Analysis & Code (from Jupyter Notebook)

In [1]:

```
import pandas as pd
import numpy as np

print("Libraries imported successfully")
```

```
Libraries imported successfully
```

In [2]:

```
users = pd.read_csv("users.csv")
campaigns = pd.read_csv("campaigns.csv")
usage = pd.read_csv("usage_metrics.csv")
```

Check datatypes and missing values for each dataset

In [3]:

```
print("=== USERS ===")
print(users.info())
```

```
=== USERS ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250 entries, 0 to 249
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   user_id        250 non-null    object
 1   signup_date    250 non-null    object
 2   country        244 non-null    object
 3   device         241 non-null    object
 4   referral_code  232 non-null    object
dtypes: object(5)
memory usage: 9.9+ KB
None
```

In [4]:

```
print("\n=== CAMPAIGNS ===")
print(campaigns.info())
```

```
=== CAMPAIGNS ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 430 entries, 0 to 429
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   campaign_id      430 non-null    object 
 1   user_id          430 non-null    object 
 2   experience_type  415 non-null    object 
 3   status           430 non-null    object 
 4   credits_used     177 non-null    float64
 5   created_date     430 non-null    object 
 6   published_date   187 non-null    object 
dtypes: float64(1), object(6)
memory usage: 23.6+ KB
None
```

In [5]:

```
print("\n=== USAGE METRICS ===")
print(usage.info())
```

```
=== USAGE METRICS ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2031 entries, 0 to 2030
Data columns (total 6 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   user_id           2031 non-null   object 
 1   week_start_date   2031 non-null   object 
 2   sessions          1963 non-null   float64
 3   avg_session_time  1951 non-null   float64
 4   engagement_score  1850 non-null   float64
 5   last_active_date  1671 non-null   object 
dtypes: float64(3), object(3)
memory usage: 95.3+ KB
None
```

Data cleaning and Preparation

Convert date columns

In [6]:

```
users['signup_date'] = pd.to_datetime(users['signup_date'])
campaigns['created_date'] = pd.to_datetime(campaigns['created_date'])
campaigns['published_date'] = pd.to_datetime(campaigns['published_date'])
usage['week_start_date'] = pd.to_datetime(usage['week_start_date'])
usage['last_active_date'] = pd.to_datetime(usage['last_active_date'])
```

Fill missing categorical values

In [7]:

```
users['country'] = users['country'].fillna('Unknown')
users['device'] = users['device'].fillna('Unknown')
users['referral_code'] = users['referral_code'].fillna('No')
campaigns['experience_type'] = campaigns['experience_type'].fillna('Unknown')
```

Fill missing numeric values

In [8]:

```
campaigns['credits_used'] = campaigns['credits_used'].fillna(0)
usage['sessions'] = usage['sessions'].fillna(usage['sessions'].mean())
usage['avg_session_time'] = usage['avg_session_time'].fillna(usage['avg_session_time'].mean())
usage['engagement_score'] = usage['engagement_score'].fillna(usage['engagement_score'].mean())
```

DATA UNDERSTANDING - USERS

In [9]:

```
# Total users
print("Total users:", users['user_id'].nunique())

# Users by country
print("\nUsers by Country:")
display(users['country'].value_counts())

# Users by device
print("\nUsers by Device:")
display(users['device'].value_counts())
```

```
Total users: 250

Users by Country:
India      106
USA         37
UAE         32
Germany     28
UK          25
Brazil      16
Unknown      6
Name: country, dtype: int64
Users by Device:
Android    152
iOS         57
Web         32
Unknown      9
Name: device, dtype: int64
```

DATA UNDERSTANDING - Campigns

In [10]:

```
# Total campaigns
print("Total campaigns:", campaigns['campaign_id'].nunique())

# Campaign status
print("\nCampaign Status Counts:")
display(campaigns['status'].value_counts())

# Experience types
print("\nExperience Type Counts (Top 5):")
display(campaigns['experience_type'].value_counts().head(5))

# Average credits used
print("\nAverage Credits Used:", round(campaigns['credits_used'].mean(),2))
```

```
Total campaigns: 430

Campaign Status Counts:
draft        207
published    187
deleted       36
Name: status, dtype: int64
```

```
Experience Type Counts (Top 5):
3D Spatial          51
Prism Extended      50
Prism Interactive   45
Spatial Alpha       43
Prism               42
Name: experience_type, dtype: int64
Average Credits Used: 1.25
```

DATA UNDERSTANDING - Usage metrics

In [11]:

```
# Total records
print("Total records:", len(usage))

# Unique users in usage data
print("Unique users in usage data:", usage['user_id'].nunique())

# Average sessions and engagement
print("Average sessions per week:", round(usage['sessions'].mean(),2))
print("Average engagement score:", round(usage['engagement_score'].mean(),2))
```

```
Total records: 2031
Unique users in usage data: 250
Average sessions per week: 3.55
Average engagement score: 28.16
```

Merge data sets

In [12]:

```
# Merge usage with users for device/country/referral info
usage_users = usage.merge(users, on='user_id', how='left')
print("Merged Usage + Users Sample:")
display(usage_users.head(2))

# Merge campaigns with users
campaigns_users = campaigns.merge(users, on='user_id', how='left')
print("Merged Campaigns + Users Sample:")
display(campaigns_users.head(2))

# Merge campaigns + usage for engagement analysis
campaign_usage = campaigns.merge(usage, on='user_id', how='left')

print("Merged Campaigns + Usage Sample:")
display(campaign_usage.head(2))
```

```
Merged Usage + Users Sample:
```

	user_id	week_start_date	sessions	avg_session_time	engagement_score	last_active_date	signup_date	country	device	referral_code
0	U0001	2025-09-08	6.0	11.62	69.72	2025-09-12	2025-09-12	Germany	iOS	No
1	U0001	2025-09-15	5.0	4.88	24.40	2025-09-21	2025-09-12	Germany	iOS	No

```
Merged Campaigns + Users Sample:
```

	campaign_id	user_id	experience_type	status	credits_used	created_date	published_date	signup_date	country	device	referral_code
0	C00001	U0002	Unknown	draft	0.0	2025-09-17	NaT	2025-08-23	Germany	iOS	No
1	C00002	U0002	Unknown	published	4.0	2025-09-07	2025-09-12	2025-08-23	Germany	iOS	No

```
Merged Campaigns + Usage Sample:
```

	campaign_id	user_id	experience_type	status	credits_used	created_date	published_date	week_start_date	sessions	avg_session_time	engage
0	C00001	U0002	Unknown	draft	0.0	2025-09-17	NaT	2025-08-18	0.0	6.81	
1	C00001	U0002	Unknown	draft	0.0	2025-09-17	NaT	2025-08-25	3.0	2.53	

Handle Missing Values After Merge

In [13]:

```
# Check missing values in merged datasets
print("=== Missing in usage_users ===")
display(usage_users.isna().sum())

print("\n=== Missing in campaigns_users ===")
display(campaigns_users.isna().sum())
```

```
print("\n=== Missing in campaign_usage ===")
display(campaign_usage.isna().sum())
```

```
=== Missing in usage_users ===
user_id               0
week_start_date       0
sessions              0
avg_session_time      0
engagement_score      0
last_active_date    360
signup_date           0
country               0
device                0
referral_code         0
dtype: int64
=== Missing in campaigns_users ===
campaign_id          0
user_id              0
experience_type      0
status               0
credits_used         0
created_date         0
published_date     243
signup_date          0
country              0
device               0
referral_code        0
dtype: int64
=== Missing in campaign_usage ===
campaign_id            0
user_id                0
experience_type        0
status                 0
credits_used           0
created_date           0
published_date      1993
week_start_date        0
sessions               0
avg_session_time       0
engagement_score       0
last_active_date     646
dtype: int64
```

In [14]:

```
# For usage users
usage_users['last_active_date'] = usage_users['last_active_date'].fillna(usage_users['signup_date'])
```

In [15]:

```
# for campaigns_users
campaigns_users['published_status'] = campaigns_users['published_date'].apply(
    lambda x: 'Published' if pd.notnull(x) else 'Not Published'
)
```

In [16]:

```
# for campaign usage
campaign_usage = campaign_usage.merge(users[['user_id','signup_date']], on='user_id', how='left')

campaign_usage['published_status'] = campaign_usage['published_date'].apply(
    lambda x: 'Published' if pd.notnull(x) else 'Not Published'
)
campaign_usage['last_active_date'] = campaign_usage['last_active_date'].fillna(campaign_usage['signup_date'])
```

EXPORT CLEANED & MERGED DATA FOR POWER BI

In [17]:

```
users.to_csv("users_clean.csv", index=False)
campaigns_users.to_csv("campaigns_users_clean.csv", index=False)
usage_users.to_csv("usage_users_clean.csv", index=False)
campaign_usage.to_csv("campaign_usage_clean.csv", index=False)

print(" Cleaned and merged datasets exported for Power BI visualization")
```

```
 Cleaned and merged datasets exported for Power BI visualization
```

Visualizations (from Power BI)