In [40]:
```
from googleapiclient.discovery import build
import pandas as pd
import seaborn as sns
```

In [74]:
```
api_key = 'AIzaSyAn3du_COtTPFbc0PSMXDHJzXxmVCWtOQY'
channel_ids = ['UCLLw7jmFsvfIVaUFsLs8mlQ', # Luke Barousse
               'UCiT9RITQ9PW6BhXK0y2jaeg', # Ken Jee
               'UC7cs8q-gJRlGwj4A8OmCmXg', # Alex the analyst
               'UC2UXDak6o7rBm23k3Vv5dww', # Tina Huang
               'UC3rY5HOgbBvGmq7RnDfwF7A' #Rishabh Mishra
              ]

youtube = build('youtube', 'v3', developerKey=api_key)
```

In [75]:
```
def get_channel_stats(youtube, channel_ids):
    all_data = []
    request = youtube.channels().list(
                part='snippet,contentDetails,statistics',
                id=','.join(channel_ids))
    response = request.execute()

    for i in range(len(response['items'])):
        data = dict(Channel_name = response['items'][i]['snippet']['titl
                    Subscribers = response['items'][i]['statistics']['su
                    Views = response['items'][i]['statistics']['viewCoun
                    Total_videos = response['items'][i]['statistics']['v
                    playlist_id = response['items'][i]['contentDetails']
        all_data.append(data)

    return all_data
```

In [76]:
```
channel_statistics = get_channel_stats(youtube, channel_ids)
```

In [77]:
```
channel_data = pd.DataFrame(channel_statistics)
```

In [78]:
```
channel_data['Subscribers'] = pd.to_numeric(channel_data['Subscribers'])
channel_data['Views'] = pd.to_numeric(channel_data['Views'])
channel_data['Total_videos'] = pd.to_numeric(channel_data['Total_videos'
```

In [114]:
```
playlist_id = channel_data.loc[channel_data['Channel_name']=='Rishabh Mi
```

In [115]:
```
playlist_id
```

Out[115]:
```
'UU3rY5HOgbBvGmq7RnDfwF7A'
```

In [116]:
```
def get_video_ids(youtube, playlist_id):

    request = youtube.playlistItems().list(
                part='contentDetails',
                playlistId = playlist_id,
                maxResults = 50)
    response = request.execute()

    video_ids = []

    for i in range(len(response['items'])):
        video_ids.append(response['items'][i]['contentDetails']['videoId

    next_page_token = response.get('nextPageToken')
    more_pages = True

    while more_pages:
        if next_page_token is None:
            more_pages = False
        else:
            request = youtube.playlistItems().list(
                        part='contentDetails',
                        playlistId = playlist_id,
                        maxResults = 50,
                        pageToken = next_page_token)
            response = request.execute()

            for i in range(len(response['items'])):
                video_ids.append(response['items'][i]['contentDetails'][

            next_page_token = response.get('nextPageToken')

    return video_ids
```

In [117]:
```
video_ids = get_video_ids(youtube, playlist_id)
```

In [118]:
```
def get_video_details(youtube, video_ids):
    all_video_stats = []

    for i in range(0, len(video_ids), 50):
        request = youtube.videos().list(
                    part='snippet,statistics',
                    id=','.join(video_ids[i:i+50]))
        response = request.execute()

        for video in response['items']:
            video_stats = dict(Title = video['snippet']['title'],
                               Published_date = video['snippet']['publis
                               Views = video['statistics']['viewCount'],
                               Likes = video['statistics']['likeCount'],
                               Comments = video['statistics']['commentCo
                               )
            all_video_stats.append(video_stats)

    return all_video_stats
```

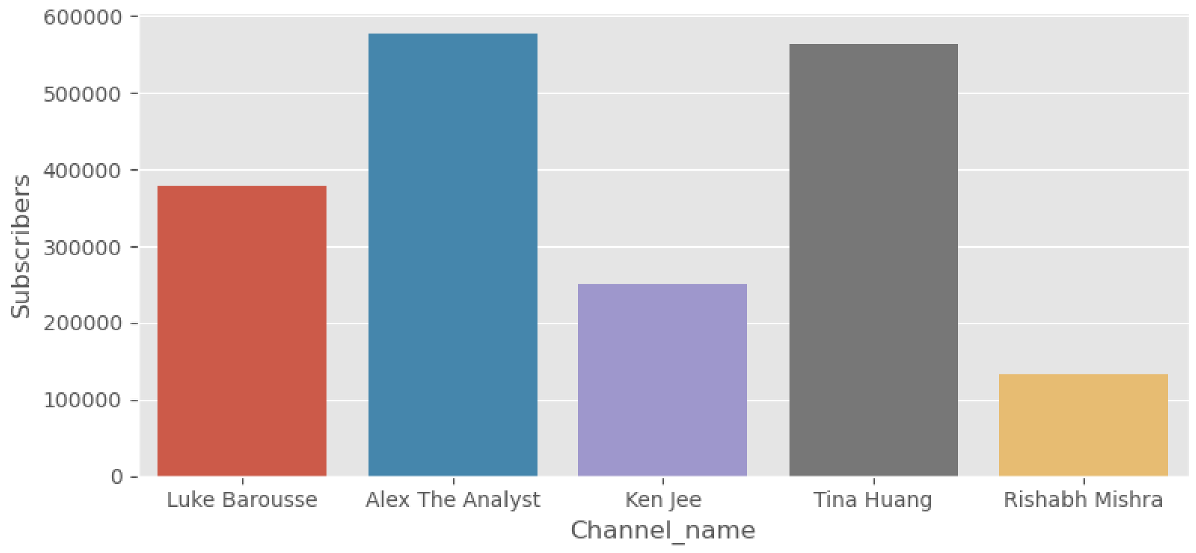
In [119]:
```
video_details = get_video_details(youtube, video_ids)
```

In [120]:
```
video_data = pd.DataFrame(video_details)
```

In [121]:
```
video_data['Published_date'] = pd.to_datetime(video_data['Published_date
video_data['Views'] = pd.to_numeric(video_data['Views'])
video_data['Likes'] = pd.to_numeric(video_data['Likes'])
video_data['Views'] = pd.to_numeric(video_data['Views'])
```

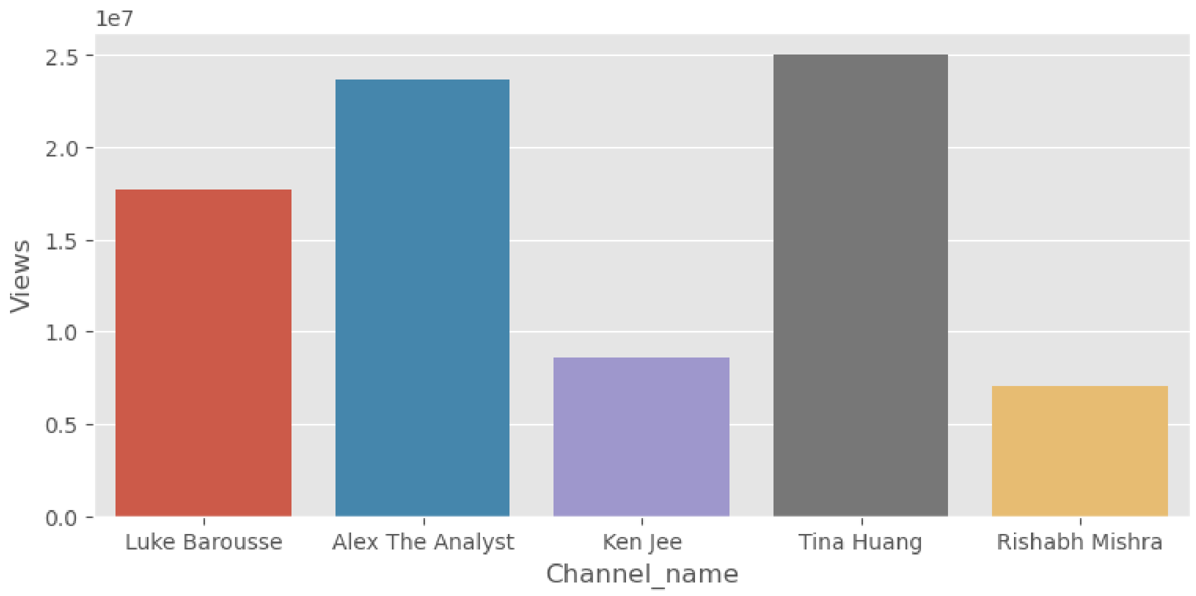
In [122]:
```
plt.figure(figsize=(9,4))
sns.barplot(data=channel_data,x="Channel_name",y="Subscribers")
```

Out[122]:
```
<Axes: xlabel='Channel_name', ylabel='Subscribers'>
```



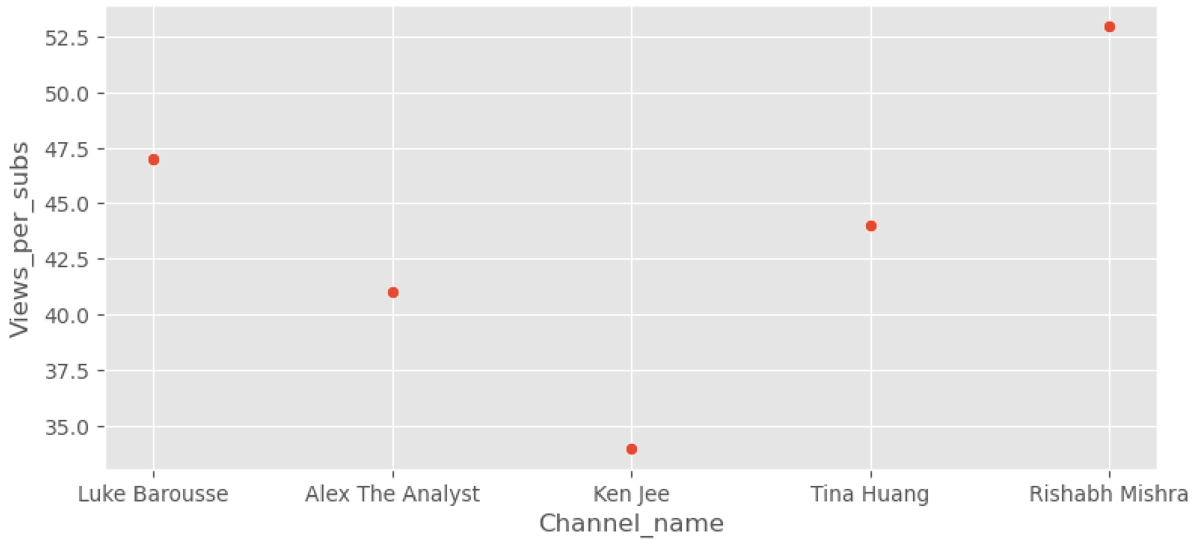
In [123]:
```
plt.figure(figsize=(9,4))
sns.barplot(data=channel_data,x="Channel_name",y="Views")
```

Out[123]:
```
<Axes: xlabel='Channel_name', ylabel='Views'>
```

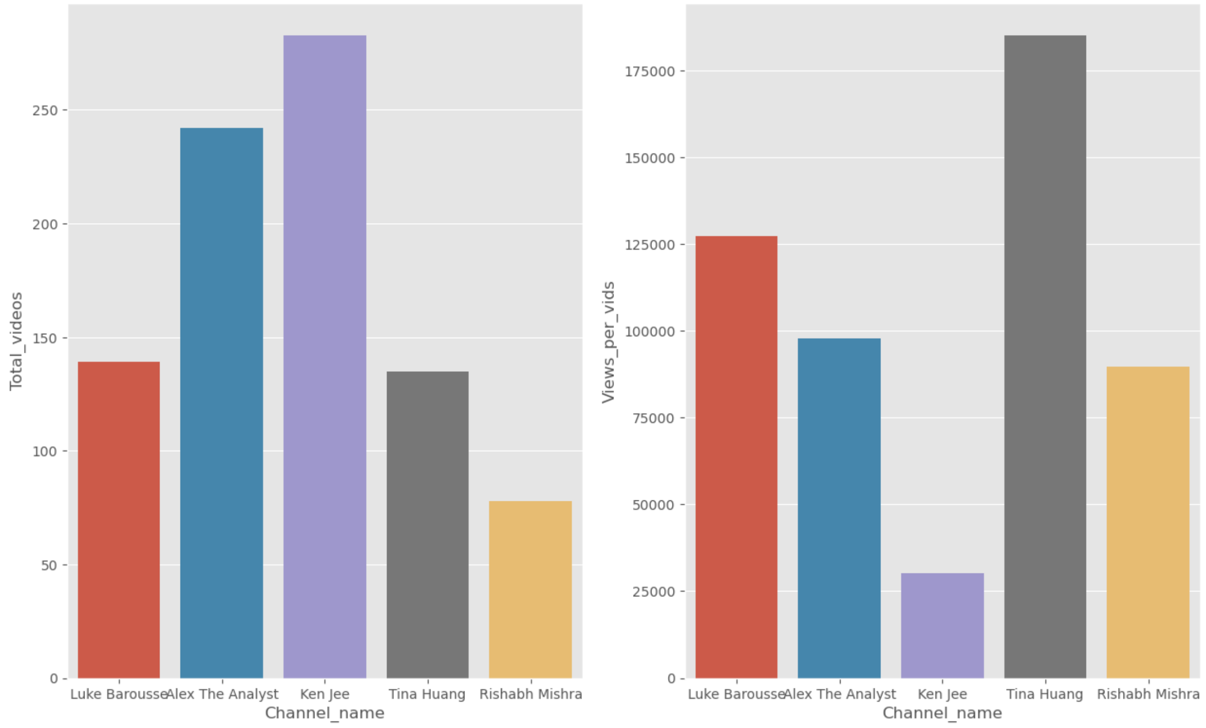


In [129]:
```
plt.figure(figsize=(9,4))
channel_data["Views_per_subs"]=channel_data["Views"]/channel_data["Subsc
channel_data["Views_per_subs"]=channel_data["Views_per_subs"].round(0)

sns.scatterplot(data=channel_data,x="Channel_name",y="Views_per_subs")
```

Out[129]:
```
<Axes: xlabel='Channel_name', ylabel='Views_per_subs'>
```



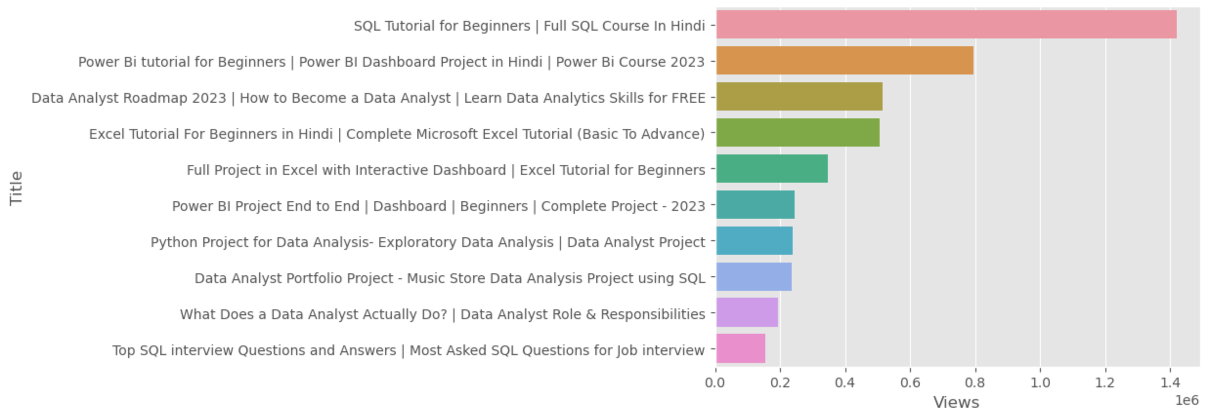
In [131]:
```
plt.style.use("ggplot")
fig, axs = plt.subplots(1,2, figsize=(15,9), sharex=True)
sns.barplot(data=channel_data,x="Channel_name",y="Total_videos",ax=axs[0


plt.subplot(1,2,2)
plt.figure(figsize=(9,4))
channel_data["Views_per_vids"]=channel_data["Views"]/channel_data["Total_
channel_data["Views_per_vids"]=channel_data["Views_per_vids"].round(0)

sns.barplot(data=channel_data,x="Channel_name",y="Views_per_vids",ax=axs

plt.tight_layout()
```



```
<Figure size 900x400 with 0 Axes>
```

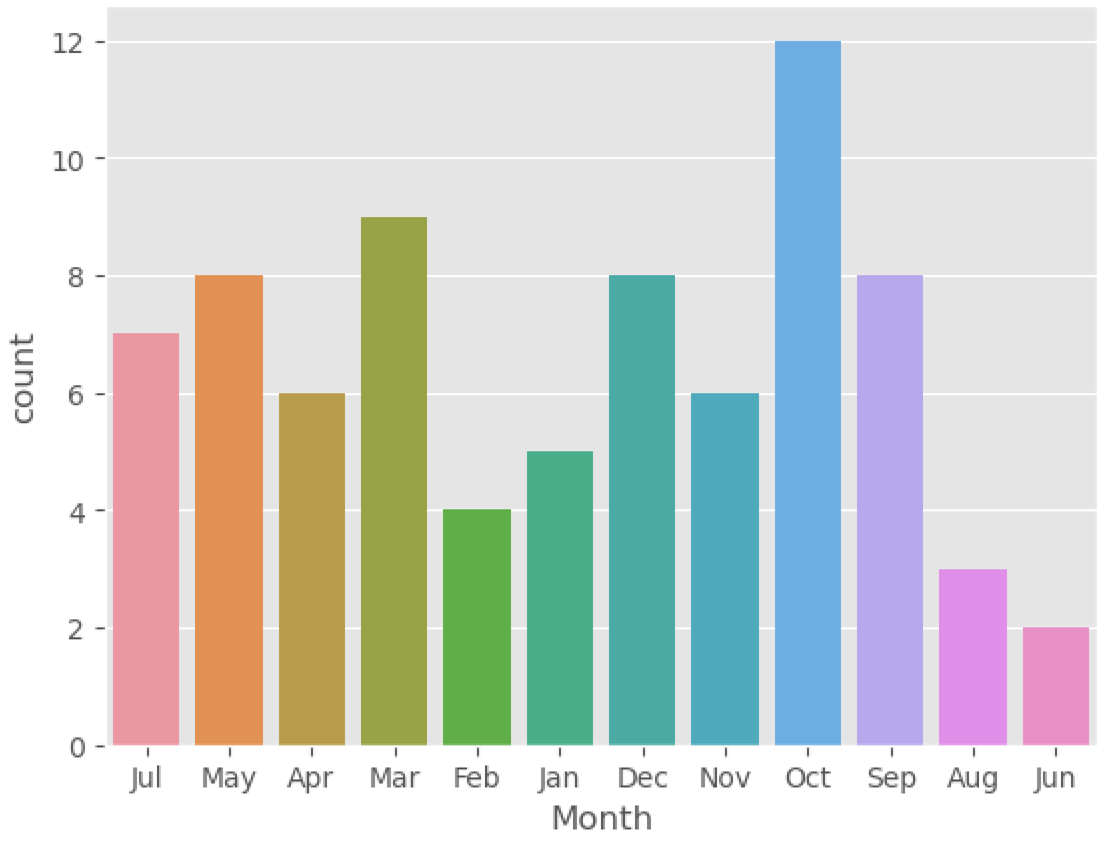
In [127]:
```
top10_videos = video_data.sort_values(by='Views', ascending=False).head(
```

In [128]:
```
ax1 = sns.barplot(x='Views', y='Title', data=top10_videos)
```



```
In [126]: video_data["Month"]=pd.to_datetime(video_data['Published_date']).dt.strf
          sns.countplot(data=video_data,x="Month")
```

```
Out[126]: <Axes: xlabel='Month', ylabel='count'>
```



```
In [ ]:
```

```
In [ ]:
```