Wppool is a WordPress products seller company whose product has been invented new level of power, aesthetics and users-centric design with its innovative offerings.

A person who has WordPress product companies include website owners, businesses, developers, agencies, digital marketers and freelancers

Wppool provides differen kind of Wordpress plugins and add-ons product with lucrative pricesWith our products, entrepreneurs can easily improve the effectiveness of their goals and that is why we have achieved a unique feature.For example, FlexTable enables real-time Google Sheets synchronization, saving you time on manual updates, while FormyChat helps convert leads seamlessly

Import Libraries

In [34]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import  seaborn as sns
import sqlite3
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
```

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
import scipy.stats as stats
```

In [2]:
```
df = pd.read_csv('wppool_growth_data_sample.csv')
df.head(5)
```

Out[2]:

	user_id	install_date	last_active_date	subscription_type	country	total_sessions	page_views	download_clicks	activation_status	days_ac
0	1	6/29/2023	7/12/2023	Free	UK	3	15	1	1	
1	2	4/10/2023	7/25/2023	Free	India	133	665	0	1	
2	3	10/25/2023	12/7/2023	Free	USA	53	106	0	1	
3	4	8/26/2023	11/9/2023	Pro	Canada	242	242	0	1	
4	5	5/14/2023	11/22/2023	Free	UK	12	48	0	1	

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   user_id            20000 non-null  int64 
 1   install_date       20000 non-null  object
 2   last_active_date   20000 non-null  object
 3   subscription_type  20000 non-null  object
 4   country            20000 non-null  object
 5   total_sessions     20000 non-null  int64 
 6   page_views         20000 non-null  int64 
 7   download_clicks    20000 non-null  int64 
 8   activation_status  20000 non-null  int64 
 9   days_active        20000 non-null  int64 
 10  pro_upgrade_date   4029 non-null   object
 11  plan_type          4029 non-null   object
 12  monthly_revenue    20000 non-null  int64 
 13  churned            20000 non-null  int64 
dtypes: int64(8), object(6)
memory usage: 2.1+ MB
```

In [4]:
```
df.describe()    # for checking statistical distribution
```

Out[4]:

	user_id	total_sessions	page_views	download_clicks	activation_status	days_active	monthly_revenue	churned
count	20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.00000	20000.000000	20000.000000
mean	10000.500000	91.914500	276.308900	0.102250	0.990550	91.28080	11.774050	0.285250
std	5773.647028	62.523862	244.775351	0.302984	0.096753	80.67644	26.845358	0.451545
min	1.000000	1.000000	1.000000	0.000000	0.000000	0.00000	0.000000	0.000000
25%	5000.750000	44.000000	96.000000	0.000000	1.000000	24.00000	0.000000	0.000000
50%	10000.500000	85.000000	208.000000	0.000000	1.000000	68.00000	0.000000	0.000000
75%	15000.250000	126.000000	396.000000	0.000000	1.000000	140.00000	0.000000	1.000000
max	20000.000000	300.000000	1500.000000	1.000000	1.000000	364.00000	99.000000	1.000000

Missing value handiling

In [62]:
```
df.isnull().sum()              #Find out the every columns null value
```

Out[62]:
```
user_id                  0
install_date             0
last_active_date         0
subscription_type        0
country                  0
total_sessions           0
page_views               0
download_clicks          0
activation_status        0
days_active              0
pro_upgrade_date     15971
plan_type            15971
monthly_revenue          0
churned                  0
dtype: int64
```

Comment:Here we have seen that there is no missing value except pro_upgrade_date and plan_type .Since pro_upgrade_date contain only date variable and plan type contain categorical vaiable.so we can fill up that by 0.Besides this, plan_type has also more missing value.since without any plan company didn't get any revenue so we can fill up the null value by 0.Beacuse,if there exist any plan then get revenue neither 0.

In [63]:
```
df['pro_upgrade_date'] = df['pro_upgrade_date'].fillna(0)    #fill up the missing value by 0
df['plan_type'] = df['plan_type'].fillna(0)
```

In [64]:
```
df.isnull().sum()              #Again check to check the missing value
```

Out[64]:
```
user_id              0
install_date         0
last_active_date     0
subscription_type    0
country              0
total_sessions       0
page_views           0
download_clicks      0
activation_status    0
days_active          0
pro_upgrade_date     0
plan_type            0
monthly_revenue      0
churned              0
dtype: int64
```

In [12]:
```
df.nunique()
```

Out[12]:
```
user_id              20000
install_date           366
last_active_date       357
subscription_type        2
country                  7
total_sessions         300
page_views             856
download_clicks          2
activation_status        2
days_active            361
pro_upgrade_date       338
plan_type                4
monthly_revenue          4
churned                  2
dtype: int64
```

2.User Engagement Analysis

In [48]:
```
# Connect to SQLite (in-memory)
conn = sqlite3.connect(":memory:")
```

```
# Store 'df' DataFrame in sqlite3
df.to_sql("df_table", conn, index=False, if_exists="replace")
```

Out[48]: 20000

Q:(2.a) Identify the average number of sessions for Free vs. Pro users

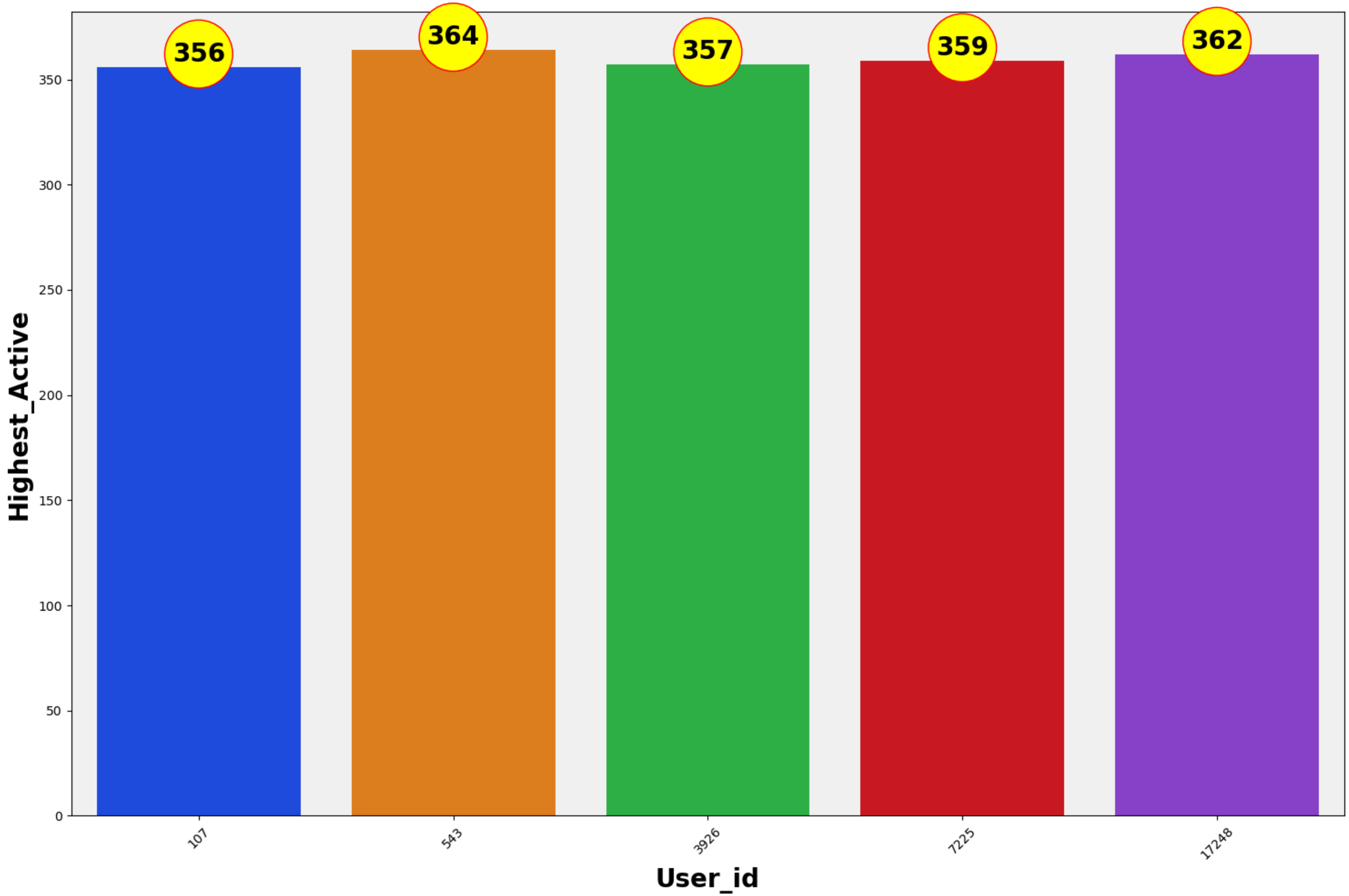
In [14]:
```
Query1="select subscription_type,avg(total_sessions) as Average_Session from df_table group by subscription_type"
Summary1= pd.read_sql(Query1,conn)
print(Summary1)
```

```
  subscription_type  Average_Session
0              Free        76.081210
1               Pro       154.677836
```

Q:(2.b) Find the top 5 most active users based on total sessions

In [15]:
```
Query2= "select user_id,max(days_active) as Highest_Active,total_sessions from df_table group by user_id,total_sessions order
Summary2= pd.read_sql(Query2,conn)
plt.figure(figsize =(15,10))
ap=sns.barplot(x='user_id',y='Highest_Active',data=Summary2,palette = "bright")
plt.xlabel('User_id',fontsize=20,fontweight='bold')
plt.ylabel('Highest_Active',fontsize=20,fontweight='bold')
plt.xticks(rotation=45)
for container in ap.containers:
    ap.bar_label(container,fontsize=20,
               bbox = {'boxstyle':'circle','facecolor':'yellow','edgecolor':'red'},fontweight = 'bold')
ap.set_facecolor("#f4f4f4")
ap.grid(False)
plt.tight_layout()
plt.show
```

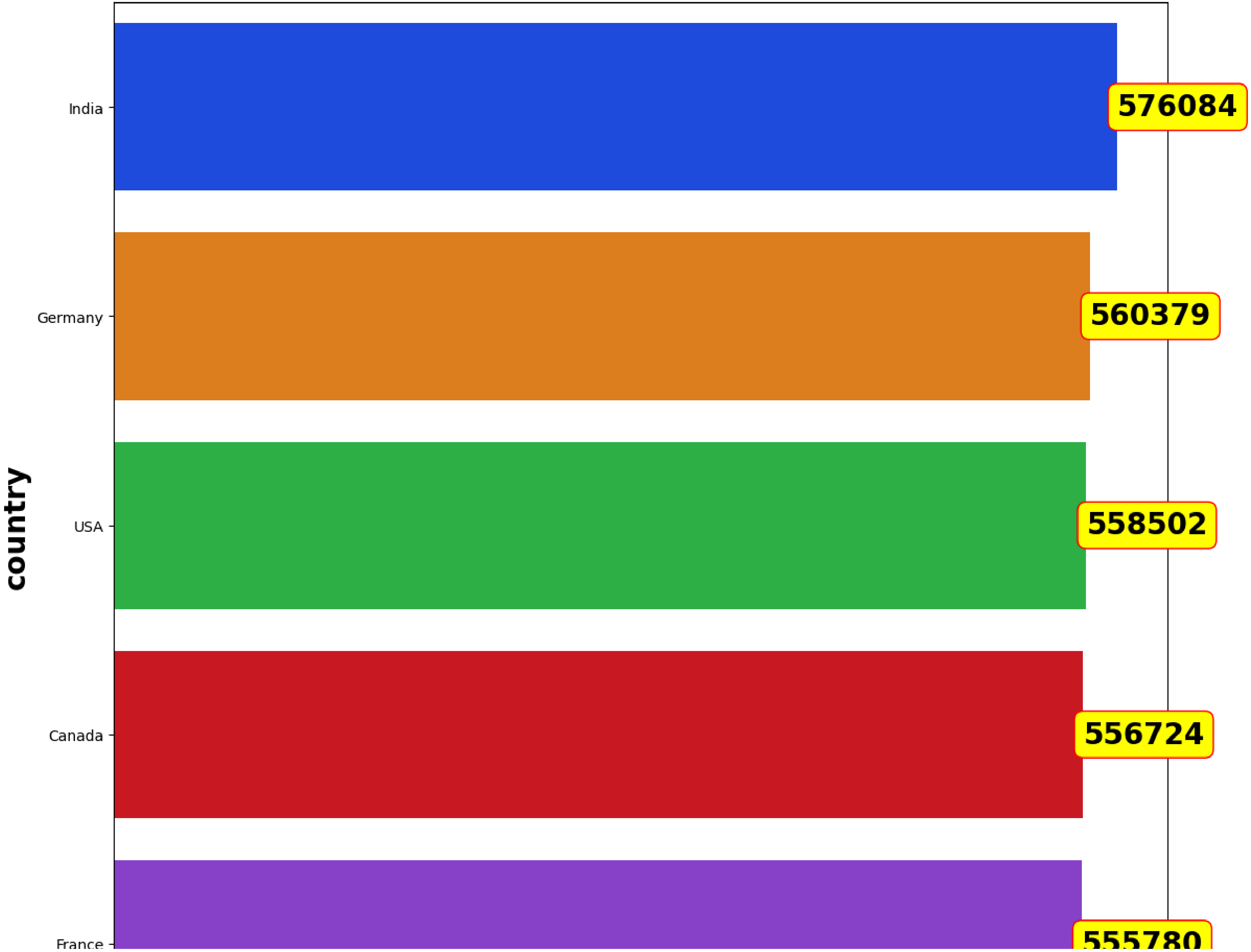
Out[15]: <function matplotlib.pyplot.show(close=None, block=None)>

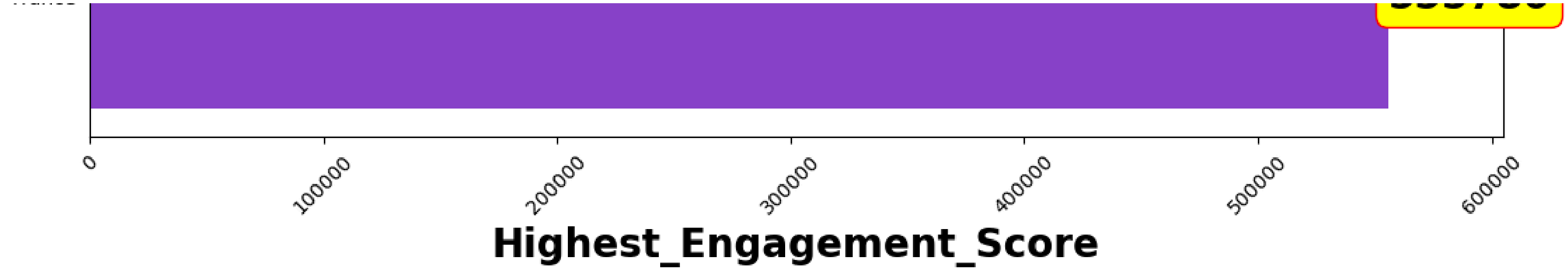


Q:(2.c) Identify the top 5 countries with the highest engagement.

In [16]:
```
Query3 = "select country,sum(total_sessions)+sum(days_active)+sum(monthly_revenue) as Highest_Engagement_Score from df_table g
Summary3 = pd.read_sql(Query3,conn)
plt.figure(figsize =(13,13))
az = sns.barplot(x='Highest_Engagement_Score',y='country',data=Summary3,palette = "bright")
plt.xlabel('Highest_Engagement_Score',fontsize=20,fontweight='bold')
plt.ylabel('country',fontsize=20,fontweight='bold')
plt.xticks(rotation=45)
for container in az.containers:
    az.bar_label(container,fontsize=20,
                bbox = {'boxstyle':'round','facecolor':'yellow','edgecolor':'red'},fontweight = 'bold')
ap.set_facecolor("#f4f4f4")
ap.grid(False)
plt.show
```

Out[16]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```


India
Germany
USA
Canada
France
country
576084
560379
558502
556724
555780



Q:(3.a) Calculate the overall churn rate for Free vs. Pro users

In [17]:
```
Query4 = "SELECT subscription_type,COUNT(*) AS total_users,SUM(churned) AS churned_users,(SUM(churned) * 1.0 / COUNT(*)) * 100
Summary4 = pd.read_sql(Query4,conn)
print(Summary4)
```

```
  subscription_type  total_users  churned_users  churn_rate_percentage
0              Free        15971           4567              28.595579
1               Pro         4029           1138              28.245222
```

In [3]:
```
df2 = pd.read_csv('wppool_growth_data_sample.csv')                  #Again import for making new dataset
```

In [5]:
```
le = LabelEncoder()                #For label encoding
df2['install_date'] = le.fit_transform(df2['install_date'])
df2['subscription_type'] = le.fit_transform(df2['subscription_type'])
df2['last_active_date'] = le.fit_transform(df2['last_active_date'])
df2['country'] = le.fit_transform(df2['country'])
df2['plan_type'] = le.fit_transform(df2['plan_type'])
df2['pro_upgrade_date'] = le.fit_transform(df2['pro_upgrade_date'])
```

In [6]:
```
#Data type checking
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   user_id            20000 non-null  int64
 1   install_date       20000 non-null  int32
 2   last_active_date   20000 non-null  int32
 3   subscription_type  20000 non-null  int32
 4   country            20000 non-null  int32
 5   total_sessions     20000 non-null  int64
 6   page_views         20000 non-null  int64
 7   download_clicks    20000 non-null  int64
 8   activation_status  20000 non-null  int64
 9   days_active        20000 non-null  int64
 10  pro_upgrade_date   20000 non-null  int32
 11  plan_type          20000 non-null  int32
 12  monthly_revenue    20000 non-null  int64
 13  churned            20000 non-null  int64
dtypes: int32(6), int64(8)
memory usage: 1.7 MB
```

Q:(3.b) Identify the top 3 factors contributing to churn using correlation or regression analysis.

In [21]:
```
correlation = df2.corr(method ='pearson')
print(correlation['churned'].sort_values(ascending=True).to_string())
```

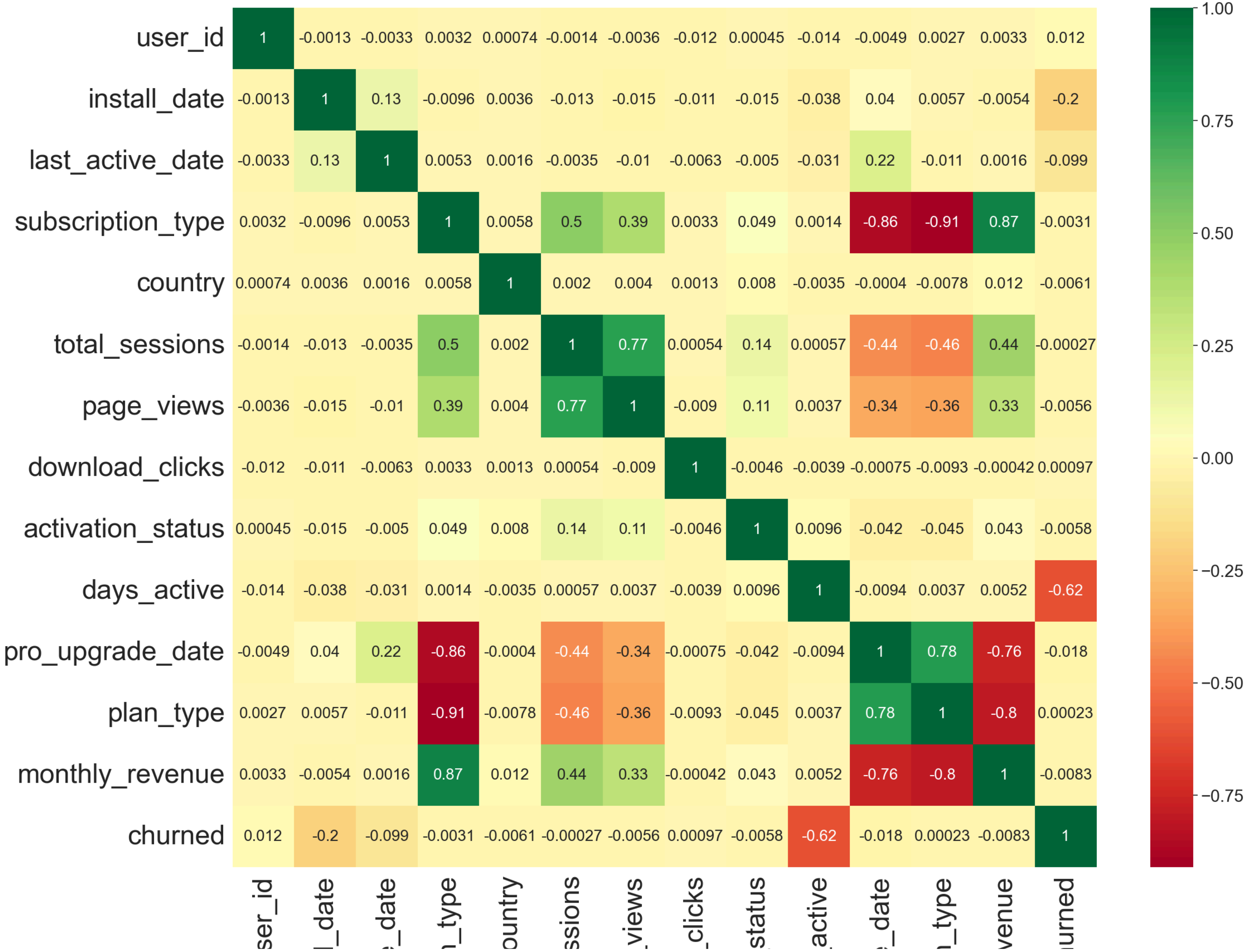
```
days_active         -0.617213
install_date        -0.196904
last_active_date    -0.099061
pro_upgrade_date    -0.018370
monthly_revenue     -0.008328
country             -0.006102
activation_status   -0.005823
page_views          -0.005605
subscription_type   -0.003112
total_sessions      -0.000273
plan_type            0.000227
download_clicks      0.000974
user_id              0.012038
churned              1.000000
```

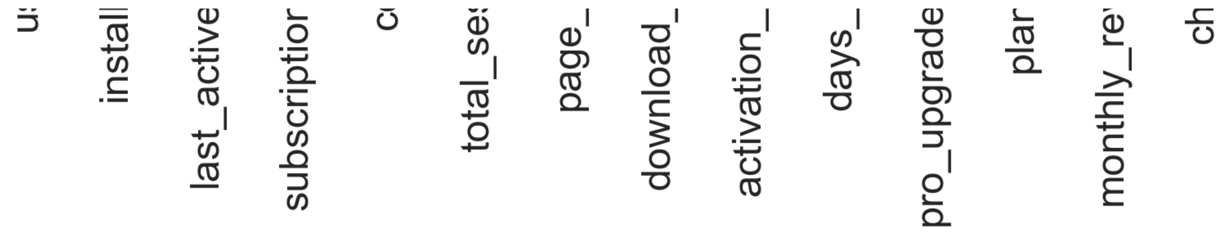
Comment: From Here,we can see that days_active,install_date and last_active_date are negetively correleted to churn

In heatmap we may see the total correlation in our dataset

In [68]:

```
sns.set(font_scale =2)
plt.subplots(figsize =(25,20))
heat_plot = sns.heatmap(df2.corr(method='pearson'),annot=True,cmap ='RdYlGn',annot_kws ={'size':20})
plt.yticks(fontsize =35)
plt.xticks(fontsize =35)

plt.show()
```


user_id
install_date
last_active_date
subscription_type
country
total_sessions
page_views
download_clicks
activation_status
days_active
pro_upgrade_date
plan_type
monthly_revenue
churned
1 -0.0013 -0.0033 0.0032 0.00074 -0.0014 -0.0036 -0.012 0.00045 -0.014 -0.0049 0.0027 0.0033 0.012
-0.0013 1 0.13 -0.0096 0.0036 -0.013 -0.015 -0.011 -0.015 -0.038 0.04 0.0057 -0.0054 -0.2
-0.0033 0.13 1 0.0053 0.0016 -0.0035 -0.01 -0.0063 -0.005 -0.031 0.22 -0.011 0.0016 -0.099
0.0032 -0.0096 0.0053 1 0.0058 0.5 0.39 0.0033 0.049 0.0014 -0.86 -0.91 0.87 -0.0031
0.00074 0.0036 0.0016 0.0058 1 0.002 0.004 0.0013 0.008 -0.0035 -0.0004 -0.0078 0.012 -0.0061
-0.0014 -0.013 -0.0035 0.5 0.002 1 0.77 0.00054 0.14 0.00057 -0.44 -0.46 0.44 -0.00027
-0.0036 -0.015 -0.01 0.39 0.004 0.77 1 -0.009 0.11 0.0037 -0.34 -0.36 0.33 -0.0056
-0.012 -0.011 -0.0063 0.0033 0.0013 0.00054 -0.009 1 -0.0046 -0.0039 -0.00075 -0.0093 -0.00042 0.00097
0.00045 -0.015 -0.005 0.049 0.008 0.14 0.11 -0.0046 1 0.0096 -0.042 -0.045 0.043 -0.0058
-0.014 -0.038 -0.031 0.0014 -0.0035 0.00057 0.0037 -0.0039 0.0096 1 -0.0094 0.0037 0.0052 -0.62
-0.0049 0.04 0.22 -0.86 -0.0004 -0.44 -0.34 -0.00075 -0.042 -0.0094 1 0.78 -0.76 -0.018
0.0027 0.0057 -0.011 -0.91 -0.0078 -0.46 -0.36 -0.0093 -0.045 0.0037 0.78 1 -0.8 0.00023
0.0033 -0.0054 0.0016 0.87 0.012 0.44 0.33 -0.00042 0.043 0.0052 -0.76 -0.8 1 -0.0083
0.012 -0.2 -0.099 -0.0031 -0.0061 -0.00027 -0.0056 0.00097 -0.0058 -0.62 -0.018 0.00023 -0.0083 1
ser_id
_date
_date
_type
ountry
ssions
views
clicks
status
active
_date
_type
venue
urned
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75



Comment: In this correlation chart,we have notice that plan type & subscription type are highly negative correlation.

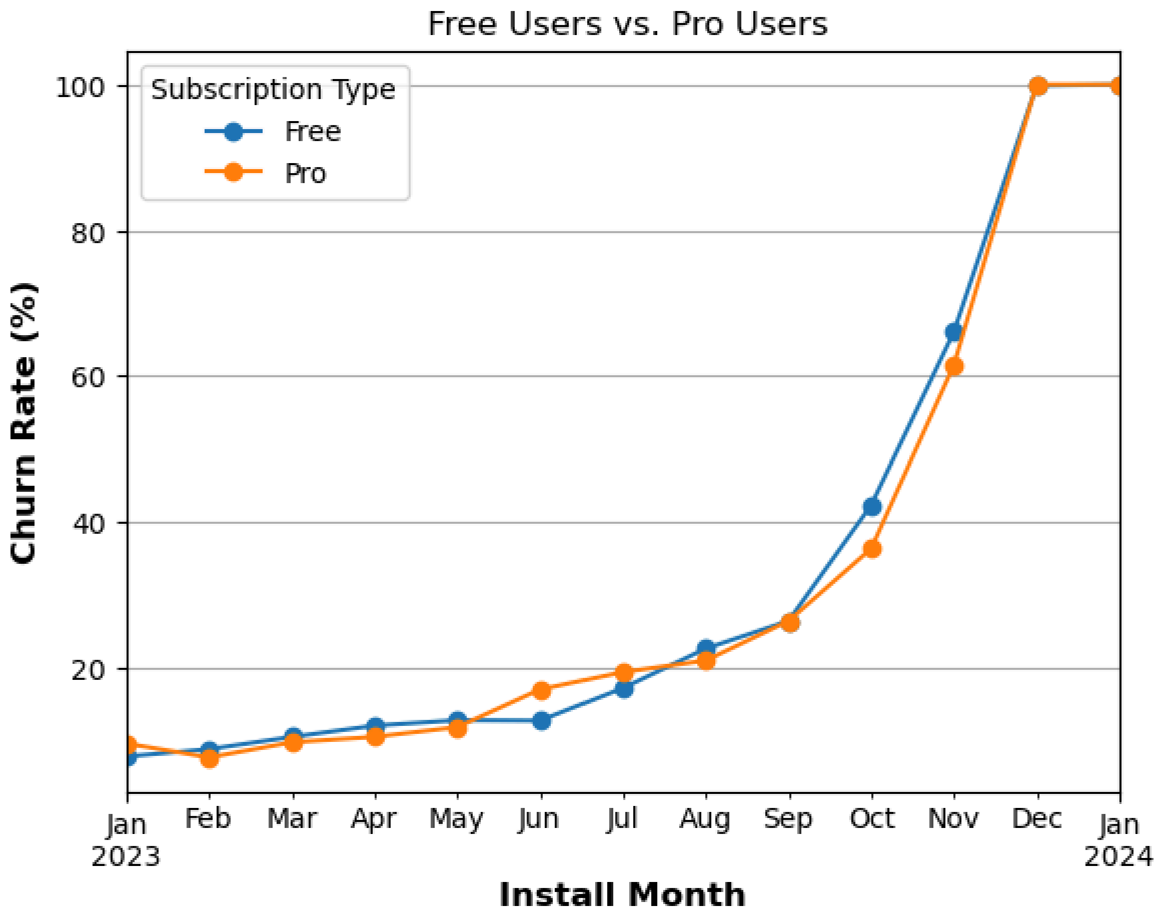
Q:(3.c) Compare churn trends between Free and Pro users.

In [22]:
```
df['install_date'] = pd.to_datetime(df['install_date'],format ='%M/%d/%Y')  ##Convert the column into datetime object
df['Month'] = df['install_date'].dt.month               ##Extracts the month from the 'order_date' column.
df['Year']  = df['install_date'].dt.year                ##Extracts the year
df['Month'] = df['Month'].astype(int)                 ##Change data type
```

In [43]:
```
df['install_date'] = pd.to_datetime(df['install_date'])
df['last_active_date'] = pd.to_datetime(df['last_active_date'])
df['month'] = df['install_date'].dt.to_period('M')
churned_trend = df.groupby(['month', 'subscription_type']).agg(total_users=('user_id', 'count'),churned_users=('churned', 'sum
).reset_index()

churned_trend['churn_rate'] = (churned_trend['churned_users'] / churned_trend['total_users']) * 100    # churn rate calculate

# to easier plotting for the pivot data
churned_trend_pivot = churned_trend.pivot(index='month', columns='subscription_type', values='churn_rate')

plt.figure(figsize=(13, 7))
churned_trend_pivot.plot(kind='line', marker='o')
plt.title(' Free Users vs. Pro Users')
plt.xlabel('Install Month',fontsize=12,fontweight='bold')
plt.ylabel('Churn Rate (%)',fontsize=12,fontweight='bold')
plt.grid(True)
```

```
plt.legend(title='Subscription Type')
plt.show()
```

<Figure size 1300x700 with 0 Axes>



In [65]:
```
df
```

Out[65]:

	user_id	install_date	last_active_date	subscription_type	country	total_sessions	page_views	download_clicks	activation_status	da
0	1	6/29/2023	7/12/2023	Free	UK	3	15	1	1	
1	2	4/10/2023	7/25/2023	Free	India	133	665	0	1	
2	3	10/25/2023	12/7/2023	Free	USA	53	106	0	1	
3	4	8/26/2023	11/9/2023	Pro	Canada	242	242	0	1	
4	5	5/14/2023	11/22/2023	Free	UK	12	48	0	1	
...	...	...	...	...	...	...	...	...	...	
19995	19996	5/6/2023	9/29/2023	Free	USA	100	300	0	1	
19996	19997	9/4/2023	9/21/2023	Pro	Germany	93	372	0	1	
19997	19998	4/1/2023	6/14/2023	Free	India	37	185	0	1	
19998	19999	1/28/2023	12/26/2023	Pro	Australia	99	198	0	1	
19999	20000	12/31/2023	1/1/2024	Free	Canada	141	282	0	1	

20000 rows × 14 columns

Q:(4.a) What percentage of users upgraded from Free to Pro?

In [73]:

```
df['pro_upgrade_date']= pd.to_datetime(df['pro_upgrade_date'])        # pro_upgrade_date convert to datetime
Qu to datetimeery5 = "select (count(case when pro_upgrade_date is not 0 then user_id end) * 100.0 / count(user_id)) as Upgrade
summary5= pd.read_sql(Query5,conn)
print(summary5)
```

```
   Upgraded_percentage
0               20.145
```

(4.b) Calculate the total monthly revenue from Pro users (Optional: Google Colab)

In [22]:
```
Query6 = "select subscription_type,sum( monthly_revenue ) as Total_revenue from df_table where subscription_type = 'Pro' group
summary6= pd.read_sql(Query6,conn)
print(summary6)
```

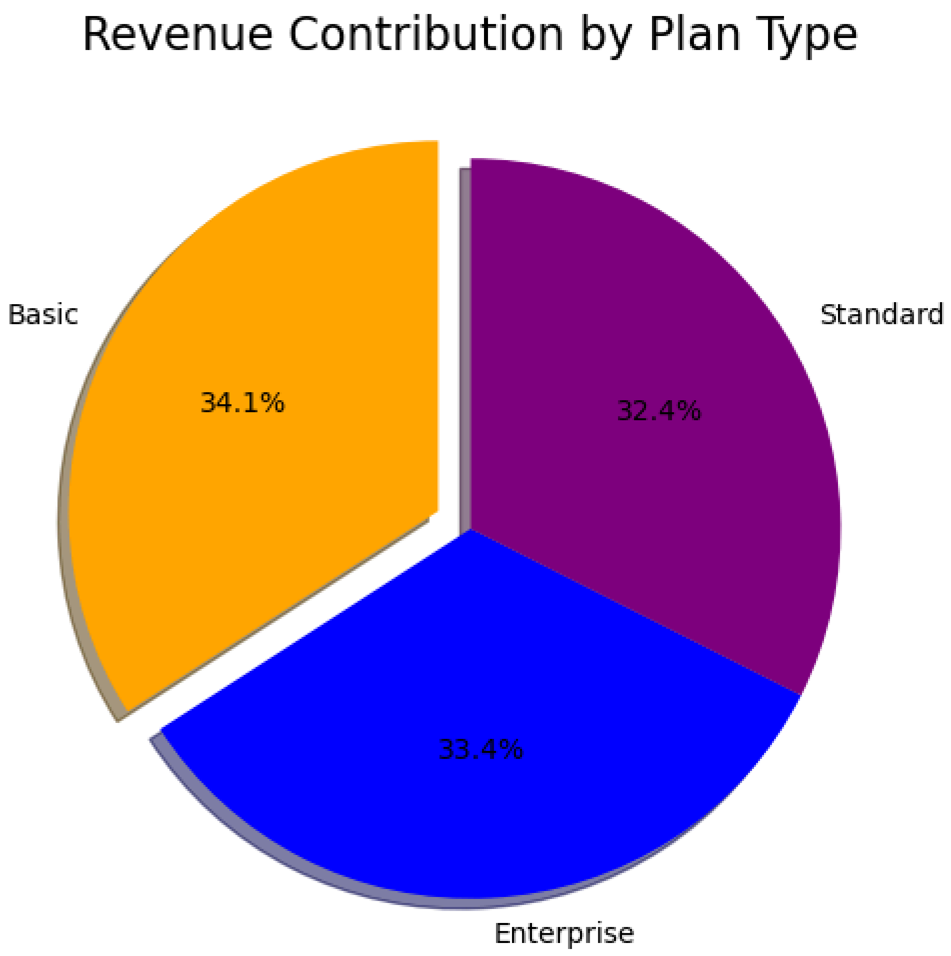
```
  subscription_type  Total_revenue
0               Pro         235481
```

Q:(4.c) Which Pro plan (Basic, Standard, or Enterprise) contributes the most revenue?

In [15]:
```
Query7 ="select plan_type,sum(monthly_revenue) as Total_revenue from df_table where plan_type is not 0 group by plan_type"
summary7= pd.read_sql(Query7,conn)
print(summary7)
```

```
    plan_type  Total_revenue
0       Basic          80339
1  Enterprise          78764
2    Standard          76378
```

In [23]:
```
Query7 ="select plan_type,sum(monthly_revenue) as Total_revenue from df_table where plan_type is not 0 group by plan_type"
summary7= pd.read_sql(Query7,conn)
labels = summary7['plan_type']
sizes = summary7['Total_revenue']
colors = ['#FFA500', '#0000FF', '#800080']
explode = (0.1, 0, 0)

# Create the pie chart
plt.figure(figsize=(8, 6))
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90, shadow=True)
plt.title('Revenue Contribution by Plan Type', fontsize=16)
plt.show()
```



(4.d) Analyze how long it takes for Free users to upgrade based on country and engagement level.

In [26]:
```
Query8 = "select country,count(pro_upgrade_date) as Upgraded_Times,sum(days_active) as Total_activated_date,sum(monthly_revenu
summary8 = pd.read_sql(Query8,conn)
print(summary8)
```

	country	Upgraded_Times	Total_activated_date	Total_revenue
0	India	2914	269647	34235
1	Canada	2899	259700	32807
2	UK	2869	253878	34032
3	France	2849	263561	32724
4	USA	2848	261495	35372
5	Germany	2832	259428	34632
6	Australia	2789	257907	31679

Q:(5.a) Suggest three strategies to reduce churn.

Step1: Users who are low engagement are more possibility to be churn.By Increasing their involvement will reduce the churn rate.we can send email or notification from our software and highligthing our offers by identifying whose total_sessions, page_views and days_active are less

Step2: By giving free trial or discount we can Incentivize to engagement themselve

Step3: Sometimes some user demonstrates the behavior to become churn.so we have to monitor by tracking the last_active_date and days_active.Meantime,we can start re-engagement campaigns and proactive support

Q:(5.b) Propose two ways to increase Free-to-Pro conversions.

Step1: We can encourage Free users to upgrade to Pro because of high risk to be churn.We can enhaunce through offer limited time discount and payment will be free for one month or 30% discounts.

Step2: we can applied these actionable activities like showcase free trial of our pro features,highlight pro benifits and send remainder before trail end.

Q:(5.c) Identify potential market expansion opportunities based on country trends.

In [32]:
```
#Identify potential market expansion opportunities based on country trend,we can analyze the dataset to find the interaction i

Query9 ="Select country,count(user_id) as Total_Users,avg(days_active) as AVG_days_active,avg(total_sessions) as AVG_Total_Ses
summary9 = pd.read_sql(Query9,conn)
print(summary9)
```

```
     country  Total_Users  AVG_days_active  AVG_Total_Session  Total_revenue  \
0      India         2914        92.535003          93.411805          34235
1     Canada         2899        89.582615          91.140738          32807
2         UK         2869        88.490066          90.328337          34032
3     France         2849        92.510004          91.082836          32724
4        USA         2848        91.817065          91.866222          35372
5    Germany         2832        91.605932          94.039195          34632
6  Australia         2789        92.472929          91.527429          31679

   Churn_Rate
0    0.002884
1    0.002891
2    0.002868
3    0.002780
4    0.002799
5    0.002791
6    0.002954
```

Comment:In India has more potential market expansion opportunist due to most number of user,less churn rate than Canada and Australia and average total session is high only less than Germany.

Q:(6.a) If WPPOOL increases the landing page conversion rate by 10%, what would be the estimated impact on Pro upgrades?

In [9]:
```
Y = df2['subscription_type']                    #target Variable
x = df2.drop(columns ='subscription_type')      #Feature Varaible
```

In [10]:
```
scaler1 = StandardScaler()
STD_scaled_df2 = scaler1.fit_transform(x)
STD_scaled_df2
```

Out[10]:
```
array([[-1.73196421e+00,  7.77606232e-01,  1.05721559e+00, ...,
         4.57030949e-01, -4.38598900e-01,  1.58293928e+00],
       [-1.73179100e+00,  9.43756846e-03,  1.18711557e+00, ...,
         4.57030949e-01, -4.38598900e-01, -6.31736172e-01],
       [-1.73161780e+00, -1.27084354e+00, -3.90241349e-01, ...,
         4.57030949e-01, -4.38598900e-01, -6.31736172e-01],
       ...,
       [ 1.73161780e+00, -4.59952841e-05,  7.97415625e-01, ...,
         4.57030949e-01, -4.38598900e-01, -6.31736172e-01],
       [ 1.73179100e+00, -1.53638332e+00, -4.83027051e-01, ...,
        -1.80354009e+00,  6.41689370e-01, -6.31736172e-01],
       [ 1.73196421e+00, -6.25961202e-01, -1.42944120e+00, ...,
         4.57030949e-01, -4.38598900e-01,  1.58293928e+00]])
```

In [11]:
```
x_train, x_test, y_train, y_test = train_test_split(x, Y, test_size=0.2, random_state=42)
```

In [12]:
```
clf = DecisionTreeClassifier(random_state=42)  #Select the DecisionTreeClassifier Train
clf.fit(x_train,y_train)
```

Out[12]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

In [13]:
```
print("train data size (feature):",len(x_train))
print("train data size (feature):",len(y_train))
print("train data size (feature):",len(x_test))
print("train data size (feature):",len(y_test))
```

```
train data size (feature): 16000
train data size (feature): 16000
train data size (feature): 4000
train data size (feature): 4000
```

In [16]:
```
y_pred = clf.predict(x_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      3162
           1       1.00      1.00      1.00       838

    accuracy                           1.00      4000
   macro avg       1.00      1.00      1.00      4000
weighted avg       1.00      1.00      1.00      4000
```

In [28]:
```
x_new = x.copy()
# increase total_sesion 10% due to new user
x_new['total_sessions'] = x_new['total_sessions'] * 1.10
x_new_data = scaler1.transform(x_new)
```

New data prediction for pro upgration

y_new_prediction = clf.predict(x_new_data) pro_new_upgrades = y_new_pred.sum()

In [33]:
```
# Compair the current pro upgrades and new pro ugrades
current_pro_upgrades = Y.sum()
impact = pro_new_upgrades - current_pro_upgrades
print(f"Estimated increase in Pro upgrades: {impact}")
```

```
Estimated increase in Pro upgrades: -4029
```

In [ ]:
```
Decission: If we we increases the landing page conversion rate by 10% which would be negatively impact for pro upgrades
```

Q(6.b) Run a simple A/B test simulation (e.g., using a chi-square test) to evaluate conversion optimization.

In [41]:
```
#Created a contingency Table
contingency_table = pd.crosstab(df['subscription_type'],df['activation_status'])
```

In [42]:
```
#Applied chi-square test
chi2,p,dof,expected = stats.chi2_contingency(contingency_table)
```

In [44]:
```
#show the result of test

print(chi2)
print("P-value:",p)
print(dof)
print(expected)
```

```
46.877929944976046
P-value: 7.554871292626058e-12
1
[[  150.92595 15820.07405]
 [   38.07405  3990.92595]]
```

In [47]:
```
if p < 0.05:
    print("There is a significant relationship between Plan Type and Activation Status (conversion).")
else:
    print("No significant relationship found.")
```

```
There is a significant relationship between Plan Type and Activation Status (Conversion).
```

Q(6.c) Suggest three A/B test ideas that could help improve the conversion rate, and explain how you would measure their success.

1) Logistic regression Test: We can use the logistic regression to identify the user characteristics to predict the probability of upgrading Free to Pro.Example:By user behaviors factor(session time, page views, download clicks) we can measures our result.

2)ANOVA Test: If we test the three or more variable we can applied ANOVA test to determine the statisticle difference

3)Survival Analysis (Kaplan-Meier Estimation): It helps to find the median time to conversion(How long does it take for free users to upgrade?) and hazard ratio(Measures the effect of engagement level on conversion)

Q(7.a) Identify 3 key performance indicators (KPIs) WPPOOL should track.

1)Conversion Rate: As my answer of question 4.a,we notice that only 20.145 % user upgradred Free to Pro subscription_type.If we evaluate the success of marketing efforts, pricing strategies and user experience optimizations that will be the key performance indicator

2)Customer Retention Rate: It's most essential part to hold on the current merket possition.we have to monitor the user retention over different periods of time.

3)Customer Acquisition Cost:Measures how much WPPOOL spends to acquire a new Pro user which helps to optimize marketing and sales spend.

Q(7.b): Suggest 2 actionable growth strategies WPPOOL can implement based on your analysis.

Stratigies No 1: Conversion Percentage from Free to Pro is very low(only 20.145%) .If we give free users limited access to Pro features for a short time and Use in-app messages & emails to show how much value users are missing by not upgrading

Stratigies No 2:If we Launced new plan_type stratigies or student service with proper information and and increase the country base merketing stratigies which would be help us reduce to be the possibility of chrned.If we notice that aproximates 89.955% user are getting the free service which is one of our lucrative example for increasing the revenue by diverting themselve.

Q(7.c): How would you measure the success of these strategies?

1) Measuring Success of Fre to Pro Conversion optimization.

2)Measuring Success of Customer Retention

3)By improving the pricing & monetization strategies

4) By increasing the customer engagement merkinting strategies

Introducing new products according to customer demand

Section 3:

A: Al-Quran,Mosnovi-Sharif & Paradise lost are three most favourite books.

B:I enjoy to show drama,thriller & adventure movie and News.

You can be a master , Don't wait for your Luck