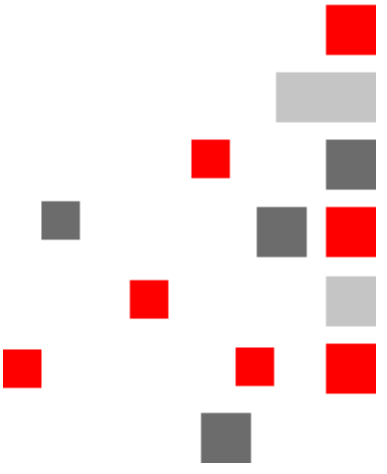
universidad de león

Exploración y visualización de datos con Python

Métodos Numéricos y Estadísticos
Grado en Ingeniería Informática / Mecánica

Curso 2023-2024

Eva María Mazcuñán Navarro



Eva María Mazcuñán Navarro
Departamento de Matemáticas
Universidad de León
E-mail: emmazn@unileon.es

Contenidos

Página

Inicio **1**

Introducción **1**
Los pingüinos del archipiélago Palmer 1
Objetivos 2

1 Librerías **2**
1.1 `pandas` 2
1.2 `seaborn` 2

2 Datos **3**
2.1 Importar los datos 4
2.2 Dimensiones 4
2.3 Primeras y últimas filas 4
2.4 Estructura 5
2.5 Variables 6

3 Seleccionar variables **10**
3.1 Seleccionar una variable 10
3.2 Seleccionar una lista de variables 11

4 Una variable categórica **12**
4.1 El método `describe()` 12
4.2 Tabla de recuentos 13
4.3 Diagrama de recuentos 14
4.4 Personalización de los gráficos (opcional) 16

5 Una variable numérica **21**
5.1 El método `describe()` 21
5.2 Histograma 22
5.3 Diagrama de caja y bigotes 23

6 Agrupar por categorías **24**
Ejemplo 1: Primeras filas de cada especie 25
Ejemplo 2: Peso máximo de cada especie 27
Ejemplo 3: Recuento del número de pingüinos de cada especie 28

7 Una variable numérica por categorías **29**
7.1 El método `describe()` 29
7.2 Histogramas 31
7.3 Diagramas de caja y bigotes 39

Bibliografía **43**

Introducción

En esta práctica aprenderás las técnicas básicas para explorar y visualizar un conjunto de datos con Python.

Los pingüinos del archipiélago Palmer

Presentaremos las diferentes técnicas a través de ejemplos trabajando con un conjunto de datos relativos a diferentes características de tres especies de pingüinos del archipiélago Palmer en la Antártida.

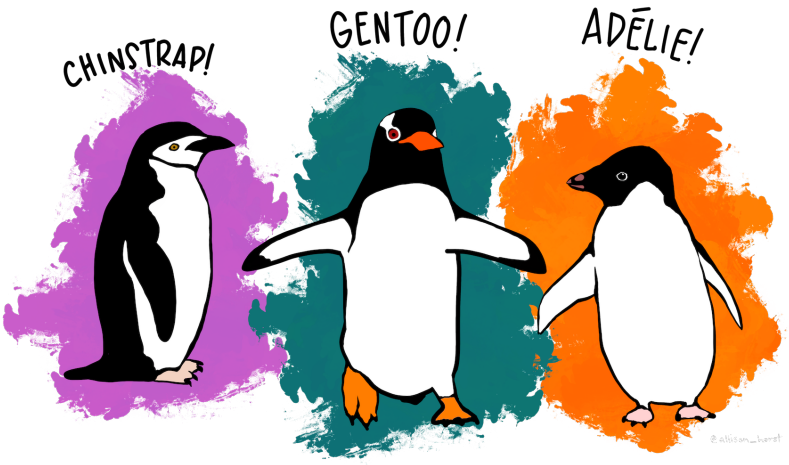


Figura 1: Ilustración de las tres especies de pingüinos del archipiélago Palmer (Artista @allison_horst)

Los datos fueron originalmente publicados en Gorman, Williams y Fraser [1]. Este conjunto de datos se hizo popular a partir de la creación del paquete palmerpenguins de R. Hoy en día los datos de los pingüinos del archipiélago Palmer se usan de forma extendida para ilustrar las técnicas de exploración y visualización de datos no solo en R, sino en muchos otros lenguajes de programación para estadística y ciencia de datos, como Python. Nosotros accederemos a los datos a través de este enlace, que proporciona los datos en formato CSV (*comma separated values*). No tienes que descargar el archivo, más adelante se explica cómo importarlo directamente de la web.

Objetivos

Aprenderás en concreto a calcular las medidas descriptivas más representativas de las características de interés y a crear diferentes tipos de gráficos o visualizaciones.

1. Librerías

Lo primero que necesitamos hacer es importar las librerías de Python que usaremos a lo largo de práctica, que son `pandas` y `seaborn`:

```
import pandas as pd
import seaborn as sns
```

1.1. `pandas`

`pandas` es una librería que permite leer datos almacenados en estructuras similares a una tabla, como las hojas de cálculo o los archivos CSV, y proporciona métodos para explorar y describir esos datos.

Usando esta librería podremos calcular por ejemplo el peso medio de los pingüinos de cada especie. Obtendremos esta tabla:

```
species
Adelie       3700.662252
Chinstrap    3733.088235
Gentoo       5076.016260
Name: body_mass_g, dtype: float64
```

Puedes consultar la documentación oficial de `pandas` aquí.

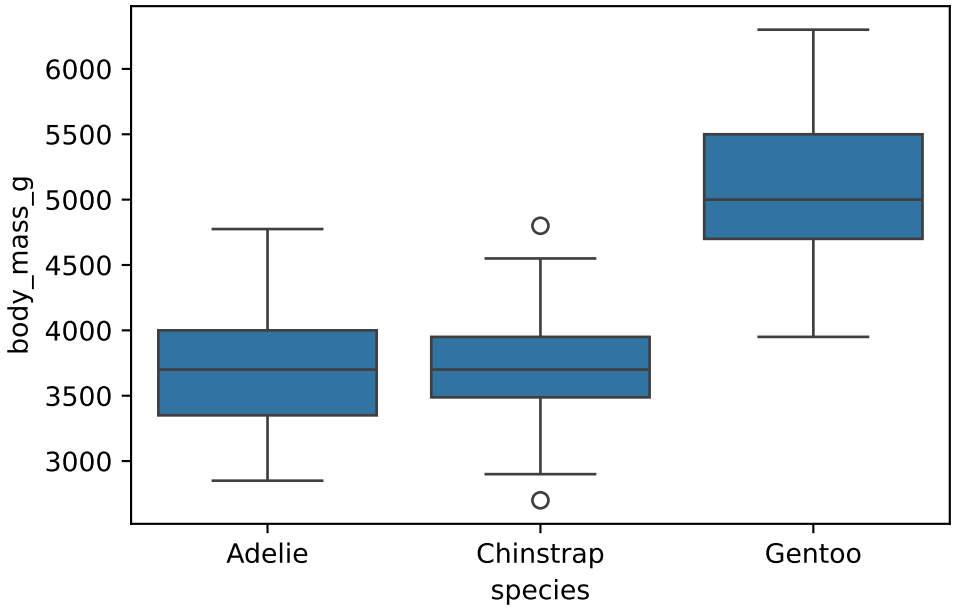
1.2. `seaborn`

`seaborn` es una librería para visualización de datos. Permite crear gráficos estadísticos muy informativos y visualmente atractivos con pocas líneas de código. Y está diseñado para trabajar con las estructuras de datos creadas con `pandas`.

El nombre de la librería y el alias canónico `sns` para importarla hacen referencia a *Samuel Norman Seaborn*, personaje de la serie de televisión *The West Wing* interpretado por Rob Lowe (lo explica el autor de la librería en la pregunta Why is seaborn imported as `sns` de la sección *FAQ* de la documentación).

Usaremos `seaborn` para realizar diferentes tipos de gráficos como diagramas de barras, histogramas, diagramas de caja y bigotes etc.

Aprenderemos por ejemplo a crear el siguiente gráfico, con los diagramas de caja y bigotes para el peso de los pingüinos de cada especie.



Puedes consultar la documentación oficial de `seaborn` aquí.

2. Datos

En esta sección importarás los datos sobre los pingüinos del archipiélago Palmer presentados en la introducción y conocerás la información que contienen.

2.1. Importar los datos

Como se indicó en la introducción, los datos con los que vamos a trabajar están disponibles en la web en un fichero de formato `CSV`.

Ejecuta las instrucciones a continuación para importar el archivo usando la función `read_csv()` y guardar el resultado en una variable de nombre `penguins`:

```
url =
↪  "https://raw.githubusercontent.com/mwaskom/seaborn-data/master/penguins.csv"
penguins = pd.read_csv(url)
```

El objeto `penguins` que acabas de crear es una **hoja de datos**, representada en `pandas` con la clase `DataFrame`.

```
type(penguins)
```

```
pandas.core.frame.DataFrame
```

En los siguientes apartados aprenderás a realizar una exploración inicial de la hoja de datos `penguins` que acabas de crear para conocer su estructura y la información que contiene.

2.2. Dimensiones

Una hoja de datos es una estructura matricial o tabular que contiene datos organizados por filas y columnas.

Para saber las dimensiones de nuestra hoja de datos `penguins` consulta su propiedad `shape`:

```
penguins.shape
```

```
(344, 7)
```

Vemos que nuestra hoja de datos tiene `344` filas y `7` columnas.

2.3. Primeras y últimas filas

Con las siguientes instrucciones puedes visualizar las cinco primeras y últimas filas de la hoja de datos `penguins` que acabas de crear.

```
penguins.head(5)
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0
3	Adelie	Torgersen	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0

```
penguins.tail(5)
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
339	Gentoo	Biscoe	NaN	NaN	NaN	NaN
340	Gentoo	Biscoe	46.8	14.3	215.0	4850.0
341	Gentoo	Biscoe	50.4	15.7	222.0	5750.0
342	Gentoo	Biscoe	45.2	14.8	212.0	5200.0
343	Gentoo	Biscoe	49.9	16.1	213.0	5400.0

2.4. Estructura

En nuestra hoja de datos `penguins`:

- Cada columna representa una variable asociada a una propiedad o característica de los pingüinos. Por ejemplo, la primera columna, de nombre `species` indica la especie (Chinstrap, Adélie o Gentoo) de pingüino. En el siguiente apartado se describen las otras seis variables.
- Cada fila se corresponde con un pingüino concreto de los 344 seleccionados en el estudio.
- Cada celda contiene el valor de la característica del pingüino en la correspondiente fila.

Por ejemplo, mirando la primera fila de la hoja de datos en la salida de la instrucción anterior `penguins.head(5)`

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0

vemos en la primera celda que el primer pingüino del listado es de la especie Adelie.

Unos mismos datos pueden organizarse o presentarse de diferentes maneras en diferentes hojas de datos. Para que sea sencillo trabajar con una hoja de datos es conveniente que haya una relación clara entre su significado y su estructura. Se considera que la hoja de datos está *ordenada* o *limpia* (en inglés se habla de *tidy data*) si está organizada de acuerdo con los siguientes principios:

- Cada **columna** representa una **variable** o característica de interés.
- Cada **fila** representa una **observación**, caso o unidad experimental.
- Cada **celda** contiene un **valor**, el de la variable en la correspondiente columna para la observación en la correspondiente fila.

De acuerdo con la descripción inicial, nuestra hoja de datos cumple con los principios anteriores.

2.5. Variables

Ejecuta la siguiente instrucción:

```
penguins.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
```

La salida del método `info()` nos da una tabla con información sobre las siete variables de nuestra hoja de datos.

2.5.1. Descripción

En la columna de la tabla de nombre `Column` se lista el nombre de las siete variables en `penguins`. El significado de las variables es el siguiente:

Nombre	Descripción
`species`	Especie de pingüinos (Chinstrap, Adélie o Gentoo)
`island`	Nombre de la isla del archipíelago Palmer (Dream, Torgersen o Biscoe)
`bill_length_mm`	Longitud del pico, en milímetros (ver Figura 2)
`bill_depth_mm`	Anchura del pico, en milímetros (ver Figura 2)
`flipper_length_mm`	Longitud de las alas
`body_mass_g`	Peso en gramos
`sex`	Sexo (MALE o FEMALE)

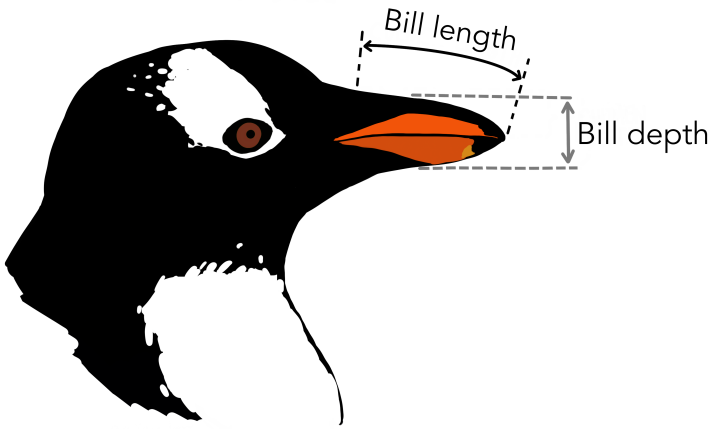


Figura 2: Ilustración de las variables `bill_length_mm` y `bill_depth_mm` (Artista @allison_horst)

Volviendo a mirar la primera fila de nuestra hoja de datos

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0

ahora sabes que el primer pingüino es de la especie Adelie, vive en la isla Torgensen, las dimensiones de su pico son 39.1×18.7 milímetros, sus alas miden 181 milímetros, pesa 3 kilos y 750 gramos, y es un macho.

Problema 1

Describe las características del tercer pingüino del estudio (índice 2) mirando la salida de la instrucción anterior `penguins.head(5)`.

2.5.2. Valores nulos

Fíjate ahora en la columna `Non-Null Count` de la salida del método `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
```

Los valores nulos o perdidos son valores no disponibles, que no han podido registrarse, se representan con el símbolo `NaN` (iniciales de *Not A Number*).

Si vuelves a mirar las cinco primeras filas de la hoja de datos verás que para el cuarto pingüino sólo sabemos que es de la especie Adelie y vive el la isla Torgersen, y se desconocen las otras cinco variables.

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_
3	Adelie	Torgersen	NaN	NaN	NaN	NaN

Los diferentes métodos de la librería `pandas` contemplan la posibilidad de que las hojas de datos tengan valores nulos y los tratan de un forma predeterminada (por ejemplo, la media los ignora por defecto).

2.5.3. Tipos de variables

Las siete variables de nuestra hoja de datos se dividen en dos clases:

1. `species`, `island` y `sex` son **variables categóricas**. Este tipo de variables representan una característica cualitativa que puede tomar un número finito y fijo de valores, denominados categorías o niveles.
2. Las cuatro restantes, `bill_length_mm`, `bill_depth_mm`, `flipper_length_mm` y `body_mass_g` son **variables numéricas**, que representan características cuantitativas que se describen con valores numéricos (números enteros o reales).

`pandas` asigna un tipo a cada variable de una hoja de datos en función de los valores que presenta, como puede verse en la columna `Dtype` de la salida del método `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
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0

“‘ ::: ::: ::: {.cell execution_count=14} ::: {.cell-output .cell-output-display execution_count=14} “‘{=html}

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_
341	Gentoo	Biscoe	50.4	15.7	222.0	5750.0
342	Gentoo	Biscoe	45.2	14.8	212.0	5200.0
343	Gentoo	Biscoe	49.9	16.1	213.0	5400.0

3. Seleccionar variables

Al estudiar un conjunto de datos, es frecuente tener que seleccionar los datos relevantes para responder a las diferentes cuestiones planteadas.

Si por ejemplo queremos saber cuál es el peso máximo de todos los pingüinos del estudio, seleccionaremos la variable `body_mass_g` y después calcularemos su máximo.

En esta sección aprenderás los métodos para seleccionar variables de una hoja de datos.

3.1. Seleccionar una variable

Utiliza la siguiente instrucción para seleccionar la variable `body_mass_g`:

```
mass = penguins["body_mass_g"]
```

> **i** Para seleccionar una sola variable, usa corchetes `[]` e indica el nombre de la columna de interés.

Ahora podemos aplicar la función `max()` para obtener el peso máximo:

```
mass.max()
```

```
6300.0
```

Vemos que el pingüino más pesado del estudio pesa 6 kilos y 300 gramos.

Podemos realizar las dos operaciones anteriores, seleccionar la variable `body_mass_g`, y calcular su máximo con una sola instrucción:

```
penguins["body_mass_g"].max()
```

```
6300.0
```

Obtenemos el mismo resultado de antes.

Problema 2

Calcula el peso medio de todos los pingüinos (función `mean()`).

Problema 3

Calcula el valor mínimo para la longitud de las alas de todos los pingüinos (función `min()`).

3.2. Seleccionar una lista de variables

Para seleccionar las dos variables relativas a las dimensiones del pico, `bill_length_mm` y `bill_depth_mm`, ejecuta la siguiente instrucción:

```
bill = penguins[["bill_length_mm", "bill_depth_mm"]]
```

> **i** Para seleccionar una lista de variables, usa corchetes `[]` adicionales para crear la lista con los nombres de las columnas de interés (los corchetes exteriores indican que se van a seleccionar datos y los interiores crean la lista).

Ahora podemos calcular la media para ambas variables con

```
bill.mean()
```

```
bill_length_mm    43.92193
bill_depth_mm     17.15117
```

```
dtype: float64
```

Vemos que los picos de los pingüinos tienen una longitud media de 43.92 milímetros y una anchura media de 17.15 milímetros.

Problema 4

Calcula el número de observaciones no nulas (función `count()`) para las variables `species` y `body_mass_g` con una sola línea de código.

4. Una variable categórica

En este apartado se describen los métodos básicos para explorar y analizar una variable categórica.

Analizaremos en concreto la variable `species` de nuestra hoja de datos, para conocer la distribución de los pingüinos por especies.

Comenzamos seleccionando la variable de interés en nuestra hoja de datos. Almacenamos el resultado en un nuevo objeto de nombre `species`.

```
species = penguins["species"]
```

4.1. El método `describe()`

Para obtener la información más relevante sobre la distribución de las especies de pingüinos aplicamos a `species` el método `describe()`:

```
species.describe()
```

```
count        344
unique         3
top       Adelie
freq         152
Name: species, dtype: object
```

De la salida anterior obtenemos la siguiente información sobre la variable `species`:

Fragmento de la salida	Significado
`count 344`	Hay 344 valores no nulos, así que se conoce la especie de todos los pingüinos del estudio.
`unique 3`	La variable toma tres valores diferentes, es decir, hay tres categorías (las tres especies Adelie, Chinstrap y Gentooo).
`top Adelie`	La categoría `top` o más frecuente, es decir, la especie más numerosa es la especie Adelie.
`freq 152`	El número de pingüinos de la especie Adelie (la categoría `top`) es 152.

Problema 5

Utiliza el método `describe()` para obtener información sobre la distribución de los pingüinos por islas, y responde a las siguiente pregunta: ¿Cuál es la isla más poblada y cuántos pingüinos viven en ella?

4.2. Tabla de recuentos

Ya sabemos que la especie más numerosa es Adelie, con 152 pingüinos, pero aún no sabemos cuántos pingüinos hay de las otras dos especies, Chinstrap y Gentoo. Para obtener una tabla con el número de pingüinos de cada especie usamos la función `value_counts()`:

```
species.value_counts()
```

```
species
Adelie       152
Gentoo       124
Chinstrap     68
Name: count, dtype: int64
```

Ahora sabemos que la segunda especie más numerosa es Gentoo, con 124 ejemplares, y que solo hay 68 pingüinos de la especie Chinstrap.

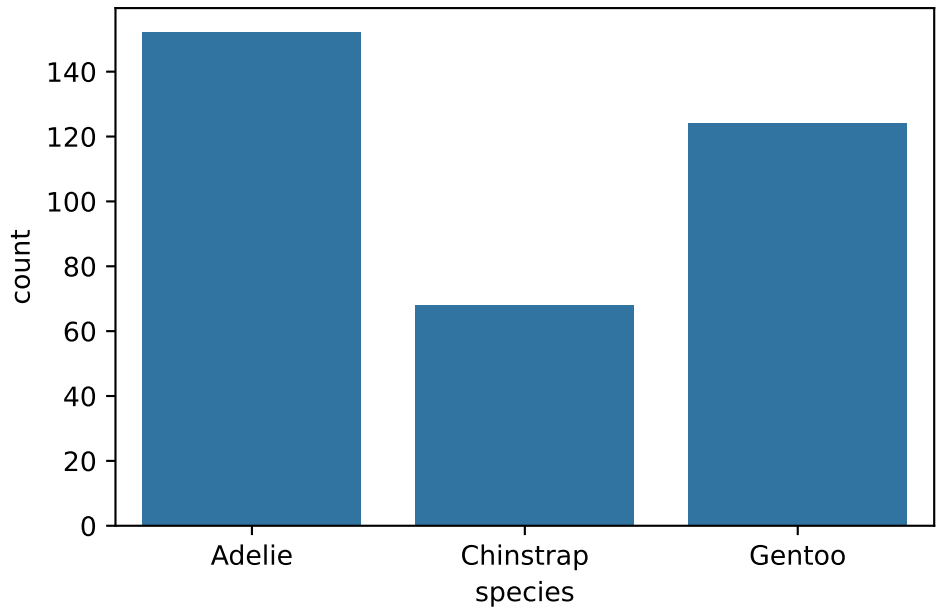
Problema 6
Determina el número total de hembras y de machos.

4.3. Diagrama de recuentos

Un **diagrama de recuentos** asocia a cada categoría una barra de longitud igual al número de observaciones en esa categoría.

Para realizar un diagrama del recuento de pingüinos por especie usamos la función `countplot()` del paquete `seaborn`

```
sns.countplot(data=penguins, x="species");
```

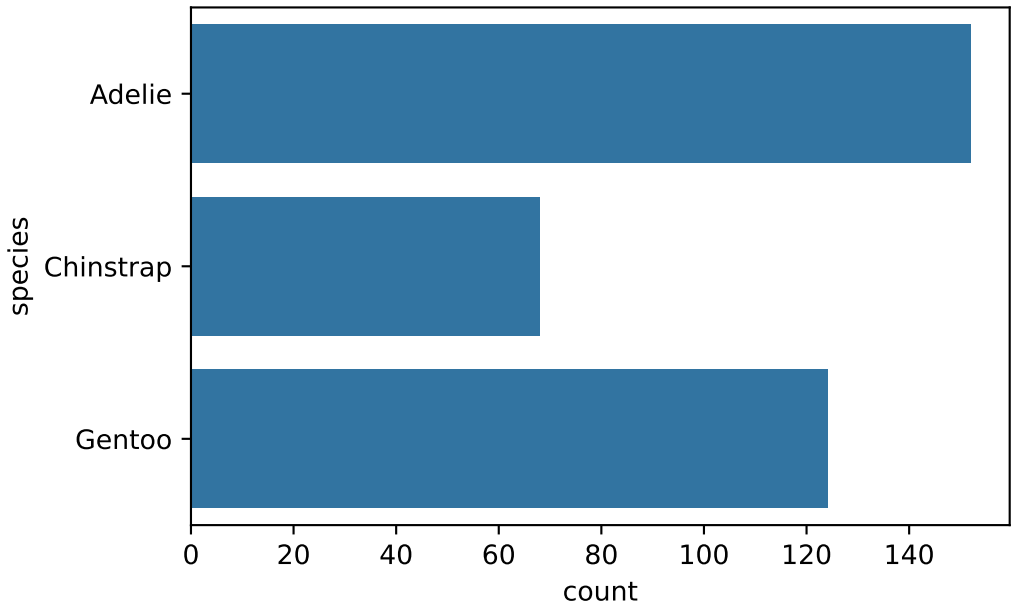


> Si quitas el punto y coma `;` al final de la instrucción anterior aparecerá el mensaje
> `<AxesSubplot:xlabel='count', ylabel='species'>`
> en la salida antes del gráfico.
> Al terminar una instrucción con punto y coma `;` se inhibe la impresión de la salida.

Nota que las alturas de las barras en el gráfico anterior coindicen con los recuentos que hemos calculado en el apartado anterior con `value_counts()`.

A veces es preferible usar barras horizontales, porque se tiene más espacio para las etiquetas de las categorías.

```
sns.countplot(data=penguins, y="species");
```



> Para realizar un diagrama de recuentos de las categorías de una variable categórica, usa la función `countplot()` de`seaborn`. E indica:
> - El nombre de la hoja de datos como valor del argumento `data`.
> - El nombre de la variable categórica como valor del argumento `x` si quieres barras verticales, o como valor del argumento `y` si quieres barras horizontales.

Problema 7

Crea un diagrama de recuentos para el número de hembras y machos, con barras verticales.

Problema 8

Crea un diagrama de recuentos para el número de pingüinos en cada isla, con barras horizontales.

4.4. Personalización de los gráficos (opcional)

Los gráficos de la librería `seaborn` admiten muchas opciones para personalizar su aspecto cambiando por ejemplo los colores, los rótulos de los ejes, etc.

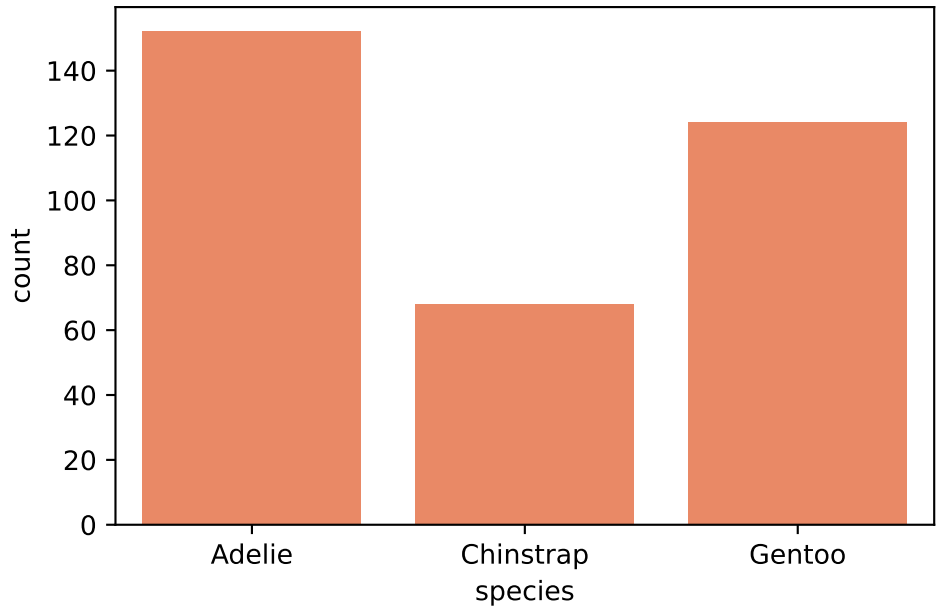
La personalización de los gráficos no carece de importancia, siendo especialmente relevante dar títulos descriptivos a los ejes. Pero en esta práctica nos centraremos en los procedimientos para realizar los gráficos y en la mayoría de ocasiones omitiremos los detalles de personalización de los mismos, que pueden consultarse en la documentación de `seaborn`.

En este apartado se da una muestra de las opciones para personalizar los diagramas de recuentos que se han presentado en el apartado anterior. Se trata de un apartado opcional y puedes por ahora omitir su lectura y volver a él al final de la práctica.

4.4.1. Colores

Si quieres cambiar el color de las barras usa el argumento `color`:

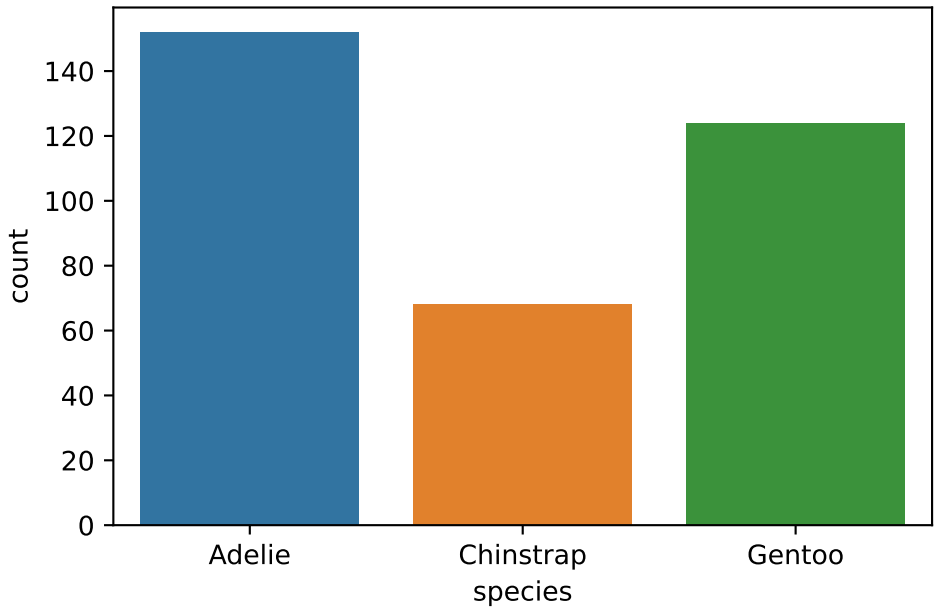
```
sns.countplot(data=penguins, x="species", color="coral");
```



Puedes ver los colores disponibles aquí.

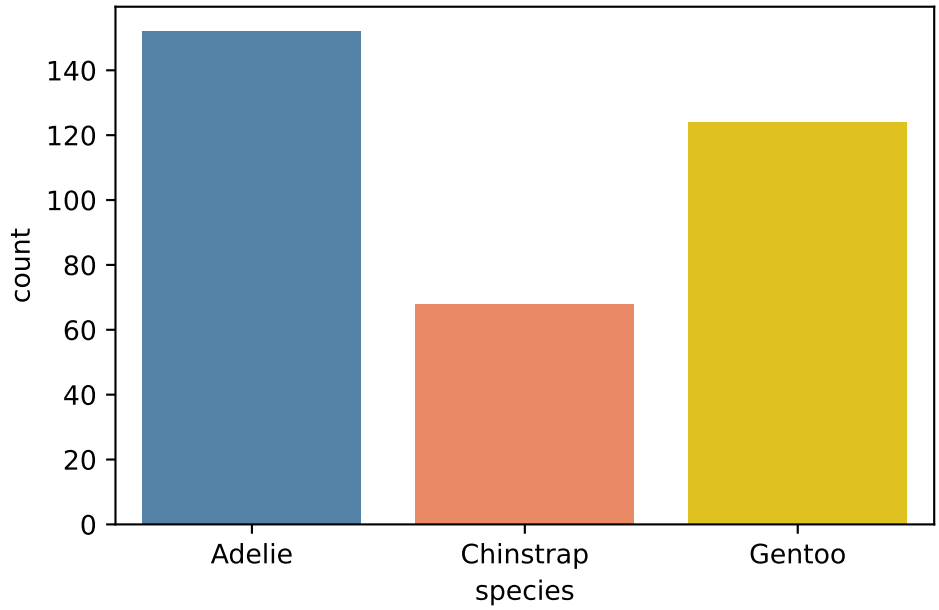
Para utilizar un color diferente para la barra de cada categoría usa el argumento `hue`:

```
sns.countplot(data=penguins, x="species", hue="species");
```



También se puede indicar una secuencia de colores personalizada usando el argumento `palette`:

```
sns.countplot(data=penguins, x="species", hue="species",
↪  palette=["steelblue", "coral", "gold"]);
```



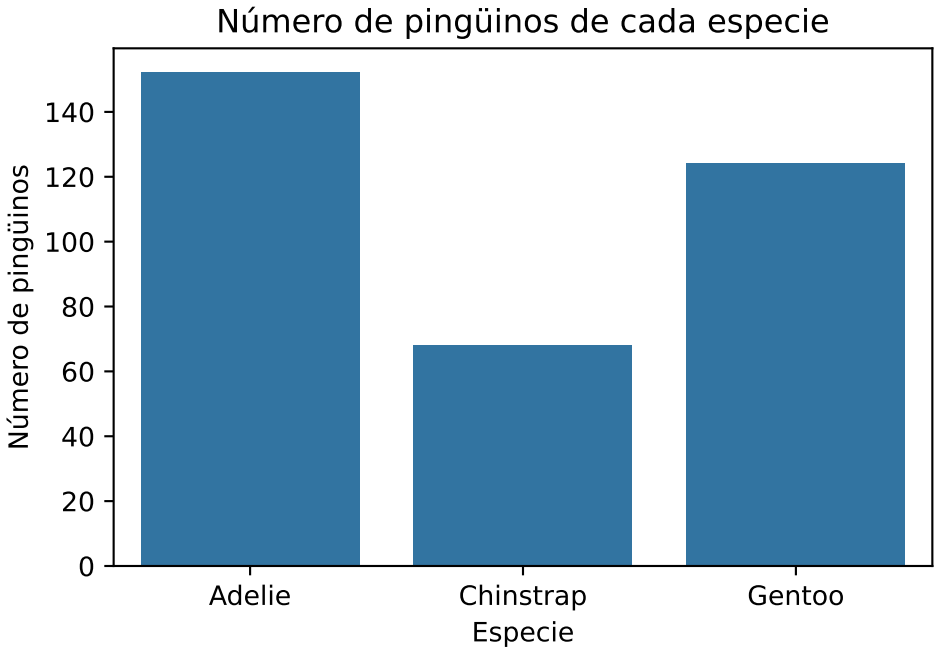
Como valor para `palette` se puede indicar una lista de colores, como en el ejemplo anterior, o el nombre de una de las paletes predefinidas (`deep`, `muted`, `pastel`, `bright`, `dark` y `colorblind`), como en el siguiente ejemplo:

```
sns.countplot(data=penguins, x="species", hue="species",
↪  palette="colorblind");
```

4.4.2. Rótulos

Con el siguiente código personalizamos los títulos de los ejes y damos un título global al gráfico

```
ax = sns.countplot(data=penguins, x="species")
ax.set(
    title="Número de pingüinos de cada especie",
    xlabel="Especie",
    ylabel="Número de pingüinos"
);
```

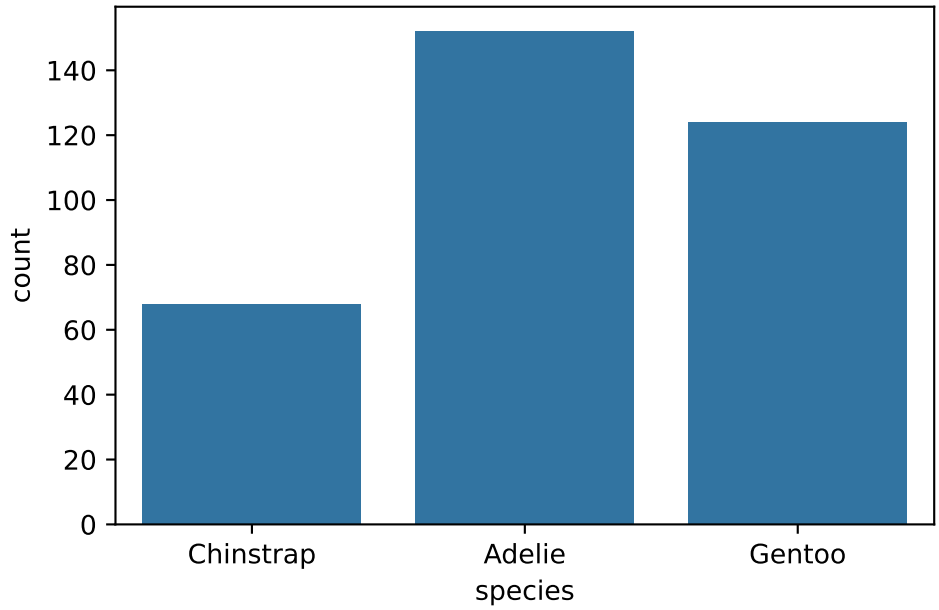


4.4.3. Orden de los niveles

Si te fijas en los gráficos que has hecho hasta ahora, apreciarás que las categorías aparecen en los gráficos en el mismo orden en el que aparecen en las filas de la hoja de datos.

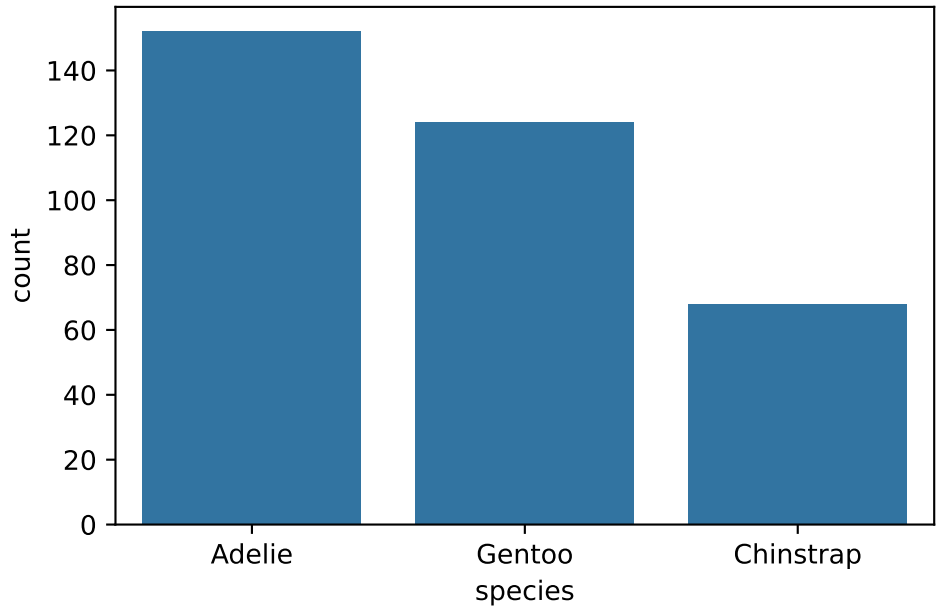
Si quieres un orden personalizado para las categorías puedes usar el argumento `order`, como en el siguiente ejemplo:

```
sns.countplot(data=penguins, x="species", order = ['Chinstrap',
↪  'Adelie', 'Gentoo']);
```



Con el siguiente código ordenamos las categorías de mayor a menor frecuencia.

```
sns.countplot(data=penguins, x="species", order =
↪  species.value_counts().index);
```



5. Una variable numérica

Igual que en el apartado anterior estudiamos una variable categórica, en este apartado se describen los métodos básicos para explorar y analizar una variable numérica.

Analizaremos en concreto la variable `body_mass_g` de nuestra hoja de datos, para conocer la distribución de los pesos de los pingüinos.

Empezamos seleccionando la variable de interés y almacenando el resultado en un nuevo objeto de nombre `mass`.

```
mass = penguins["body_mass_g"]
```

5.1. El método `describe()`

El método `describe()` que usamos en la sección anterior para una variable categórica, también puede aplicarse a variables numéricas.

Si lo aplicamos a `mass` obtenemos el siguiente resultado:

```
mass.describe()
```

```
count     342.000000
mean     4201.754386
std       801.954536
min      2700.000000
25%      3550.000000
50%      4050.000000
75%      4750.000000
max      6300.000000
Name: body_mass_g, dtype: float64
```

La información que obtenemos de la salida anterior sobre la distribución del peso de los pingüinos es la siguiente:

Fragmento de la salida	Significado
`count 342.000000`	Hay 342 valores no nulos, así que no se conoce el peso de dos pingüinos.
`mean 4201.754386`	El peso medio de los pingüinos es 4 kilogramos y 201 gramos.

Fragmento de la salida		Significado
`std`	`801.954536`	La desviación estandar del peso de los pinguinos es 801 gramos.
`min`	`2700.000000`	El pingüino que menos pesa, pesa 2 kilos y 700 gramos.
`25%`	`3550.000000`	El 25% de los pingüinos pesa menos de 3 kilos y 550 gramos (y el 75% restante más). Este valor se conoce como **cuantil 0.25**, **percentil 25** o **primer cuartil**.
`50%`	`4050.000000`	El 50% de los pingüinos pesa menos de 4 kilos y 50 gramos (y el 50% restante más). Este valor se llama **cuantil 0.5**, **percentil 50**, **segundo cuartil** o **mediana**.
`75%`	`4750.000000`	El 75% de los pingüinos pesa menos de 4 kilos y 750 gramos (y el 25% restante más). Este valor se llama **cuantil 0.75**, **percentil 75**,o **tercer cuartil**.
`max`	`6300.000000`	El pingüino que más pesa, pesa 6 kilos y 300 gramos.

Problema 9

Utiliza el método `describe()` para obtener información sobre la distribución de la longitud de las alas de los pingüinos, y responde a las siguientes preguntas:

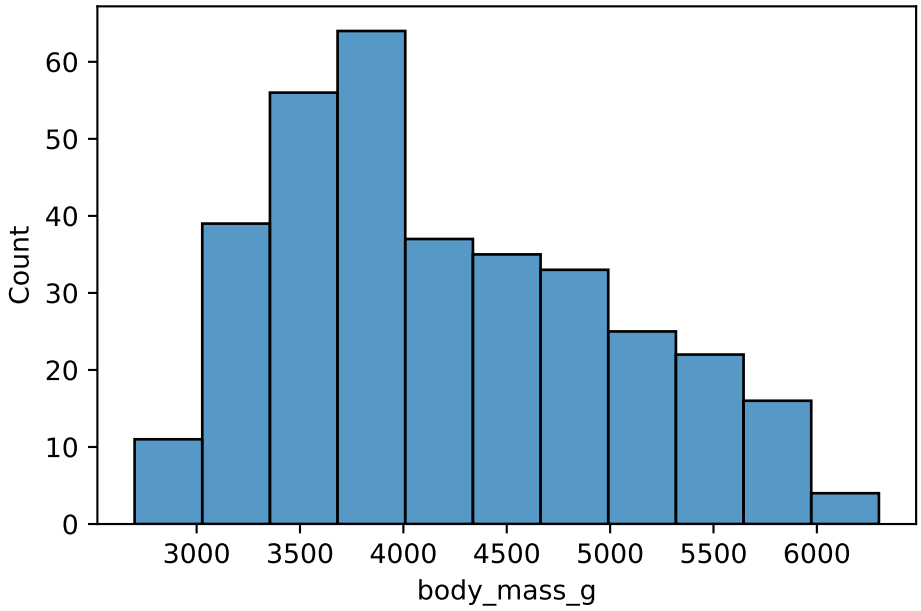
- ¿Para cuántos pingüinos falta el dato de la longitud de sus alas?
- ¿Cuál es el percentil 25 de la longitud de las alas de los pingüinos?

5.2. Histograma

Los **histogramas** son uno de los gráficos más comunes e informativos para describir la distribución de una variable continua. Para crear un histograma se representa en el eje x el rango de valores de la variable, se divide ese rango en intervalos de la misma longitud, y se dibuja para cada intervalo una barra de altura igual al número de valores que caen en ese intervalo.

El siguiente código crea un histograma para el peso de los pingüinos:

```
sns.histplot(data=penguins, x="body_mass_g");
```



> Para realizar un histograma usa la función `histplot()` del paquete `seaborn`, e indica:
> - El nombre de la hoja de datos como valor del argumento `data`.
> - El nombre de la variable como valor del argumento `x`.

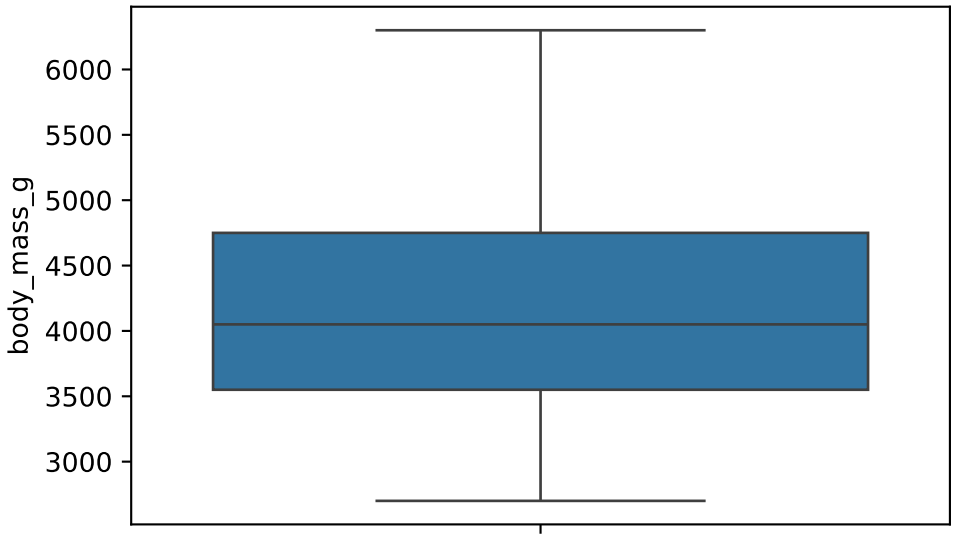
Problema 10
Realiza un histograma para la longitud de las alas de los pingüinos.

5.3. Diagrama de caja y bigotes

Otro tipo de gráficos para variables numéricas son los llamados **diagramas de caja** o **de caja y bigotes**.

Ejecuta el siguiente código para crear un diagrama de caja y bigotes para el peso de los pingüinos.

```
sns.boxplot(data=penguins, y="body_mass_g");
```



La caja se construye usando los cuartiles de la variable.

Y los bigotes se extienden desde los extremos de la caja hasta los valores mínimo y máximo, exceptuando los valores que se clasifican como **outliers**.

> Para realizar un diagrama de caja y bigotes usa la función `boxplot()` del paquete `seaborn` e indica:
> - El nombre de la hoja de datos como valor del argumento `data`.
> - El nombre de la variable como valor del argumento `y`.

Problema 11

Realiza un diagrama de caja y bigotes para la longitud de las alas de los pingüinos.

6. Agrupar por categorías

En la Sección 2.3 visualizamos los datos de los cinco primeros pingüinos con la instrucción

```
penguins.head(5)
```

Y en la Sección 3.1 calculamos el peso máximo de todos los pingüinos con la orden

```
penguins["body_mass_g"].max()
```

Ahora nos planteamos realizar las operaciones anteriores, pero para cada especie de pingüinos (Adelie, Chinstrap y Gentoo), es decir, queremos:

- Ver los datos de los primeros pingüinos de cada una de las tres especies.
- Calcular el peso máximo en cada una de las tres especies de pingüinos.

`pandas` proporciona el método `groupby()` para resolver este tipo de problemas.

El método `groupby()` divide una hoja de datos en los grupos o categorías definidos por una variable categórica (en nuestro caso las tres especies dadas por la variable `species`), de forma que al realizar una operación (como `head()` o `máx()`) en la hoja de datos por grupos, obtendremos el resultado de la operación en cada uno de los grupos (en nuestro caso en cada especie).

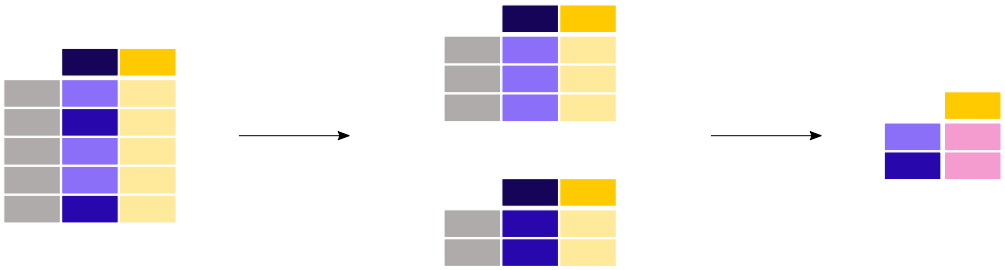


Figura 3: Ilustración del método `groupby()` en el manual de `pandas`

Ejemplo 1: Primeras filas de cada especie

Para ver los datos de los tres primeros pingüinos de cada especie, ejecuta la instrucción siguiente:

```
penguins.groupby("species").head(3)
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_m
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_
152	Chinstrap	Dream	46.5	17.9	192.0	3500.0
153	Chinstrap	Dream	50.0	19.5	196.0	3900.0
154	Chinstrap	Dream	51.3	19.2	193.0	3650.0
220	Gentoo	Biscoe	46.1	13.2	211.0	4500.0
221	Gentoo	Biscoe	50.0	16.3	230.0	5700.0
222	Gentoo	Biscoe	48.7	14.1	210.0	4450.0

Observa que en la salida anterior se muestran:

- las filas de índices 0, 1 y 2 correspondientes a los tres primeros pingüinos de la especie Adelie
- las filas de índices 152, 153 y 154 correspondientes a los tres primeros pingüinos de la especie Chinstrap
- y las filas de índices 220, 221 y 222 correspondientes a los tres primeros pingüinos de la especie Gentoo.

Analicemos, paso por paso, lo que ocurre al ejecutar la instrucción `penguins.groupby("species").head(3)`:

1. `penguins.groupby("species")` parte la hoja de datos `penguins` en tres grupos correspondientes a las tres especies. Esta operación devuelve un objeto de un nuevo tipo llamado `DataFrameGroupBy`:

   ```
   type(penguins.groupby("species"))
   ```

   ```
   pandas.core.groupby.generic.DataFrameGroupBy
   ```

   Puedes pensar que después de aplicar `groupby()` tenemos los datos virtualmente divididos en tres hojas de datos:

   - una hoja de datos con las 152 filas de los pingüinos de la especie Adelie
   - otra con las 68 filas de los pingüinos de la especie Chinstrap
   - y una tercera con las 124 filas de los pingüinos de la especie Gentoo.

2. Al aplicar `head(3)` sobre la hoja de datos agrupada creada con `penguins.groupby("species")`, el efecto es que se aplica `head(3)` en cada uno de los tres grupos (como si se aplicara en cada una de las tres hojas de datos virtuales que ha creado `groupby()` a partir de `penguins`).

La salida que hemos obtenenido es la combinación de los tres resultados en los tres grupos.

Ejemplo 2: Peso máximo de cada especie

Para obtener el peso máximo de cada especie ejecuta

```
penguins[["species", "body_mass_g"]].groupby("species").max()
```

species	body_mass_g
Adelie	4775.0
Chinstrap	4800.0
Gentoo	6300.0

Vemos que:

- el peso máximo entre los pingüinos de la especie Adelie es de 4 kg y 775 gramos.
- el peso máximo entre los pingüinos de la especie Chinstrap es de 4 kg y 800 gramos.
- el peso máximo entre los pingüinos de la especie Gentoo (y entre todos los pingüinos) es de 6 kg y 300 gramos.

Analicemos la instrucción anterior paso por paso:

1. Como nos interesa el peso máximo por especie, primero hemos escrito `penguins[["species", "body_mass_g"]]`, para seleccionar las dos variables asociadas a las características de interés, `species` y `body_mass_g`, conforme aprendimos en la Sección 3.2.

2. Después `groupby("species")` divide la selección de la hoja de datos del paso anterior en tres grupos para cada especie.

3. `max()` aplicada a la hoja de datos por grupos (de tipo `DataFrameGroupBy`) creada en el paso anterior, calcula el máximo en cada uno de los grupos para cada especie.

   La salida que hemos obtenido es una tabla con los tres pesos máximos.

En las hojas de datos agrupadas por categorías (objetos de tipo `DataFrameGroupBy`) también funcionan los mecanismos de selección de variables usando corchetes `[]`. Así, una forma equivalente de calcular el peso máximo de cada especie sería:

```
penguins.groupby("species")["body_mass_g"].max()
```

```
species
Adelie       4775.0
Chinstrap    4800.0
Gentoo       6300.0
Name: body_mass_g, dtype: float64
```

Los pasos seguidos en esta segunda variante serían:

1. `groupby("species")` crea los tres grupos o categorías definidos por la variable `species`.

2. `["body_mass_g"]` selecciona la variable con el peso de los pingüinos en cada grupo.

3. `max()` calcula el máximo en cada grupo.

Ejemplo 3: Recuento del número de pingüinos de cada especie

El método `value_counts()` que usamos en la Sección 4.2 para obtener una tabla de recuentos del número de pingüinos de cada especie es en el fondo la combinación de una operación de agrupación y la aplicación del método `count()`. De hecho es equivalente a

```
penguins.groupby("species")["species"].count()
```

```
species
Adelie       152
Chinstrap     68
Gentoo       124
Name: species, dtype: int64
```

Problema 12

Combina los métodos `groupby()` y `mean()` para calcular el peso medio de los pingüinos que viven en cada isla.

Problema 13

Combina los métodos `groupby()` y `median()` para calcular la mediana de la longitud del pico de las hembras y de los machos.

Problema 14

Combina los métodos `groupby()` y `value_counts()` para calcular cuántos pingüinos de cada especie viven en cada isla.
Analiza el resultado para responder a las siguientes preguntas:

- ¿cuál es la única especie de pingüinos presente en las tres islas?
- ¿en qué islas viven los pingüinos de la especie Chinstrap?
- ¿qué especies de pingüinos hay en la isla Biscoe?

7. Una variable numérica por categorías

En la Sección 5 estudiamos la distribución del peso de todos los pingüinos del estudio. Ahora estudiaremos si hay diferencias en la distribución del peso dependiendo de la especie.

7.1. El método `describe()`

El método `describe()` que usamos en la Sección 5.1 para obtener un resumen de la distribución del peso de todos los pingüinos, también puede aplicarse al peso agrupado por especies:

```
penguins.groupby("species")["body_mass_g"].describe()
```

species	count	mean	std	min	25%	50%	75%	max
Adelie	151.0	3700.662252	458.566126	2850.0	3350.0	3700.0	4000.0	4775.0
Chinstrap	68.0	3733.088235	384.335081	2700.0	3487.5	3700.0	3950.0	4800.0
Gentoo	123.0	5076.016260	504.116237	3950.0	4700.0	5000.0	5500.0	6300.0

	count	mean	std	min	25%	50%	75%	max
species								

Vemos que obtenemos una tabla con el resultado combinado de aplicar el método `describe()` en el grupo de datos correspondiente a cada una de las tres especies.

Interpretamos los resultados en las columnas de rótulo `count`, `mean`, `50%` y `max`:

Fragmento de la salida	Significado
`count`	De la Sección 4.1 sabíamos que hay 152 pingüinos de la especie Adelie, 124 de la especie Gentoo, y 68 de la especie Chinstrap. Y de la Sección 5.1 sabíamos que se desconoce el peso de dos pingüinos. Ahora sabemos que esos dos pingüinos son uno de la especie Adelie y otro de la especie Gentoo, y que se conoce el peso de todos los pingüinos de la especie Chinstrap.
`mean`	El peso medio de los pingüinos de la especie Gentoo es de 5 kilogramos y 76 gramos, superior a la media global de 4 kilos y 201 gramos (que conocíamos de la Sección 5.1). Y los pesos medios de las especies Adelie y Chinstrap son inferiores a la media global y muy similares: 3 kilos 700 gramos, y 3 kilos y 733 gramos respectivamente.
`50%`	La mediana del peso es la misma para las especies Adelie y Chinstrap, 3 kilos y 700 gramos, y sustancialmente más grande, 5 kilos, para la especie Gentoo.

Fragmento de la salida	Significado
`max`	El pingüino de mayor peso del estudio, con 6 kilos y 300 gramos, es de la especie Gentoo. Mientras que el peso máximo para las especies Adelie y Chinstrap no supera los 5 kilos (4 kilos y 775 gramos y 4 kilos y 800 gramos respectivamente).

Problema 15

Usa los métodos `groupby()` y `describe()` para responder a las siguientes preguntas:

a. Indica el peso medio de hembras y machos y compáralos con el peso medio global.
b. ¿Cuál es el peso máximo de las hembras?
c. ¿Cuál es la mediana del peso de los machos?
d. El pingüino de mayor peso ¿es macho o hembra?

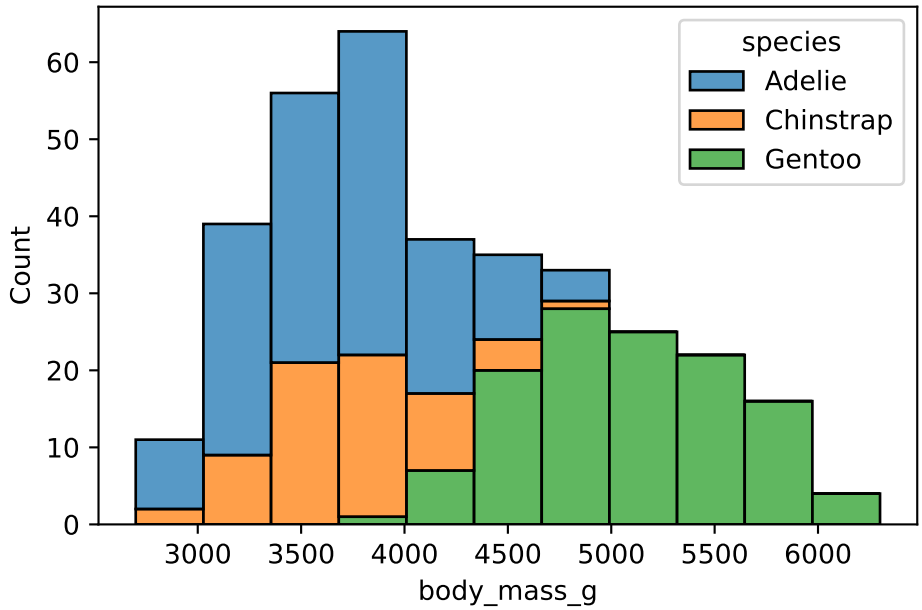
7.2. Histogramas

En este apartado aprenderemos a construir diferentes histogramas para comparar la distribución del peso de las tres especies de pingüinos.

7.2.1. Histograma de barras apiladas

Para crear un histograma del peso distinguiendo la especie mediante barras apiladas ejecuta la siguiente instrucción:

```
sns.histplot(
    data=penguins,
    x="body_mass_g",
    hue="species",
    multiple="stack"
);
```



Observa que la forma del diagrama es la misma que la del histograma que se realizó en la Sección 5.2. La novedad es que cada una de las barras que representa el número de pingüinos en cada uno de los intervalos de peso se ha fragmentado en tres barras indicando la contribución de cada especie. Aprecia que conforme va aumentando el peso, va disminuyendo la contribución de las especies Adelie (azul) y Chinstrap (naranja), y aumentando la contribución de la especie Gentoo (verde).

Con el argumento `hue="species"` conseguimos que los datos se dividan según las categorías de la variable `species` y se asigne un color diferente a cada categoría, que se indica en la leyenda de la esquina superior derecha del gráfico.

Y con el argumento `multiple="stack"` indicamos que las barras de cada especie se dibujen apiladas en cada intervalo.

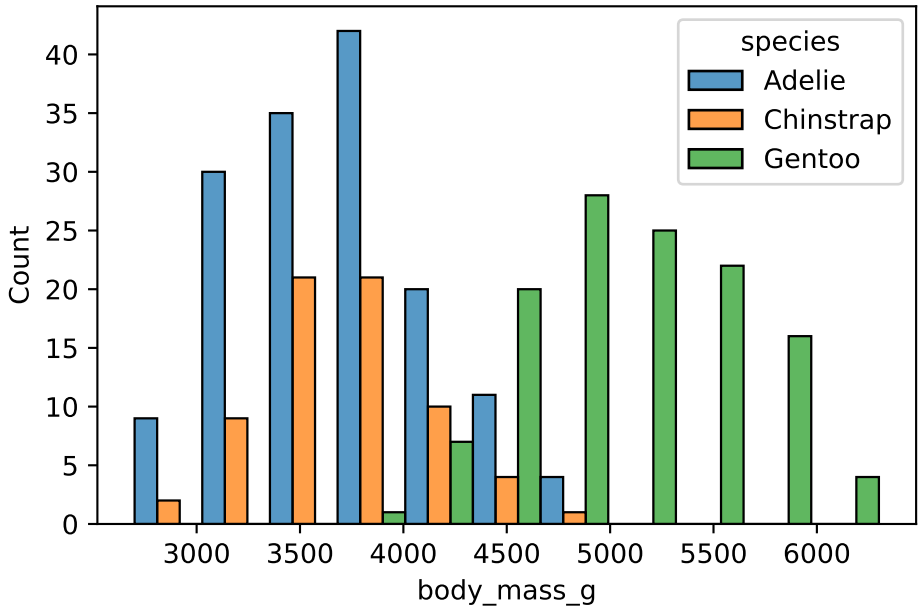
> Para crear un histograma de barras apiladas de una variable numérica por categorías, usa la función `histplot()` del paquete `seaborn` conforme aprendiste en la sección Sección 5.2 y además:
>
> - indica el nombre de la variable que determina las categorías como valor del parámetro `hue`.
> - usa `multiple="stack"` para que las barras de las diferentes categorías se muestren apiladas.

Problema 16
Crea un histograma de barras apiladas para el peso de los pingüinos por sexo.

7.2.2. Histograma de barras separadas

Para crear un histograma del peso con barras separadas para cada especie ejecuta la siguiente instrucción:

```
sns.histplot(
    data=penguins,
    x="body_mass_g",
    hue="species",
    multiple="dodge"
);
```



En este tipo de histograma fijándonos en las barras de un solo color vemos el histograma de la especie correspondiente.

> Para crear un histograma de barras separadas procede de la misma forma que para crear un histograma de barras apiladas pero cambia el valor del parámetro `multiple` a `"dodge"`.

Problema 17

Crea un histograma de barras separadas para la longitud del pico de los pingüinos por sexo.

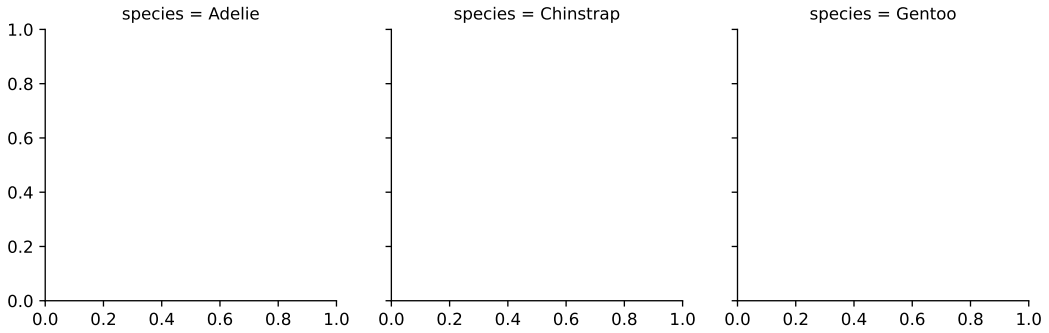
7.2.3. Mosaico de histogramas

En los dos apartados anteriores aprendiste a crear dos tipos de histogramas para visualizar la distribución de una variable numérica distinguiendo varias categorías con colores diferentes en un mismo gráfico. Pero a veces puede resultar más útil realizar histogramas independientes para cada categoría. En este apartado aprenderás a crear, de forma simultánea, tres histogramas con la distribución del peso de los pingüinos de cada especie.

Con este fin el paquete `seaborn` proporciona la clase `FacetGrid`, que permite crear un mosaico o una malla en la que ubicar los diferentes gráficos para los datos de cada categoría.

Para entender cómo se usa la clase `FacetGrid` empieza ejecutando la siguiente instrucción:

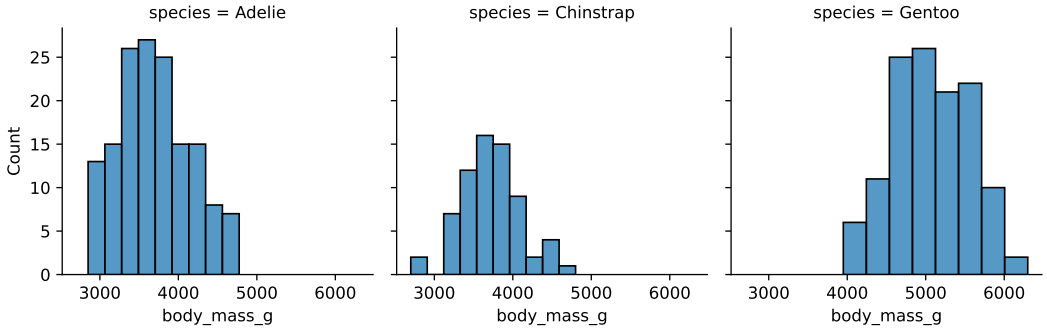
```
sns.FacetGrid(data=penguins, col="species")
```



Observa el resultado y aprecia que se han creado tres ejes de coordenadas, uno para cada especie. El efecto del argumento `col="species"` es que se ha creado una columna (`col`=*column*) para cada categoría de la variable `species`.

De momento hemos inicializado el mosaico pero aún no hemos dibujado nada en cada una de las celdas del mosaico. Para hacerlo tenemos que aplicar al mosaico el método `map()`, como en el siguiente ejemplo:

```
grid = sns.FacetGrid(data = penguins, col="species")
grid.map(sns.histplot, "body_mass_g");
```



> **!** Para que puedas visualizar el gráfico anterior en tu cuaderno de *Colab* tienes que escribir las dos instrucciones anteriores en la misma celda.

Fíjate en las dos etapas que se han seguido en el ejemplo anterior para la construcción de los tres histogramas:

1. La primera instrucción

   ```
   grid = sns.FacetGrid(data = penguins, col="species")
   ```

   crea el mosaico con las tres columnas para las tres especies, y lo almacena en un nuevo objeto de nombre `grid` (puedes usar cualquier otro nombre).

   > **i** Para inicializar un mosaico de gráficos con una columna para cada valor de una variable categórica, inicializa la clase `FacetGrid` indicando:
   > - La hoja de datos como valor del parámetro `data`
   > - El nombre de la variable categórica para crear los grupos como valor del parámetro `col`

   En este punto tenemos preparadas tres celdas para crear en cada celda un gráfico de los datos de la especie correspondiente a esa celda. La creación efectiva de los gráficos se realiza en el siguiente paso.

2. La segunda instrucción

   ```
   grid.map(sns.histplot, "body_mass_g")
   ```

   aplica el método `map()` al mosaico creado en el paso anterior, especificando qué tipo de gráfico queremos en cada celda del mosaico, y a qué variable queremos aplicar ese gráfico:

- El primer argumento `sns.histplot` indica que en cada celda queremos realizar un histograma (para el grupo de datos de la especie correspondiente a la celda).
- El segundo argumento `"body_mass_g"` indica la variable de la que queremos hacer los histogramas.

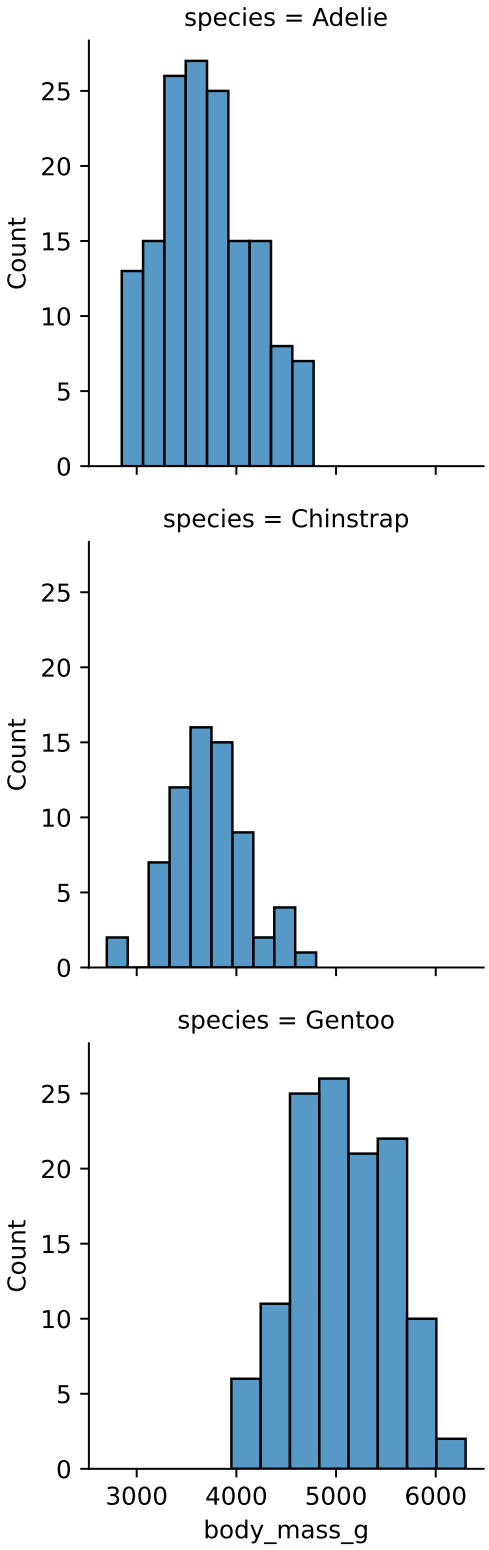
> **i** Para crear sendos gráficos de los datos asociados a cada celda de un mosaico creado con `FacetGrid` usa el método `map()` con los siguientes argumentos:
>
> - La función para construir los gráficos
> - El nombre de la variable a la que aplicar esa función gráfica

Problema 18

Usa la clase `FacetGrid` para realizar sendos histogramas de la distribución del peso para hembras y machos.

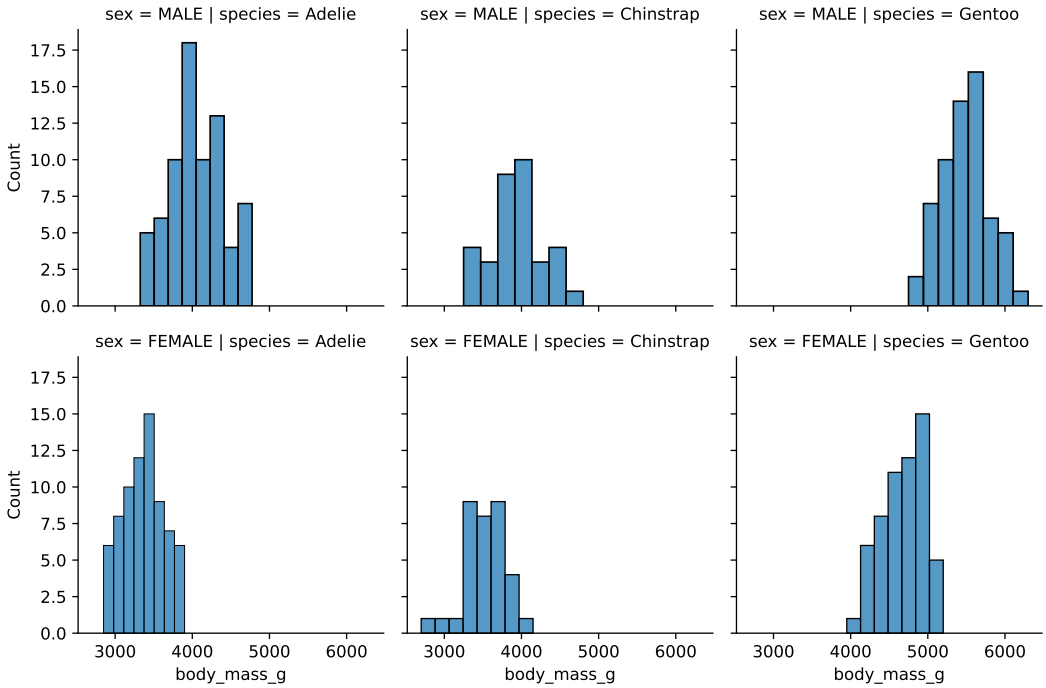
Si al inicializar la clase `FacetGrid` se usa el parámetro `row` en lugar de `col` los diferentes gráficos aparecerán dispuestos por filas en lugar de por columnas. Por ejemplo:

```
grid = sns.FacetGrid(data = penguins, row="species")
grid.map(sns.histplot, "body_mass_g");
```


species = Adelie
Count
0
5
10
15
20
25
species = Chinstrap
Count
0
5
10
15
20
25
species = Gentoo
Count
0
5
10
15
20
25
3000
4000
5000
6000
body_mass_g

De hecho, se pueden usar los argumentos `row` y `col` al mismo tiempo para crear un mosaico bidimensional. Las siguientes instrucciones crean un mosaico de histogramas del peso de los pingüinos, distinguiendo la especie por filas y el sexo por columnas, de dimensiones 3×2:

```
grid = sns.FacetGrid(data = penguins, row="sex", col="species")
grid.map(sns.histplot, "body_mass_g");
```

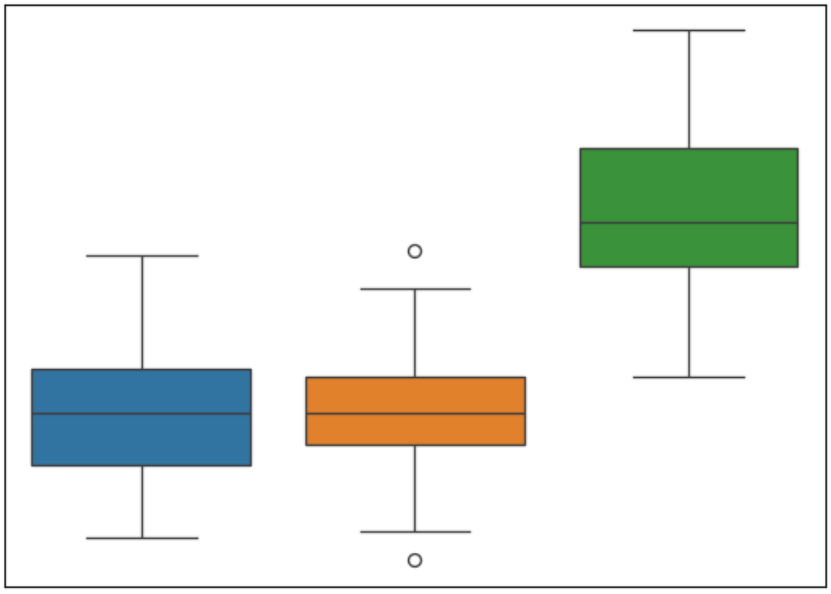


Problema 19

Usa `FacetGrid` para crear un mosaico de histogramas para la longitud de las alas de los pingüinos distingüiendo la isla en la que viven y su sexo.

7.3. Diagramas de caja y bigotes

Para acabar la práctica realizaremos el gráfico de la portada de este documento



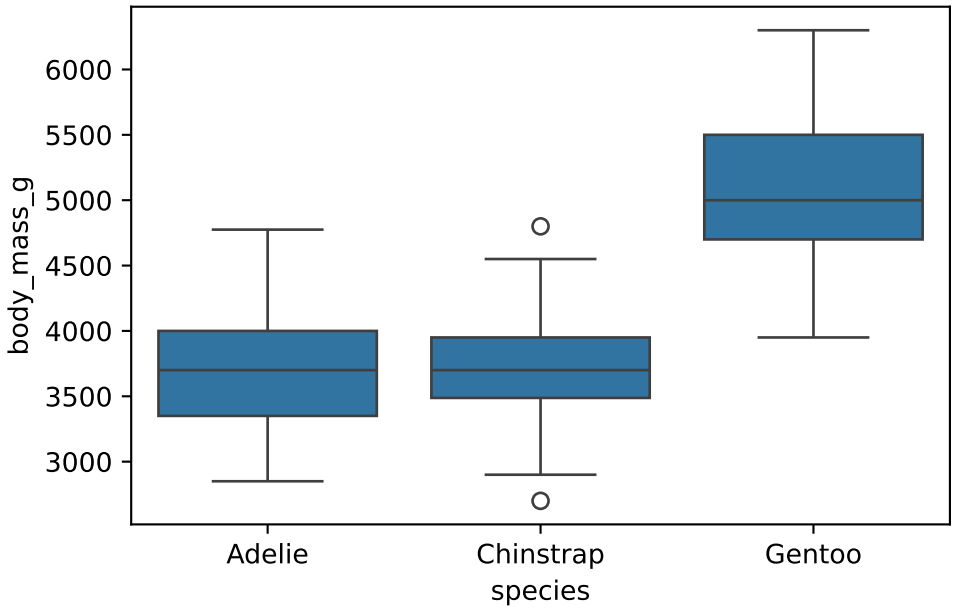
en el que se representa un diagrama de caja y bigotes para el peso de los pingüinos de cada especie.

En la Sección 5.3 hicimos un diagrama de caja y bigotes para el peso de todos los pingüinos (sin distinguir la especie) con la instrucción

```
sns.boxplot(data=penguins, y="body_mass_g");
```

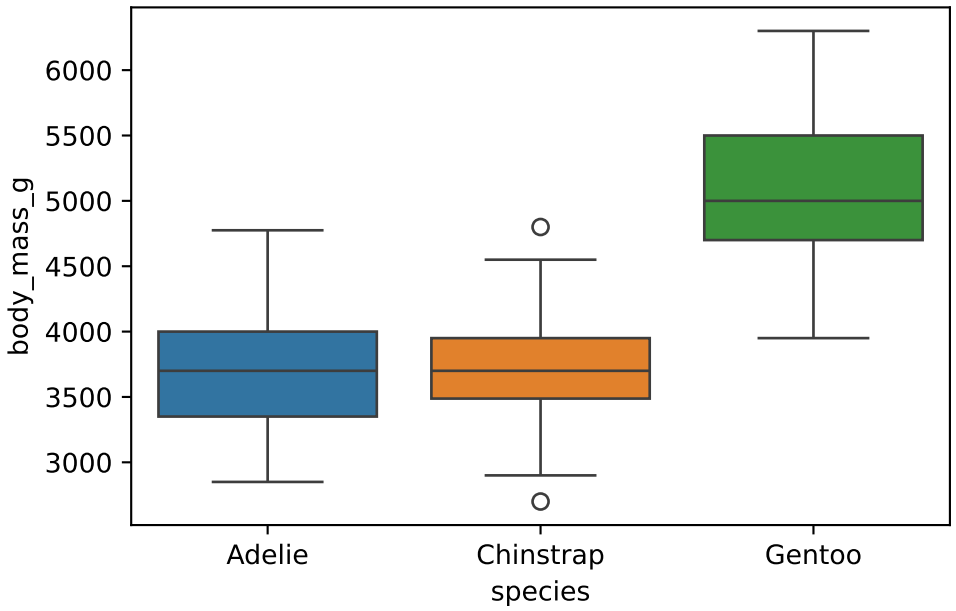
Para diferenciar por especie no tenemos más que añadir el argumento `x="species"`:

```
sns.boxplot(data=penguins, x="species", y="body_mass_g");
```



Para asignar un color diferente a cada especie añadimos `hue="species"`:

```
sns.boxplot(data=penguins, x="species", y="body_mass_g",
↪  hue="species");
```



Los dos puntos que aparecen resaltados con un círculo fuera de los bigotes en el diagrama de la especie Chinstrap (en naranja) son **outliers**. El outlier

por debajo del bigote inferior representa un pingüino de la especie Chinstrap con peso excesivamente bajo, y el outlier por encima del bigote superior se corresponde con un pingüino de la especie Chinstrap con peso excesivamente alto (excesivamente bajo/alto respecto a la distribución del peso en los pingüinos de la especie Chinstrap).

Problema 20

Realiza un gráfico que muestre los diagramas de caja y bigotes para el peso de las hembras y de los machos.
¿Aparecen outliers en alguno de los dos grupos?

Bibliografía

[1] KB Gorman, TD Williams y WR Fraser. "Ecological Sexual Dimorphism and Environmental Variability within a Community of Antarctic Penguins (Genus Pygoscelis)". en. En: *PLoS ONE* 9.3 (2014). DOI: 10.1371/journal.pone.0090081.

[2] Wes McKinney. "Data Structures for Statistical Computing in Python". En: *Proceedings of the 9th Python in Science Conference*. Ed. por Stéfan van der Walt y Jarrod Millman. 2010, págs. 56-61. DOI: 10.25080/Majora-92bf1922-00a.

[3] The pandas development team. *pandas-dev/pandas: Pandas*. Ver. latest. Feb. de 2020. DOI: 10.5281/zenodo.3509134. URL: 10.5281/zenodo.3509134.

[4] Michael L. Waskom. "seaborn: statistical data visualization". En: *Journal of Open Source Software* 6.60 (2021), pág. 3021. DOI: 10.21105/joss.03021. URL: https://doi.org/10.21105/joss.03021.