# Top 50 Songs Argentina Data Analysis

In [ ]:
```python
import pandas as pd
import plotly.graph_objects as go
import sys
sys.path.append('../')
from engine import engine
```

In [ ]:
```python
top_50_df = pd.read_sql_table('top_50_arg_songs', engine)
top_50_df.head(4)
```

Out[ ]:

| | song_name | artist | rank | extract_date |
|---|---|---|---|---|
| **0** | Un Finde \| CROSSOVER #2 | Big One | 1 | 2023-05-10 |
| **1** | Pobre Corazón - En Vivo | Ke Personajes | 2 | 2023-05-10 |
| **2** | un x100to | Grupo Frontera | 3 | 2023-05-10 |
| **3** | La Bebe - Remix | Yng Lvcas | 4 | 2023-05-10 |

In [ ]:
```python
unique = top_50_df.song_name.nunique()
print(f"""In 1 month, the max number of different songs that could have been in the ranking is {50*30},
and during may, there were: {unique} different songs""")
```

```
In 1 month, the max number of different songs that could have been in the ranking is 1500,
and during may, there were: 69 different songs
```

## Top 10 songs during May

Calculating Points: the metric I chose for total_points for each song is summing the inverse of the rank each time the song appeared on the top.

e.g, rank 5 = 1/5 = 0.2

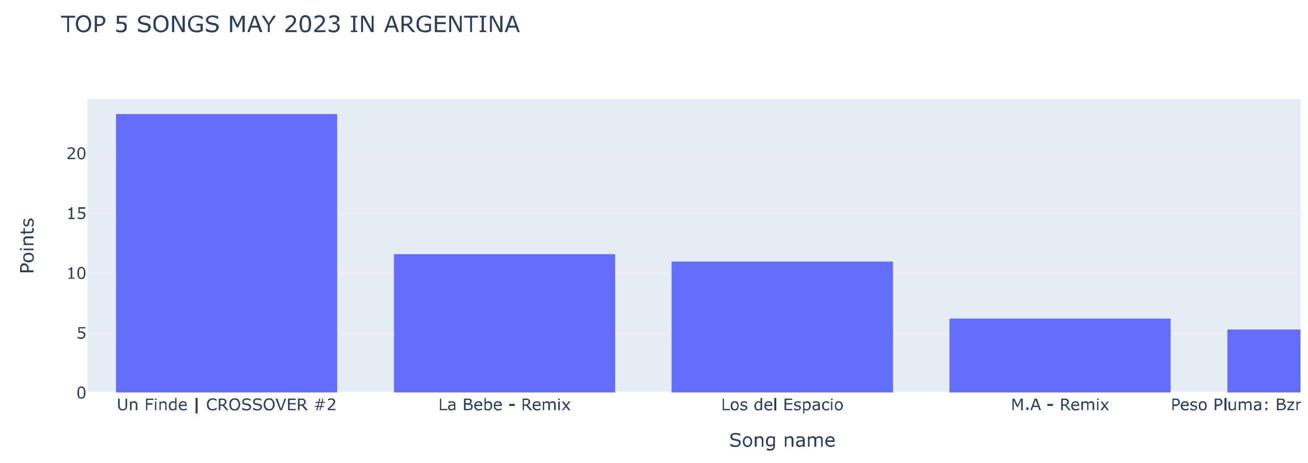
In [ ]:
```python
def calculate_points(df):
    return (1/df['rank'])
top_50_df['points'] = top_50_df.apply(calculate_points, axis=1)
```

In [ ]:
```python
top_5 = top_50_df.drop('extract_date',axis=1).groupby('song_name', as_index=False).agg('sum').sort_values('points', ascending=False)[:5]
top_5[['song_name','points']].head()
```

Out[ ]:

| | song_name | points |
|---|---|---|
| **60** | Un Finde \| CROSSOVER #2 | 23.333333 |
| **27** | La Bebe - Remix | 11.633333 |
| **30** | Los del Espacio | 11.000000 |
| **33** | M.A - Remix | 6.253882 |
| **49** | Peso Pluma: Bzrp Music Sessions, Vol. 55 | 5.333333 |

In [ ]:
```python
fig1 = go.Figure(data=go.Bar(x= top_5['song_name'], y= top_5['points']))
fig1.update_layout(title='TOP 5 SONGS MAY 2023 IN ARGENTINA',
                  xaxis_title='Song name',
                  yaxis_title='Points',
                  width=1200,
                  height=400)
fig1.show()
```

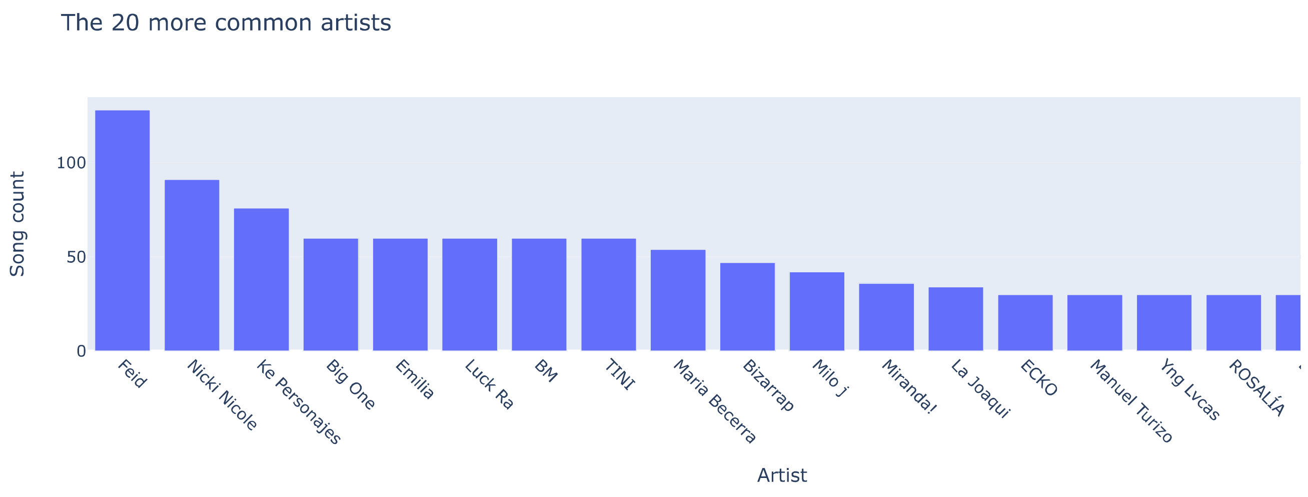


## Who were the artists that appeared most of the time on the top?

```
In [ ]: artists_count = top_50_df[['song_name','artist']].groupby('artist',as_index=False).agg('count').rename(columns={'song_name':'total'}).sort
        artists_count['total'].mean()
```

```
Out[ ]: 31.914893617021278
```

```
In [ ]: fig2= go.Figure(data=go.Bar(x= artists_count['artist'][:20], y= artists_count['total'][:20]))
        fig2.update_layout(title='The 20 more common artists',
                          xaxis_title='Artist',
                          yaxis_title='Song count',
                          xaxis_tickangle=45,
                          width=1200,
                          height=400)
        fig2.show()
```

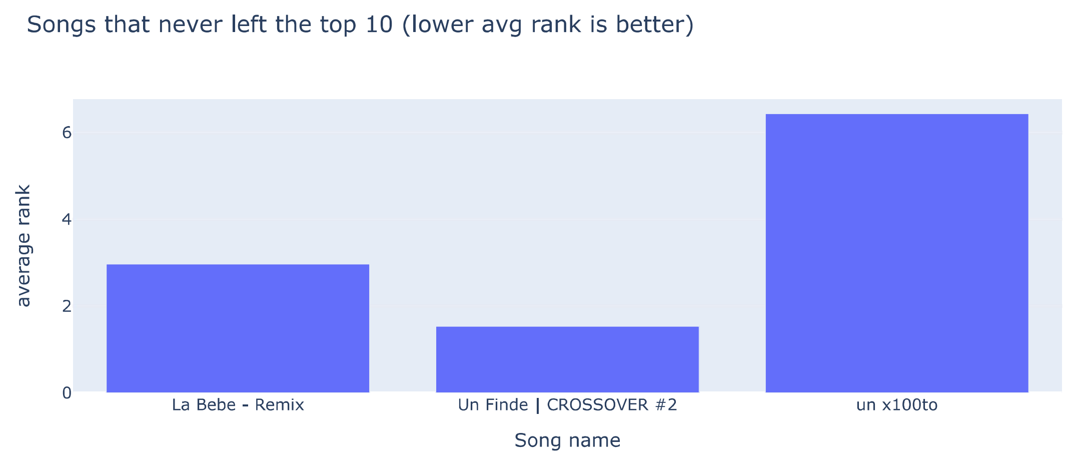


## Songs that never left the top 10

```
In [ ]: aggs = {'extract_date': 'count', 'rank' : 'sum'}
        always_on_top = top_50_df[top_50_df['rank'] <= 10].groupby('song_name',as_index=False).agg(aggs)

        always_on_top = always_on_top[always_on_top['extract_date'] == 30]

        fig3= go.Figure(data=go.Bar(x= always_on_top['song_name'], y= always_on_top['rank']/30))
        fig3.update_layout(title='Songs that never left the top 10 (lower avg rank is better)',
                          xaxis_title='Song name',
                          yaxis_title='average rank',
                          xaxis_tickangle=0,
                          width=900,
                          height=400)
        fig3.show()
```



## Personal played songs Analysis

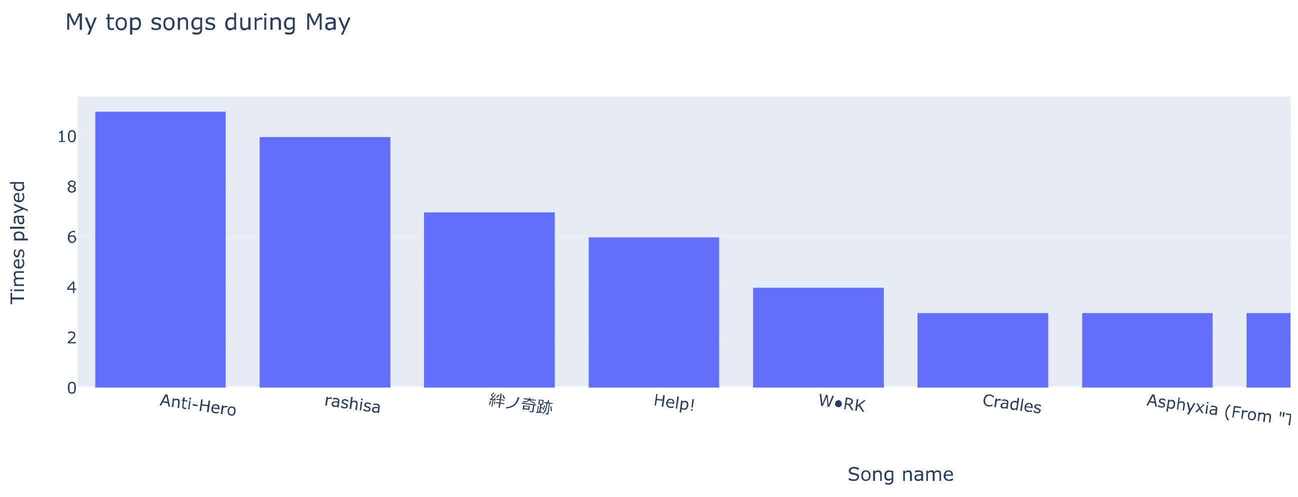
```
In [ ]: personal_played = pd.read_sql_table('my_song_history', engine)
        personal_played.head()
```

Out[ ]:

| | song_name | album | artist | duration_sec | played_at |
|---|---|---|---|---|---|
| **0** | Anti-Hero | Midnights | Taylor Swift | 200 | 2023-05-12 22:24:56 |
| **1** | Anti-Hero | Midnights | Taylor Swift | 200 | 2023-05-12 22:21:34 |
| **2** | Anti-Hero | Midnights | Taylor Swift | 200 | 2023-05-12 21:06:37 |
| **3** | 突破口 | 突破口 / 自慢になりたい | SUPER BEAVER | 255 | 2023-05-13 22:03:01 |
| **4** | 絆ノ奇跡 | 絆ノ奇跡 | MAN WITH A MISSION | 223 | 2023-05-13 21:58:58 |

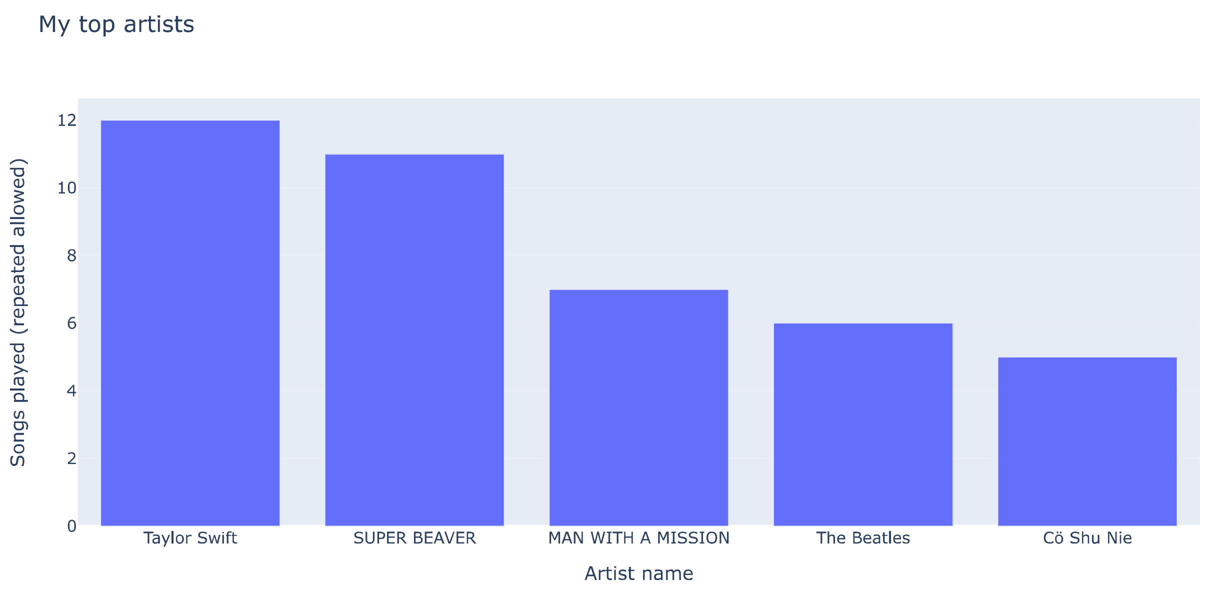
### The 10 songs I listened to the most

```
In [ ]: top_10_personal = personal_played.groupby('song_name',as_index=False).agg({'played_at':'count'}).sort_values('played_at',ascending=False)
        fig4= go.Figure(data=go.Bar(x= top_10_personal['song_name'], y= top_10_personal['played_at']))
        fig4.update_layout(title='My top songs during May',
                          xaxis_title='Song name',
                          yaxis_title='Times played',
                          xaxis_tickangle=8,
                          width=1400,
                          height=400)
        fig4.show()
```



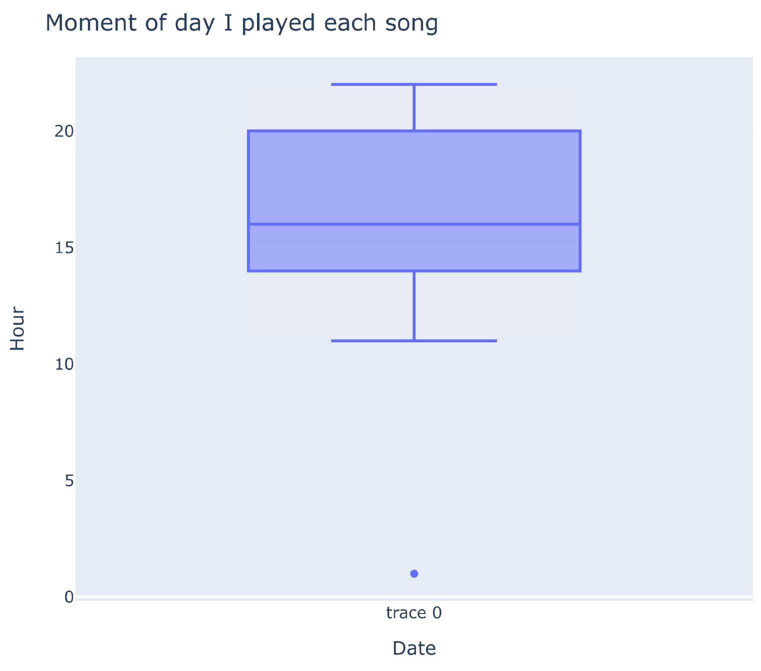
### 5 artists I listened to the most

```
In [ ]: my_top_artists = personal_played.groupby('artist',as_index=False).agg({'played_at':'count'}).rename(columns={'played_at':'songs_listened'}
        my_top_artists
        fig5= go.Figure(data=go.Bar(x= my_top_artists['artist'], y= my_top_artists['songs_listened']))
        fig5.update_layout(title='My top artists',
                          xaxis_title='Artist name',
                          yaxis_title='Songs played (repeated allowed)',
                          xaxis_tickangle=0,
                          width=1000,
                          height=500)
        fig5.show()
```



## Songs played by Hour

```
In [ ]: fig6= go.Figure(data=go.Box(y= personal_played['played_at'].dt.hour))
        fig6.update_layout(title='Moment of day I played each song',
                          xaxis_title='Date',
                          yaxis_title='Hour',
                          xaxis_tickangle=0,
                          width=600,
                          height=500,margin=dict(t=40, r=40, b=20, l=40)
        )
        fig6.show()
```



Conclusion: Most of the times I listen songs on the afternoon or at early night

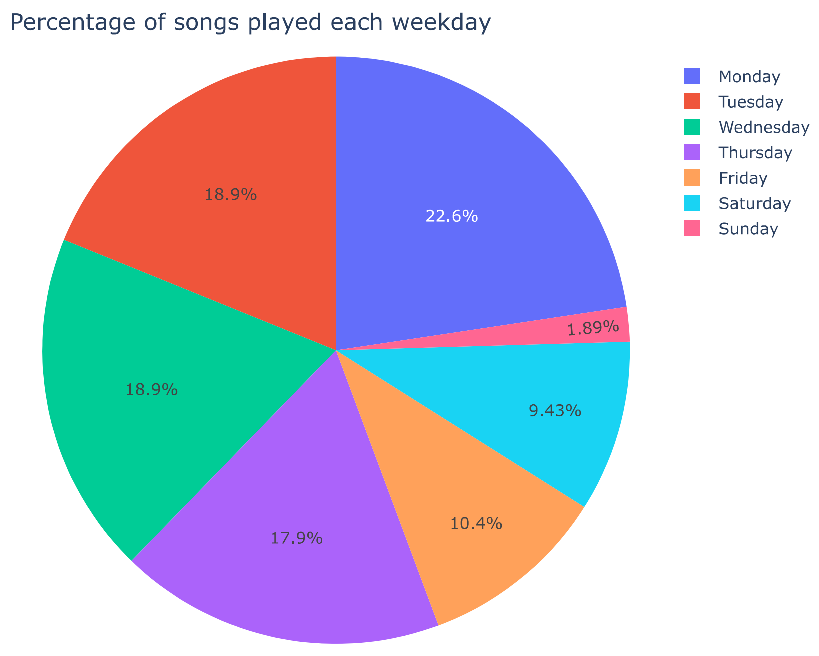
```
In [ ]: daytime_cat = pd.cut(personal_played['played_at'].dt.hour, bins=[0,6,13,19,24], labels=['night','morning','afternoon','night'], ordered=Fa
        daytime_cat.value_counts()
```

```
Out[ ]: played_at
        afternoon    56
        night        30
        morning      20
        Name: count, dtype: int64
```

## Songs played by Day of the Week

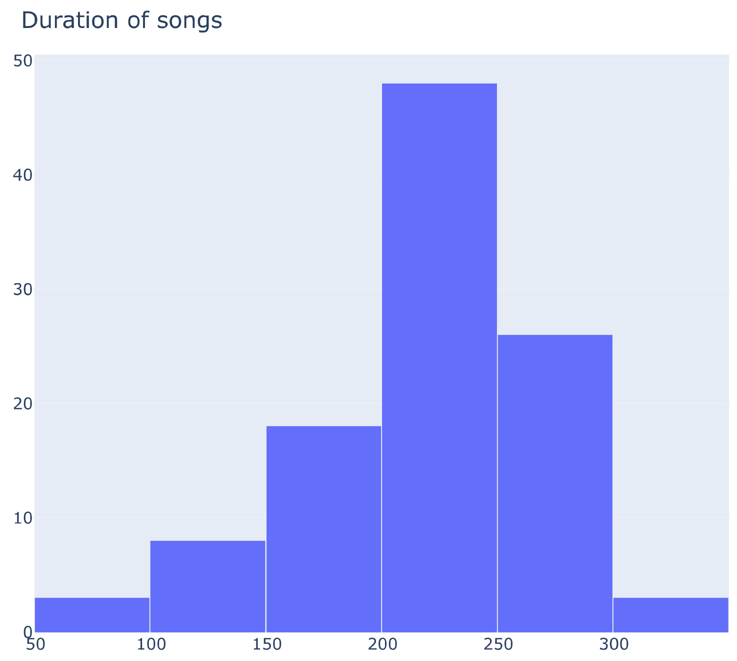
```
In [ ]: songs_by_weekday = personal_played['played_at'].dt.day_of_week
        weekday_map = ['Monday','Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday',  'Sunday']
```

```
In [ ]: fig7= go.Figure(data=go.Pie(labels=weekday_map, values= songs_by_weekday.value_counts()))
        fig7.update_layout(title='Percentage of songs played each weekday',
                          width=650,
                          height=500,margin=dict(t=40, r=40, b=20, l=40)
        )
        fig7.show()
```



## Average song duration

```
In [ ]: durations = personal_played['duration_sec']
        fig8= go.Figure(data=go.Histogram(nbinsx=6, x=durations))
        fig8.update_layout(title='Duration of songs',
                          bargap=0.01,
                          width=600,
                          height=500,margin=dict(t=40, r=40, b=20, l=40)
        )
        fig8.show()
```



Conclusion: Most of the songs I listen to are 4 minutes long