SOE THAN Hein Htet

Introduction After reviewing the dataset, I will answer two questions. First question is which country or region has the highest number of job postings in the dataset? Second question is what is their average number of application received for different job roles?

```
!pip install opendatasets
```

```
Requirement already satisfied: opendatasets in /usr/local/lib/python3.1
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: kaggle in /usr/local/lib/python3.12/dis
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-
Requirement already satisfied: bleach in /usr/local/lib/python3.12/dis
Requirement already satisfied: certifi>=14.05.14 in /usr/local/lib/pyt
Requirement already satisfied: charset-normalizer in /usr/local/lib/py
Requirement already satisfied: idna in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: protobuf in /usr/local/lib/python3.12/d
Requirement already satisfied: python-dateutil>=2.5.3 in /usr/local/li
Requirement already satisfied: python-slugify in /usr/local/lib/python
Requirement already satisfied: requests in /usr/local/lib/python3.12/d
Requirement already satisfied: setuptools>=21.0.0 in /usr/local/lib/py
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.12/
Requirement already satisfied: text-unidecode in /usr/local/lib/python
Requirement already satisfied: urllib3>=1.15.1 in /usr/local/lib/pytho
Requirement already satisfied: webencodings in /usr/local/lib/python3.1
```

```
import opendatasets as od
od.download("https://www.kaggle.com/datasets/shashankshukla123123/linke
```

```
Skipping, found downloaded files in "./linkedin-job-data" (use force=T
```

```
#data manipulation
import numpy as np
import pandas as pd

#data visualization
import seaborn as sns
import matplotlib.pyplot as plt
```

```
linkedin_jobs = pd.read_csv('/content/linkedin-job-data/linkdin_Job_dat
```

Data Exploration

```
#check the first few rows of the dataset
linkedin_jobs.head()
```

	job_ID	job	location	company_id	company_name	work_type
0	3471657636	Data Analyst, Trilogy (Remote) - $60,000/year USD	Delhi, Delhi, India	NaN	Crossover	Remote
1	3471669068	Data Analyst, Trilogy (Remote) - $60,000/year USD	New Delhi, Delhi, India	NaN	Crossover	Remote
2	3474349934	Data Analyst - WFH	Greater Bengaluru Area	NaN	Uplers	Remote
3	3472816027	Data Analyst	Gurugram, Haryana, India	NaN	PVAR SERVICES	On-site
4	3473311511	Data Analyst	Mohali district, Punjab, India	NaN	Timeline Freight Brokers	On-site

```
#check summary information about the dataset
linkedin_jobs.describe(include='all')
```

	job_ID	job	location	company_id	company_name	work_ty
count	7.927000e+03	7894	7894	0.0	7892	77
unique	NaN	2991	151	NaN	2495	
top	NaN	Lead Java Software Engineer	Bengaluru, Karnataka, India	NaN	EPAM Anywhere	On-
freq	NaN	172	1324	NaN	1517	32
mean	3.466724e+09	NaN	NaN	NaN	NaN	N
std	5.778011e+07	NaN	NaN	NaN	NaN	N
min	1.419216e+08	NaN	NaN	NaN	NaN	N
25%	3.467367e+09	NaN	NaN	NaN	NaN	N
50%	3.471882e+09	NaN	NaN	NaN	NaN	N
75%	3.476181e+09	NaN	NaN	NaN	NaN	N
max	3.477823e+09	NaN	NaN	NaN	NaN	N

```
#check info about the dataset
linkedin_jobs.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7927 entries, 0 to 7926
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   job_ID              7927 non-null   int64  
 1   job                 7894 non-null   object 
 2   location            7894 non-null   object 
 3   company_id          0 non-null      float64
 4   company_name        7892 non-null   object 
 5   work_type           7736 non-null   object 
 6   full_time_remote    7848 non-null   object 
 7   no_of_employ        7603 non-null   object 
 8   no_of_application   7887 non-null   object 
 9   posted_day_ago      7920 non-null   object 
 10  alumni              4858 non-null   object 
 11  Hiring_person       5720 non-null   object 
 12  linkedin_followers  4814 non-null   object 
 13  hiring_person_link  5720 non-null   object 
 14  job_details         7881 non-null   object 
 15  Column1             0 non-null      float64
dtypes: float64(2), int64(1), object(13)
memory usage: 991.0+ KB
```

Data Cleaning

```
#remove unnecessary columns
linkedin_jobs_cleaned = linkedin_jobs.drop(columns=['company_id', 'jol
```

I drop some columns which are not going to be used in the analysis.

```
#check the cleaned dataset
linkedin_jobs_cleaned.head()
```

	job	location	company_name	work_type	full_time_remote	no_o
0	Data Analyst, Trilogy (Remote) - $60,000/year USD	Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	1 emp Serv
1	Data Analyst, Trilogy (Remote) - $60,000/year USD	New Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	1 emp Serv

```
#removing duplicates
linkedin_jobs.drop_duplicates(inplace=True)
```

```
#check how many duplicates were removed
print("Number of rows after removing duplicates:", linkedin_jobs.shap
```

```
Number of rows after removing duplicates: 7848
```

```
#job title and company name are not consistent in terms of letter cas
#standardize job titles and company names to title case
linkedin_jobs_cleaned['job'] = linkedin_jobs_cleaned['job'].str.title
linkedin_jobs_cleaned['company_name'] = linkedin_jobs_cleaned['compan
```

```
#check the changes
linkedin_jobs_cleaned[['job', 'company_name']].head()
```

```
                                                  job              company_name
0  Data Analyst, Trilogy (Remote) - $60,000/Year Usd                 Crossover
1  Data Analyst, Trilogy (Remote) - $60,000/Year Usd                 Crossover
2                                 Data Analyst - Wfh                    Uplers
3                                       Data Analyst             Pvar Services
4                                       Data Analyst  Timeline Freight Brokers
```

```
#check the clean dataset first 10 rows
linkedin_jobs_cleaned.head(10)
```

	job	location	company_name	work_type	full_time_remote	r
0	Data Analyst, Trilogy (Remote) - $60,000/Year Usd	Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	S
1	Data Analyst, Trilogy (Remote) - $60,000/Year Usd	New Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	S
2	Data Analyst - Wfh	Greater Bengaluru Area	Uplers	Remote	Full-time · Mid-Senior level	S
3	Data Analyst	Gurugram, Haryana, India	Pvar Services	On-site	Full-time	1
4	Data Analyst	Mohali district, Punjab, India	Timeline Freight Brokers	On-site	Full-time	1

```
#Removing unnecessary whitespaces and elements from 'job' column by sp
linkedin_jobs_cleaned['job'] = linkedin_jobs_cleaned['job'].str.split

#remove qutation marks from 'job' column
linkedin_jobs_cleaned['job'] = linkedin_jobs_cleaned['job'].str.repla
```

```
#check the cleaned dataset
linkedin_jobs_cleaned.head(10)
```

	job	location	company_name	work_type	full_time_remote	no_
0	Data Analyst	Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	em Ser
1	Data Analyst	New Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	em Ser
2	Data Analyst	Greater Bengaluru Area	Uplers	Remote	Full-time · Mid-Senior level	em Ser
3	Data Analyst	Gurugram, Haryana, India	Pvar Services	On-site	Full-time	1-1(
4	Data Analyst	Mohali district, Punjab, India	Timeline Freight Brokers	On-site	Full-time	1-1(
5	Data Analyst	Gurugram, Haryana, India	Airtel	On-site	Full-time · Entry level	10,001- Telecom

```
#check first 20 rows of 'job' column
linkedin_jobs_cleaned['job'].head(20)
```

```
                                                       job
 0                                            Data Analyst
 1                                            Data Analyst
 2                                            Data Analyst
 3                                            Data Analyst
 4                                            Data Analyst
 5                                            Data Analyst
 6                                            Data Analyst
 7                                       Shopify Developer
 8                                      Database Developer
 9                                       Shopify Developer
10                                       Shopify Developer
11                                           Data Engineer
12                                                 Tableau
13                                           Data Engineer
14                                    Python Data Engineer
15                                          Zoho Developer
16                              Salesforce Cpq Developer
17                                        Golang Developer
18   Sap Customer Data Cloud Business Analyst Techn...
19                                     Quickbase Developer

dtype: object
```

```
#check row 18 only
linkedin_jobs_cleaned.iloc[18]
```

	18
job	Sap Customer Data Cloud Business Analyst Techn...
location	Bengaluru, Karnataka, India
company_name	Mobile Programming Llc
work_type	On-site
full_time_remote	Full-time
no_of_employ	1,001-5,000 employees
no_of_application	13
posted_day_ago	20 hours
linkedin_followers	269,352 followers

dtype: object

```
#check 'job' column unique values
linkedin_jobs_cleaned['job'].unique()
```

```
array(['Data Analyst', 'Shopify Developer', 'Database Developer', ...,
       'Aws Cloud Specialist', 'Vue Js', 'Power Train Function
Developer'],
      dtype=object)
```

```
#see what rows contain 'Data Scientist' in 'job' column
linkedin_jobs_cleaned[linkedin_jobs_cleaned['job'].str.contains('Data
```

	job	location	company_name	work_type	full_time_remote	no
188	Data Scientist	Bengaluru, Karnataka, India	Tata Consultancy Services	On-site	Full-time · Associate	e Se
193	Data Scientist	Chennai, Tamil Nadu, India	Tata Consultancy Services	On-site	Full-time · Associate	e Se
240	Data Scientist	India	A.P. Moller - Maersk	Remote	NaN	
638	Data Scientist	Bengaluru, Karnataka, India	Tredence Inc.	Hybrid	Full-time · Associate	Co
1258	Data Scientist/ Business Analyst/ Data Analyst...	Mumbai, Maharashtra, India	Indiafirst Life	On-site	Full-time · Mid-Senior level	
...	...	...	...	...	...	
6013	Lead Data Scientist	Vadodara, Gujarat, India	Nielseniq	On-site	Full-time · Entry level	

```
#check the tail of the cleaned dataset
linkedin_jobs_cleaned.tail(20)
```

	job	location	company_name	work_type	full_time_rem
7907	Ibm Mdm Developer	Pune, Maharashtra, India	Ltimindtree	Hybrid	Full-t
	Java	Mumbai			

7908	Java Microservices	Metropolitan Region	Ltimindtree	Hybrid	Full-time · Assoc
7909	Sr. Data Engineer	Hyderabad, Telangana, India	Bizacuity Solutions Pvt Ltd	On-site	Full-time · Mid-Se l
7910	Power Platform Developer	Noida, Uttar Pradesh, India	Globallogic	Hybrid	Full-time · Assoc
7911	Integration Developer	Hyderabad, Telangana, India	Daysmart	Remote	Full-time · Mid-Se l
7912	Java Spring Boot	Bengaluru, Karnataka, India	Tata Consultancy Services	On-site	Full-time · Mid-Se l
7913	Informatica Developer	Kochi, Kerala, India	Tata Consultancy Services	On-site	Full-t
7914	Business Support/Partner Alliance Coordinator	Navi Mumbai, Maharashtra, India	Cloudxchange.Io	On-site	Full-time · Assoc
7915	Xceptor Developer	Chennai, Tamil Nadu, India	Virtusa	Hybrid	Full-time · Mid-Se l
7916	Senior Qa Engineer	Noida, Uttar Pradesh, India	Gojoko Technologies	On-site	Full-time · Assoc
7917	Linux Device Driver Developer	Bengaluru, Karnataka, India	Smartsoc Solutions Pvt Ltd	On-site	Contract · Mid-Se l
7918	Senior Java Software Developer	Pune, Maharashtra, India	Epmware	Remote	Full-t

Data Analysis

```
#split location to get country
linkedin_jobs_cleaned['country'] = linkedin_jobs_cleaned['location'].s
```

```
#chekc the changes
linkedin_jobs_cleaned[['location', 'country']].head()
```

	location	country
0	Delhi, Delhi, India	India
1	New Delhi, Delhi, India	India
2	Greater Bengaluru Area	Greater Bengaluru Area
3	Gurugram, Haryana, India	India
4	Mohali district, Punjab, India	India

```
#unique countries in the dataset
linkedin_jobs_cleaned['country'].unique()
```

```
array(['India', 'Greater Bengaluru Area', 'Mumbai Metropolitan
Region',
       'Greater Kolkata Area', nan, 'Pune/Pimpri-Chinchwad Area',
       'Greater Hyderabad Area', 'Greater Delhi Area', 'APAC',
       'Greater Coimbatore Area', 'Greater Vadodara Area',
       'Greater Chennai Area', 'Greater Ahmedabad Area',
       'Greater Nagpur Area'], dtype=object)
```

I figured out that there is only one country which is India. So, I will change my approach to figure out highest number of job posting by region(city, state).

```
#Split location to get city
linkedin_jobs_cleaned['city'] = linkedin_jobs_cleaned['location'].str
```

```
#Check the changes
linkedin_jobs_cleaned[['location', 'city']].head()
```

	location	city
0	Delhi, Delhi, India	Delhi
1	New Delhi, Delhi, India	New Delhi
2	Greater Bengaluru Area	Greater Bengaluru Area
3	Gurugram, Haryana, India	Gurugram
4	Mohali district, Punjab, India	Mohali district

```
#split the location into a list
loc_split = linkedin_jobs_cleaned['location'].str.split(',')
```

```
#create a region column by joining the first two elements of location
linkedin_jobs_cleaned['region_state'] = loc_split.str[0]+', '+loc_spl
```

```
#fixing the NaN values in 'region_state' column
linkedin_jobs_cleaned['region_state'] = linkedin_jobs_cleaned['region_
```

```
#check the changes
linkedin_jobs_cleaned[['location', 'region_state']].head()
```

	location	region_state
0	Delhi, Delhi, India	Delhi, Delhi
1	New Delhi, Delhi, India	New Delhi, Delhi
2	Greater Bengaluru Area	Greater Bengaluru Area
3	Gurugram, Haryana, India	Gurugram, Haryana
4	Mohali district, Punjab, India	Mohali district, Punjab

```
#check unique regions/states
linkedin_jobs_cleaned['region_state'].unique()
```

```
array(['Delhi, Delhi', 'New Delhi, Delhi', 'Greater Bengaluru Area',
       'Gurugram, Haryana', 'Mohali district, Punjab',
       'Bengaluru, Karnataka', 'Delhi, India', 'Noida, Uttar
Pradesh',
       'Hyderabad, Telangana', 'India', 'Chennai, Tamil Nadu',
       'Kolkata, West Bengal', 'Madurai, Tamil Nadu', 'Pune,
Maharashtra',
       'Mumbai, Maharashtra', 'Kochi, Kerala',
       'Visakhapatnam, Andhra Pradesh', 'Ahmedabad, Gujarat',
       'Vadodara, Gujarat', 'Hosur, Tamil Nadu', 'Coimbatore, Tamil
Nadu',
       'Chengannur, Kerala', 'Mumbai Metropolitan Region',
       'Bhopal, Madhya Pradesh', 'Maharashtra, India', 'Surat,
Gujarat',
       'Ludhiana, Punjab', 'Jaipur, Rajasthan', 'Chandigarh,
Chandigarh',
       'Trivandrum, Kerala', 'Delhi Cantonment, Delhi',
       'Thane, Maharashtra', 'Gujarat, India', 'Navi Mumbai,
Maharashtra',
       'Greater Kolkata Area', 'Bangalore Urban, Karnataka',
       'Gurgaon, Haryana', 'Nagpur, Maharashtra', 'Kalyan,
Maharashtra',
       'Dehradun, Uttarakhand', 'Kozhikode, Kerala',
       'Tenkasi, Tamil Nadu', 'Amritsar, Punjab', 'Patna, Bihar',
       'Vijayawada, Andhra Pradesh', 'Tamil Nadu, India',
       'Borivali, Maharashtra', 'Udipi, Karnataka',
       'Thiruvananthapuram, Kerala', 'Sahibzada Ajit Singh Nagar,
Punjab',
       nan, 'Telangana, India', 'Indore, Madhya Pradesh',
       'Pune/Pimpri-Chinchwad Area', 'Kerala, India',
       'Lucknow, Uttar Pradesh', 'Greater Hyderabad Area',
       'Guindy, Tamil Nadu', 'Vapi, Gujarat', 'Alwar, Rajasthan',
       'Greater Delhi Area', 'Rajkot, Gujarat',
       'Bangalore Urban district, India', 'Faridabad, Haryana',
'APAC',
       'Kanpur, Uttar Pradesh', 'Bandra, Maharashtra',
       'Thiruvarur, Tamil Nadu', 'Puducherry, Puducherry',
       'Mangaluru, Karnataka', 'Chandigarh, India',
       'Aurangabad, Maharashtra', 'Gurgaon Rural, Haryana',
       'Pimpri Chinchwad, Maharashtra',
       'Kolkata metropolitan area, West Bengal', 'Goa, India',
       'Gandhinagar, Gujarat', 'Ambala, Haryana',
       'Greater Coimbatore Area', 'Ghaziabad, Uttar Pradesh',
       'Hoshiarpur, Punjab', 'Manesar, Haryana', 'Bhubaneshwar,
Odisha',
       'South Delhi, Delhi', 'Saket, Delhi', 'Karunagappally,
Kerala',
       'Rohini, Delhi', 'Dumka, Jharkhand', 'Srinagar, Jammu &
Kashmir',
       'Jabalpur, Madhya Pradesh', 'Greater Vadodara Area',
       'Haryana, India', 'Greater Chennai Area', 'Ernakulam,
Kerala',
       'Rupnagar, Punjab', 'Thrissur, Kerala', 'Siwani, Haryana',
       'Vishakhapatnam, Andhra Pradesh', 'Ulhasnagar, Maharashtra',
       'Bengaluru East, Karnataka', 'Itanagar, Arunachal Pradesh',
       'Punjab, India', 'Kangra, Himachal Pradesh',
       'Kanpur Nagar, Uttar Pradesh', 'Alipur, Delhi',
       'Nagercoil, Tamil Nadu', 'Jammu, Jammu & Kashmir',
```

```
#check how many Nans in 'region_state' column
linkedin_jobs_cleaned['region_state'].isna().sum()
```

```
np.int64(33)
```

```
#see one Nan value in region_state for the whole row
print(linkedin_jobs_cleaned[linkedin_jobs_cleaned['region_state'].isna
```

```
job                          NaN
location                     NaN
company_name                 NaN
work_type                    NaN
full_time_remote             NaN
no_of_employ                 NaN
no_of_application            NaN
posted_day_ago        11 minutes
linkedin_followers           NaN
country                      NaN
city                         NaN
region_state                 NaN
Name: 317, dtype: object
```

I am encountering a problem. I created a region_state(city, state) column but there are several NaN value in my new column because in the location(the original column), some row only has one value like 'Greater Bengaluru Area'. So, my approach is to manipulate the location string to get city and state regions.

```
#drop existing 'region_state' column if exists
if 'region_state' in linkedin_jobs_cleaned.columns:
    linkedin_jobs_cleaned = linkedin_jobs_cleaned.drop(columns=['regi

#function to format region_state column
def format_region_state(loc):
    #handle NaN or non-string values
    if pd.isna(loc) or not isinstance(loc, str):
        return "Unknown, Unknown"

    #manipulate location string to get city and state/region
    if loc == "Greater Bengaluru Area":
        city, state = "Bengaluru", "Greater Bengaluru Area"

    #the list comprehension
    #if there is a comma in the location string, it will take the firs
    else:
        parts = [p.strip() for p in loc.split(',')]

        if len(parts) >= 2:
            city, state = parts[0], parts[1]
        else:
            city, state = loc, loc #if there is no comma, it will use

    return f"{city}, {state}"
```

```
#apply the function to the 'region_state' column
linkedin_jobs_cleaned['region_state'] = linkedin_jobs_cleaned['locati
```

```
#check any NaN values in 'region_state' column
print("Number of NaN values in 'region_state' column:", linkedin_jobs_
```

```
Number of NaN values in 'region_state' column: 0
```

```
linkedin_jobs_cleaned['region_state']
```

	region_state
0	Delhi, Delhi
1	New Delhi, Delhi
2	Bengaluru, Greater Bengaluru Area
3	Gurugram, Haryana
4	Mohali district, Punjab
...	...
7922	Kochi, Kerala
7923	Gurugram, Haryana
7924	Hyderabad, Telangana
7925	Bengaluru, Karnataka
7926	Bengaluru, Karnataka

7927 rows × 1 columns

dtype: object

Which country or region has the highest number of job postings in the dataset?

1. Which region is hiring the most?

```
#count hired jobs by region/state
most_hired_jobs_by_region = linkedin_jobs_cleaned['region_state'].valu

#which region/state has the highest number of job postings
top_region = most_hired_jobs_by_region.idxmax()
top_region_count = most_hired_jobs_by_region.max()

print(f"The top REGION is: {top_region} with {top_region_count} postir
```

```
The top REGION is: Bengaluru, Karnataka with 1324 postings.
```

The below code is the same result but using 'groupby'.

```
#Using group by region, then count the 'job' column specifically
region_count_group_by = linkedin_jobs_cleaned.groupby('region_state')

top_region_group_by = region_count_group_by.idxmax()
top_region_count_group_by = region_count_group_by.max()
print(f"The top REGION is: {top_region_group_by} with {top_region_cou
```

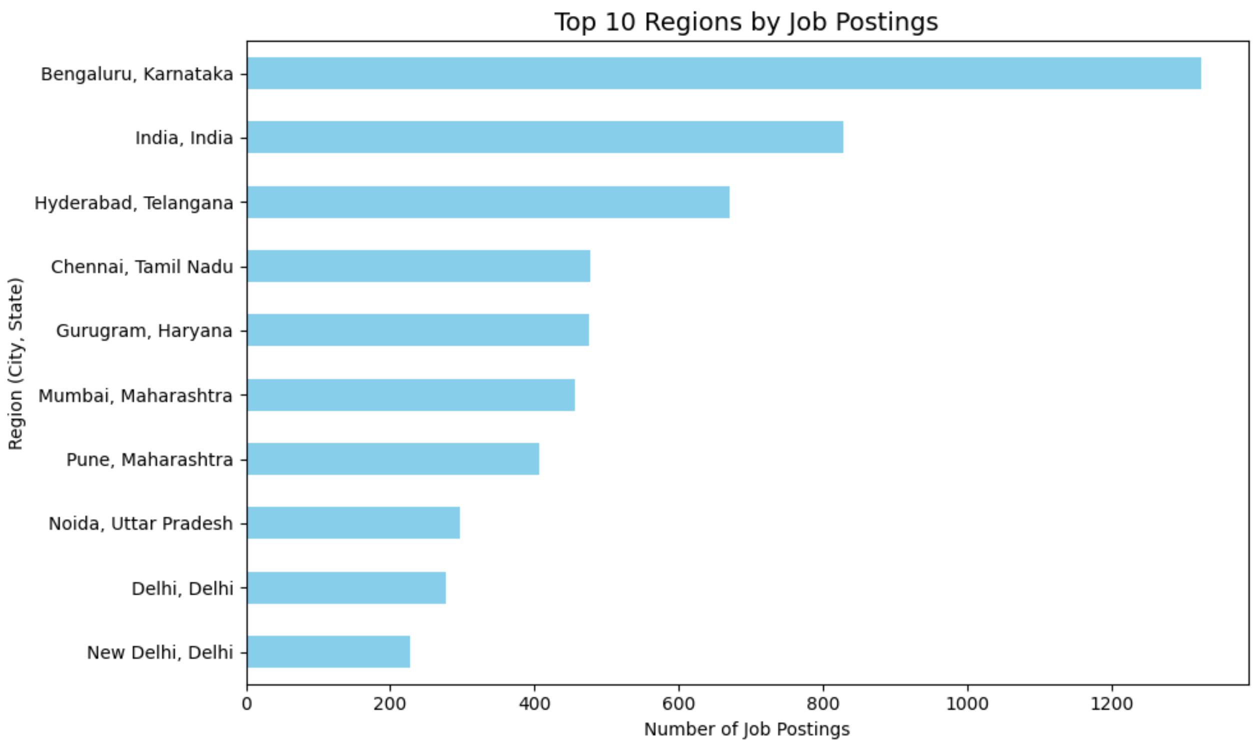
```
The top REGION is: Bengaluru, Karnataka with 1324 postings.
```

Now I will put the result in the visualization by using matplotlib.plot.

```
#to embed plots within the notebook
%matplotlib inline

#import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
```

Visualization

```
# Top 10 Regions
plt.figure(figsize=(10, 6)) #width = 10, height = 6
linkedin_jobs_cleaned['region_state'].value_counts().head(10).sort_va

plt.title('Top 10 Regions by Job Postings', fontsize=14)
plt.xlabel('Number of Job Postings')
plt.ylabel('Region (City, State)')
plt.tight_layout() # This ensures your long labels aren't cut off at
plt.show()
```



The analysis shows that Bengaluru, Karnataka acts as the primary engine for employment, significantly outperforming all other areas in job volume. This high concentration suggests a centralized hub for the data science job market industry, making it the most critical destination for talent looking for diverse opportunities. According to the analysis, the job seekers might have better opportunity searching job in that region.

My second research question. What is the average number of applications received for data analyst job role? As I somehow clean the job column, I will start with extracting number.

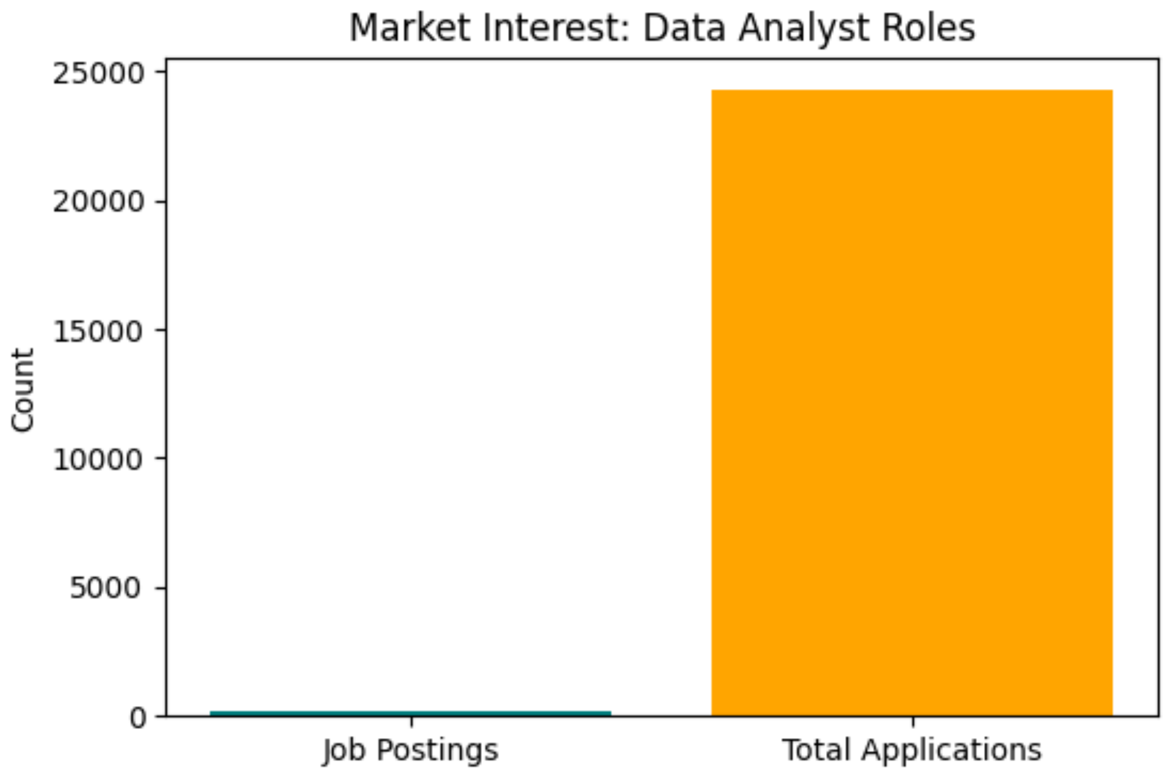
```
linkedin_jobs_cleaned
```

	job	location	company_name	work_type	full_time_remote	no_
0	Data Analyst	Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	en Se
1	Data Analyst	New Delhi, Delhi, India	Crossover	Remote	Full-time · Associate	en Se
2	Data Analyst	Greater Bengaluru Area	Uplers	Remote	Full-time · Mid-Senior level	en Se
3	Data Analyst	Gurugram, Haryana, India	Pvar Services	On-site	Full-time	1-1
4	Data Analyst	Mohali district, Punjab, India	Timeline Freight Brokers	On-site	Full-time	1-1
...	...	...	...	...	...	
7922	Back End Developer	Kochi, Kerala, India	Orion Innovation	Hybrid	Full-time · Associate	5 en Se

7923	Software Engineer	Gurugram, Haryana, India	Uplers	On-site	Full-time · Mid-Senior level	en Sei
7924	Vue Js	Hyderabad, Telangana, India	Tata Consultancy Services	On-site	Full-time · Mid-Senior level	en Sei

```
# 1. Filter for just Data Analyst roles
# Using .str.contains handles "Senior Data Analyst" and "Junior Data A
da_filter = linkedin_jobs_cleaned['job'].str.contains('Data Analyst',
da_data = linkedin_jobs_cleaned[da_filter].copy()

# 2. Extract application numbers (dealing with the "onion"/null values
da_data['apps_clean'] = (
    da_data['no_of_application']
    .astype(str)
    .str.extract('(\d+)') # Extracts just the numbers
    .astype(float)
    .fillna(0) # Treats missing values as 0 for the total
)

# 3. Calculate your two specific numbers
total_da_postings = len(da_data)
total_da_applications = da_data['apps_clean'].sum()

print(f"Number of Data Analyst postings: {total_da_postings}")
print(f"Total Applications for these roles: {int(total_da_applications

# 4. Visualization: A simple 'Summary Card' style plot
plt.figure(figsize=(6, 4))
plt.bar(['Job Postings', 'Total Applications'], [total_da_postings, to
plt.title('Market Interest: Data Analyst Roles')
plt.ylabel('Count')
plt.show()
```

```
<>:10: SyntaxWarning: invalid escape sequence '\d'
<>:10: SyntaxWarning: invalid escape sequence '\d'
/tmp/ipython-input-2245833006.py:10: SyntaxWarning: invalid escape sequ
  .str.extract('(\d+)') # Extracts just the numbers
Number of Data Analyst postings: 224
Total Applications for these roles: 24283
```



```
print(f"Total Data Analyst Job Postings: {total_da_postings}")
print(f"Total Applications for Data Analyst Roles: {int(total_da_appl
print(f"Applications per Posting: {total_da_applications / total_da_po
```

```
Total Data Analyst Job Postings: 224
Total Applications for Data Analyst Roles: 24283
Applications per Posting: 108.41
```

I found that the Data Analyst role is incredibly crowded, with 24,283 applications pouring into just 224 job postings. This means every single opening is being chased by about 108 people, which shows just how competitive this field has become. By matching these two numbers, it's clear that while the demand for data analysts is high, the number of people trying to get in is even higher.

```
#drop unnecessary columns
linkedin_jobs_final= linkedin_jobs_cleaned.drop(columns=['location',

linkedin_jobs_final.head()
```

	job	company_name	work_type	full_time_remote	no_of_employ	no_o
0	Data Analyst	Crossover	Remote	Full-time · Associate	1,001-5,000 employees · IT Services and IT Con...	
1	Data Analyst	Crossover	Remote	Full-time · Associate	1,001-5,000 employees · IT Services and IT Con...	

```
#export the final cleaned dataset to a new CSV file
linkedin_jobs_final.to_csv('linkedin_jobs_final.csv', index=False)
```

AI Usage Log

Tool Used: Gemini

Purpose & Process: I used AI as a technical assistant to streamline the data-cleaning phase of this project.

Specifically, the AI helped with: Code Debugging: Resolving data type errors when calculating the average number of applications. Syntax Support: Writing the Regex (Regular Expression) logic needed to strip non-numeric text from the "no_of_application" column. Data Transformation: Standardizing job titles and company names to Title Case for better consistency.

Statement of Integrity: While the AI assisted with specific Python syntax and formatting, all research questions, data interpretations, and the discovery of the 108:1 competition ratio were my own. I have reviewed every line of code to ensure I fully understand the logic used to generate the final results.

```
print("The project is completed successfully.")
```

```
The project is completed successfully.
```

Conclusion

This project successfully identified that while job availability is high, the competition for 'Data Analyst' roles is even higher, with over 24,000 people competing for just 224 spots. By cleaning the data and matching these two metrics, I’ve shown that this field is a major entry point for talent but requires candidates to stand out in a heavily saturated market. Lastly, these numbers prove that the volume of interest in data careers currently far outweighs the number of open vacancies.