

Introduction:

Python

Analyse de donnée

Pratique avec
Google Colab

Présenté par
Kevin Rosamont

Qu'est-ce que Python?

Python

Un langage de programmation

La première version réalisée sort en 1991

Facile à lire

Quand utilise t'on Python?

Utilisation de Python

Développement web (Google, Fb)

Application de bureau (BitTorrent, Dropbox)

Développement de jeux video (EVE Online, Doki Doki)

Data Science (Imagerie, système de recommandation)

Utilisation de Python

Développement web (Google, Fb)

Application de bureau (BitTorrent, Dropbox)

Développement de jeux video (EVE Online, Doki Doki)

Data Science (Imagerie, système de recommandation)

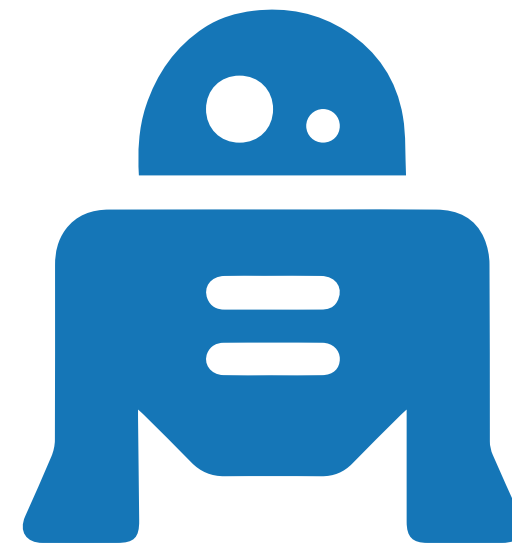
Dans la Data Science



Analyse de
Données



Visualisation



Machine
Learning



Statistiques



Big
Data

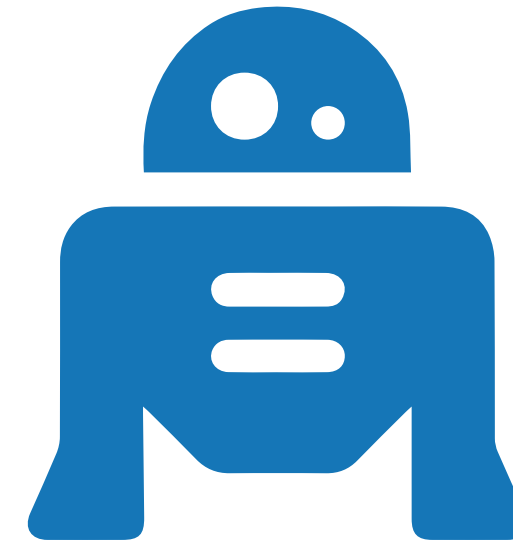
Dans la Data Science



Analyse de
Données



Visualisation



Machine
Learning



Statistiques



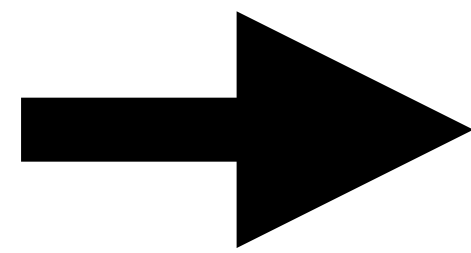
Big
Data

Analyse de données

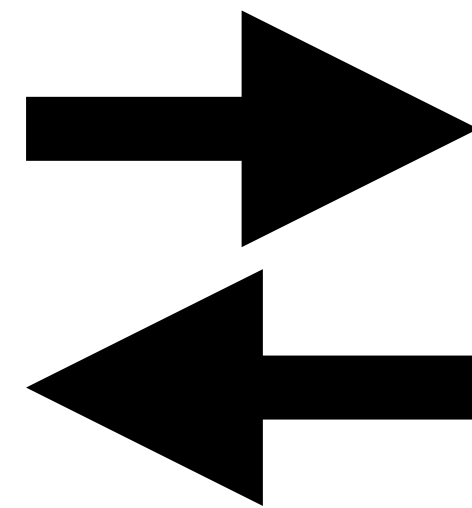
4 étapes basiques



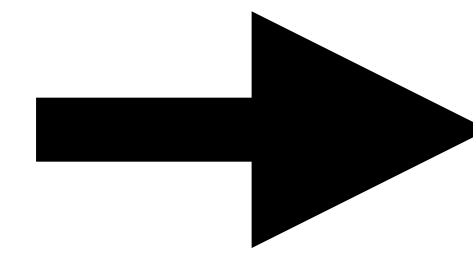
Lecture des
données



Transformation
et description



Visualisation

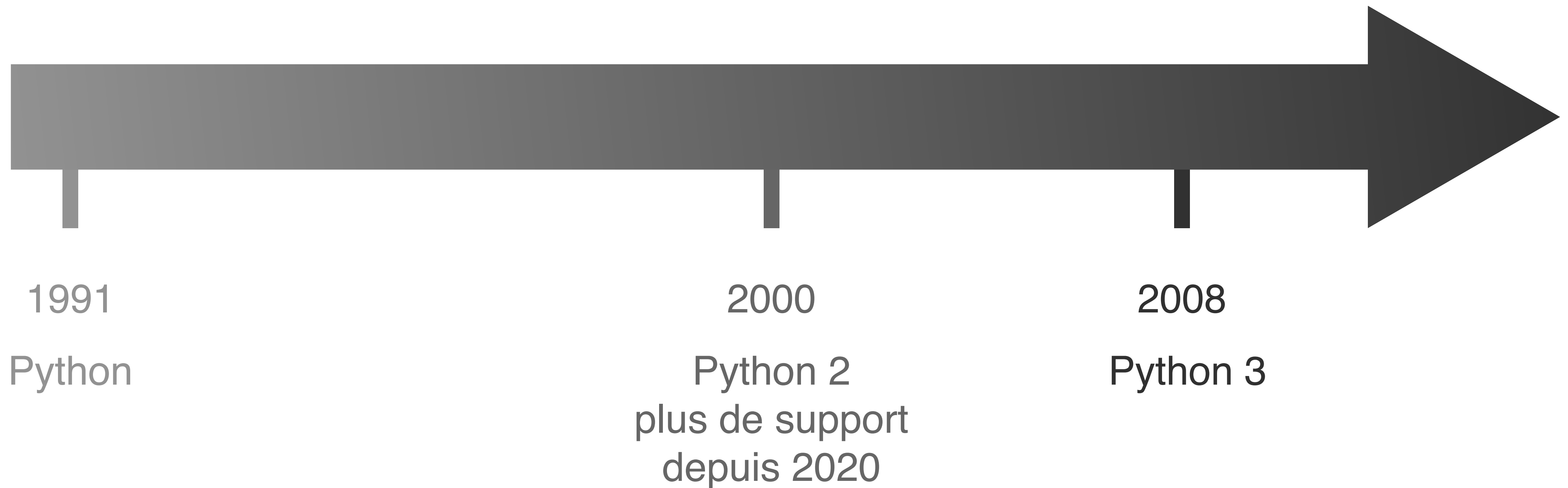


Sauvegarde

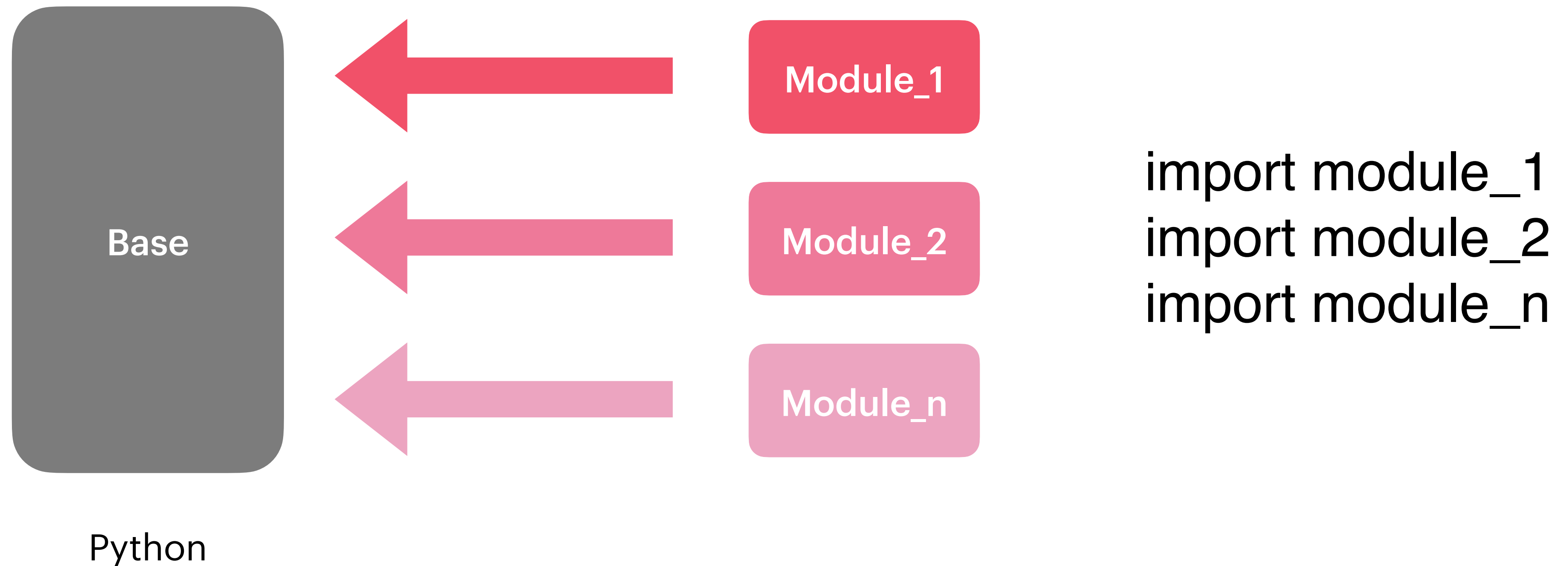
Fonctionnement du langage

Évolution du langage

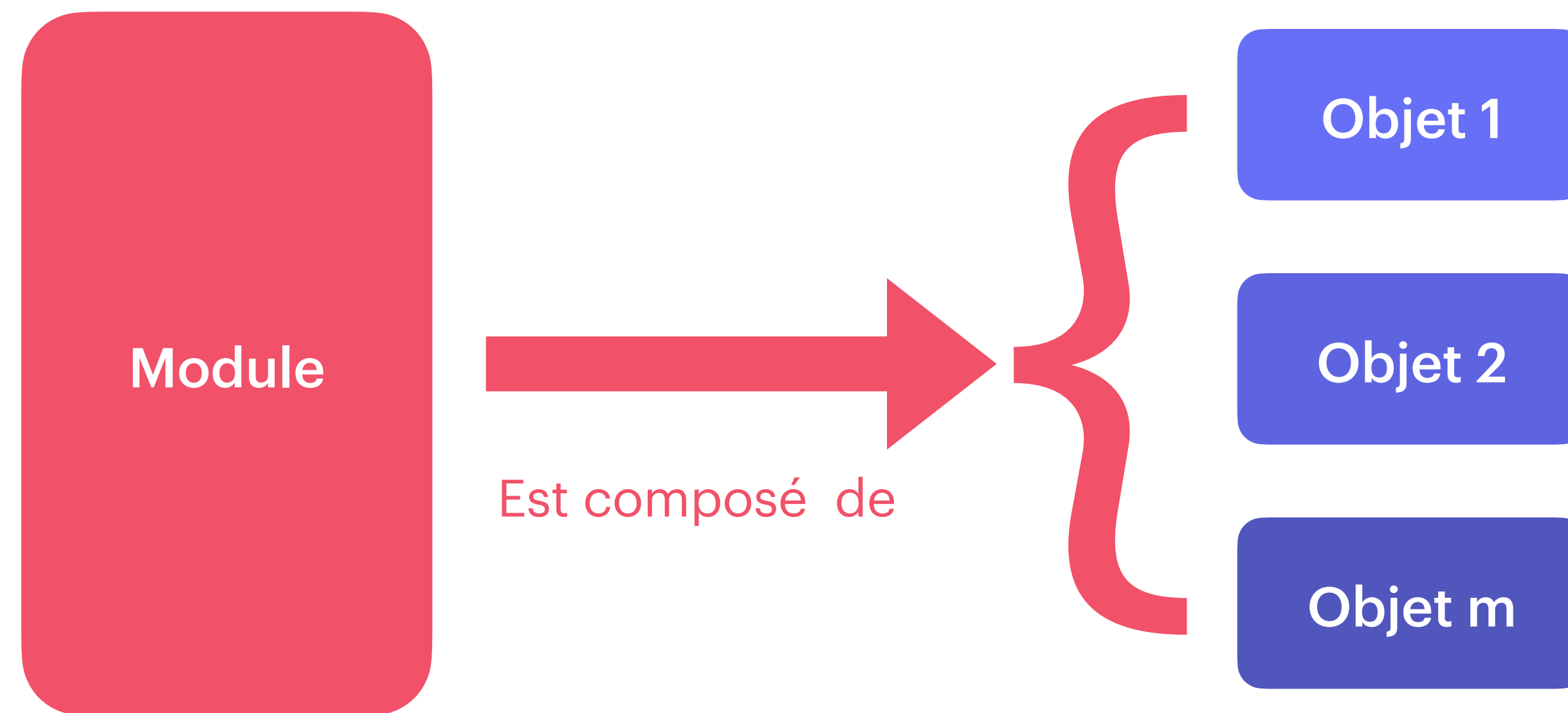
3 étapes dans l'évolution



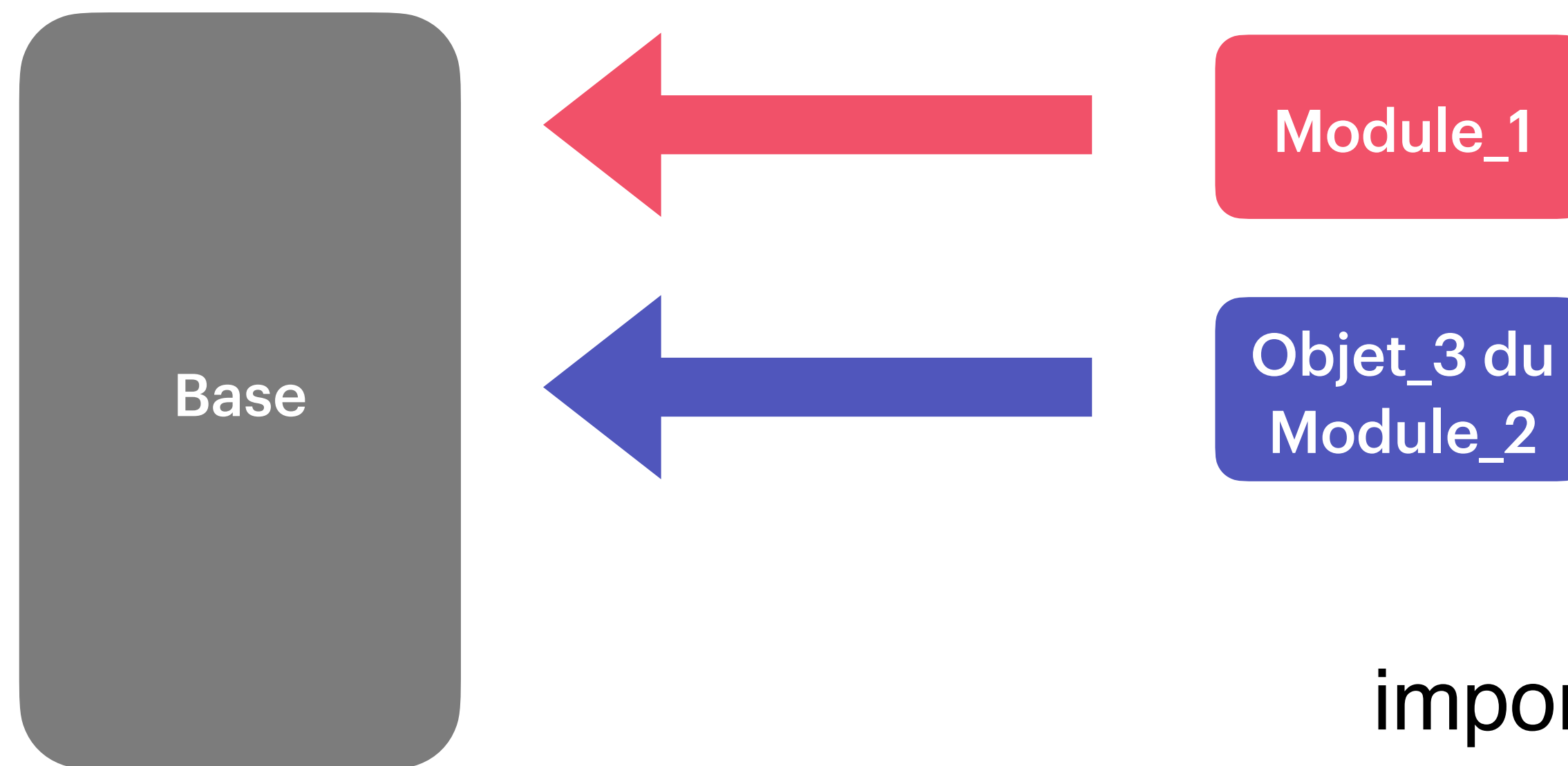
Enrichissement des fonctionnalités



Chaque module est composé d'éléments



Dans la pratique



Python

```
import module_1 as md1
# md1: alias comme en SQL
import objet_3 from module_2
```

Les modules utilisés

Nos modules principaux



Pandas



Numpy



Seaborn

Utilisation de Google Colab (gratuit)

Taper “Colab” sur Google



Colab



All

Images

Maps

News

Videos

More

Tools

About 60,800,000 results (0.42 seconds)

<https://colab.research.google.com>

Google Colab

Sign in.

Colaboratory

With Colab you can import an image dataset, train an image ...

Google Colab

Sign in. Loading... Loading...

Google Colab Notebook

Overview of Colaboratory - Markdown Guide - External data

Colab

Colab vous permet de tirer pleinement parti des ...











[More results from google.com »](#)

Créer un nouveau script

1- Allez dans Nouveau notebook

ExemplesRécentsGoogle DriveGitHubImporter

Filtrer les notebooks

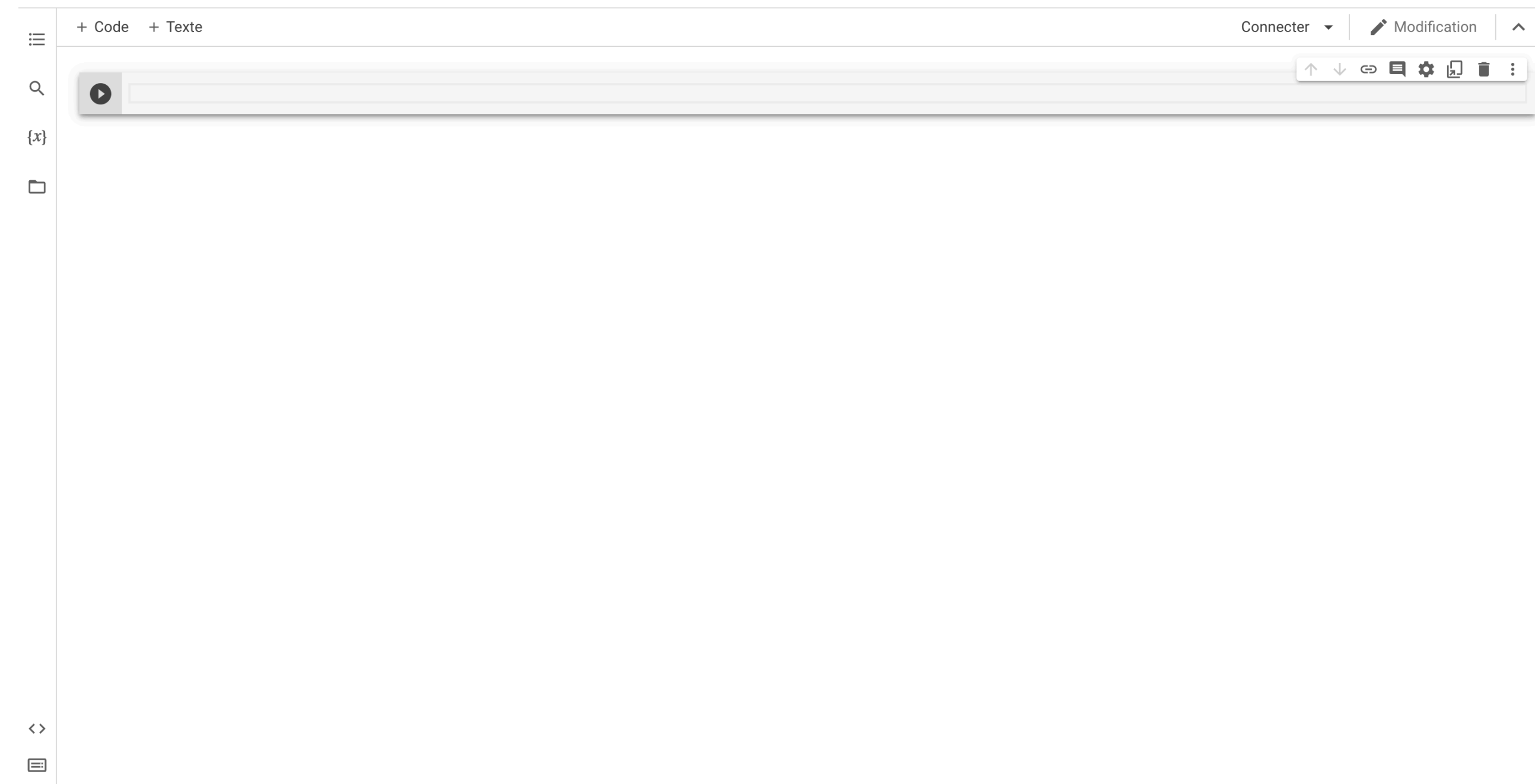
Titre	Dernière ouverture ▲	Première ouverture ▼	
 Bienvenue dans Colaboratory	01:15	01:01	
 Data Table Display	01:13	01:13	
 Untitled0.ipynb	01:02	01:02	 
 GNN_overview.ipynb	10 août 2021	10 août 2021	 

Nouveau notebook

Annuler

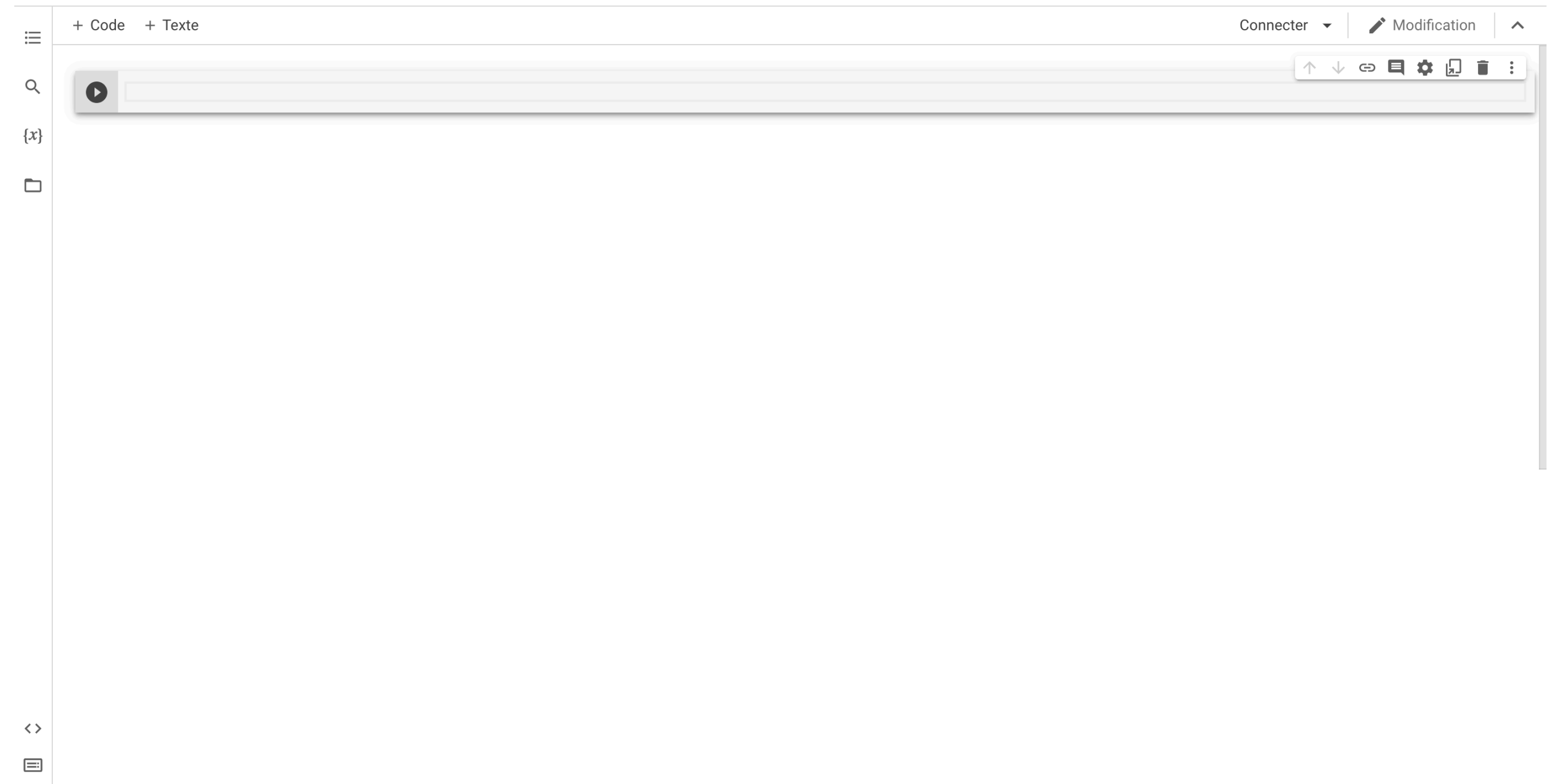
Page de script

2- Nouvelle page



Jupyter Notebook

Possibilité d'ajouter du texte et du code dans un document



Importation et manipulation de données

Lecture de fichier csv

```
import pandas as pd
df = pd.read_csv('chemin/nom_du_fichier.csv')
# df est le nom donné au dataset
df.head() # En tapant df, on a un aperçu du dataset

# '#' Permet d'écrire des commentaires, du code non lu
```

Utiliser `pd.read_excel()` pour lire des fichiers `.xls`, `.xlsx`

Rappel en SQL

Requête type

```
SELECT DISTINCT column1 [, column2, AGG_FUNC(column_or_expression), ...]  
FROM mytable  
WHERE constraint_expression1 [AND/OR constraint_expression2...]  
GROUP BY column1 [, column2,...]  
HAVING constraint_having_expression1 [AND/OR constraint_having_expression2...]  
ORDER BY column1 ASC/DESC [,column2,...]  
LIMIT n_observations;
```

Filtrer les observations

Les même opérateurs qu'en SQL:

<, <=, >, >=,

| (or), & (and), ==,

isnull(), notnull(), ~(not)

```
df[
```

```
# On choisit le dataset
```

Filterer les observations

Les même opérateurs qu'en SQL:

<, <=, >, >=,

| (or), & (and), ==,

isnull(), notnull(), ~(not)

```
df[df['col_name']    ]
```

```
# On choisit le dataset
```

```
# On choisit la colonne sur laquelle on filtre
```

Filterer les observations

Les même opérateurs qu'en SQL:

<, <=, >, >=,

| (or), & (and), ==,

isnull(), notnull(), ~(not)

```
df[df['col_name']<= 5]
```

```
# On choisit le dataset
```

```
# On choisit la colonne sur laquelle on filtre
```

```
# On y applique la règle
```

Un cas concret sur Colab

Filtrer des observations

```
import pandas as pd
df = pd.read_csv('sample_data/california_housing_test.csv')

df.head(n)
# Conserver les n premières lignes

df[df["population"] > 4000]
# Toutes les observations (lignes) où la population est sup. à 4000

df[(df["population"] == 809) | (df["population"] > 4000)]
# Toutes les observations (lignes) où la population est égale à 809
# == pour vérifier une égalité
```

Questions

Expliquer la signification:

1. `df[df['total_rooms'] <= 500]`
2. `df[(df['total_rooms'] <= 500) & (df['median_income'] > 10)]`
3. `df[(df['total_rooms'] <= 500) | (df['median_income'] > 10)]`
4. Les observations où le nombre total de chambre est 14
5. Les observations où la latitude est entre 34.05 et 34.06 et la longitude entre -118.24 et -118.25

Modifier des observations

```
df[df["population"] == 809] = 809 * 2  
# Toutes les observations (lignes) où la population est égale à 809  
# == pour vérifier une égalité
```


Création d'une nouvelle colonne

```
df["median_income_month"] = df["median_income"] * 1000  
# Nouvelle colonne nommé median_income_month qui est  
# la valeur de median_income multiplié par 1 000
```

Filterer les observations nulles

```
vehicles = pd.read_csv('vehicles.csv')
```

```
vehicles.head() # Avoir un aperçu du dataset
```

```
vehicles[vehicles["num_veh"].isnull()]
```

```
# Toutes les observations (lignes) où le numéro du véhicule est nul
```

```
vehicles[vehicles["num_veh"].str.isnull()]
```

```
# Si la colonne représente est une chaîne de caractère
```

Filtre avancé des observations de type string

Utiliser `startswith()`, `endswith()`, `contains()`

```
vehicles[vehicles["Num_Acc"].astype(str).startswith("2012")]
```

À ajouter `astype(str)` si la colonne n'est pas un string

Sélectionner des colonnes

```
df[['col_name1', 'col_name2', 'col_name3']]  
# Mettre le nom des colonnes entre guillemets
```

```
sub_df = df[['col_name1', 'col_name2', 'col_name3']]  
# sub_df est un nouvel objet basé sur df avec  
# uniquement 3 colonnes de df : col_name1, col_name2, col_name3
```

Le double `[]` permet de continuer à manipuler des datasets.
Le simple `[]` pour sortir des datasets et manipuler une colonne.

Renommer des colonnes

```
df.rename(columns= {'ex1':'new1', 'ex2':'new2'})
```

Se lit: l'ancien nom ex1 correspond au nouveau new1

```
df2 = df.rename(columns= {'ex1':'new1', 'ex2':'new2'})
```

df2 est un nouvel objet basé sur df avec les nouveaux noms

Questions

1. Importer la table “characteristics.csv”
2. Créer un nouvel objet `sub_car` composé des colonnes `Num_Acc`, `adr`, `dep`
3. Dans `sub_car`, sélectionnez les adresses contenant `Victor Hugo` dans le département `59`
4. Renommez `adr` en `adresse` et `Num_Acc` en `num_acc`

Aggregat

mean, sum, max, min, count, median, quantile

```
df['col_name'].mean()
```

```
df['col_name'].sum()
```

Calculer la somme ou la moyenne sur une colonne

```
df['col_name'].nunique()
```

Compter le nombre d'observations uniques

Compter le nombre de départements qui s'affichent dans `sub_car`

Description Statistiques du dataset

```
df.describe()
```

```
# En tapant df, on a un aperçu du dataset :  
moyenne, somme, médiane, quantile
```

Essayez sur un des jeux de données importés

Calculer les agrégats par groupe

```
df.groupby('key')[['colname']]  
# Grouper colname par key
```

```
df.groupby('key')[['colname']].mean()  
# Afficher une moyenne de colname groupé par key
```

```
df.groupby(['key1', 'key2'])[['colname']].mean()  
# Afficher une moyenne de colname par key1 et key2
```

Concrètement

```
df.groupby('sellers_id')[['price']]  
# Grouper les prix par vendeur
```

```
df.groupby('sellers_id')[['price']].sum()  
# Afficher une moyenne de colname groupé par key
```

Questions

1. Importer la table “soccer_all.xlsx”
2. Avoir un aperçu descriptif du dataset
3. Regarder le budget moyen et median par année
4. Regarder le budget moyen et median par championnat
5. Regarder le budget moyen et median par année et par championnat

Rappel des jointures en SQL

SQL INNER JOIN

Velo

id_velo	marque	lot
1	A	20B
2	A	21A
4	C	23R

Station

id_stn	place	name
A	12	Bastille
B	15	Halles
D	8	Voltaire

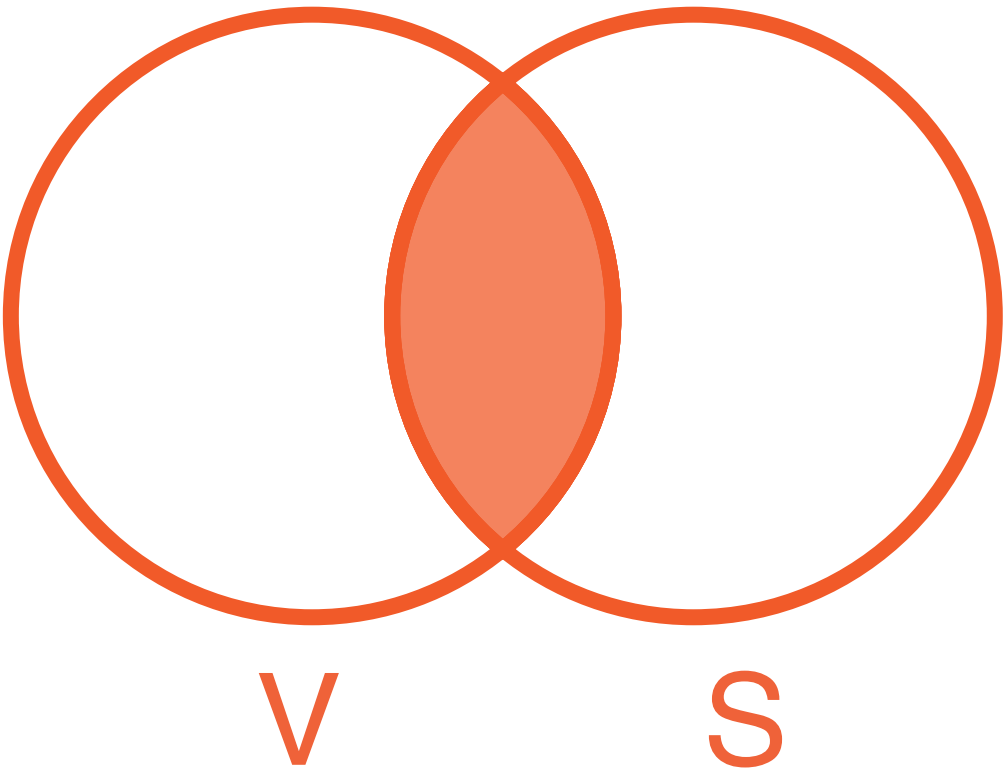
SQL INNER JOIN

Velo

id_velo	id_stn	lot
1	A	20B
2	A	21A
4	C	23R

Station

id_stn	place	name
A	12	Bastille
B	15	Halles
D	8	Voltaire



Intersection de **V** et **S**

Résultat

id_velo	id_stn	name	place	lot
1	A	Bastille	12	20B
2	A	Bastille	12	21A

Exemple :

```
SELECT *  
FROM Velo as V  
INNER JOIN Station as S  
ON S.id_stn = V.id_stn
```

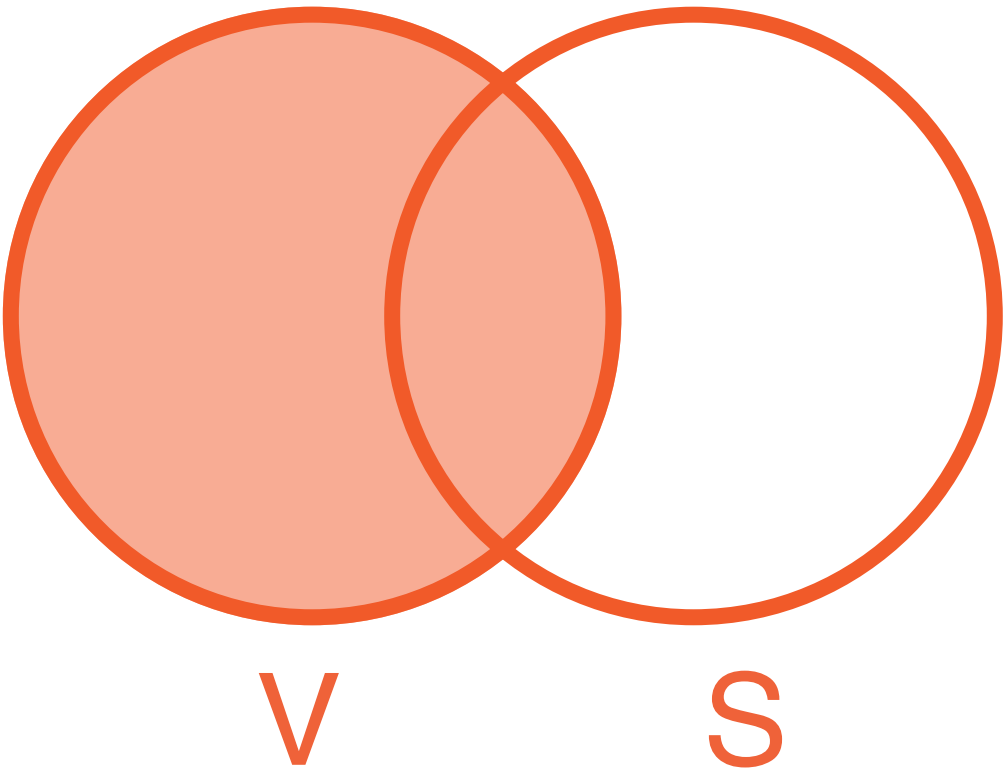
SQL LEFT JOIN

Velo

id_velo	id_stn	lot
1	A	20B
2	A	21A
4	C	23R

Station

id_stn	place	name
A	12	Bastille
B	15	Halles
D	8	Voltaire



Tous les éléments de **V** et les éléments de **S** qui se retrouve dans **V**

Résultat

Le nombre de ligne du résultat dépendra du nombre de ligne dans la table du **FROM**

Exemple :

```
SELECT *  
FROM Velo as V  
LEFT JOIN Station as S  
ON S.id_stn = V.id_stn
```

id_velo	id_stn	name	place	lot
1	A	Bastille	12	20B
2	A	Halles	12	21A
4	C			23R

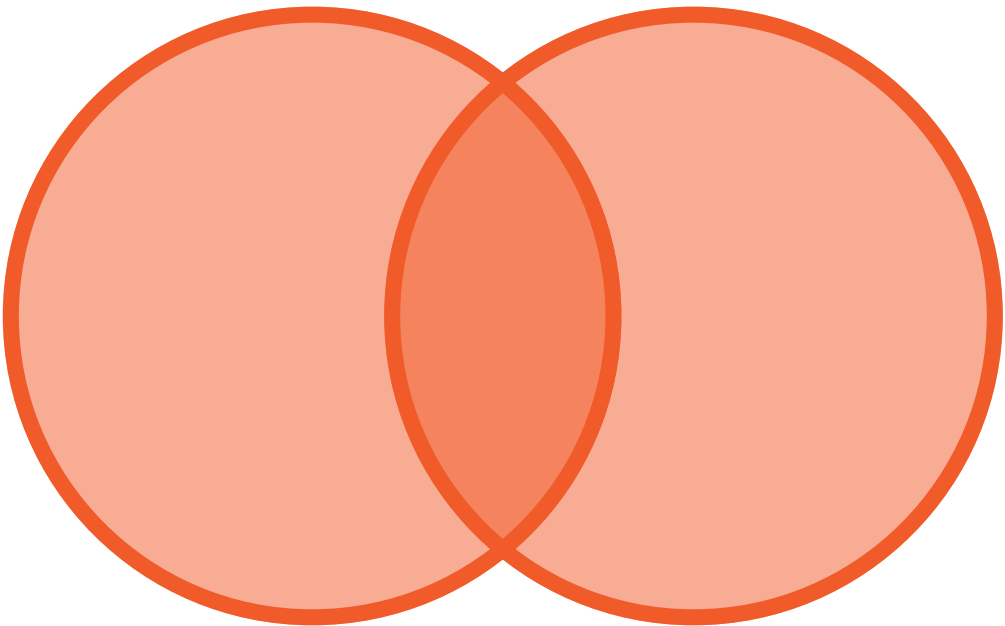
SQL FULL OUTER JOIN

Velo

id_velo	id_stn	lot
1	A	20B
2	A	21A
4	C	23R

Station

id_stn	place	name
A	12	Bastille
B	15	Halles
D	8	Voltaire



Tous les éléments de
V et de **S**

Résultat

id_velo	id_stn	name	place	lot
1	A	Bastille	12	20B
2	A	Halles	12	21A
4	C			23R
	B	Halles	15	
	D	Voltair	8	

```
Exemple :  
SELECT *  
FROM Velo as V  
FULL OUTER JOIN Station as S  
ON S.id_stn = V.id_stn
```


Jointure en python

```
join_df = df.merge(df2, on='key', how='inner')
```

```
# join_df est la jointure interne de df et df2
```

```
# la clef de jointure est la colonne key
```

```
# how= inner, left, right ou outer
```

```
join_df
```

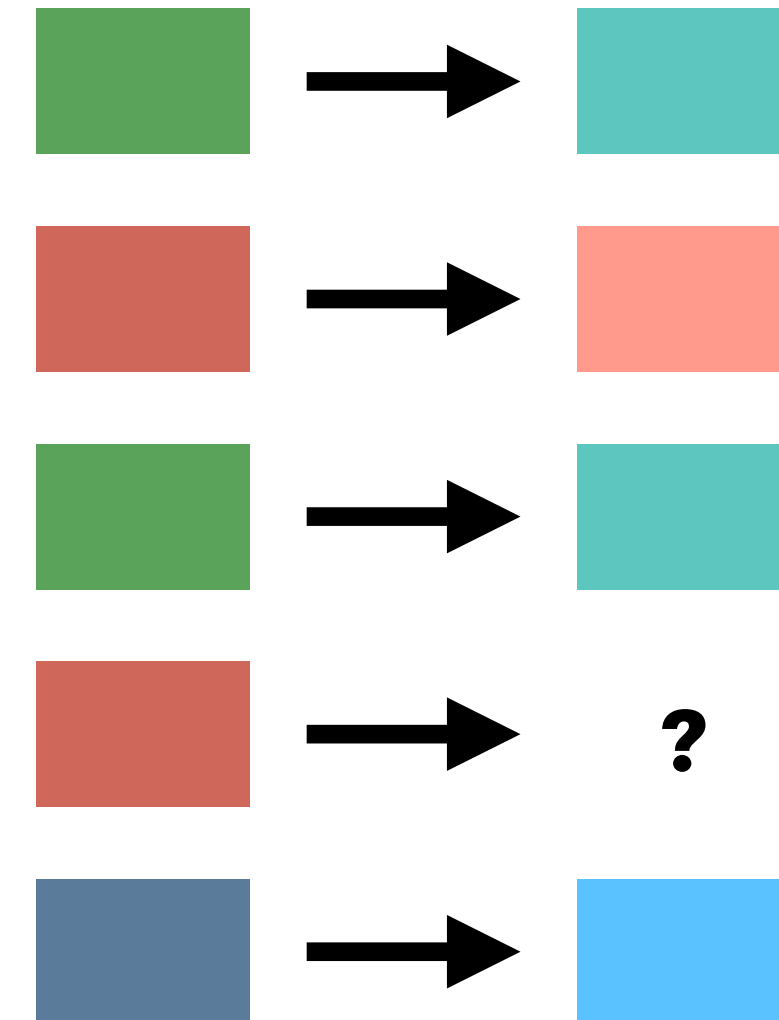
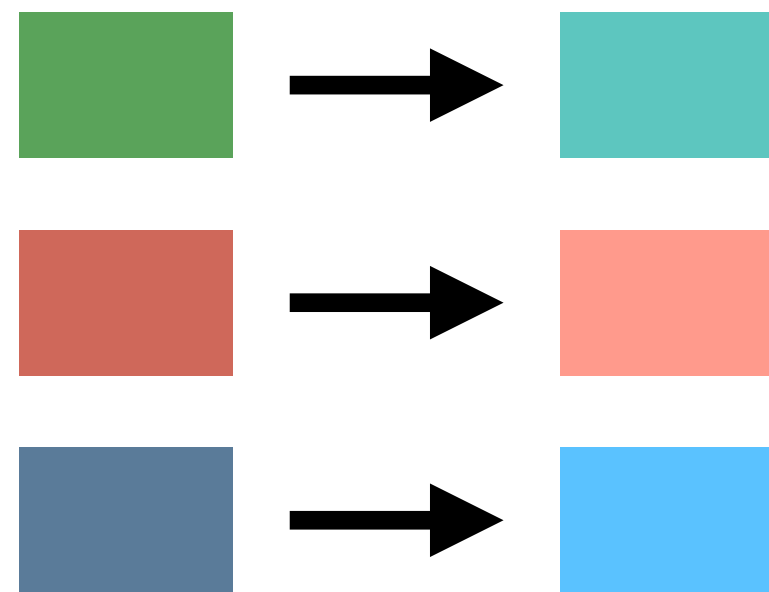
```
# En tapant join_df, on a un aperçu du dataset
```

Cas concret

```
sell_prod = products.merge(orders, on='product_id',  
                           how='inner')  
# sell_prod est la jointure interne des produits vendus  
# et des caractéristiques des produits  
  
sell_prod.head()  
# En tapant sell_prod, on a un aperçu du dataset
```

Associer une colonne à une autre - le mapping

Règle:



Associer une variable à une autre en python

characteristics = carac
carat

Que représente **col**?

pour le savoir :

<https://www.kaggle.com/datasets/ahmedlhlou/accidents-in-france-from-2005-to-2016>

	Num_Acc	an	mois	jour	hrmn	lum	agg	int	atm	col	com
0	2016000000001	16	2	1	1445	1	2	1	8	3.0	5
1	2016000000002	16	3	16	1800	1	2	6	1	6.0	5
2	2016000000003	16	7	13	1900	1	1	1	1	6.0	11
3	2016000000004	16	8	15	1930	2	2	1	7	3.0	477
4	2016000000005	16	12	23	1100	1	2	3	1	3.0	11
...
105579	201500046148	15	12	17	900	1	1	1	1	3.0	28
105580	201500046149	15	12	17	1810	5	1	1	1	6.0	27
105581	201500046150	15	12	18	1740	5	1	1	1	2.0	582
105582	201500046151	15	12	18	1830	3	1	1	1	4.0	18
105583	201500046152	15	12	19	300	5	1	1	1	2.0	27

105584 rows × 16 columns

Associer une variable à une autre en python

Grâce au **dictionnaire**, on crée une association {**clef**: **valeur**}

```
col_to_label = {  
1: 'Two vehicles frontal',  
2: 'Two vehicles rear',  
...,  
7: 'Without collision'}
```

col : Type of collision:

- 1 - Two vehicles - frontal
- 2 - Two vehicles - from the rear
- 3 - Two vehicles - by the side
- 4 - Three vehicles and more - in chain
- 5 - Three or more vehicles - multiple collisions
- 6 - Other collision
- 7 - Without collision

Associer une variable à une autre en python

On va créer cette association

```
carac['col_label'] = carac['col'].map(col_to_label)
```

```
col_to_label = {  
1: 'Two vehicles frontal',  
2: 'Two vehicles rear',  
...,  
7: 'Without collision'}
```

	Num_Acc	col	col_label
0	201600000001	3.0	Two vehicles - by the side
1	201600000002	6.0	Other collision
2	201600000003	6.0	Other collision
3	201600000004	3.0	Two vehicles - by the side
4	201600000005	3.0	Two vehicles - by the side
...
105579	201500046148	3.0	Two vehicles - by the side
105580	201500046149	6.0	Other collision
105581	201500046150	2.0	Two vehicles - from the rear
105582	201500046151	4.0	Three vehicles and more - in chain
105583	201500046152	2.0	Two vehicles - from the rear
105584	201500046153	2.0	Two vehicles - from the rear

Tableau de Contingence

<div>Ville</div> <div>Age</div>	Paris	Marseilles
25-34ans	45	67
35-44ans	55	23
>44 ans	43	55

Crosstab pour les tableaux de contingence

```
pd.crosstab(df.col1, df.col2, margins=True)  
# nom_du_dataset.nom_colonne
```

Faites un cas concret

Questions

1. Importer la table “**user.csv**”
2. Compter le nombre de personnes impliquées par accident
3. Joindre la table user à vehicles
4. Compter le nombre de personnes impliquées en moyenne par type de véhicule **catv**
5. Trouvez les accidents où il y a eu le plus de personnes impliquées
6. Faire un tableau de contingence entre gravité et place (avec les labels)

Sauvegarder votre nouveau jeu de données

```
new_df.to_csv('chemin/nom_fichier.csv')
```

```
# Enregistré sous le nom choisi
```

```
# Si le chemin n'est pas précisé, le fichier à la source
```

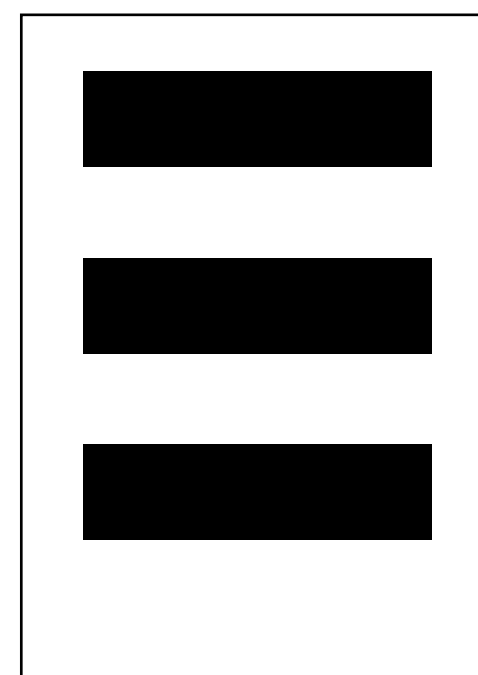
```
new_df.to_excel('chemin/nom_fichier.xlsx',  
                sheet_name='Sheet_name_1')
```

```
# sheet_name n'est pas obligatoire
```

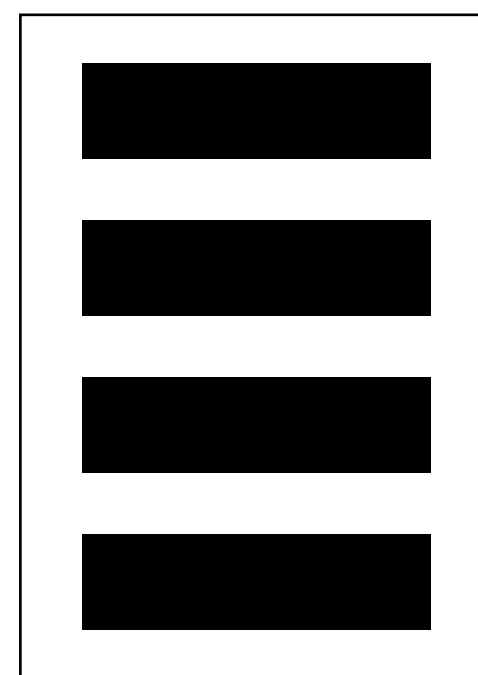
Analyse concrète de panier de consommation

Rappel des notions:

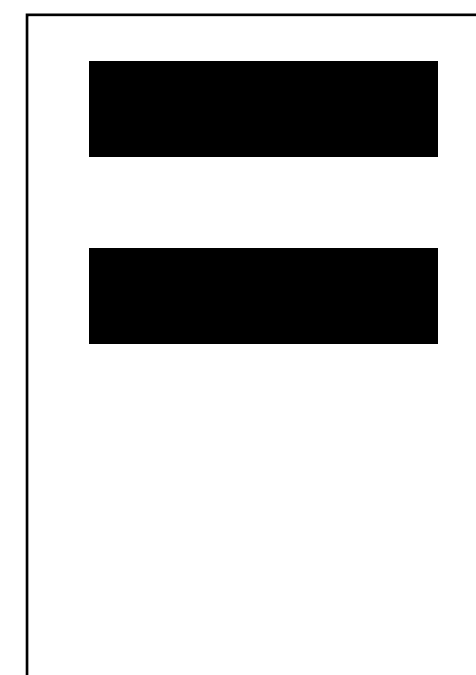
Support : Le taux de présence d'un article par rapport à l'ensemble des tickets de caisses



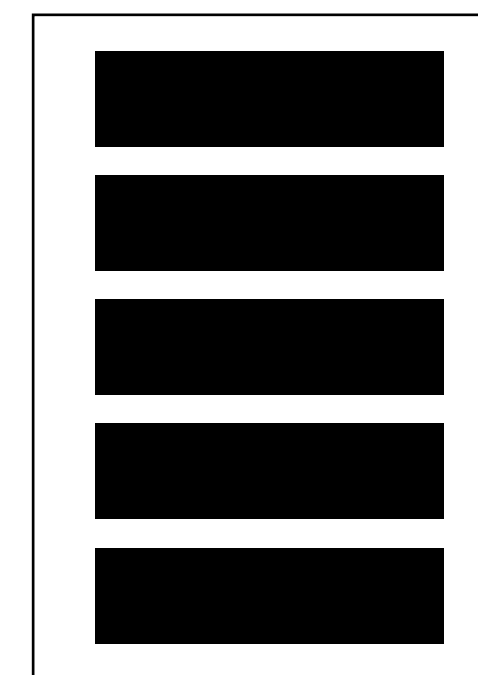
Ticket 1



Ticket 2



Ticket 3



Ticket 4

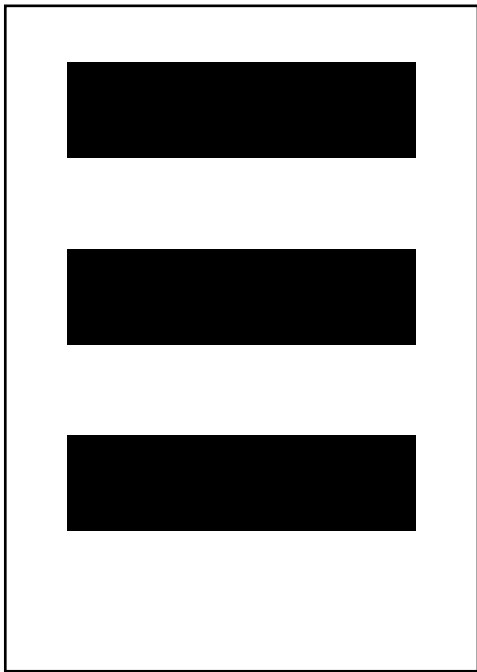
Rappel des notions:

Support : Le taux de présence d'un article par rapport à l'ensemble des tickets de caisses

Article jaune

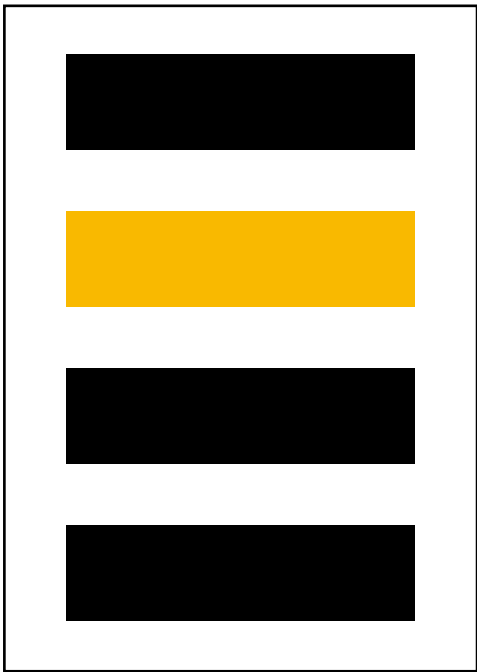


0



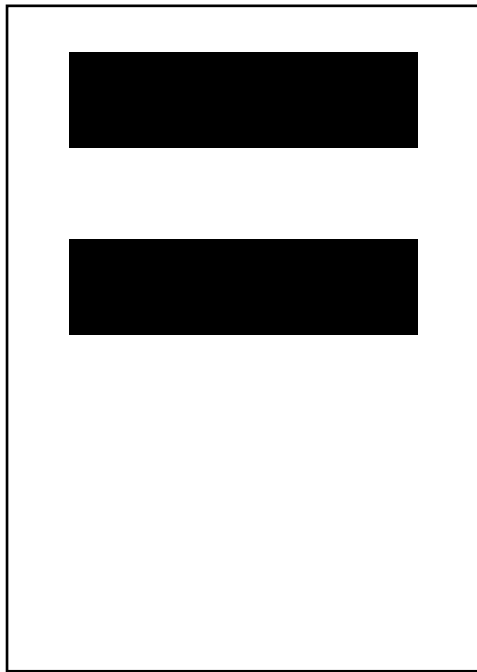
Ticket 1

1



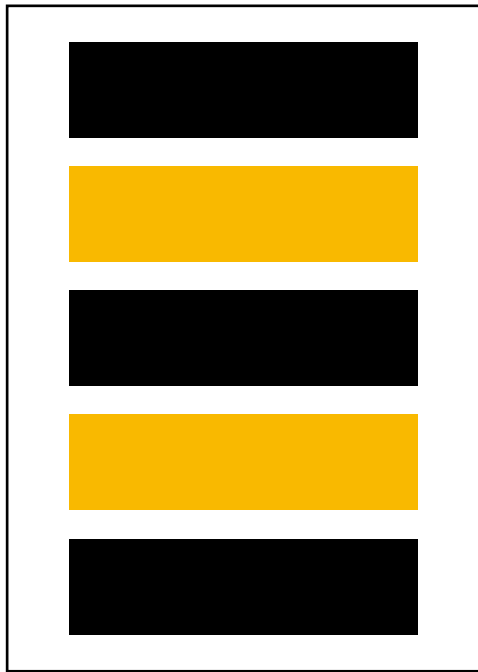
Ticket 2

0



Ticket 3

1



Ticket 4

Support de
l'article jaune:

present sur 2 tickets,
ensemble de 4 tickets

 $2/4 = 50\%$

Rappel des notions:

Confidence : Le taux de présence de l'item B quand l'item A est acheté

Article jaune



Article bleu



Quel est la chance de voir l'article bleu quand on achète le jaune?

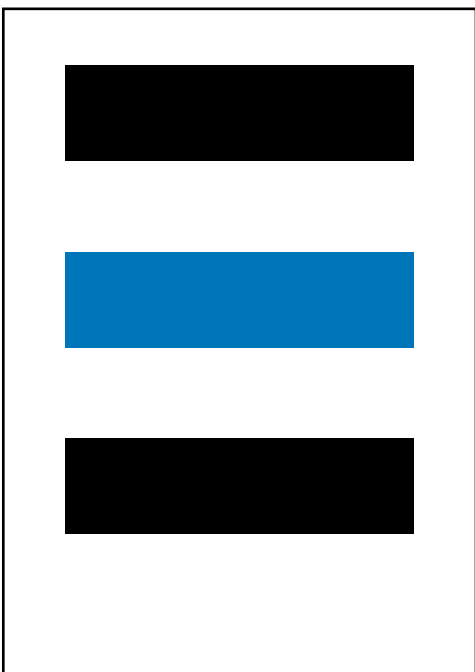
Rappel des notions:

Confidence : Le taux de présence de l'item B quand l'item A est acheté

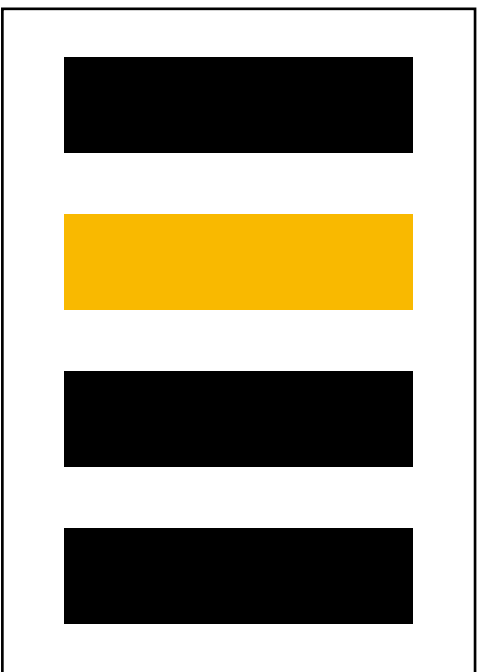
Article jaune



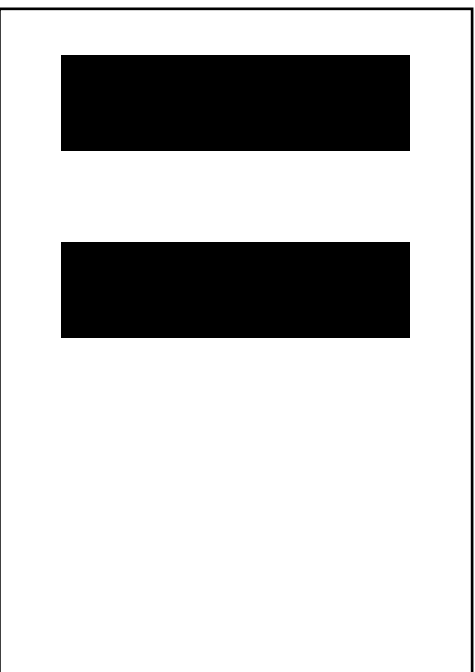
Article bleu



Ticket 1



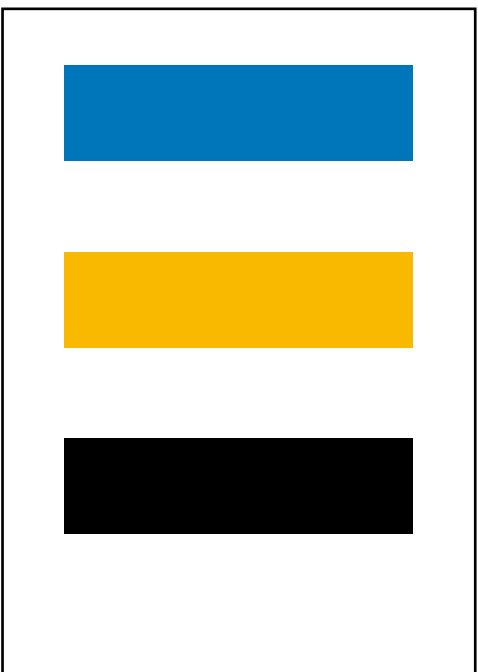
Ticket 2



Ticket 3



Ticket 4



Ticket 5

Jaune -> Bleu

bleu et jaune présents sur 2
jaune présent sur 3 tickets

$2/3 = 66,6\%$

Début de l'activité

Questions Partie 1

1. Importez la table “`transactions_by_dept.csv`”
2. Comptez le nombre de tickets de caisse
3. Comptez le nombre d'articles
4. Renommez Ticket de Caisse, Type d'article et l'id avec des noms en minuscule et sans espace
5. Réalisez un tableau de contingence entre ticket de caisse et type d'article (`sans les marges`)
6. Créez une colonne qui correspond à la somme des article par ticket de caisse. `La réponse est ci-dessous`
`reponse_Q_precedente['all'] = reponse_Q_precedente.sum(axis=1)`
`reponse_Q_precedente`

Questions Partie 2

1. Gardez uniquement les tickets de plus d'un type d'articles (basez vous sur la réponse de la question précédente)
2. Supprimez la variable qui somme les articles par ticket de caisse
`df.drop(columns=['col1', 'col2'])` #Remplacez col1, col2 par les véritables noms
3. Importez les fonctions `apriori` et `association_rules` du module `mlxtend.frequent_patterns`
4. Testez vos résultats en exécutant la commande suivante
`apriori(response_Question2, min_support=0.05, use_colnames=True)`

Visualisation

5 graphiques de base



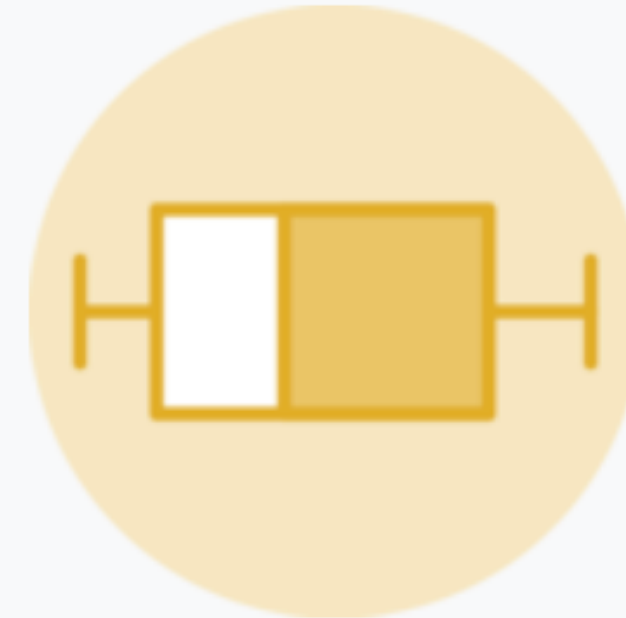
Scatter



Line plot



Histogram



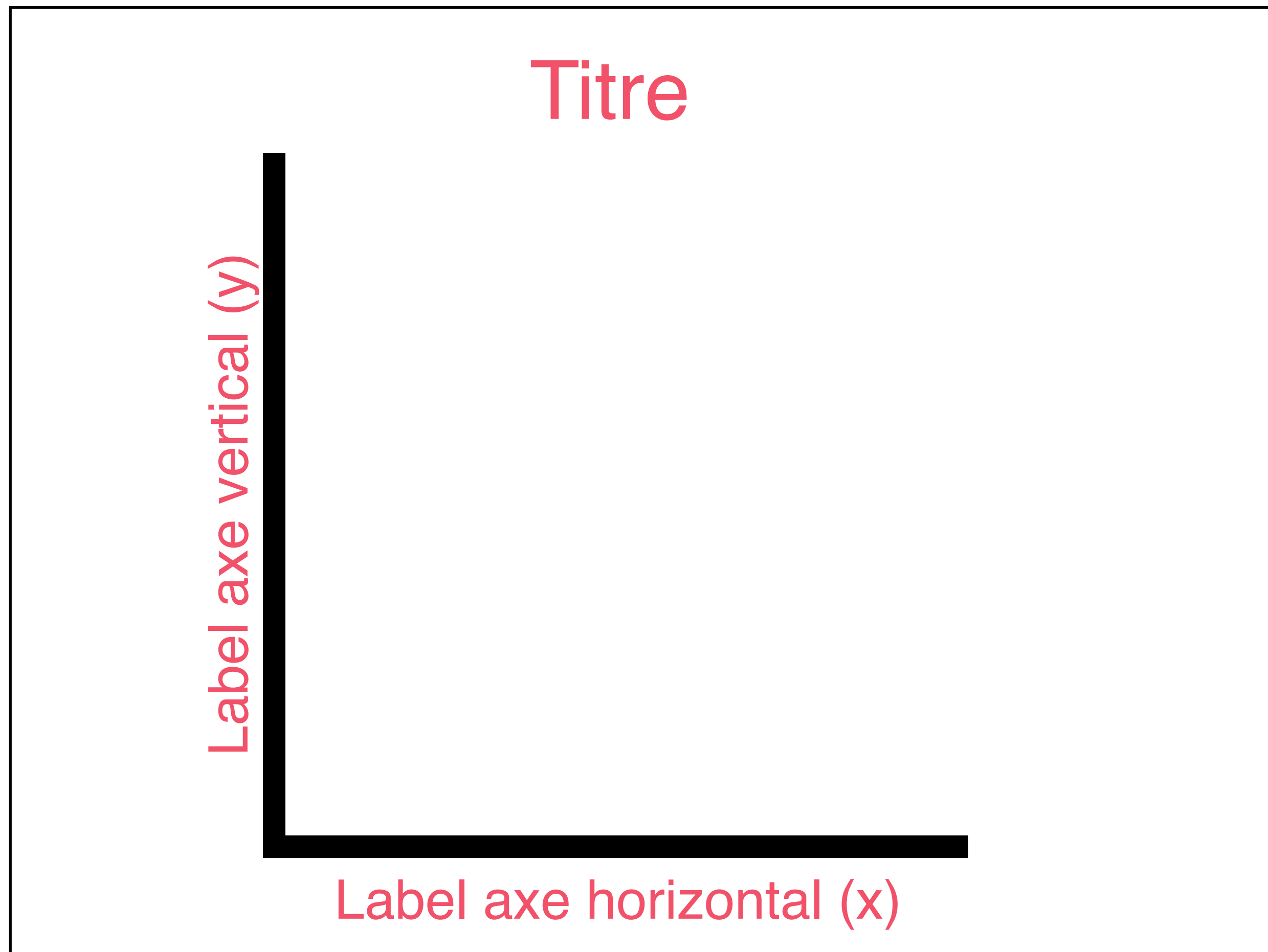
Boxplot



Pie chart

Quelques Rappels

Structure d'un graph



Titre

Choisir un titre qui décrit ce que vous voulez mettre en avant

Axe x

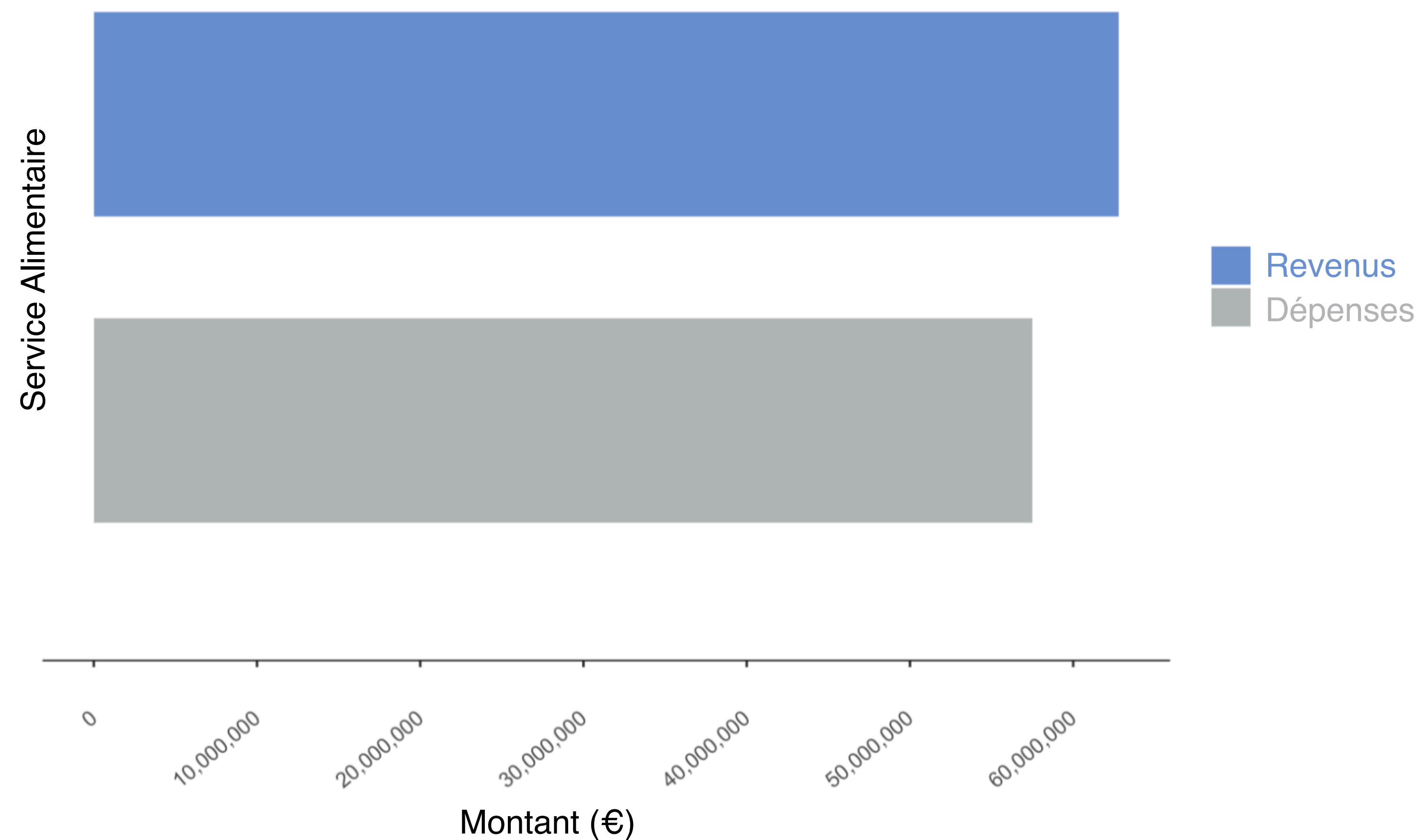
Choisir la variable qui est la cause, celle qui est indépendante

Axe y

Choisir la variable qui est la conséquence, l'effet

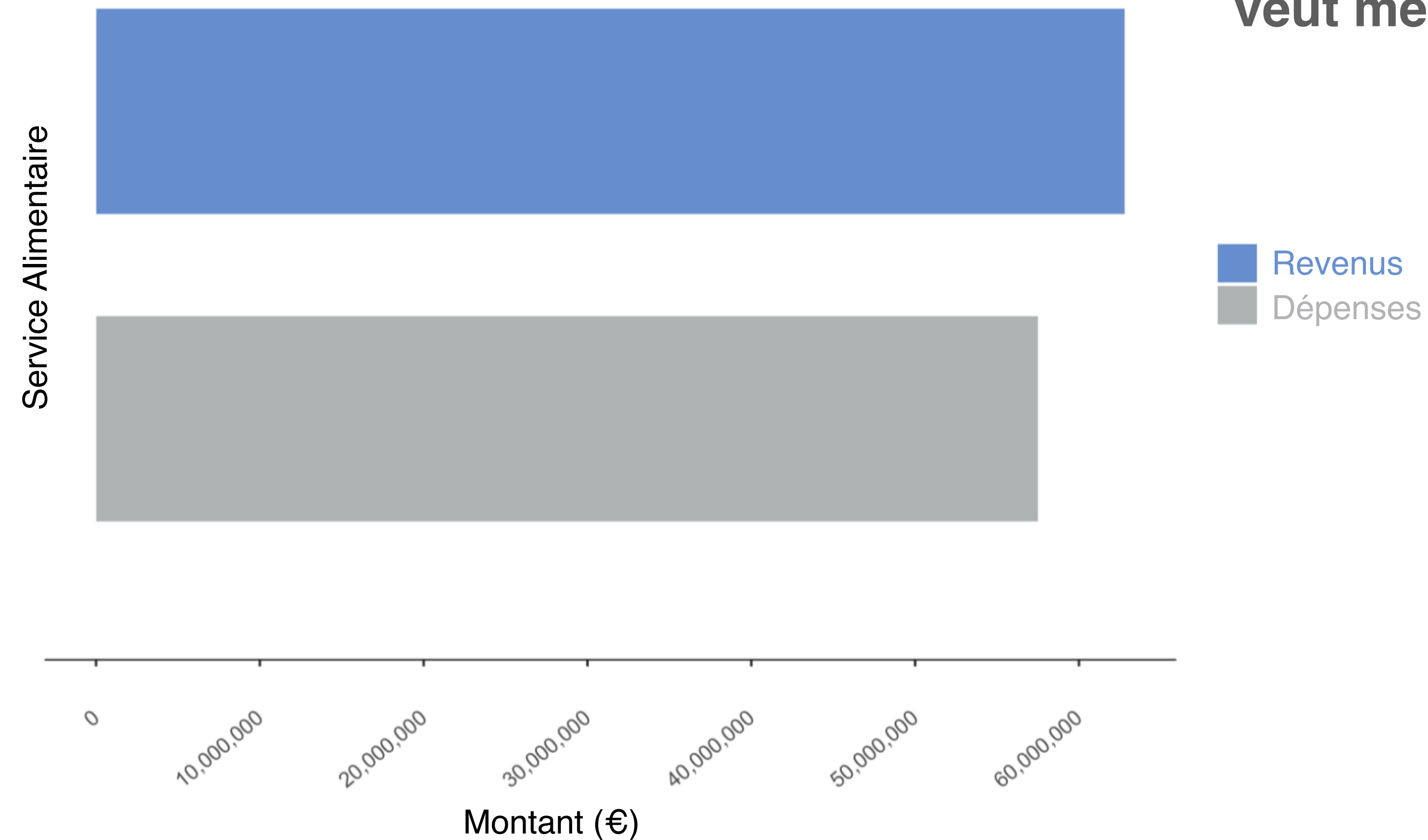
Exemples

Revenus et Dépenses liés
au service alimentaire en 2021



Exemples

Revenus et Dépenses liés
au service alimentaire en 2021



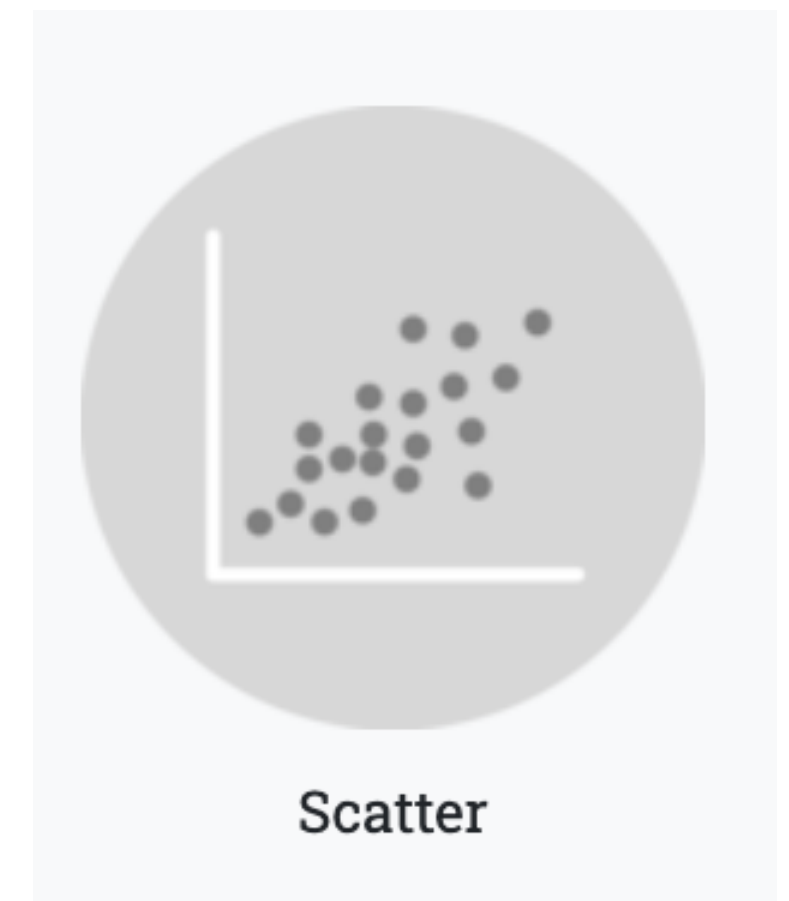
Titre décrivant ce que l'on voit décrit
mais ne décrivant pas ce que l'on
veut mettre en avant

Les graphes

Scatterplot (nuage de points)

Utiliser pour représenter 2 variables continues

Pour montrer une tendance, visualiser des groupes



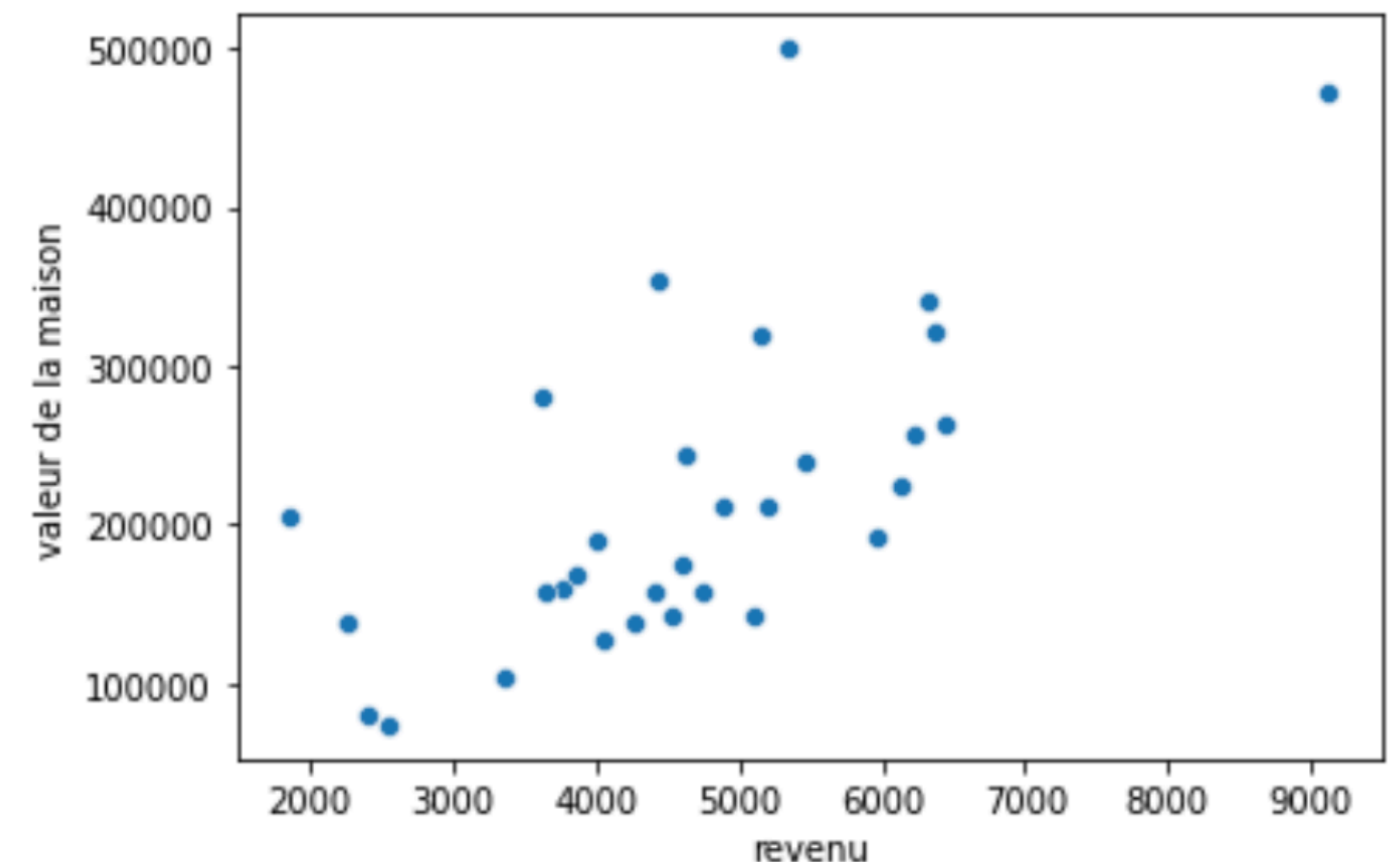
Exemples:

Immo: valeur maison vs revenus

```
import seaborn as sb
from matplotlib import pyplot as plt

sb.scatterplot(data=california,
               x="median_income",
               y="median_house_value").
set(title="Plus les revenus...",
     xlabel='revenu',
     ylabel='valeur de la maison')
```

Tendance: Les ménages aux revenus élevés ont des biens plus valorisés



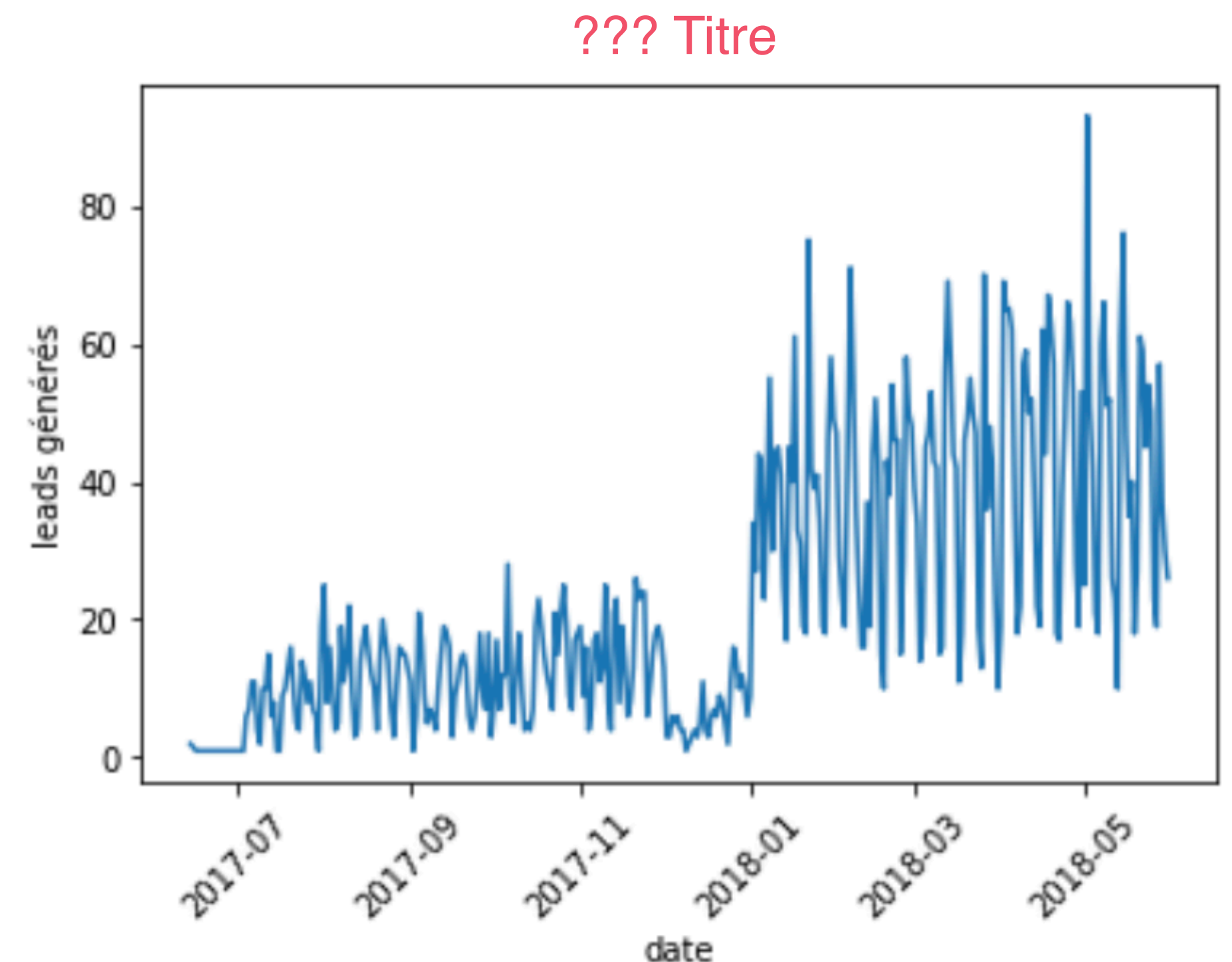
Lineplot (diagramme en lignes)

Utiliser pour représenter 2 variables continues
Parfait pour montrer une évolution

Exemples:

marketing: Nombre leads par jours

```
sb.lineplot(data=nb_leads,  
            x="date",  
            y="count").  
set(title="???",  
    xlabel='date',  
    ylabel='leads générés')
```



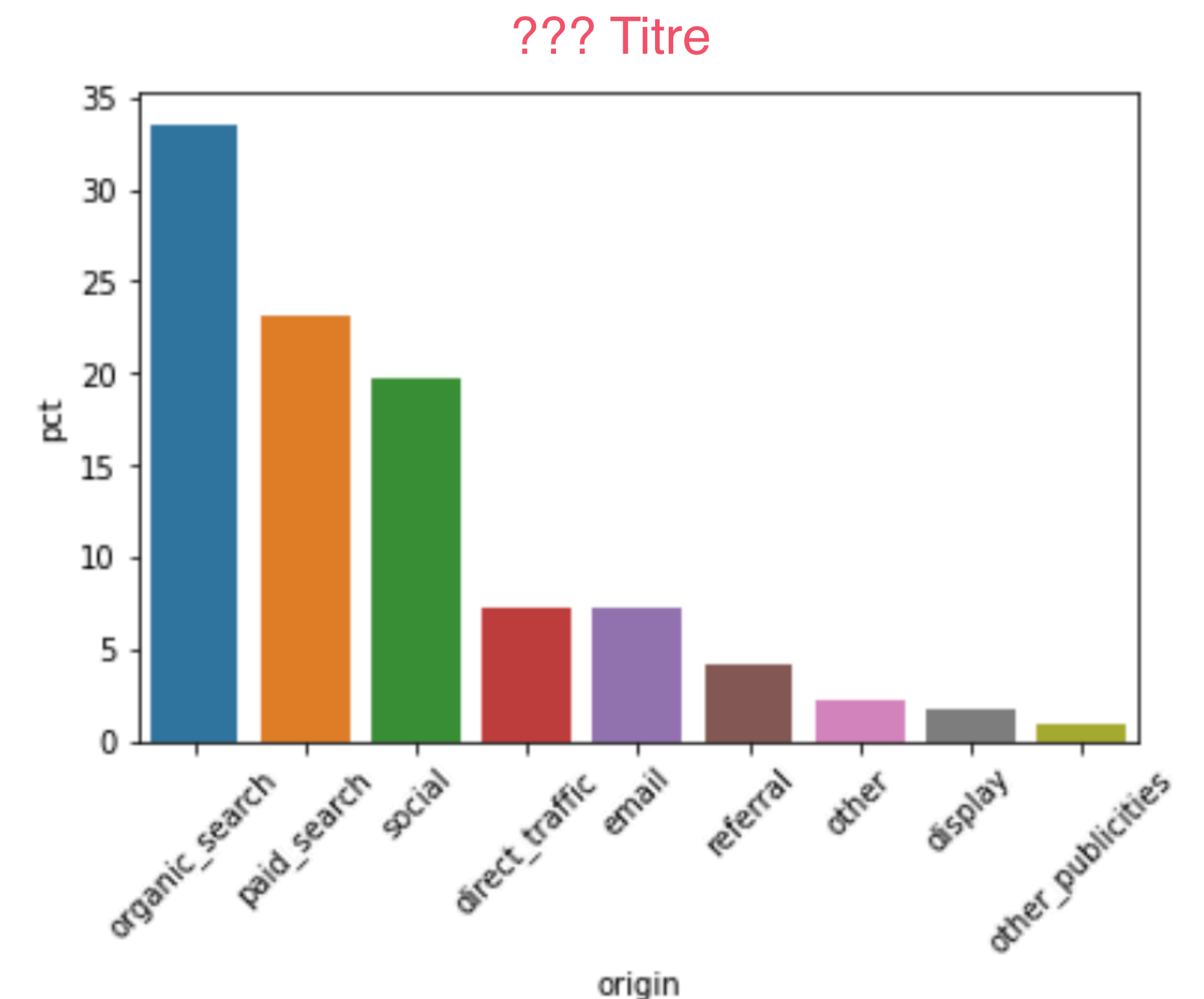
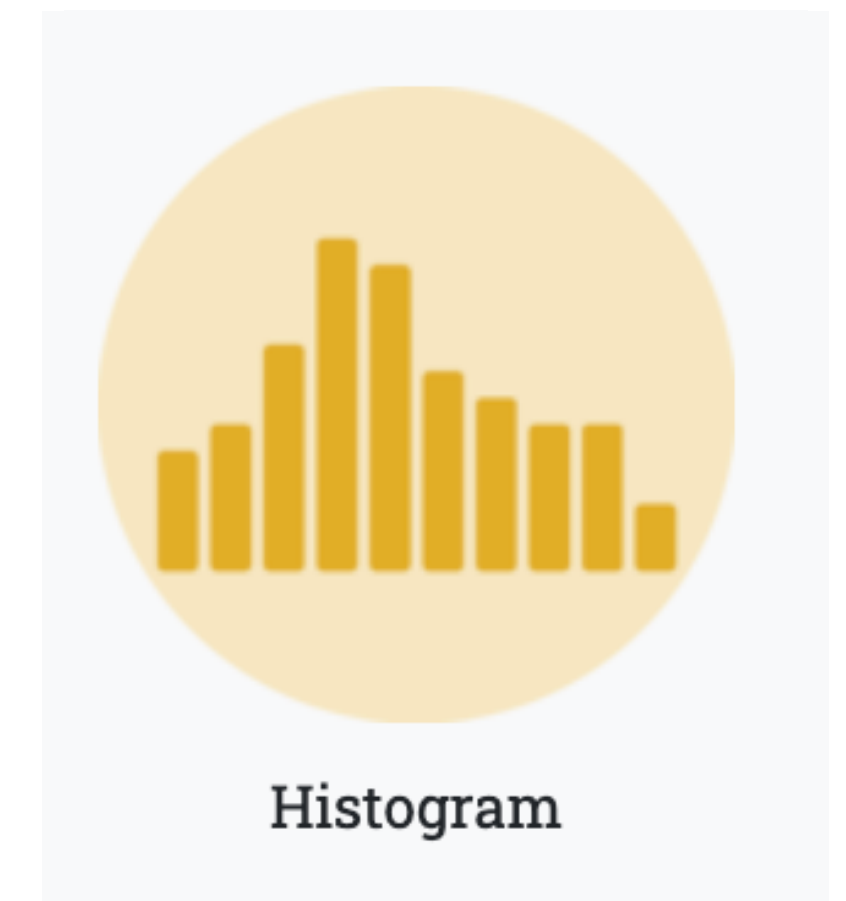
barplot (diagramme en baton)

Utiliser pour représenter :
Une variable catégorielle et une continue
Voir des proportions, comparer des catégories

Exemples:

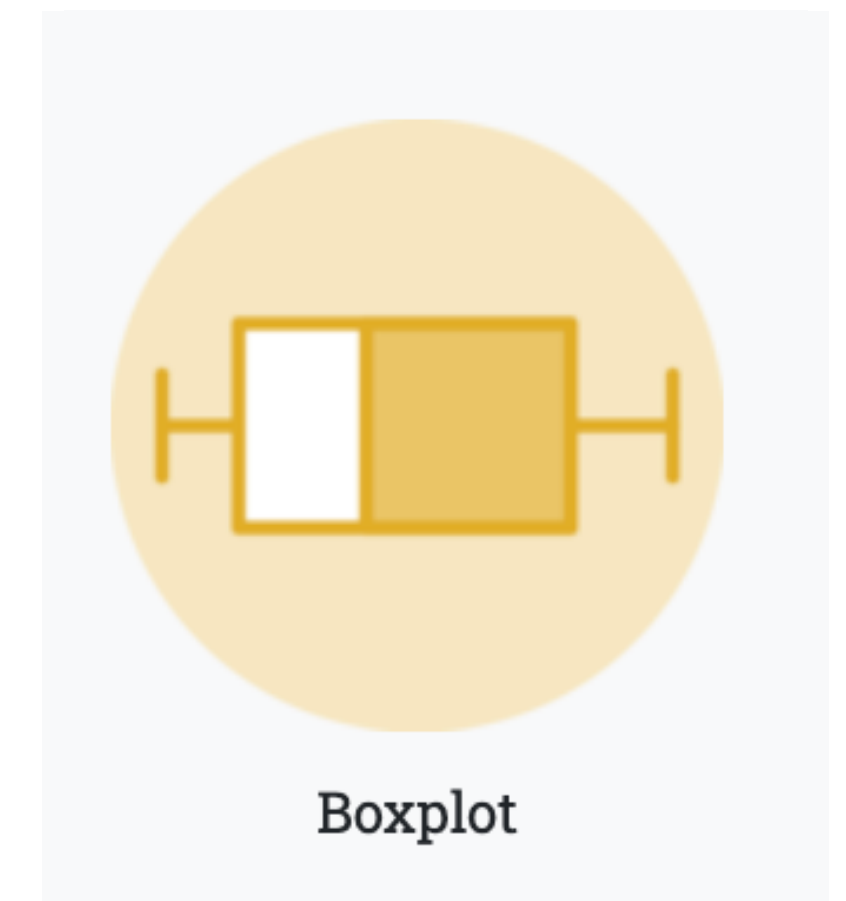
Provenance des leads (en pct)

```
sb.barplot(data=nb_origin,  
           x="origin",  
           y="pct").  
set(title="???",  
    xlabel='date',  
    ylabel='leads générés')
```



Boxplot (Boîte à moustaches)

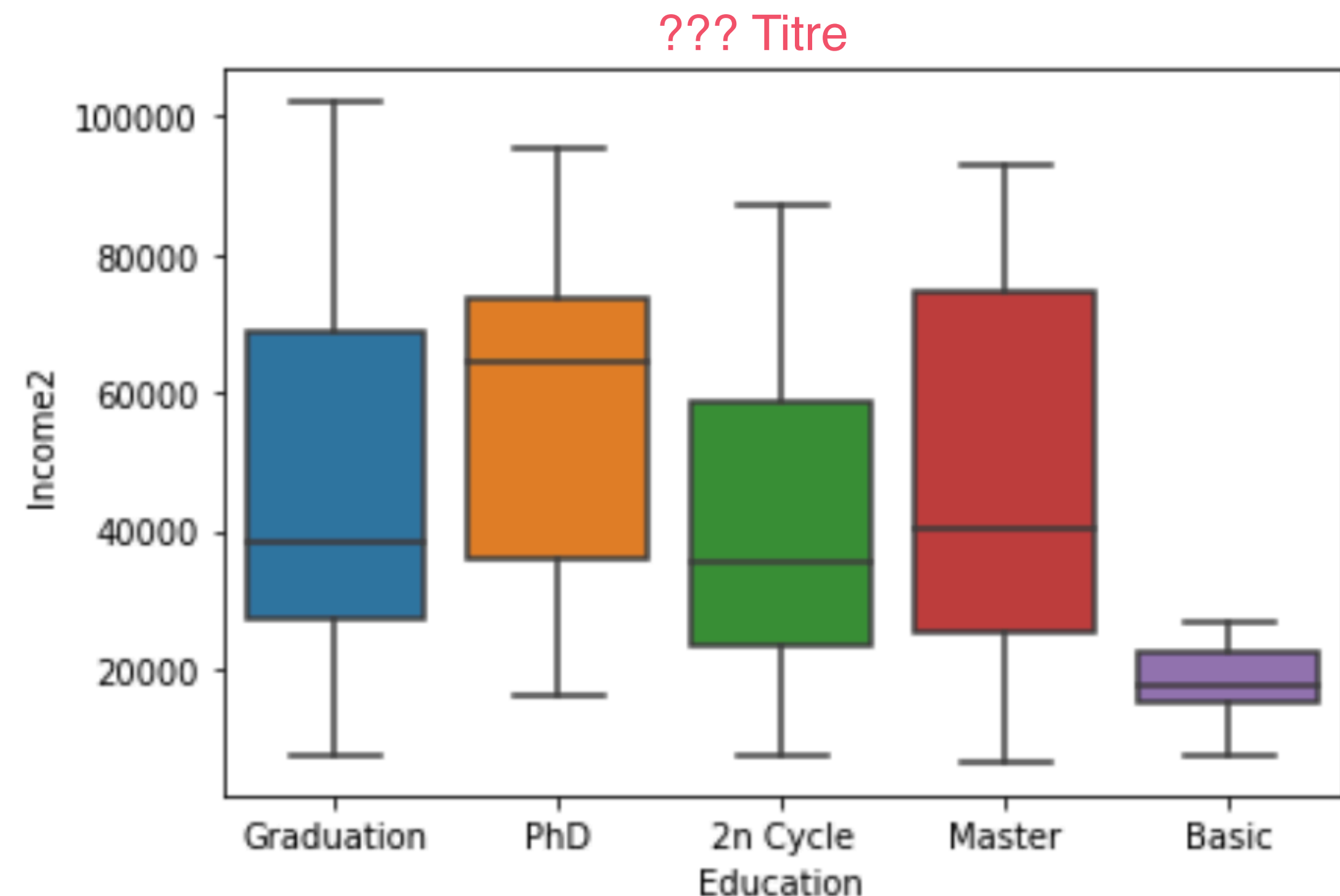
Utiliser pour représenter la répartition des données d'une continue par rapport à une catégorielle



Exemples:

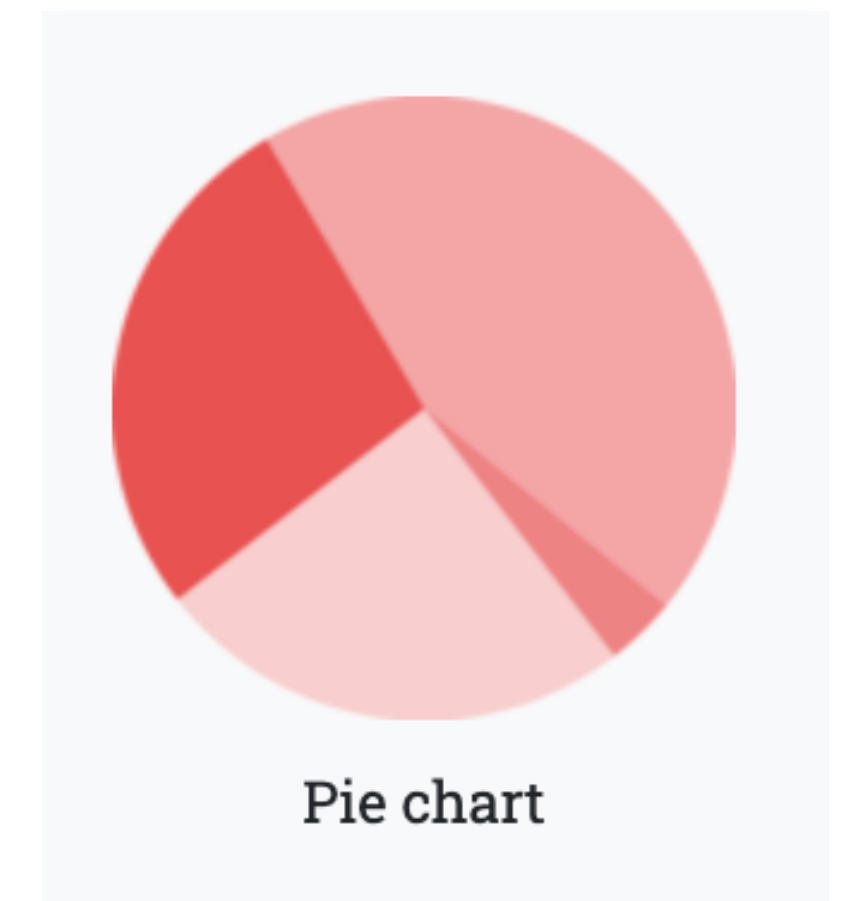
Répartition: Salaire en fonction des études

```
sb.boxplot(data=market,  
x="Education",  
y="Income2").  
set(title="???",  
xlabel='date',  
ylabel='leads générés')
```



Piechart (Camembert)

Utiliser pour voir la proportion des occurrences d'une variable

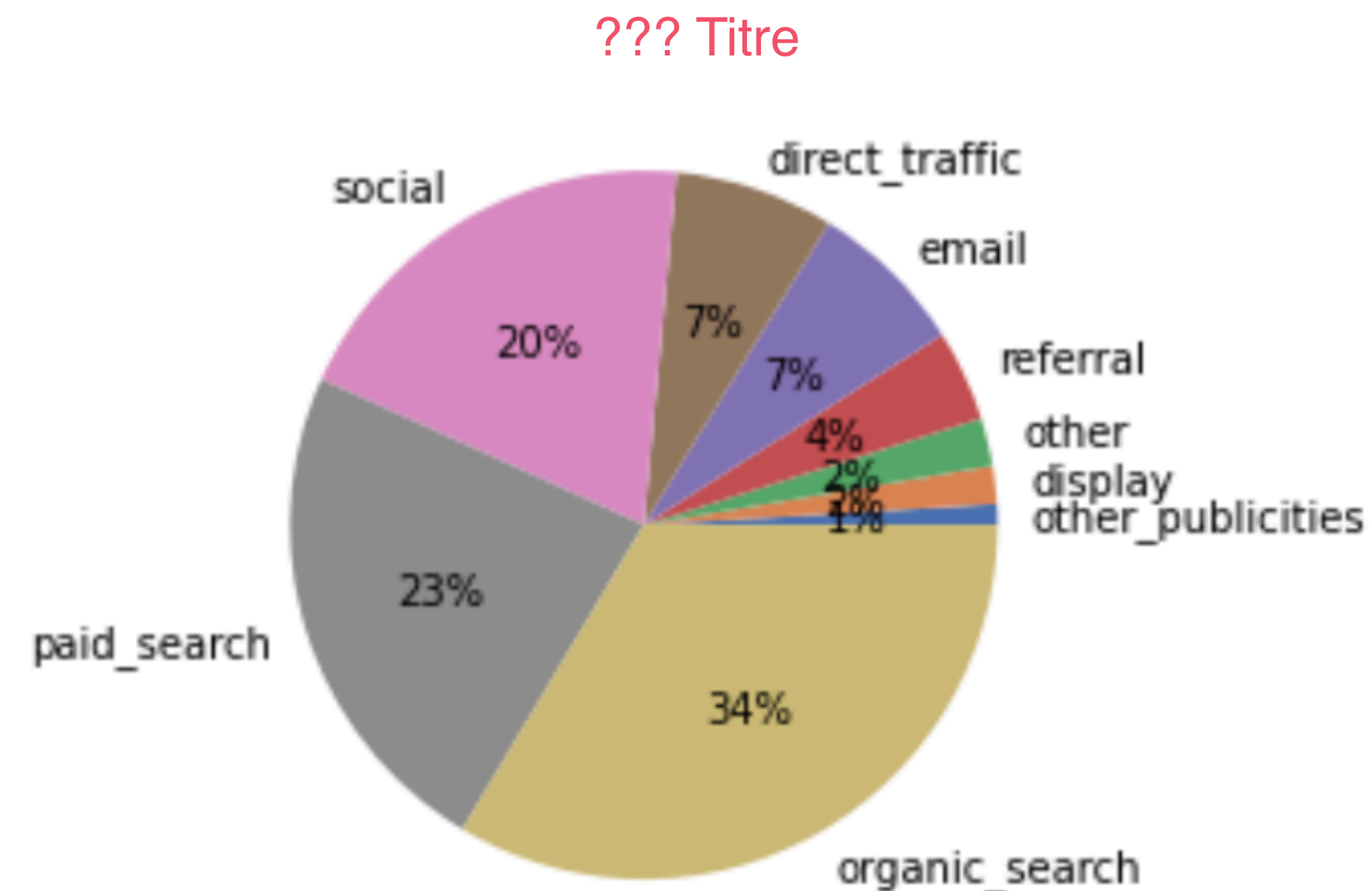


Exemples:

Provenance des leads (en pct)

```
plt.pie(data=market,  
x=nb_origin2['n'],  
labels=nb_origin2['origin'],  
autopct='%0f%%')
```

Syntaxe différente car package
différent **plt** au lieu de **sb**



Ne recommande pas l'usage des piecharts, utilisez plutôt les barplots

La pratique

Ressources pour aller plus loin

Formation en ligne

<https://www.codecademy.com/catalog/language/python>

Livre

[Data Analysis with Python - Wes McKinney](#)



linkedin.com/in/krosamont/

MERCI

Contactez moi par message
si vous avez des questions

Kevin Rosamont