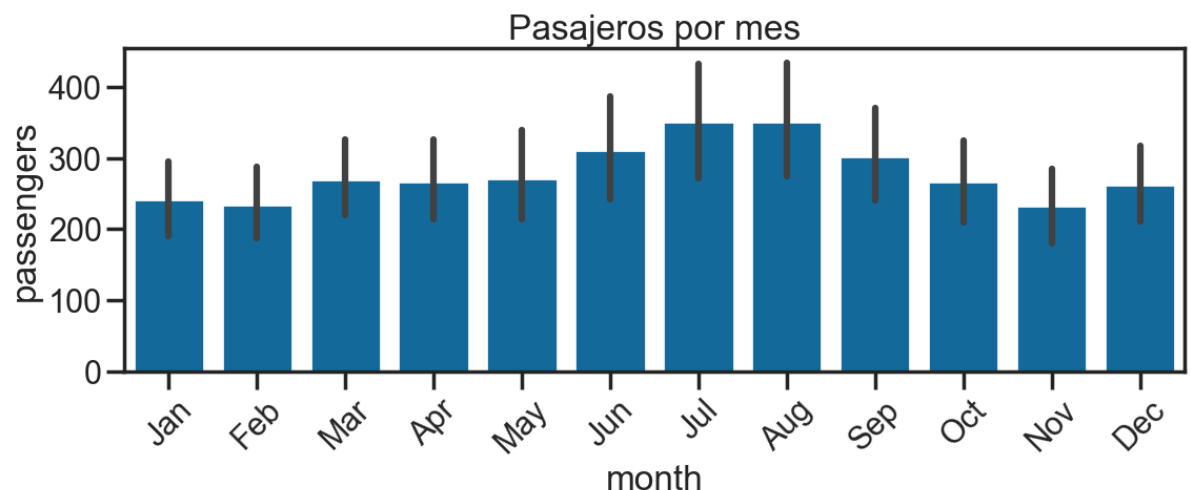
- Para mejorar la legibilidad cuando las etiquetas son largas

In [144...]

```
df=sns.load_dataset("flights")
```

In [146...]

```
plt.figure(figsize=(13, 4))
sns.barplot(x='month', y='passengers', data=df)
plt.xticks(rotation=45)
plt.title('Pasajeros por mes')
plt.show()
```

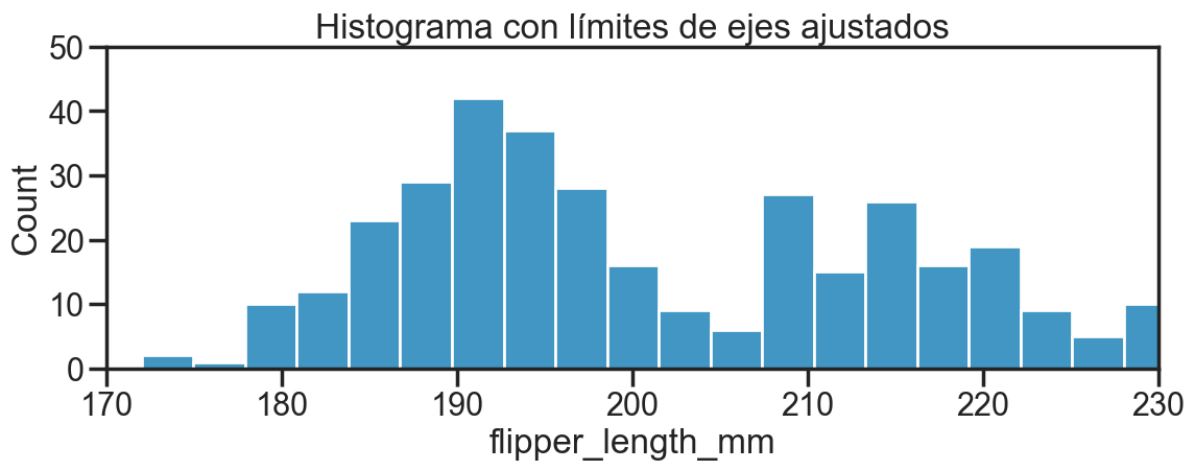


Ajuste de límites en ejes (xlim, ylim)

- Para enfocar un rango específico de datos de un gráfico
- xlim(min, max), ylim(min,max) permite enfocar la visualización en una rango específico

In [148...]

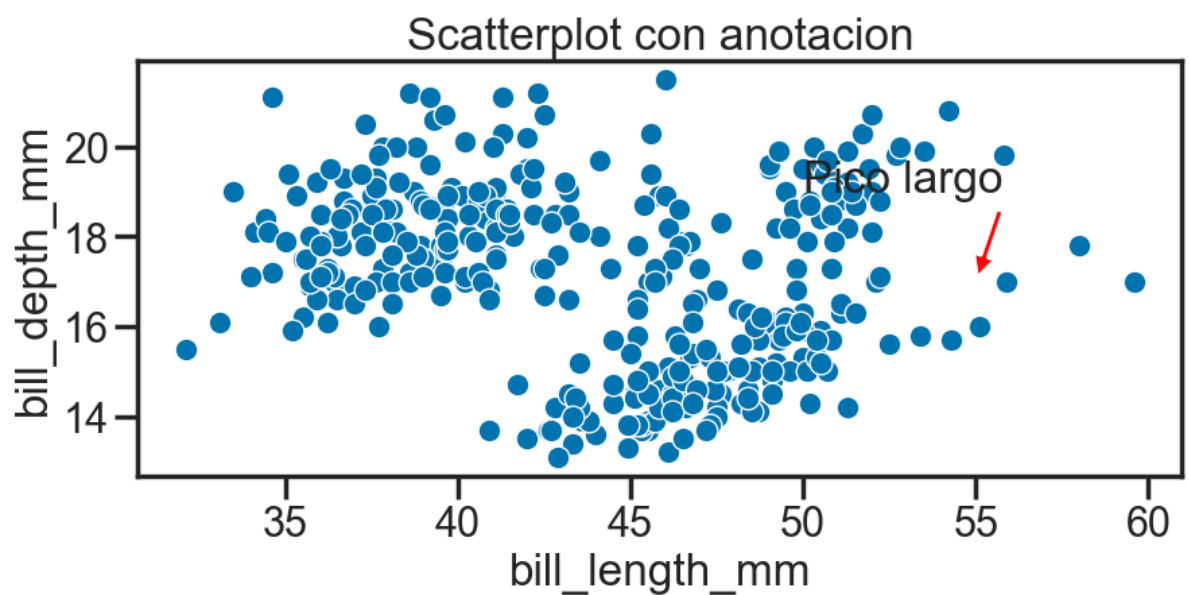
```
plt.figure(figsize=(13,4))
sns.histplot(df['flipper_length_mm'], bins=20)
plt.xlim(170,230)
plt.ylim(0, 50)
plt.title('Histograma con límites de ejes ajustados')
plt.show()
```



Agregar anotaciones manuales

- Para destacar valores importantes dentro de un gráfico

```
In [151... plt.figure(figsize=(10,4))
sns.scatterplot(x='bill_length_mm', y='bill_depth_mm', data=df)
plt.annotate('Pico largo', xy=(55,17), xytext=(50,19), arrowprops=dict(facecolor='red'))
plt.title('Scatterplot con anotacion')
plt.show()
```



Personalización por tipo gráfico

barplot		boxplot	
x="categoria", y="valor"	data=df	x="categoria", y="valor"	data=df
hue="categoria"	Variable para colorear puntos	hue="grupo"	Variable para colorear cajas
saturation=0.8	Saturación del color	palette="Set3"	Paleta de colores
style="grupo"	Variable para estilo de marcadores	saturation=0.75	Saturación del color
palette="viridis"	Paleta de colores	width=0.8	Ancho de las cajas
dodge=True	Si las barras se agrupan o apilan	linewidth=1	Grosor de línea

barplot		boxplot	
capsize=0.2	Tamaño de las barras de error	fliersize=5	Tamaño de los valores atípicos
errwidth=2	Grosor de las barras de error	whis=1.5	Longitud de los whiskers
markeredgecolor="black"	Color del borde del marcador	notch=False	Si se muestra una muesca
ci=95	Intervalo de confianza	showcaps=True	Si se muestran las tapas
		showfliers=True	Si se muestran los valores atípicos
		boxprops={'facecolor': 'lightblue'}	Propiedades de las cajas
		whiskerprops={'linestyle': '--'}	Propiedades de los whiskers
Scatterplot		lineplot	
x="x", y="y"	data=df	x="x", y="y"	data=df
hue="categoria"	Variable para colorear puntos	hue="categoria"	Variable para colorear puntos
size="valor"	Variable para tamaño de puntos	size="valor"	Variable para tamaño de puntos
style="grupo"	Variable para estilo de marcadores	style="grupo"	Variable para estilo de marcadores
palette="viridis"	Paleta de colores	palette="viridis"	Paleta de colores
sizes=(20, 200)	Rango de tamaños	dashes=[(2,2), (1,1)]	Patrones de línea personalizados
markers=["o", "s", "D"]	Tipos de marcadores	markers=["o", "^"]	Tipos de marcadores
alpha=0.7	Transparencia	markersize=10	Tamaño de marcadores
edgecolor="white"	Color del borde	markeredgecolor="black"	Color del borde del marcador
linewidth=0.5	Ancho de línea del borde	ci=95	Intervalo de confianza
violinplot	stripplot		swarmplot
x="categoria", y="valor"	data=df	x="categoria", y="valor"	data=df
hue="grupo"	Variable para colorear violines	hue="grupo"	Variable para colorear puntos
palette="muted"	Paleta de colores	palette="deep"	Paleta de colores
split=False	Si se dividen los violines por hue	size=5	Tamaño de los puntos

violinplot		stripplot		swarmplot
inner="quartile"	Representación interna	jitter=True		Si se añade ruido horizontal
scale="width"	Escala de los violines	dodge=True		Si los puntos se agrupan por hue
scale_hue=True	Si se escalan los violines por hue	alpha=0.7		Transparencia
gridsize=100	Precisión de la estimación de densidad KDE	marker="o"		Tipo de marcador
width=0.8	Ancho de los violines	edgecolor="gray"		Color del borde
linewidth=1	Grosor de línea	linewidth=0.5		Ancho de línea del borde
saturation=0.75	Saturación del color			
countplot		histplot		
x="categoria"	data=df	data=df	x="valor"	
hue="grupo"	Variable para colorear barras	hue="grupo"	Variable para colorear histogramas	
palette="pastel"	Paleta de colores	palette="Set1"	Paleta de colores	
saturation=0.75	Saturación del color	bins=20	Número de bins	
dodge=True	Si las barras se agrupan o apilan	binwidth=None	Ancho de bins	
		kde=True	Si se añade estimación de densidad KDE	
		stat="density"	Estadística	
		cumulative=False	Si el histograma es acumulativo	
		multiple="layer"	Cómo mostrar múltiples distribuciones	
		alpha=0.5	Transparencia	
		linewidth=0	Ancho de línea del borde	
		element="bars"	Tipo de elemento	
		fill=True	Si se rellenan las barras	

Ejercicio Práctico

- Vamos a hacer exploración de un conjunto de datos con Seaborn
- Cargar un dataset y generar gráficos básicos para explorarlo.
- Cargaremos el dataset penguins de Seaborn
- Mostraremos las primeras filas
- Generaremos un histograma para visualizar la distribución de la variable body_mass_g
- Usaremos un boxplot para ver cómo varía la masa corporal según la especie

```
In [24]: df = sns.load_dataset("penguins")
data = sns.load_dataset("penguins")
```



```
In [13]: print(df.info())
```

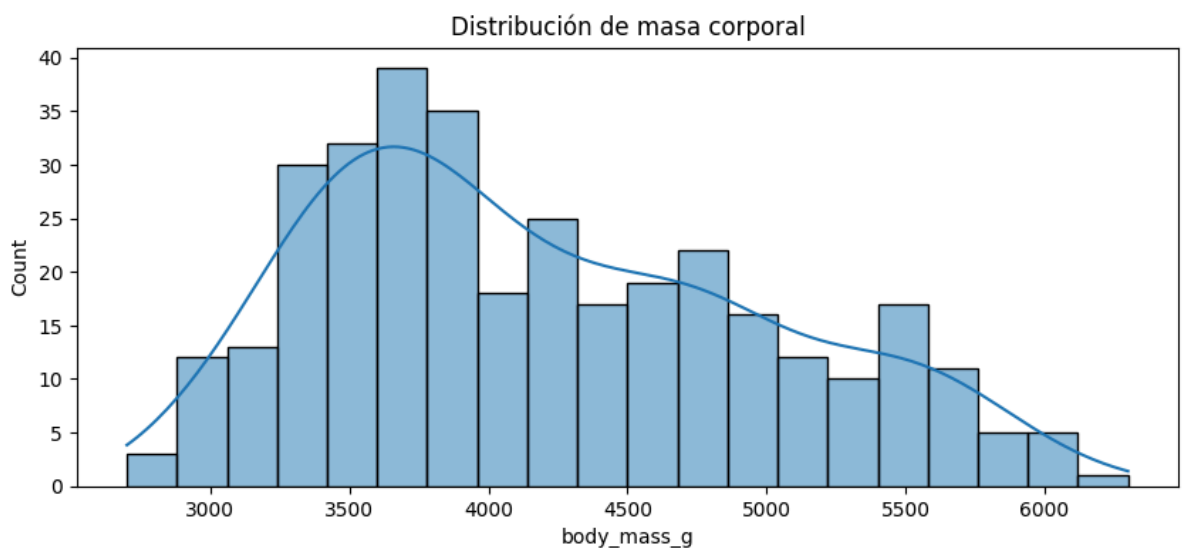
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   species               344 non-null   object 
1   island                344 non-null   object 
2   bill_length_mm        342 non-null   float64
3   bill_depth_mm         342 non-null   float64
4   flipper_length_mm     342 non-null   float64
5   body_mass_g           342 non-null   float64
6   sex                   333 non-null   object 
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
None
```

```
In [14]: print(df.head())
```

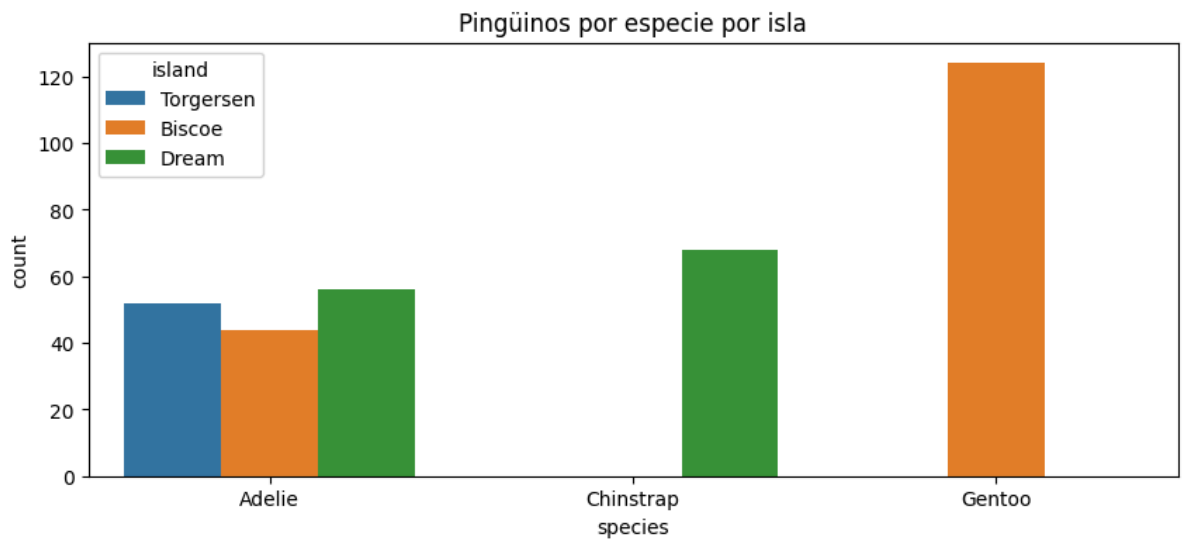
```
   species  island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
0  Adelie  Torgersen             39.1           18.7             181.0  
1  Adelie  Torgersen             39.5           17.4             186.0  
2  Adelie  Torgersen             40.3           18.0             195.0  
3  Adelie  Torgersen             NaN            NaN              NaN  
4  Adelie  Torgersen             36.7           19.3             193.0  

   body_mass_g  sex
0      3750.0  Male
1      3800.0  Female
2      3250.0  Female
3         NaN   NaN
4      3450.0  Female
```

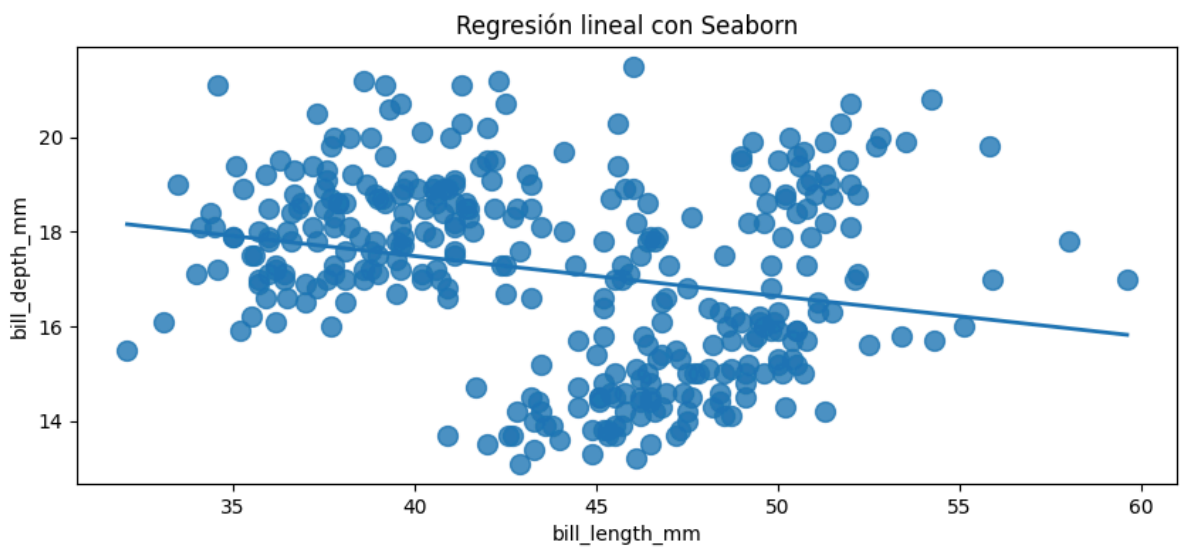
```
In [17]: plt.figure(figsize=(10,4))
sns.histplot(df['body_mass_g'], bins=20, kde=True)
plt.title('Distribución de masa corporal')
plt.show()
```



```
In [19]: plt.figure(figsize=(10,4))
sns.countplot(x='species', hue='island', data=df)
plt.title('Pingüinos por especie por isla')
plt.show()
```



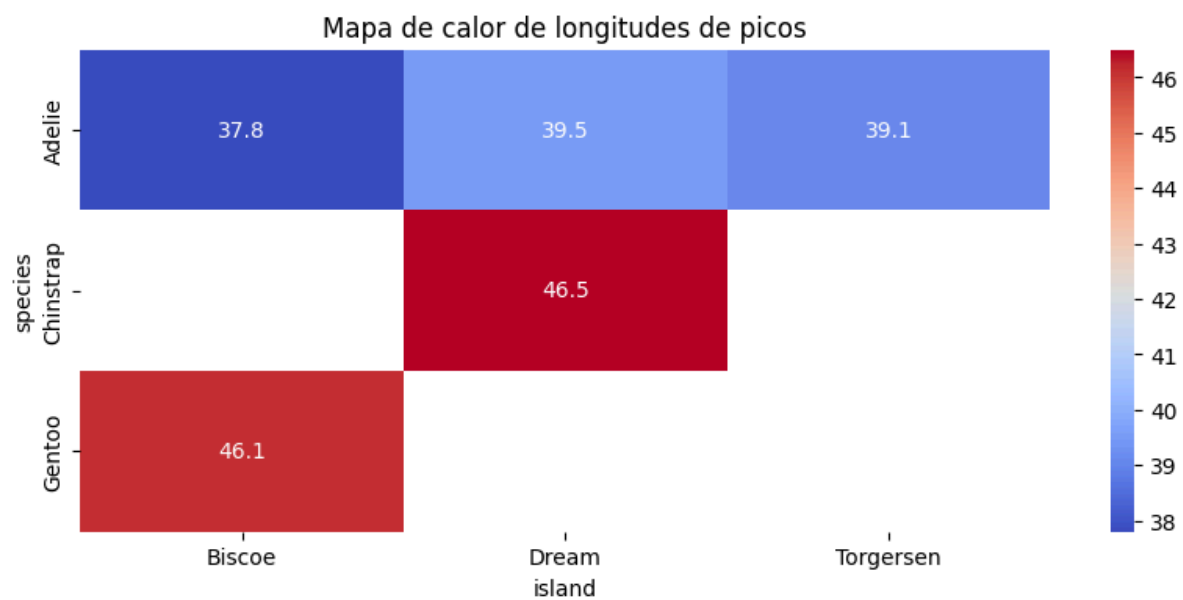
```
In [23]: plt.figure(figsize=(10,4))
sns.regplot(
    data=df,
    x='bill_length_mm',
    y='bill_depth_mm',
    scatter_kws={'s':100},
    line_kws={'lw':2},
    ci=None
)
plt.title('Regresión lineal con Seaborn')
plt.show()
```



```
In [27]: data_unique = data.drop_duplicates(subset=['species', 'island'])

mapa = (
    data_unique
    .pivot(index='species', columns='island', values='bill_length_mm')
)

f, ax = plt.subplots(figsize=(10,4))
sns.heatmap(mapa, annot=True, fmt='.1f', cmap='coolwarm', ax=ax)
plt.title('Mapa de calor de longitudes de picos')
plt.show()
```



Por Edurdo Soto ING de Software