IMPORT LIBRARIES

In [1]:
```python
# import library yang di butuhkan

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

import sqlite3
```

KONEKSIKAN KE DATABASE

In [2]:
```python
# koneksikan data ke sqlite3 database

conn = sqlite3.connect('chinook.db')
print("Database created and Successfully Connected to SQLite")
```

```
Database created and Successfully Connected to SQLite
```

In [3]:
```python
# test koneksi

test = conn.execute('select * from customers limit 10')
for i in test :
    print(i)
```

```
(1, 'Luís', 'Gonçalves', 'Embraer - Empresa Brasileira de Aeronáutica S.A.', 'Av. Brigadeiro
Faria Lima, 2170', 'São José dos Campos', 'SP', 'Brazil', '12227-000', '+55 (12) 3923-5555',
'+55 (12) 3923-5566', 'luisg@embraer.com.br', 3)
(2, 'Leonie', 'Köhler', None, 'Theodor-Heuss-Straße 34', 'Stuttgart', None, 'Germany', '7017
4', '+49 0711 2842222', None, 'leonekohler@surfeu.de', 5)
(3, 'François', 'Tremblay', None, '1498 rue Bélanger', 'Montréal', 'QC', 'Canada', 'H2G 1A7',
'+1 (514) 721-4711', None, 'ftremblay@gmail.com', 3)
(4, 'Bjørn', 'Hansen', None, 'Ullevålsveien 14', 'Oslo', None, 'Norway', '0171', '+47 22 44 2
2 22', None, 'bjorn.hansen@yahoo.no', 4)
(5, 'František', 'Wichterlová', 'JetBrains s.r.o.', 'Klanova 9/506', 'Prague', None, 'Czech R
epublic', '14700', '+420 2 4172 5555', '+420 2 4172 5555', 'frantisekw@jetbrains.com', 4)
(6, 'Helena', 'Holý', None, 'Rilská 3174/6', 'Prague', None, 'Czech Republic', '14300', '+420
2 4177 0449', None, 'hholy@gmail.com', 5)
(7, 'Astrid', 'Gruber', None, 'Rotenturmstraße 4, 1010 Innere Stadt', 'Vienne', None, 'Austri
a', '1010', '+43 01 5134505', None, 'astrid.gruber@apple.at', 5)
(8, 'Daan', 'Peeters', None, 'Grétrystraat 63', 'Brussels', None, 'Belgium', '1000', '+32 02
219 03 03', None, 'daan_peeters@apple.be', 4)
(9, 'Kara', 'Nielsen', None, 'Sønder Boulevard 51', 'Copenhagen', None, 'Denmark', '1720', '+
453 3331 9991', None, 'kara.nielsen@jubii.dk', 4)
(10, 'Eduardo', 'Martins', 'Woodstock Discos', 'Rua Dr. Falcão Filho, 155', 'São Paulo', 'S
```

In [4]:
```python
# query untuk membuat master table

query = """
with cte1 as(
select invoices.InvoiceId, invoice_items.TrackId, invoices.CustomerId, invoices.InvoiceDate,
customers.FirstName || ' ' || customers.LastName as 'Customer Name',
employees.FirstName || ' ' || employees.LastName as 'Employee Representative',
invoices.BillingCountry, invoices.Total
from invoices
join invoice_items on invoices.InvoiceId = invoice_items.InvoiceId
join customers on invoices.CustomerId = customers.CustomerId
join employees on customers.SupportRepId = employees.EmployeeId
order by invoices.InvoiceId
), cte2 as(
select playlists.Name as 'Playlist Name', playlist_track.TrackId from playlists
join playlist_track on playlists.PlaylistId = playlist_track.PlaylistId
), cte3 as(
select artists.ArtistId, artists.Name as 'Artist Name', tracks.TrackId,
tracks.Name as 'Track Name', albums.Title as 'Album Title',
cte2.[Playlist Name], tracks.UnitPrice
from artists
join albums on artists.ArtistId = albums.ArtistId
join tracks on albums.AlbumId = tracks.AlbumId
join cte2 on tracks.TrackId = cte2.TrackId group by tracks.Name order by artists.ArtistId
)
select cte1.InvoiceId, cte1.InvoiceDate, cte1.[Customer Name], cte1.[Employee Representativ
e], cte1.BillingCountry,
cte3.[Artist Name], cte3.[Track Name], cte3.[Album Title], cte3.[Playlist Name], cte1.Total
from cte1 join cte3 on cte1.TrackId = cte3.TrackId
```

In [5]:
```python
# eksekusi query master table

master_table = conn.execute(query)
```

In [6]:
```python
# mengambil semua values kedalam list

li = []
for i in master_table:
    li.append(i)
```

In [7]:
```python
# membuat dataframe dari values yang ada pada master table

colname = ['Invoice Id', 'Invoice Date', 'Customer Name', 'Employee Representative',
           'Billing Country', 'Artist Name', 'Track Name', 'Album Title', 'Playlist Type',
'Total Payment']
```

In [8]:
```python
# melihat 5 data teratas pada dataframe

df.head()
```

Out[8]:

	Invoice Id	Invoice Date	Customer Name	Employee Representative	Billing Country	Artist Name	Track Name	Album Title	Playlist Type	Total Payment
0	1	2009-01-01 00:00:00	Leonie Köhler	Steve Johnson	Germany	Accept	Balls to the Wall	Balls to the Wall	Music	1.98
1	1	2009-01-01 00:00:00	Leonie Köhler	Steve Johnson	Germany	Accept	Restless and Wild	Restless and Wild	Music	1.98
2	2	2009-01-02 00:00:00	Bjørn Hansen	Margaret Park	Norway	AC/DC	Breaking The Rules	For Those About To Rock We Salute You	Music	3.96
3	2	2009-01-02 00:00:00	Bjørn Hansen	Margaret Park	Norway	AC/DC	Evil Walks	For Those About To Rock We Salute You	Music	3.96
4	2	2009-01-02 00:00:00	Bjørn Hansen	Margaret Park	Norway	AC/DC	Inject The Venom	For Those About To Rock We Salute You	Music	3.96

In [9]:
```python
# memeriksa info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2063 entries, 0 to 2062
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Invoice Id               2063 non-null   int64  
 1   Invoice Date             2063 non-null   object 
 2   Customer Name            2063 non-null   object 
 3   Employee Representative  2063 non-null   object 
 4   Billing Country          2063 non-null   object 
 5   Artist Name              2063 non-null   object 
 6   Track Name               2063 non-null   object 
 7   Album Title              2063 non-null   object 
 8   Playlist Type            2063 non-null   object 
 9   Total Payment            2063 non-null   float64
dtypes: float64(1), int64(1), object(8)
memory usage: 161.3+ KB
```

In [10]:
```python
# merubah datatype InvoiceId menjadi string

df['Invoice Id'] = df['Invoice Id'].astype('str')

print(df['Invoice Id'].dtype)
```

```
object
```

In [11]:
```python
# merubah datatype Invoice Date menjadi datetime

df['Invoice Date'] = pd.to_datetime(df['Invoice Date'])

print(df['Invoice Date'].dtype)
```

```
datetime64[ns]
```

In [12]:
```python
# memeriksa kembali info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2063 entries, 0 to 2062
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype         
---  ------                   --------------  -----         
 0   Invoice Id               2063 non-null   object        
 1   Invoice Date             2063 non-null   datetime64[ns]
 2   Customer Name            2063 non-null   object        
 3   Employee Representative  2063 non-null   object        
 4   Billing Country          2063 non-null   object        
 5   Artist Name              2063 non-null   object        
 6   Track Name               2063 non-null   object        
 7   Album Title              2063 non-null   object        
 8   Playlist Type            2063 non-null   object        
 9   Total Payment            2063 non-null   float64       
dtypes: datetime64[ns](1), float64(1), object(8)
memory usage: 161.3+ KB
```

In [13]:
```python
# statistik deskriptif untuk data numerical

df.describe().T
```

Out[13]:

	count	mean	std	min	25%	50%	75%	max
Total Payment	2063.0	9.393684	5.122044	0.99	5.94	8.91	13.86	25.86

> Dari analisis deskriptif diatas dapat diketahui bahwa rata - rata `Total Payment` dalam dollar adalah 9.39, dengan standar deviasi 5.12. Transaksi paling minimal adalah 0.99, dengan median 8.91 dan maksimum nya adalah 25.86 dollar.

In [14]:
```python
# statistik deskriptif untuk data categorical

df.describe(include = 'object').T
```

Out[14]:

	count	unique	top	freq
Invoice Id	2063	396	82	14
Customer Name	2063	59	Diego Gutiérrez	38
Employee Representative	2063	3	Jane Peacock	732
Billing Country	2063	24	USA	460
Artist Name	2063	164	Iron Maiden	93
Track Name	2063	1833	Lamento De Carnaval	2
Album Title	2063	298	Minha Historia	27
Playlist Type	2063	2	Music	1955

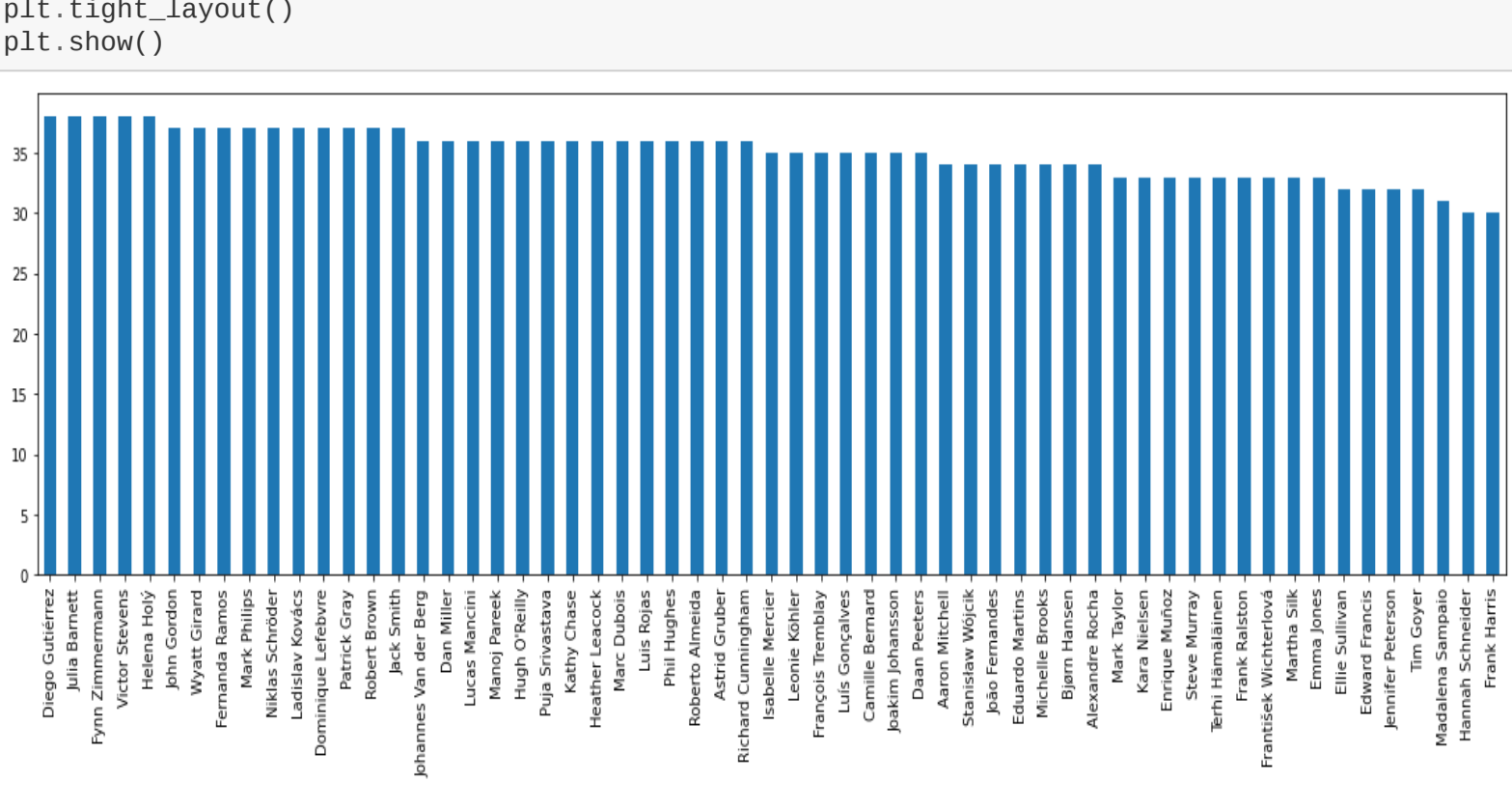
> Analisis deskriptif untuk data categorical memberikan gambaran umum mengenai data - data tersebut secara general. Misalnya pada kolom `Employee Representative` yang mewakili karwayan yang melayani customer, dimana kita lihat terdapat 3 unique ; artinya hanya terdapat 3 nama employee, dan yang paling banyak adalah Jane Peacock sebanyak 732 kali. Pembacaan informasi untuk semua kolom adalah sama. Misalnya, `Playlist Type` yaitu mewakili tipe dari playlist yang di beli. Terdapat 2 unique ; artinya terdapat 2 jenis tipe, dimana Music adalah merupakan tipe yang paling dominan di beli dengan frekuensi sebanyak 1955 kali.

VISUALISASI DAN ANALISIS

In [15]:
```python
# visualisasi customer name

cust = df['Customer Name'].value_counts().sort_values(ascending=False)

plt.figure(figsize=(16,6))
cust.plot(kind = 'bar')
plt.tight_layout()
plt.show()
```



In [16]:
```python
cust.describe()
```

Out[16]:
```
count    59.000000
mean     34.966102
std       2.050787
min      30.000000
25%      33.000000
50%      35.000000
75%      36.500000
max      38.000000
Name: Customer Name, dtype: float64
```

> Dari visualisasi dan describe pada `Customer Name` diatas, kita dapat mengetahui bahwa para customer melakukan transaksi di range antara 30 sampai 38 kali, dengan rata - rata per customer melakukan transaksi sekitar 34 kali.

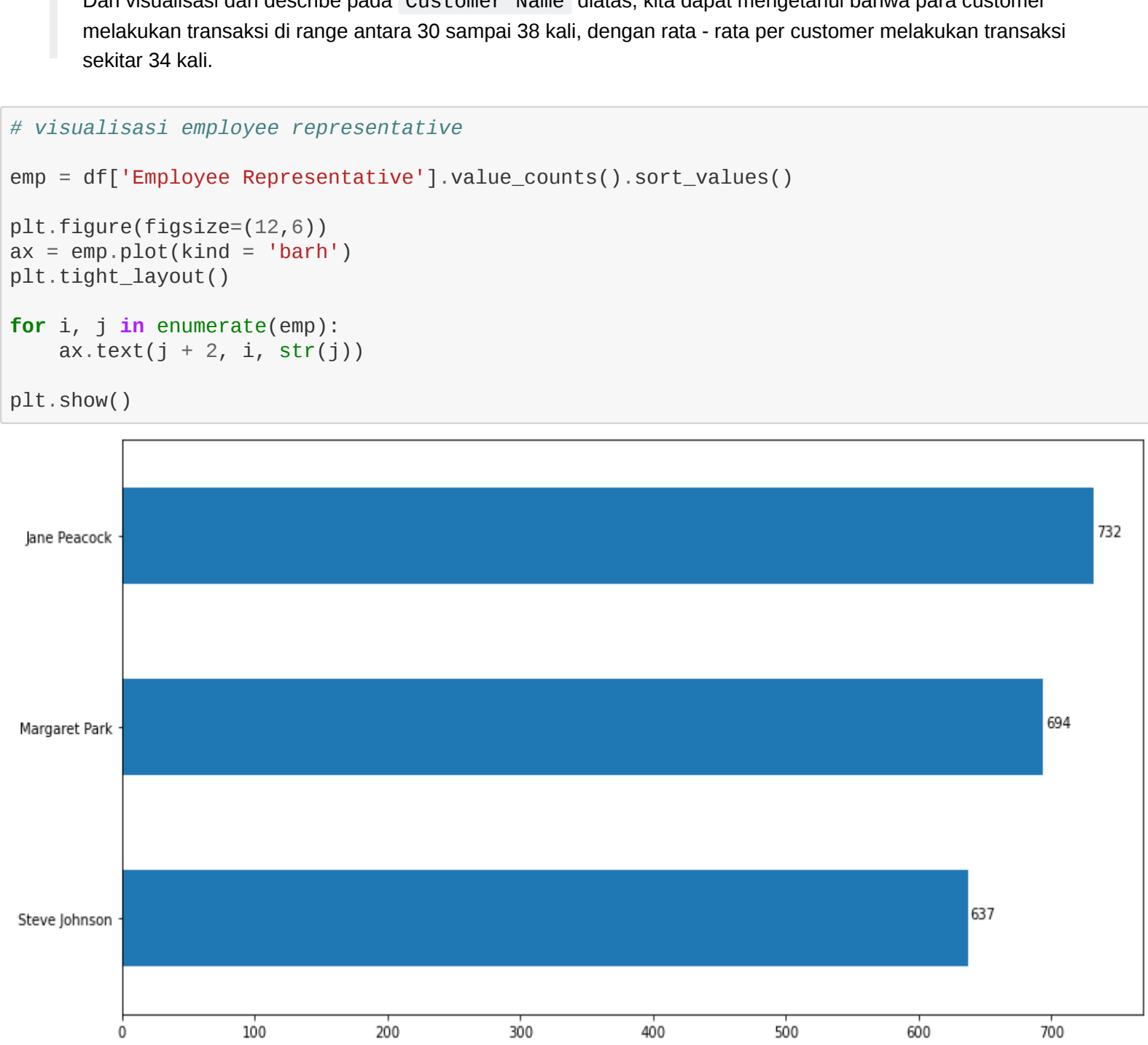
In [17]:
```python
# visualisasi employee representative

emp = df['Employee Representative'].value_counts().sort_values()

plt.figure(figsize=(12,6))
ax = emp.plot(kind = 'barh')
plt.tight_layout()

for i, j in enumerate(emp):
    ax.text(j + 2, i, str(j))

plt.show()
```



> Dari visualisasi `Employee Representative` diatas dapat disimpulkan bahwa memang benar hanya terdapat 3 nama, dimana Jane Peacock melayani lebih banyak pelanggan sebanyak 732 kali, di ikuti oleh Margaret Park sebanyak 694, dan Steve Johnson sebanyak 637 kali.

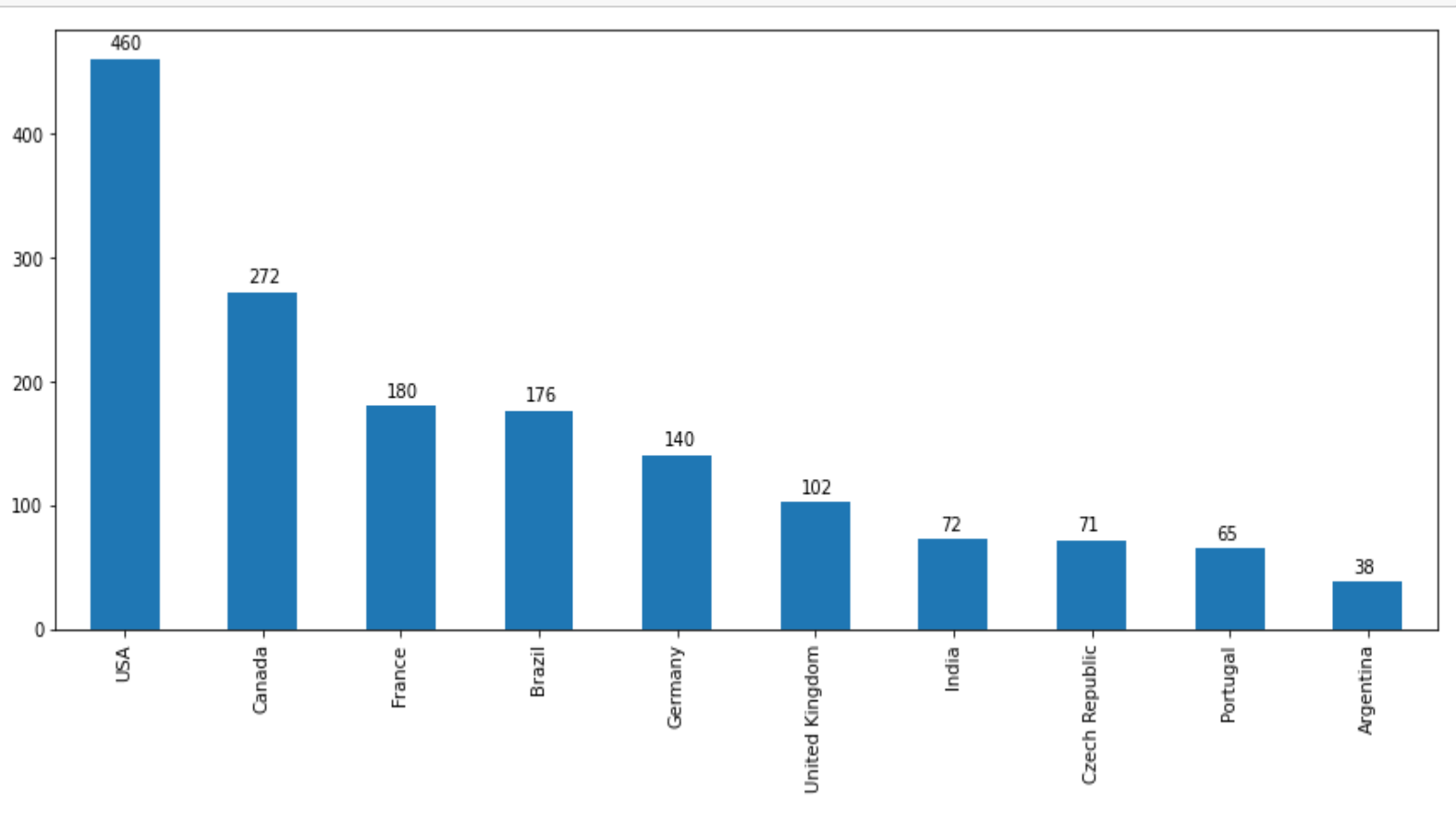
In [18]:
```python
# visualisasi top 10 billing country

cou = df['Billing Country'].value_counts().head(10).sort_values(ascending = False)

plt.figure(figsize=(12,6))
ax = cou.plot(kind = 'bar')
plt.tight_layout()

for i in ax.patches:
        ax.annotate('{}'.format(i.get_height()), (i.get_x()+0.15, i.get_height()+8))

plt.show()
```



> Visualisasi diatas adalah top 10 `Billing Country`, dan dapat diketahui bahwa memang sebagian besar customer berasal atau berdomisili dari USA sebanyak 460 transaksi, selanjutnya Canada dengan 272 transaksi, France dengan 180 transaksi, Brazil dengan 176 transaksi, dst.

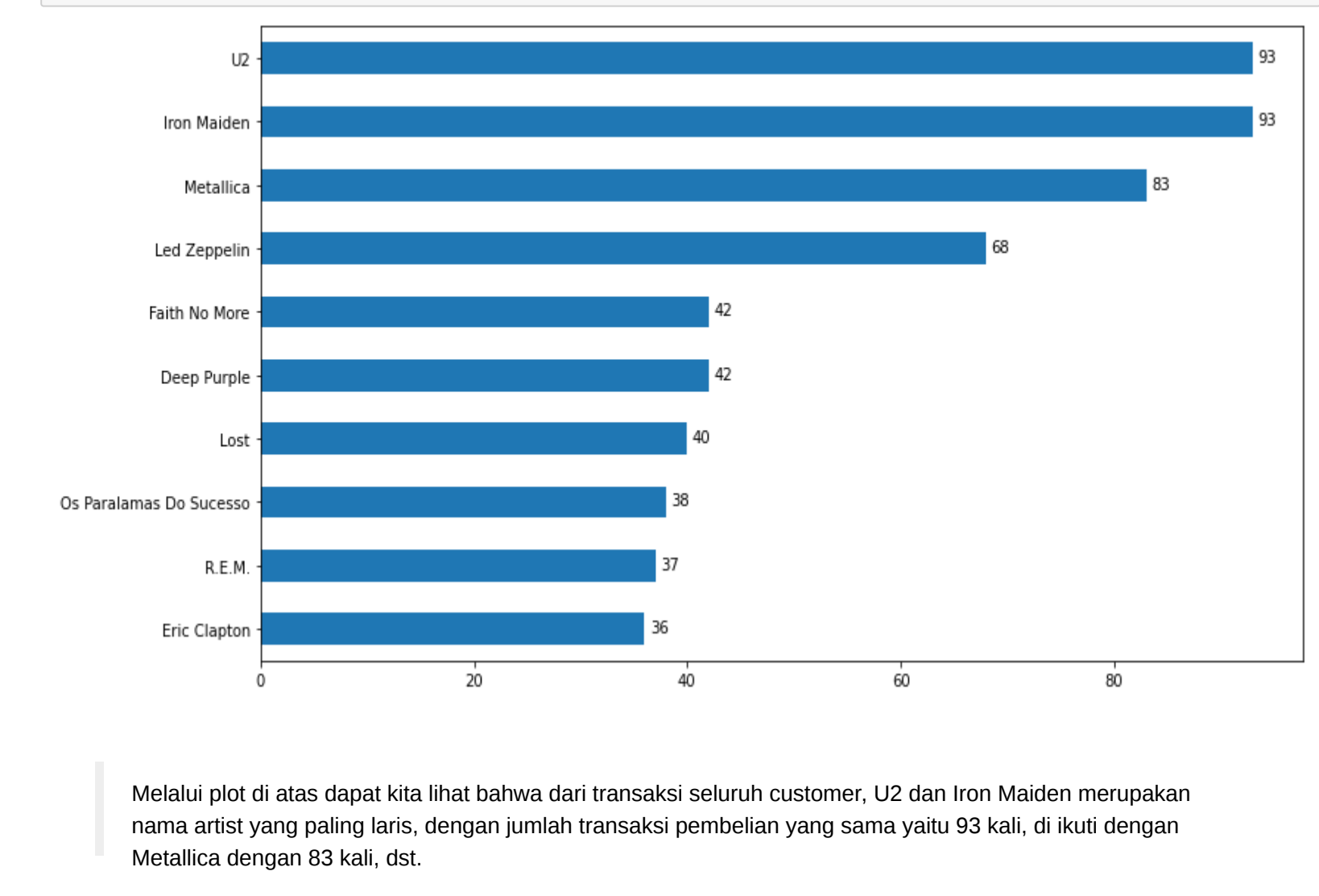
In [19]:
```python
# visualisasi top 10 artist name

art = df['Artist Name'].value_counts().head(10).sort_values()

plt.figure(figsize=(12,6))
ax = art.plot(kind = 'barh')
plt.tight_layout()

for i, j in enumerate(art):
    ax.text(j+0.5, i-0.05, str(j))

plt.show()
```



> Melalui plot di atas dapat kita lihat bahwa dari transaksi seluruh customer, U2 dan Iron Maiden merupakan nama artist yang paling laris, dengan jumlah transaksi pembelian yang sama yaitu 93 kali, di ikuti dengan Metallica dengan 83 kali, dst.

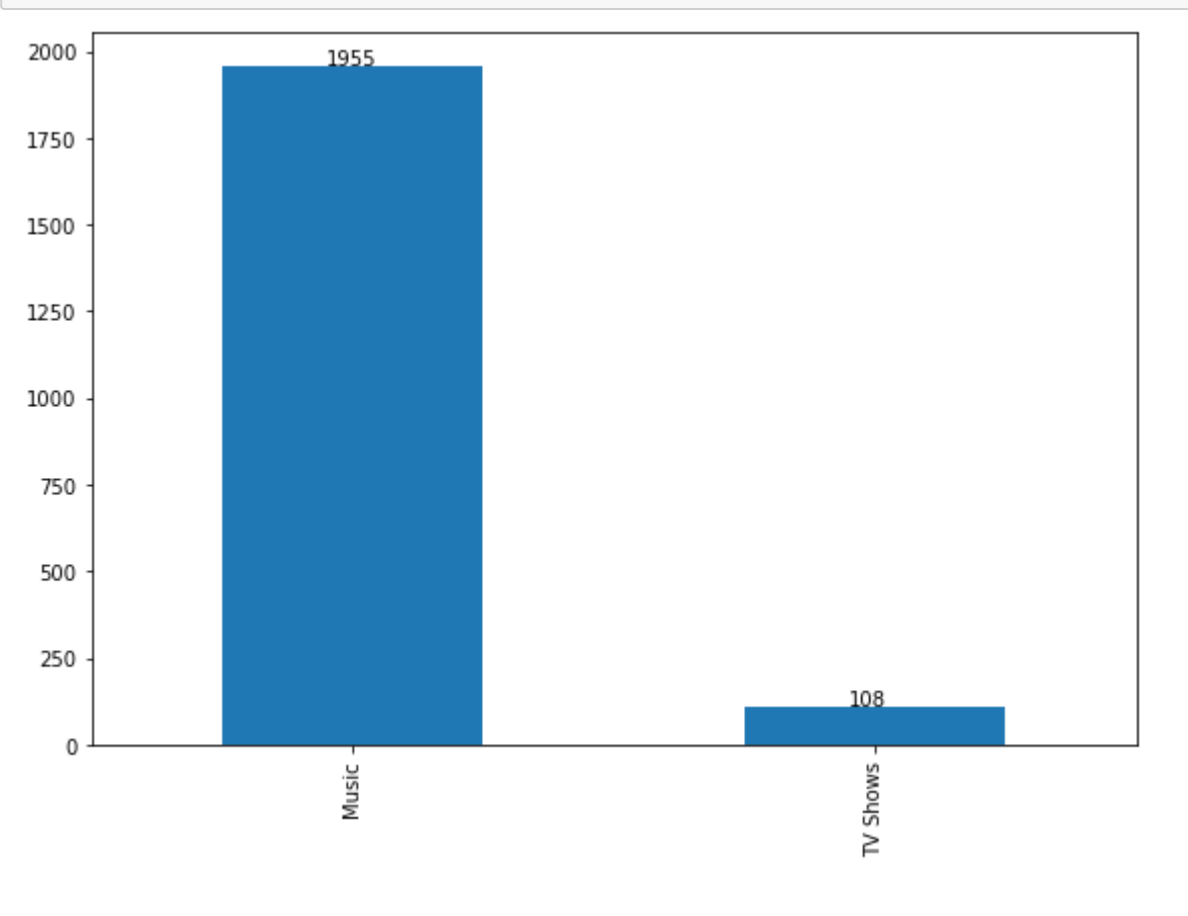
In [20]:
```python
# visualisasi playlist type

pla = df['Playlist Type'].value_counts().sort_values(ascending = False)

plt.figure(figsize=(8,6))
ax = pla.plot(kind = 'bar')
plt.tight_layout()

for i in ax.patches:
        ax.annotate('{}'.format(i.get_height()), (i.get_x()+0.2, i.get_height()+8))

plt.show()
```



> Pada visualisasi `Playlist Type` di atas dapat dilihat jelas bahwa Music merupakan jenis yang paling sering di beli oleh customer, dengan jumlah sebanyak 1955 kali transaksi. Hal ini sangat jauh jika dibandingkan dengan TV Shows yang hanya 108 kali.