In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_csv('spotify_history.csv')
```

In [3]:
```
print(df.head())
```

```
       spotify_track_uri           ts    platform  ms_played  \
0  2J3n32GeLmMjwuAzyhcSNe  7/8/13 2:44  web player        3185
1  1oHxIPqJyvAYHy0PVrDU98  7/8/13 2:45  web player       61865
2  487OPlneJNni3NWC8SYqhW  7/8/13 2:50  web player      285386
3  5IyblF777jLZj1vGHG2UD3  7/8/13 2:52  web player      134022
4  0GgAAB0ZMllFhbNc3mAodO  7/8/13 3:17  web player           0

                                    track_name        artist_name  \
0                             Say It, Just Say It       The Mowgli's
1  Drinking from the Bottle (feat. Tinie Tempah)      Calvin Harris
2                                   Born To Die       Lana Del Rey
3                              Off To The Races       Lana Del Rey
4                                     Half Mast  Empire Of The Sun

                          album_name reason_start reason_end  shuffle
\
0               Waiting For The Dawn     autoplay   clickrow    False
1                           18 Months     clickrow   clickrow    False
2  Born To Die - The Paradise Edition     clickrow    unknown    False
3  Born To Die - The Paradise Edition    trackdone   clickrow    False
4                 Walking On A Dream     clickrow    nextbtn    False

   skipped
0    False
1    False
2    False
3    False
4    False
```

In [4]:
```
print(df.describe())
```

```
          ms_played
count  1.498600e+05
mean   1.283166e+05
std    1.178401e+05
min    0.000000e+00
25%    2.795000e+03
50%    1.388400e+05
75%    2.185070e+05
max    1.561125e+06
```

In [5]:
```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 149860 entries, 0 to 149859
Data columns (total 11 columns):
 #   Column             Non-Null Count   Dtype 
---  ------             --------------   ----- 
 0   spotify_track_uri  149860 non-null  object
 1   ts                 149860 non-null  object
 2   platform           149860 non-null  object
 3   ms_played          149860 non-null  int64 
 4   track_name         149860 non-null  object
 5   artist_name        149860 non-null  object
 6   album_name         149860 non-null  object
 7   reason_start       149717 non-null  object
 8   reason_end         149743 non-null  object
 9   shuffle            149860 non-null  bool  
 10  skipped            149860 non-null  bool  
dtypes: bool(2), int64(1), object(8)
memory usage: 10.6+ MB
None
```

In [ ]:
```
Data set overview

Shape:
Data set has 149860 data with 11 columns

Variable type:
comprising of 8 objects, 1 interger and 2 bool

Basic Statistics:
Count: The number of non-null enteries are consistent across variables
Mean Standard Deviation Min and Max Values:  Ms_played has a mean of 1
Pencentile(25%,50%, 75%) These indicate the duration of data for insta
unique values:  Some columns are showing unqiue values like reason_sta
```

In [6]:
```
print(df.isnull().sum())
```

```
spotify_track_uri      0
ts                     0
platform               0
ms_played              0
track_name             0
artist_name            0
album_name             0
reason_start         143
reason_end           117
shuffle                0
skipped                0
dtype: int64
```

In [ ]:
```
Checking for missing Values.  reason_start has 143 mising values and r
```

In [19]:
```
df.rename(columns={'ts': 'timestamp', 'ms_played': 'milliseconds_playe
```

In [20]:
```
df['minutes_played'] = df['milliseconds_played'] / (1000 * 60)
```

In [ ]:
```
Analyzing the data:
```

In [21]:
```
top_tracks = df.groupby('track_name')['milliseconds_played'].sum().sor
print(top_tracks)
```

```
track_name
Ode To The Mets
67431580
The Return of the King (feat. Sir James Galway, Viggo Mortensen and Ren
ee Fleming)     64401661
The Fellowship Reunited (feat. Sir James Galway, Viggo Mortensen and Re
née Fleming)    44756730
19 Dias y 500 Noches - En Directo
42914042
In the Blood
38427087
Claudia's Theme - Version Eight
37120900
Dying Breed
36182653
The Breaking of the Fellowship (feat. "In Dreams")
35990898
All These Things That I've Done
35754915
Caution
35619945
Name: milliseconds_played, dtype: int64
```

In [ ]:
```
These are the most played songs
```

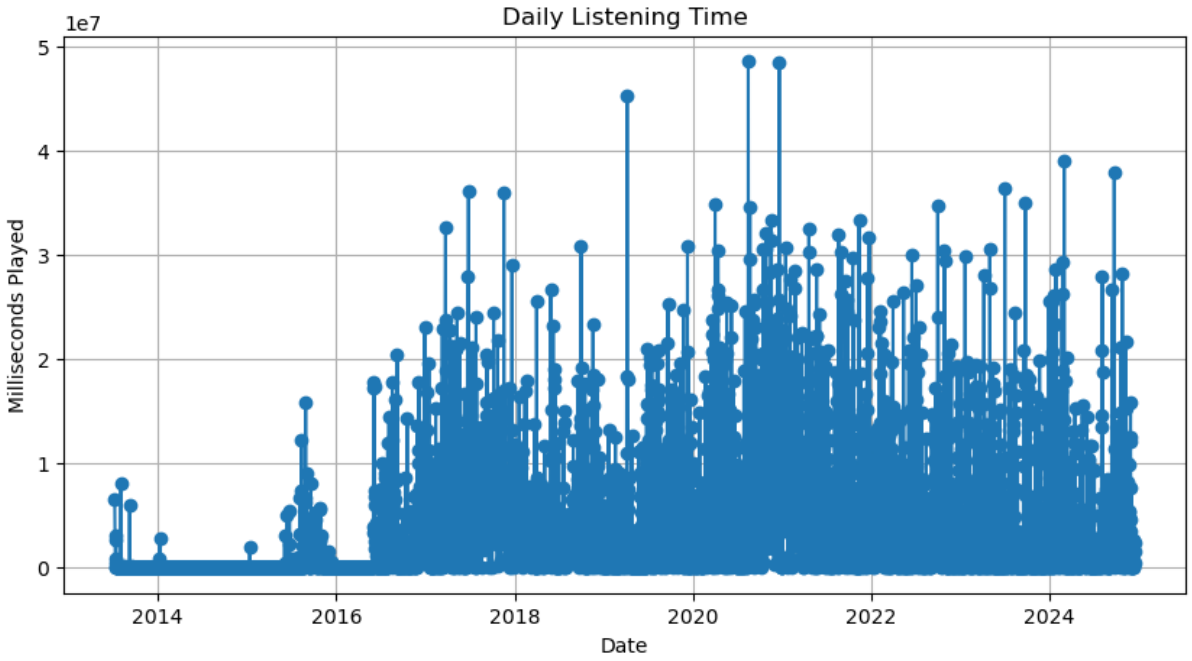
In [22]:
```
top_artists = df.groupby('artist_name')['milliseconds_played'].sum().s
print(top_artists)
```

```
artist_name
The Beatles           1210184552
The Killers           1059556516
John Mayer             725219443
Bob Dylan              569456396
Paul McCartney         357354370
Howard Shore           348930675
The Strokes            317508419
The Rolling Stones     307917009
Pink Floyd             260531842
Led Zeppelin           248338279
Name: milliseconds_played, dtype: int64
```

In [ ]:
```
These are the most played artists
```

In [23]:
```
df.set_index('timestamp', inplace=True)
daily_playtime = df['milliseconds_played'].resample('D').sum()
```

```
plt.figure(figsize=(10,5))
plt.plot(daily_playtime, marker='o', linestyle='-')
plt.title('Daily Listening Time')
plt.xlabel('Date')
plt.ylabel('Milliseconds Played')
plt.grid()
plt.show()
```



In [ ]:
```
This shows the listening trends over time.
```

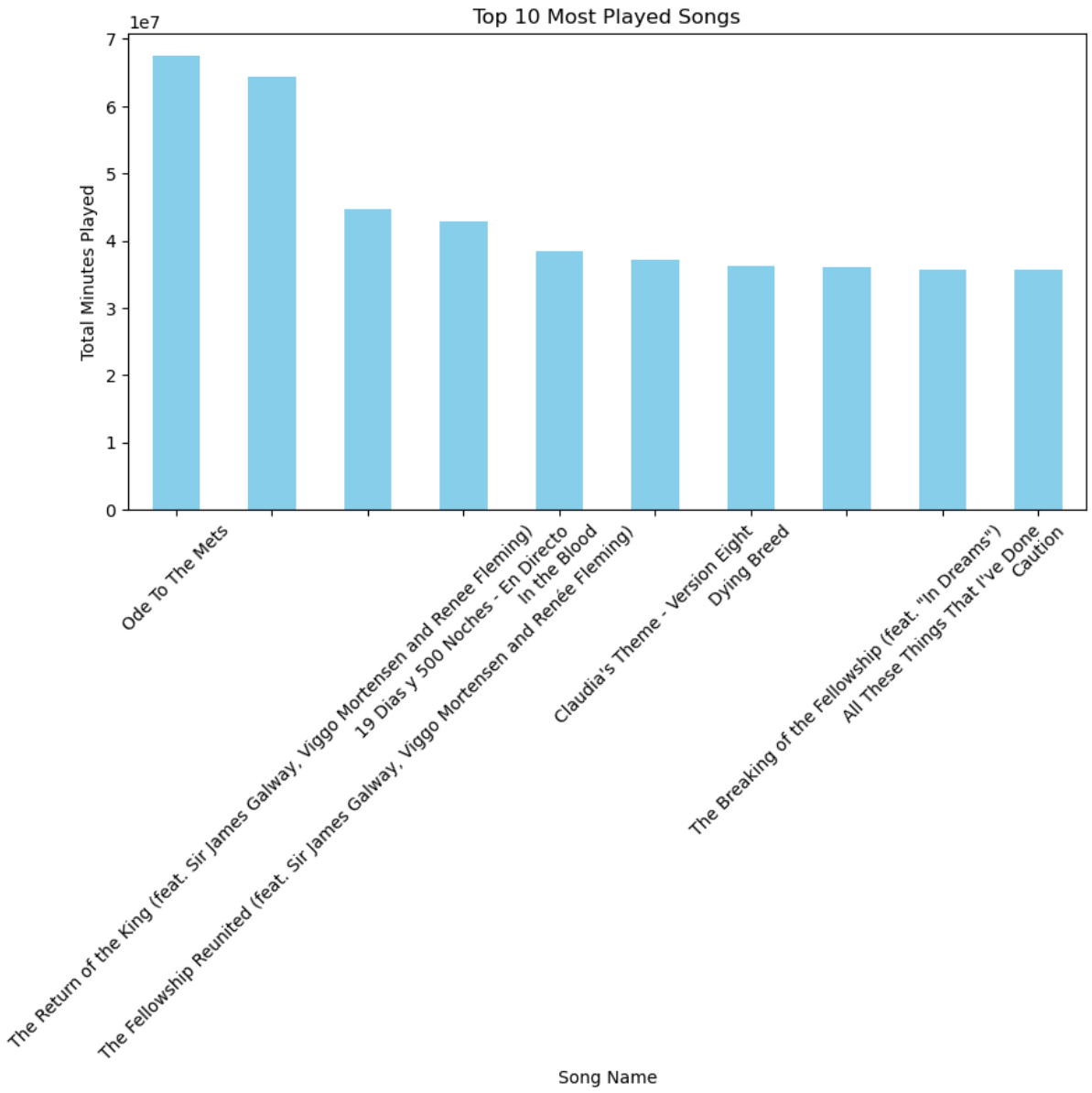
In [25]:
```
skip_rate = df['skipped'].value_counts(normalize=True) * 100
print("Percentage of Skipped vs Completed Tracks:\n", skip_rate)
```

```
Percentage of Skipped vs Completed Tracks:
 skipped
False    94.749099
True      5.250901
Name: proportion, dtype: float64
```

In [ ]:
```
This shows the most skipped tracks.
```

In [ ]:
```
Data Visualization
```

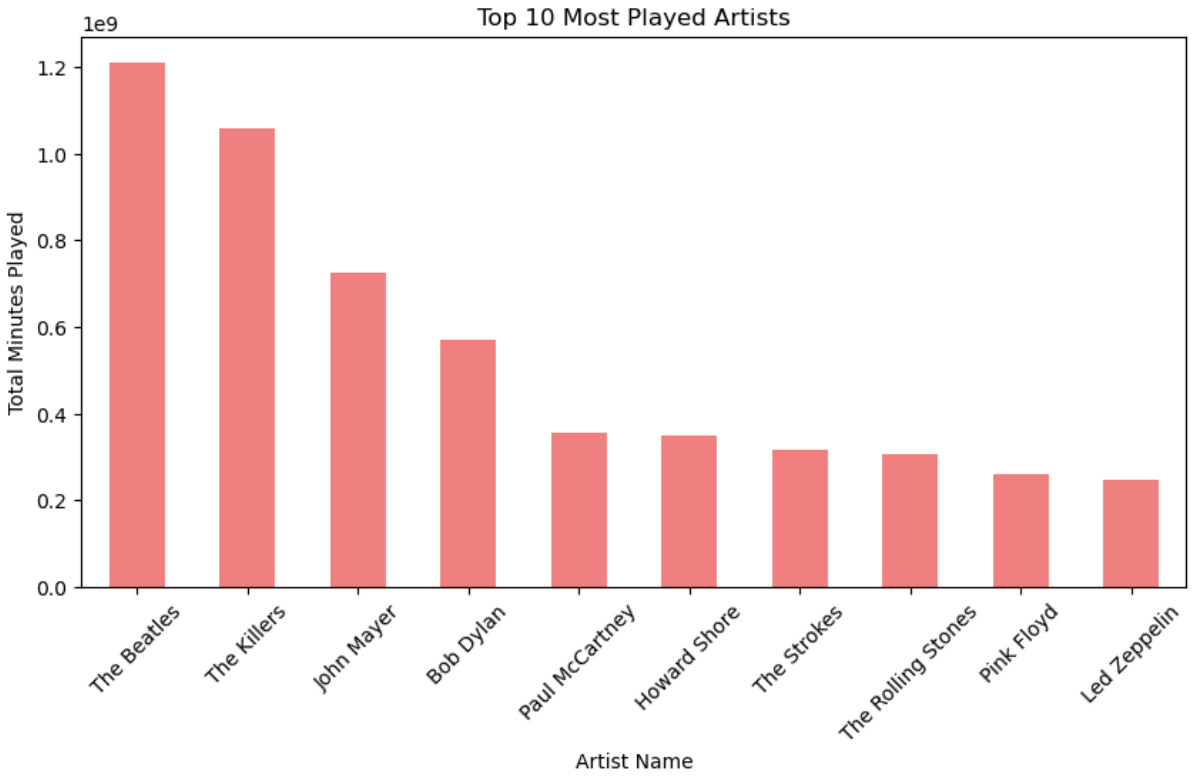
In [29]:
```
plt.figure(figsize=(10,5))
top_tracks.plot(kind='bar', color='skyblue')
plt.title("Top 10 Most Played Songs")
plt.xlabel("Song Name")
plt.ylabel("Total Minutes Played")
plt.xticks(rotation=45)
plt.show()
```



In [ ]:

In [ ]: This analysis show that certain songs were played more frequently. Od

In [30]:
```
plt.figure(figsize=(10,5))
top_artists.plot(kind='bar', color='lightcoral')
plt.title("Top 10 Most Played Artists")
plt.xlabel("Artist Name")
plt.ylabel("Total Minutes Played")
plt.xticks(rotation=45)
plt.show()
```

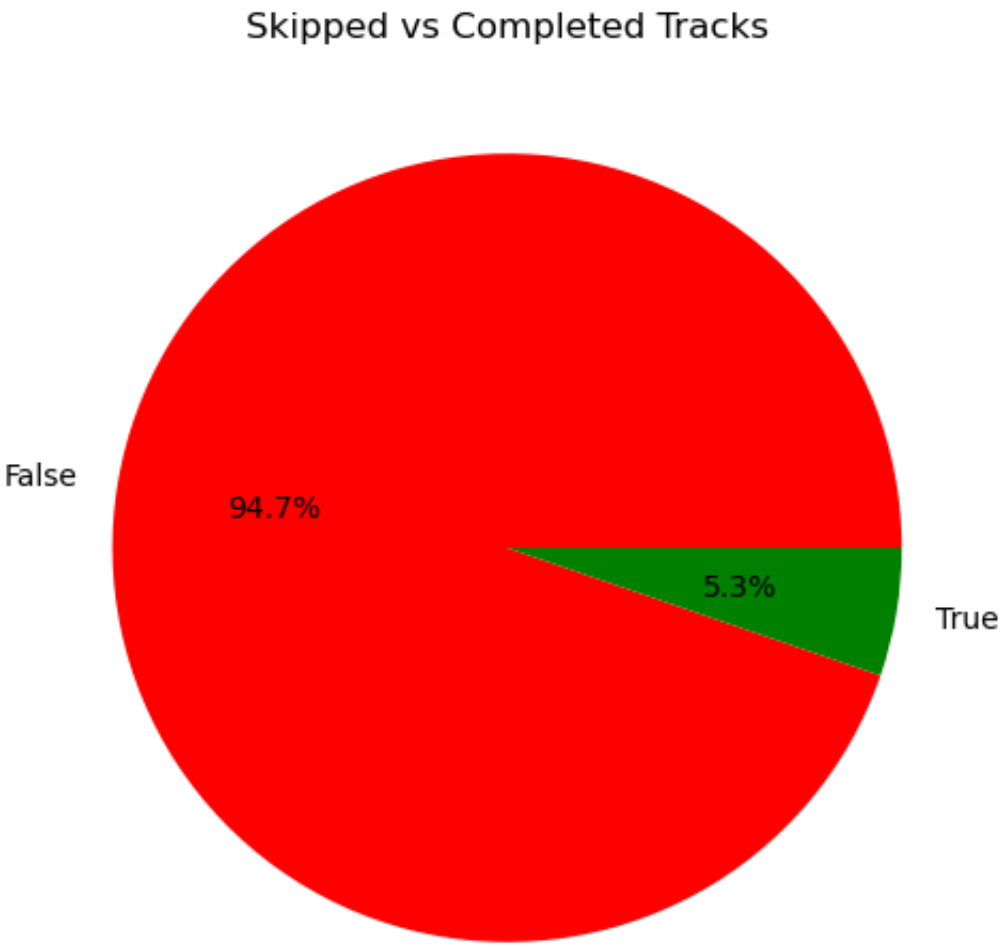


In [ ]:
```
This analysis shows that Beatles group was the most listened to.
```

In [ ]:
```

```

In [31]:
```
plt.figure(figsize=(6,6))
skip_rate.plot(kind='pie', autopct='%1.1f%%', colors=['red', 'green'])
plt.title("Skipped vs Completed Tracks")
plt.ylabel("")
plt.show()
```



In [ ]:
```
94.7% of the tracks were skipped, indicating that users often move to
Only about 5.3% of the tracks were fully played, which suggests users
```

In [ ]:
```
Business Question:  Why is the skip rate so high, and how can users en
From our analysis:

94.7% of tracks were skipped before completion.
Only 5.3% of tracks were completed, which indicates low engagement.
This raises key questions:

Are users skipping songs due to bad recommendations?
Are certain artists, genres, or songs skipped more often?
Do skip rates vary based on platform (mobile vs desktop)?
How does shuffle mode impact song engagement?
```

In [ ]:
```
Recommedndations:
```

In [32]:
```
skipped_songs = df[df['skipped'] == True]
top_skipped_artists = skipped_songs['artist_name'].value_counts().head
print("Top 10 Most Skipped Artists:\n", top_skipped_artists)
```

```
Top 10 Most Skipped Artists:
 artist_name
The Beatles           388
The Killers           197
Bob Dylan             163
John Mayer            153
Led Zeppelin          128
The Rolling Stones    125
The Script            121
Imagine Dragons       116
Paul McCartney        107
Radiohead             102
Name: count, dtype: int64
```

In [ ]:
```
1. Will recommend a better system that will reduce skip rates and incr
```

In [33]:
```
low_skip_songs = df[df['skipped'] == False]['track_name'].value_counts
print("Top 10 Least Skipped Songs:\n", low_skip_songs)
```

```
Top 10 Least Skipped Songs:
 track_name
Ode To The Mets                        206
In the Blood                           180
Dying Breed                            164
Caution                                162
For What It's Worth                    145
19 Dias y 500 Noches - En Directo      143
All These Things That I've Done        139
Concerning Hobbits                     135
Come Together - Remastered 2009        135
Yesterday - Remastered 2009            133
Name: count, dtype: int64
```

In [ ]:
```
2. Users will likely spend more time streaming music.
```

In [34]:
```
skip_by_shuffle = df.groupby('shuffle')['skipped'].mean() * 100
print("Skip Rate with Shuffle Mode:\n", skip_by_shuffle)
```

```
Skip Rate with Shuffle Mode:
 shuffle
False    4.323745
True     5.568949
Name: skipped, dtype: float64
```

In [ ]:
```
3.  If users favorite tracks are the preferred tracks, skip rates will
```