# 📈 Travail Dirigé (TD) - Visualisation

---

## Consignes

### - Jeu de données : data_exploration.csv.

---

In [2]:

```
#Importation des bibliothèques nécessaires.
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Data Description (very Quick). Rename the DataFrame variable (df) if needed.
df = pd.read_csv('./data_exploration.csv', sep=';')

print(f'\n ** Noms et Types des attributs **: \n{df.dtypes}\n\n------\n')
print(f'\n ** Apercu des données **: \n{df.head(5)}')
```

```
 ** Noms et Types des attributs **:
Username        object
Subscribers    float64
Views            int64
Country         object
dtype: object

------


 ** Apercu des données **:
                   Username  Subscribers        Views Country
0                  Colors TV         77.6  76490031656      IN
1              shripraashant         53.9   3237975581      IN
2                Alan Walker         46.2  14996804659      NO
3          SonyMusicIndiaVEVO         53.2  33399550869      US
4  LUCCAS NETO - LICCAS TOON         50.1  29698114509      BR
```

---

## Question 1

### - Nous souhaitons utiliser un nuage de points (scatterplot) pour vérifier si le nombre de vues des chaînes YouTube est proportionnel à celui des abonnés.

### - Figure à réaliser (voir exemple 1):

Figure 2D (nuage de points) avec :

- en abscisses les Youtubers classés du plus suivi au moins suivi (attribut Subscribers)
- et en ordonnées les nombres de vues des Youtubers

https://seaborn.pydata.org/generated/seaborn.scatterplot.html

### - Exemple 1

title

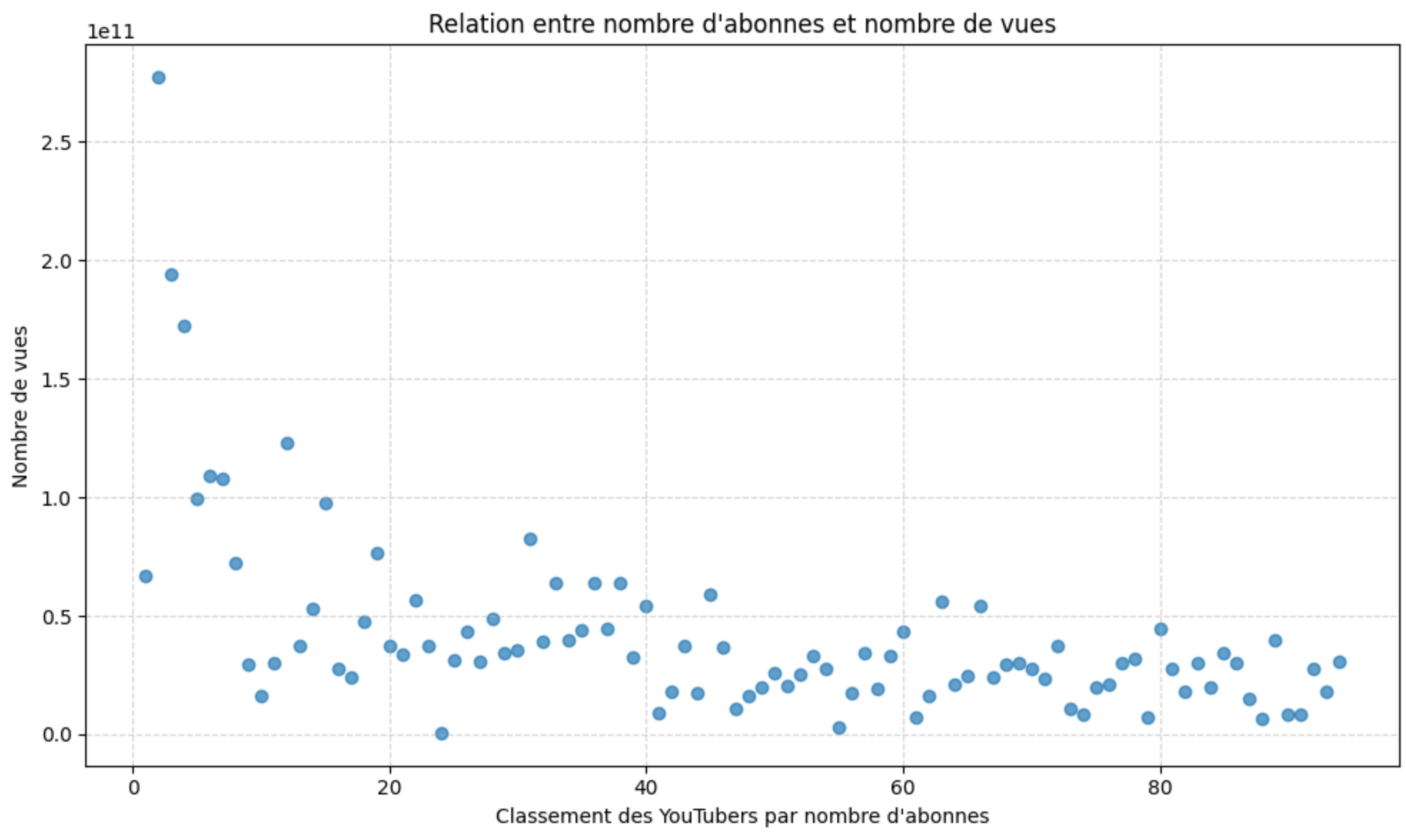
## Réponse 1

In [3]:

```
# Trier par nombre d'abonnés (du plus au moins suivi)
df_sorted = df.sort_values(by="Subscribers", ascending=False).reset_index(drop=True)

df_sorted["Rank"] = df_sorted.index + 1
```

In [16]:

```
# Scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(df_sorted["Rank"], df_sorted["Views"], alpha=0.7)

# Mise en forme
plt.xlabel("Classement des YouTubers par nombre d'abonnes")
plt.ylabel("Nombre de vues")
plt.title("Relation entre nombre d'abonnes et nombre de vues")

plt.grid(True, linestyle="--", alpha=0.5)
plt.tight_layout()
plt.show()
```



## Question 2

### - Nous souhaitons utiliser la visualisation en boîtes à moustaches (boxplots) pour examiner la distribution des vues sur YouTube en fonction de pays spécifiques, afin de réaliser une comparaison.

### - Figure à réaliser (voir exemple 2):

Figure 2D (boxplots) avec :

- en abscisses les deux classes de YouTubers (1st Class et 2nd Class), groupées par pays (Canada, États-Unis, Japon et autres pays),
- et en ordonnées le nombre de vues des YouTubers,
- sachant qu'un YouTuber appartient à la 1st Class si le nombre de vues est strictement supérieur à 60Milliards, sinon il est classé dans la 2nd Class."

https://seaborn.pydata.org/generated/seaborn.boxplot.html#

### - Exemple 2

title

## Réponse 2

In [17]:

```
df["Famous_Youtuber"] = np.where(df['Views'].astype(int) > 60000000000, 1, 2)
conditions = [
    (df['Country'] == 'US'),
    (df['Country'] == 'JP'),
    (df['Country'] == 'CA')
]
choices = ['United States', 'Japan', 'Canada']

# Create the new 'result' column using np.select()
df['Country2'] = np.select(conditions, choices, default='Other Countries')
df.head()
```

Out[17]:

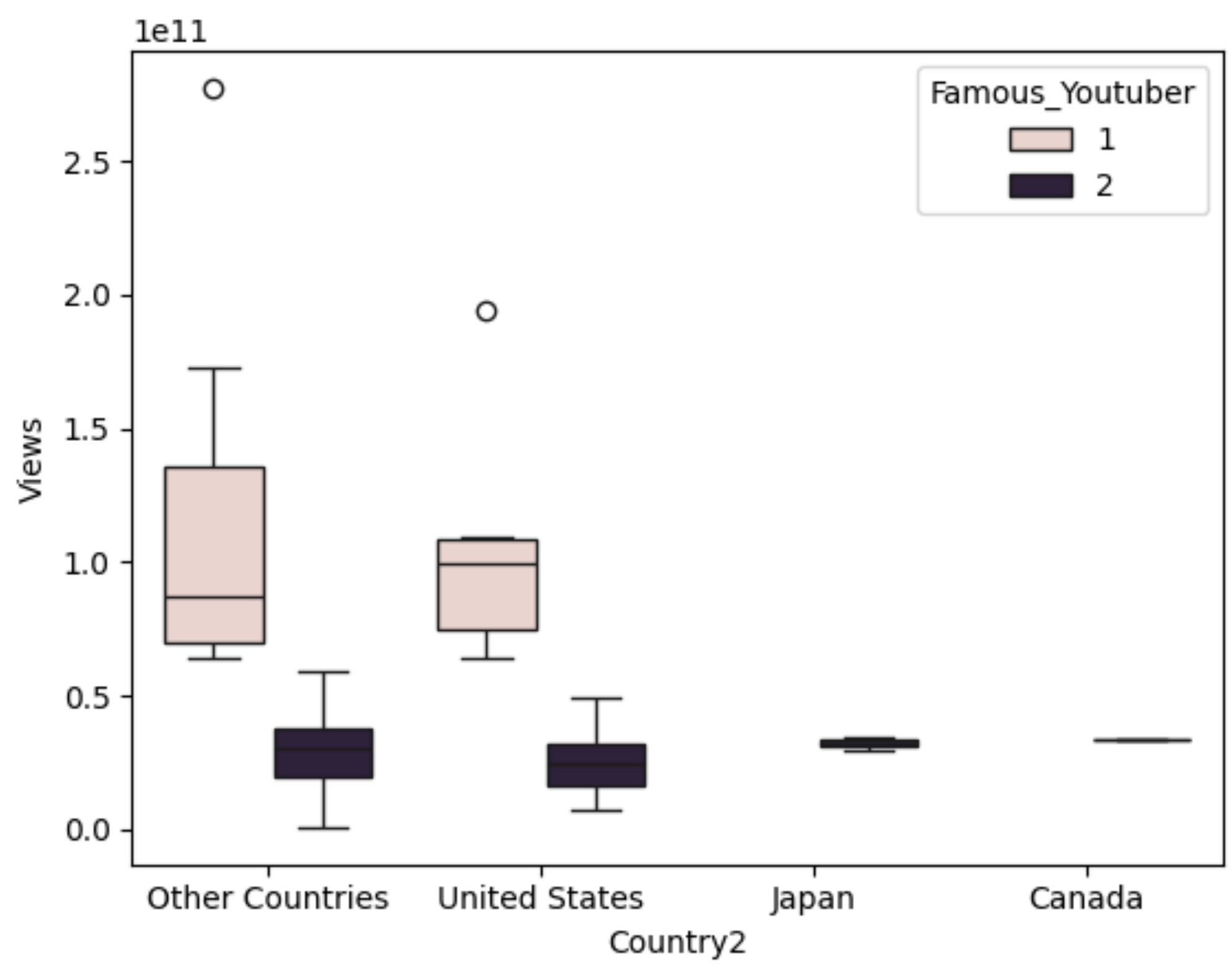
| | Username | Subscribers | Views | Country | Famous_Youtuber | Country2 | Country3 |
|---|---|---|---|---|---|---|---|
| **0** | Colors TV | 77.6 | 76490031656 | IN | 1 | Other Countries | India |
| **1** | shripraashant | 53.9 | 3237975581 | IN | 2 | Other Countries | India |
| **2** | Alan Walker | 46.2 | 14996804659 | NO | 2 | Other Countries | Other Countries |
| **3** | SonyMusicIndiaVEVO | 53.2 | 33399550869 | US | 2 | United States | United States |
| **4** | LUCCAS NETO - LICCAS TOON | 50.1 | 29698114509 | BR | 2 | Other Countries | Other Countries |

In [18]:

```
sns.boxplot(data=df, x='Country2', y='Views', hue="Famous_Youtuber", fill=True, gap=.1)
```

Out[18]:

```
<Axes: xlabel='Country2', ylabel='Views'>
```

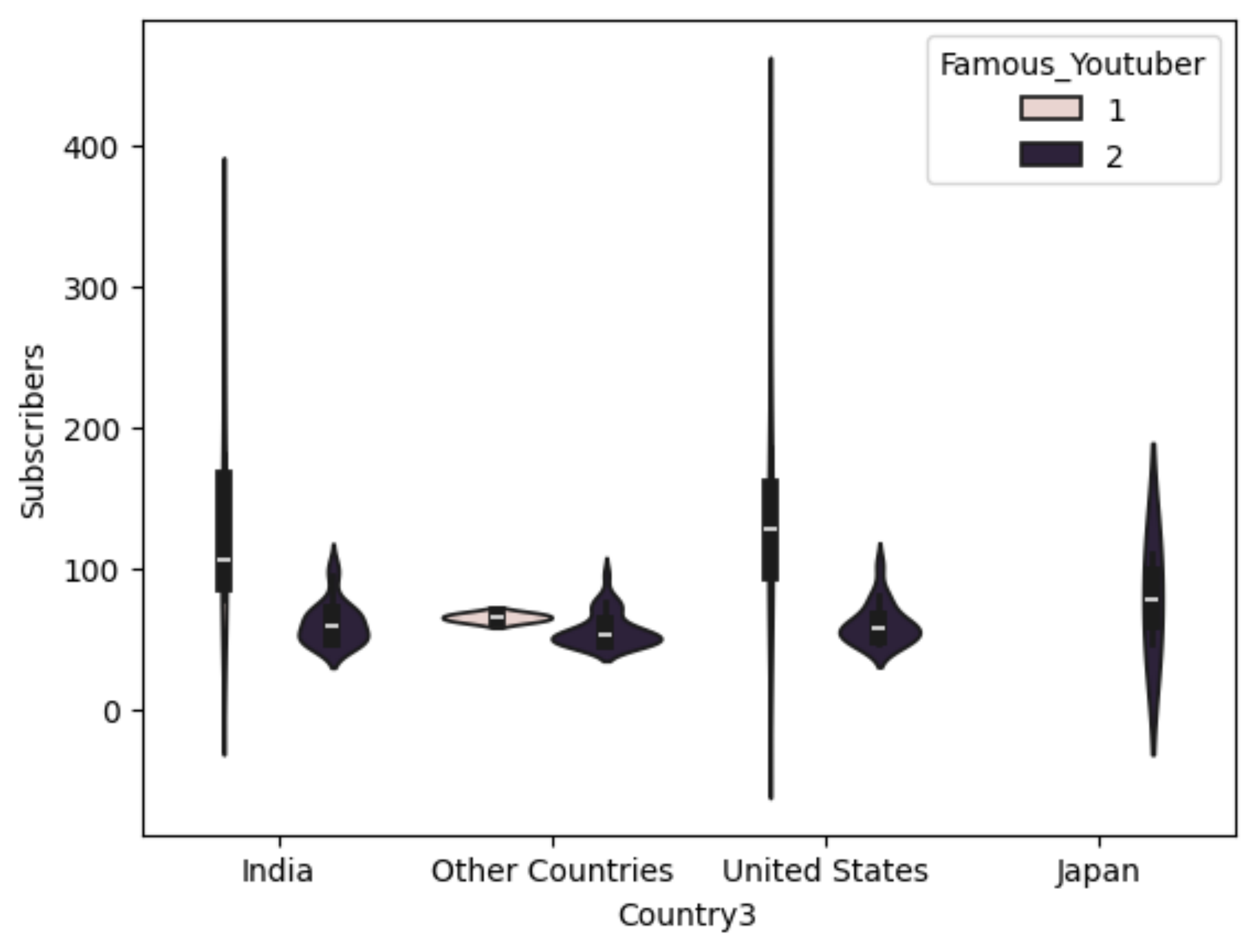


In [19]:

```
conditions = [
    (df['Country'] == 'US'),
    (df['Country'] == 'JP'),
    (df['Country'] == 'IN')
]
choices = ['United States', 'Japan', 'India']

df['Country3'] = np.select(conditions, choices, default='Other Countries')
sns.violinplot(data=df, x='Country3', y='Subscribers', hue="Famous_Youtuber")
```

Out[19]:

```
<Axes: xlabel='Country3', ylabel='Subscribers'>
```



In [ ]:

In [ ]: