BI-DBS: HW0

Jonathan Mahesh Najare
Michael A. Ay
Tereza Kudláčková

October 12, 2025

Contents

1 Domain and Participant Description

1.1 Platform Type

Our project is a social media application designed for the hiking and outdoor community. It is meant to accompany the rest of our business, which is creating hiking routes. The application's main purpose is to let Users share their experiences, which is more than just tracking physical activities.

As shown in the ER diagram, the main feature is the Post. It allows a User to record a completed Activity. Each post can include Photos and be shared with friends or in specific Groups. The platform is built to help users interact through social features like friending, commenting on posts, and tagging other users. The goal is to provide a space where hikers can record their achievements, share stories, find new trails, and connect with other people who have similar interests.

1.2 Key User Roles

The platform has two main types of users:

- **Hiker:** This is the primary user of the platform. A User is the central entity that creates content (like a Post or Photo), joins Groups, and interacts with others through Comments, Tags, and friendships. They perform and log an Activity, which is a core feature of the app.
- **Advertiser:** This is a commercial user role. The main function of this classification of user to be able to flag posts that are advertisements. The platform's business model relies on analyzing how hikers interact with this content to improve ad targeting.

1.3 Object Types

The platform's objects and their relationships are defined by the ER diagram. The following list describes each entity shown in the diagram.

- **User:** The central entity in the model. A user can be connected with other users (*has_friend*), join and create groups (*joins*, *creates*), produce content (*writes*, *shares*) and tag someone (*create* **Tag**).
- **Group:** A community space that a **User** can *create* (first **User** joining also creates the **Group**) or *join*. Also **Post** can be made only inside a **Group** (Main page is also a **Group**, which includes every **User**).
- **Post:** The main content object. A **Post** is *shared* by a **User** and is used to share an **Activity** (*includes*). It can contain a **Photo** (*is_part*) and receive **Comments** and **User** can also be tagged (*contains* a **Tag**).
- **Activity:** A predetermined hike or route that serves as a template experience. A **Post** is linked to it via the *includes* relationship to show a user has completed that specific hike.

- **Photo:** An image entity that is included as part of a **Post**.
- **Comment:** A text response that is attached to a specific **Post**.
- **Tag:** An entity used to link a **User** to a **Post** through the *was_tagged* relationship.

1.4 Events and Actions

Users on the platform perform a range of actions that generate data. These events are centered around completing activities, creating content, and interacting within the community.

1.4.1 Account Management

- **User Registration:** A new user creates an account.

1.4.2 Content and Activity

- **Post Creation:** A user creates a new **Post** to document their completion of a predefined **Activity**.
- **Photo Addition:** A user adds a **Photo** to their **Post**.

1.4.3 Social Interaction

- **Comment Addition:** A user writes a **Comment** on a **Post**.
- **Post Liked:** A user gives a "like" to a **Post**.
- **Post Shared:** A user shares a **Post**.
- **User Tagged:** A user is mentioned in a **Post** via a **Tag**.
- **Friend Added:** A user establishes a friendship with another user.

1.4.4 Group Interaction

- **Group Creation:** A user creates a new **Group**.
- **Group Joined:** A user becomes a member of a **Group**.

2 ER Diagram

The following ER diagram visually represents the key entities and the relationships between them in our platform's data model.

This is a test edit.

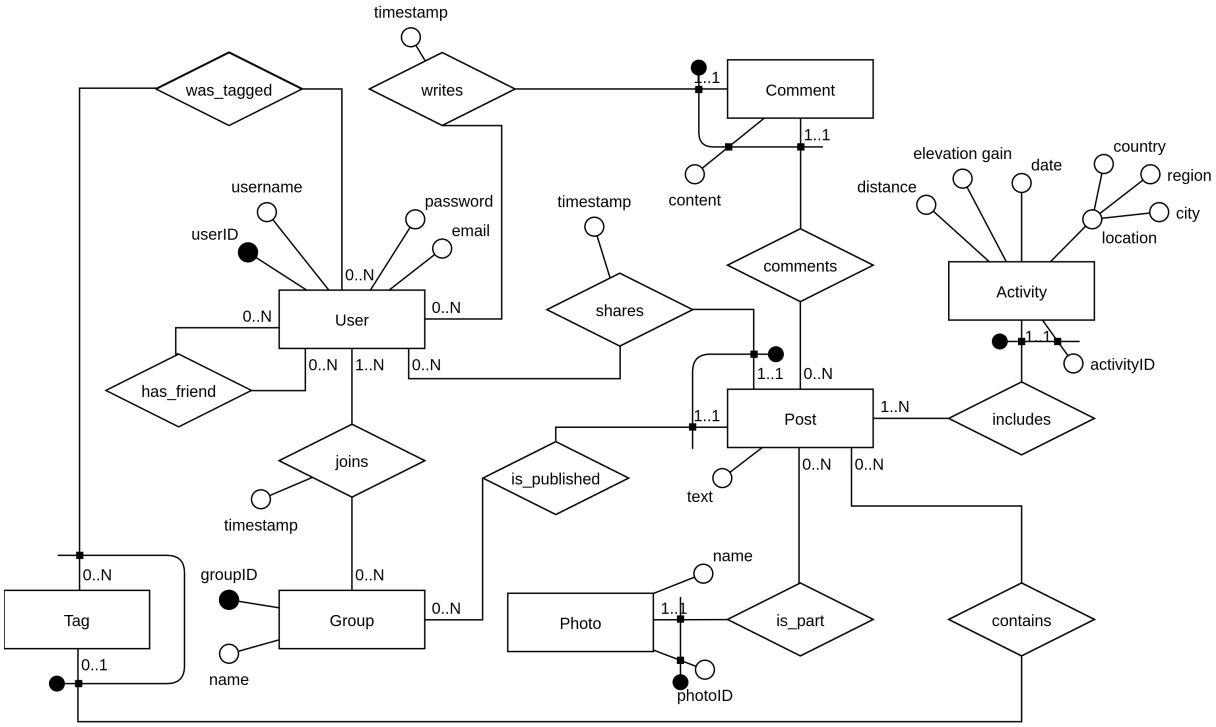

Figure 1: Entity-Relationship diagram of the platform.

3 Analytical Tasks

Task 1: Find Inactive Users for Removal

Business Goal: To identify and manage dormant accounts to maintain a clean and active user base, which can also help reduce data storage costs.

Input: A target date. The query will scan all user events (Post Creation, Comment Addition, Group Joined) from the beginning of time up to the target date.

Output: A list of users with their user_id, username, and days_inactive.

Granularity: By user.

Sorting / Top-N: Sorted by days_inactive in descending order. The full list of inactive users is returned (no Top-N).

Execution Frequency: Annually.

Acceptance Criterion: The output list contains only users whose most recent event timestamp is more than 365 days before the target date. A manual check of several users from the list against the raw event logs confirms their inactivity. The expected result size is 5-15% of the total user base.

Task 2: Top 10 Most Active Groups

Business Goal: To identify highly engaged communities for targeted advertising or promotional campaigns.

Input: All Post Creation events from the last 7 days, along with the current membership count for each group.

Output: A list of the top 10 groups with their group_id, group_name, and a calculated activity score (average posts per member).

Granularity: By group.

Sorting / Top-N: Sorted by the activity score in descending order, returning the top 10.

Execution Frequency: Weekly.

Acceptance Criterion: The list contains exactly 10 groups. The activity score is correctly calculated as the total number of posts in the group over the last 7 days divided by the total number of members in that group. A manual calculation for 2-3 groups on the list confirms the metric is accurate.

Task 3: Top 10 Groups by New Member Growth

Business Goal: To identify fast-growing or trending groups to recommend to users, increasing community engagement.

Input: All Group Joined events from the last 7 days.

Output: A list of the top 10 groups with their group_id, group_name, and new_member_count.

Granularity: By group.

Sorting / Top-N: Sorted by new_member_count in descending order, returning the top 10.

Execution Frequency: Weekly.

Acceptance Criterion: The list contains exactly 10 groups. The new_member_count for each group correctly matches the number of unique users who joined that group in the last 7 days. The counts for a few groups on the list are manually verified against the raw event logs.

Task 4: Top 20 Users by Monthly Distance

Business Goal: To create a leaderboard recognizing the most active users, fostering a sense of competition and motivating the community.

Input: All Post Creation events from the previous calendar month. Each post contains distance and step data from the user's hike.

Output: A list of the top 20 users with their user_id, username, total_distance, and total_steps for the month.

Granularity: By user.

Sorting / Top-N: Sorted by total_distance in descending order, returning the top 20.

Execution Frequency: Monthly, on the first day of each month.

Acceptance Criterion: The list contains exactly 20 users. The total_distance and total_steps correctly sum the metrics from all Posts created by each user during the previous calendar month. The aggregated values for a few users on the list are manually verified against their post history.

Task 5: Monthly Post Count by Region

Business Goal: To analyze the seasonal popularity of hiking in different regions to inform targeted advertising and content promotion.

Input: A specific Region and Year. The query uses all Post Creation events, which are linked to Activities that have a location.

Output: A list of 12 rows, each containing the month and the total post_count for that month.

Granularity: By month, for the specified region.

Sorting / Top-N: Sorted chronologically by month.

Execution Frequency: On-demand, allowing for analysis of any region at any time.

Acceptance Criterion: The output contains exactly 12 rows. The post_count for each month correctly matches the number of Posts created in that month linked to an Activity in the specified Region. The sum of all monthly counts equals the total post count for that region in that year.

Task 6: Top 10 Most Tagged Users (Influencers)

Business Goal: To identify influencers or popular users who appear often in others' content, which is useful for influencer partnerships or engagement campaigns.

Input: All User Tagged events from the last 90 days.

Output: A list of the top 10 users with their user_id, username, and total tag_count.

Granularity: By user.

Sorting / Top-N: Sorted by total tag_count in descending order, returning the top 10.

Execution Frequency: Monthly.

Acceptance Criterion: The list contains exactly 10 users. The tag_count for each user correctly matches the number of times they were tagged in the last 90 days. The counts are verified for a few users by checking the raw event logs.

Task 7: Top 10 Groups by Tag Interactions

Business Goal: To identify the most socially interactive groups where members frequently tag each other, for targeted community engagement or promotions.

Input: All User Tagged events that occurred in posts made within a group context over the last 30 days.

Output: A list of the top 10 groups with their group_id, group_name, and tag_interaction_count.

Granularity: By group.

Sorting / Top-N: Sorted by tag_interaction_count in descending order, returning the top 10.

Execution Frequency: Monthly.

Acceptance Criterion: The list contains exactly 10 groups. The tag_interaction_count correctly sums all tags within posts created in each group over the last 30 days. The counts for a few groups on the list are manually verified against the raw event logs.

Task 8: User Engagement with Advertisement Posts

Business Goal: To understand which types of users are most engaged with advertisements (based on activity level, geography, etc.) to improve ad targeting and monetization.

Input: All ad interaction events (Post Liked, Comment Added, Post Shared, Ad Clicked) on advertisement Posts from the last 90 days. User profile data is also required.

Output: A list of all users who interacted with an ad, with their user_id, country, account_age_days, total_posts_created, and ad_interaction_count.

Granularity: By user.

Sorting / Top-N: Sorted by ad_interaction_count in descending order. The full list of interacting users is returned.

Execution Frequency: Monthly.

Acceptance Criterion: The output contains one row for every user with at least one ad interaction in the period. The ad_interaction_count for each user must match the sum of their ad-related events in the raw logs. The expected result size is in the thousands of users.

Task 9: Find User Community Clusters (Graph Analysis)

Business Goal: To identify natural user communities and their key influencers, which can be used to strengthen network effects and design group-based engagement strategies.

Input: The user interaction graph built from events in the last 180 days (e.g., Post Liked, Comment Added, User Tagged) and existing social connections (friendships).

Output: A list of all users, each assigned a cluster_id. The output includes metrics like cluster_size, the user's centrality_score within the cluster, and the overall modularity_score of the clustering.

Granularity: By user, within a cluster.

Sorting / Top-N: The output is typically sorted by cluster_size in descending order, then by user_id. The full list of users is returned.

Execution Frequency: Quarterly.

Acceptance Criterion: Every active user is assigned to exactly one cluster. A manual review of a few clusters shows that the members have plausible connections (e.g., they interact frequently or are part of the same groups). The expected result is thousands of clusters, with sizes ranging from 10 to 500 members each.

Task 10: Identify Influential Users by Activity Engagement

Business Goal: To identify influential users whose Posts motivate others to complete the same predefined Activities, in order to highlight and reward community influencers.

Input: All Post Creation events from the last 12 months. The query looks for patterns where one user posts about an Activity, and other users subsequently post about the same Activity within a defined time window (e.g., 30 days).

Output: A list of influential users with their user_id, username, total_posts_created, inspired_hike_count, and a calculated influence_score.

Granularity: By user (the potential influencer).

Sorting / Top-N: Sorted by influence_score in descending order, returning the top 50 users.

Execution Frequency: Monthly.

Acceptance Criterion: The list contains up to 50 users. The inspired_hike_count correctly counts Posts from other users about the same Activity within the 30-day window following the influencer's Post. The logic is manually verified for a few users on the list.

Task 11: Daily Post Count for Top 10 Activities

Business Goal: To monitor demand for the most popular predefined Activities and detect daily or weekly patterns in their usage.

Input: All Post Creation events from the last 30 days.

Output: A daily time series for each of the top 10 Activities, containing the activity_id, date, daily_post_count, and a list of post_timestamps for that day.

Granularity: By Activity, by day.

Sorting / Top-N: The query first identifies the top 10 Activities based on total posts in the last 30 days. The final output is sorted by activity_id, then chronologically by date.

Execution Frequency: Daily.

Acceptance Criterion: The list of top 10 Activities is correctly identified. For those Activities, the daily_post_count is accurate. The sum of all daily counts in the output matches the total number of Posts for those top 10 Activities in the raw event logs.

Task 12: User Profile for a Specific Activity

Business Goal: To understand the characteristics (e.g., overall activity level, country) of users who complete a specific predefined Activity, in order to better target recommendations and assess the Activity's audience.

Input: A specific activity_id. The query uses all Post Creation events linked to this Activity and the corresponding user profile data.

Output: A list of all users who completed the Activity, with their user_id, username, country, registration_date, and total_posts_created (as a measure of their overall activity).

Granularity: By user, for the specified Activity.

Sorting / Top-N: The full list is returned, sorted by total_posts_created in descending order to show the most active users first.

Execution Frequency: On-demand.

Acceptance Criterion: The output contains a complete list of all unique users who have created a Post for the given activity_id. All profile and activity metrics for each user are accurate. A spot check confirms that a few users on the list have posted about that Activity.

Task 13: Weekly User Retention Cohort Analysis

Business Goal: To track how well the platform retains new users over time. This helps measure the "stickiness" of the app and the impact of product changes or marketing campaigns on long-term user engagement.

Input: All User Registration events to define the weekly cohorts. All activity events (e.g., Post Creation, Comment Addition) for all users.

Output: A cohort matrix where each row is a weekly registration cohort and each column is a week since registration (Week 0, Week 1, etc.). The values are the percentage of users in a cohort who were active in that subsequent week.

Granularity: By weekly user cohort.

Sorting / Top-N: The output matrix is sorted chronologically by the cohort's registration week.

Execution Frequency: Monthly.

Acceptance Criterion: The Week 0 column for each cohort shows 100%. The percentages in subsequent columns correctly represent the portion of the original cohort that had at least one activity event in that week. The percentages are expected to decrease over time for each cohort. A manual calculation for one cohort's first few weeks confirms the data is correct.

Task 14: Impact of Weather and Season on Activity Popularity

Business Goal: To identify which predefined Activities are popular under various weather conditions and seasons, in order to provide better recommendations and safety tips.

Input: All Post Creation events over the last 2 years, joined with an external historical weather dataset based on the date and the location of the associated Activity.

Output: A list containing activity_id, season, weather_condition (e.g., Sunny, Rain), and the total post_count for that combination.

Granularity: By Activity, by season, and by weather condition.

Sorting / Top-N: The full list is returned, typically unsorted, for analytical export.

Execution Frequency: Quarterly.

Acceptance Criterion: The external weather data is correctly joined to each Post based on its date and location. The post_count for each combination is accurate. The analysis only includes combinations with a statistically significant number of posts (e.g., more than 30) to ensure valid conclusions.

Task 15: New User Onboarding Analysis (First 7 Days)

Business Goal: To understand how new users engage with key features during their first week, in order to evaluate and improve the onboarding process and long-term retention.

Input: All users who registered in the last 30 days, and all of their subsequent events.

Output: A list of these new users with their user_id, registration_date, country, and boolean flags indicating key actions: created_first_post_in_7d, joined_group_in_7d.

Granularity: By user.

Sorting / Top-N: Sorted by registration_date in descending order. The full list of new users from the period is returned.

Execution Frequency: Daily.

Acceptance Criterion: The output contains one row for every user who registered in the last 30 days. The boolean flags are correctly calculated based on the user's event history within 7 days of their registration. A manual check for a few users confirms the flags are accurate.

Task 16: Top 5 Busiest Time Slots by User Interaction

Business Goal: To identify the peak hours of user activity. This helps content creators maximize post visibility and allows the platform to schedule maintenance during off-peak hours.

Input: All Post Liked and Comment Added events from the last 90 days.

Output: A list of the top 5 time slots, each with its day_of_week, hour_of_day, and total_interaction_count.

Granularity: By day of the week and hour of the day.

Sorting / Top-N: Sorted by total_interaction_count in descending order, returning the top 5.

Execution Frequency: Quarterly.

Acceptance Criterion: The output contains exactly 5 time slots. The total_interaction_count for each slot is the correct sum of all likes and comments that occurred during that hour over the last 90 days. The sum of interactions across all 168 possible time slots (7 days x 24 hours) equals the total number of like and comment events in the period.

Task 17: Top 20 Group-Activity Combinations by Member Engagement

Business Goal: To measure the influence of Groups on discovering and completing new Activities, by identifying which Groups are most effective at encouraging members to participate in specific hikes.

Input: All Group Joined events and all Post Creation events that are associated with a Group.

Output: A list of the top 20 combinations with their group_id, activity_id, and influenced_post_count.

Granularity: By Group and Activity combination.

Sorting / Top-N: Sorted by influenced_post_count in descending order, returning the top 20.

Execution Frequency: Quarterly.

Acceptance Criterion: The list contains 20 Group-Activity pairs. The influenced_post_count for a pair only includes Posts created by a user after they joined that specific Group. This attribution logic is manually verified for a few pairs by tracing individual user event histories.

Task 18: New Group Performance in First 30 Days

Business Goal: To measure the initial engagement and success of newly created Groups, in order to identify which new communities are gaining traction and which might need promotion.

Input: All Group Creation events to identify new groups. All Post Creation and Group Joined events to measure their activity.

Output: A list of recently created groups with their group_id, group_name, creator_user_id, age_in_days, member_count_in_first_30d, and post_count_in_first_30d.

Granularity: By Group.

Sorting / Top-N: The full list of new groups is returned, sorted by post_count_in_first_30d in descending order.

Execution Frequency: Monthly.

Acceptance Criterion: The output lists all Groups created in a recent period (e.g., last 60 days). The member and post counts are correctly calculated for events occurring within the first 30 days of each group's creation date. The logic is manually verified for a few new groups.

Task 19: Top 10 Regions by User-to-Activity Ratio

Business Goal: To identify regions with a high number of active users but a low number of available Activities, in order to prioritize adding new predefined routes in those areas.

Input: The list of all active users (e.g., users with at least one Post in the last 90 days) and their regions. The list of all predefined Activities and their regions.

Output: A list of the top 10 regions, with their region_name, active_user_count, activity_count, and the calculated user_to_activity_ratio.

Granularity: By region.

Sorting / Top-N: Sorted by user_to_activity_ratio in descending order, returning the top 10.

Execution Frequency: Quarterly.

Acceptance Criterion: The list contains 10 regions. The active_user_count and activity_count for each region are correct. The ratio is calculated correctly. The counts for a few regions on the list are manually verified against the source user and activity data.

Task 20: New User Engagement Funnel (First 90 Days)

Business Goal: To track the new user journey from registration to key engagement actions (joining a group, creating a post), in order to identify potential drop-off points in the onboarding process.

Input: All users registered in the last 90 days, and their subsequent Group Joined and Post Creation events.

Output: A list of funnel steps: 'Registered', 'Joined First Group', 'Created First Post'. Each step will have a user_count and a conversion_rate from the previous step.

Granularity: By funnel step.

Sorting / Top-N: The order of the steps is fixed as it represents a user journey.

Execution Frequency: Monthly.

Acceptance Criterion: The user_count for the 'Registered' step matches the total number of new users in the period. The counts for subsequent steps are less than or equal to the previous step. A manual check of several user journeys confirms they are correctly categorized in the funnel.

Task 21: Identify Inactive (Dead) Groups

Business Goal: To identify inactive groups that have had no recent engagement, so they can be considered for archival or removal to improve the user experience and platform health.

Input: All Post Creation and Comment Added events that occurred in a group context.

Output: A list of inactive groups with their group_id, group_name, member_count, and days_since_last_activity.

Granularity: By group.

Sorting / Top-N: The full list is returned, sorted by days_since_last_activity in descending order.

Execution Frequency: Monthly.

Acceptance Criterion: The output lists only groups whose most recent Post or Comment is older than 60 days. The days_since_last_activity is correctly calculated from the timestamp of the last known event in the group. A manual check of a few groups on the list confirms their inactivity in the event logs.

4 Data Structure

This section outlines the structure of the data entities for the platform, as defined by the data generation scripts.

4.1 Users

- **Fields:** The following fields are generated for each user:
  - user_id
  - username
  - is_commercial (a flag indicating an Advertiser account)
  - email
  - password (a randomly generated string for synthetic purposes)
  - join_date
  - country
- **Identifiers:** The user_id is the unique primary identifier for each user. The username is also expected to be unique.
- **Cardinality and Distribution:** The script generates users (tested up to 30k users). This is split into 85% regular (Hiker) users and 15% commercial (Advertiser) users. User registrations show a seasonal bias towards summer months and are distributed unevenly across a predefined list of countries.

4.2 Groups

- **Fields:** The following fields are generated for each group:
  - group_id
  - group_name
  - group_country
  - creator_id (the user who created the group)
  - creation_date
- **Identifiers:** The group_id is the unique primary identifier.
- **Cardinality and Distribution:** The script generates 3,000 groups (10% of the user count). Groups are assigned a country based on the user country distribution and are categorized as 'popular', 'big', 'small', or 'mostly_dead' to create a skewed distribution of member activity.

4.3 Posts

- **Fields:** The following fields are generated for each post:
  - post_id
  - author_user_id
  - created_at (timestamp of the post)
  - text (randomly generated string content)
  - activity_id (optional link to a predefined Activity)

  - photo_id (optional link to a Photo)

- **Identifiers:** The post_id is the unique primary identifier.
- **Cardinality and Distribution:** The script generates posts (tested up to 75k). The number of posts per user follows a Zipf-like (heavy-tail) distribution, meaning a small number of users create a large percentage of the posts. Post creation times are biased towards evenings and weekends.

4.4 Other Objects

- **Comments:** Generated with fields: comment_id, post_id, author_user_id, written_at, and content. The distribution of comments is heavily skewed, with popular posts and active users receiving disproportionately more comments.
- **Photos:** Generated with fields: photo_id and raw image_data. About half of all posts have an associated photo.

4.5 Relationships

These entities represent the direct connections between other objects.

- **Friendship:** A bidirectional link between two users, with fields user_id_1 and user_id_2. The probability of friendship is higher between users from the same country, creating realistic social clusters.
- **Group Membership:** A record of a user joining a group, with fields: user_id, group_id, and join_date. The joining probability is influenced by user activity, country, and the group's popularity category.
- **Tags:** A record of user_ids tagged in a post (given by post_id)

5 Dataset Generation Plan

The project's dataset is synthetically generated using a series of parameterized Python scripts. This method ensures the data contains realistic distributions and that the entire dataset can be perfectly reproduced.

5.1 Generation Rules and Distributions

The generation logic incorporates several rules to create a lifelike social network dataset.

- **Parameters:** The main script is controlled by key parameters, including a fixed random seed (SEED = 150) for reproducibility, the total number of users (N_users = 6000), and the time period.
- **Users:** The script generates 30,000 users, split into 85% regular (Hiker) users and 15% commercial (Advertiser) users. User registrations follow an exponential growth trend over time and have a seasonal bias, with more users joining in the summer.

- **Skewed Distributions:** To mimic real-world patterns, nearly all data is generated with a heavy-tail (Zipf-like) distribution:
  - The distribution of users across countries is skewed.
  - A small number of highly active users create a large percentage of all posts and comments.
  - A small number of "popular" groups attract a disproportionate number of members.
  - Popular posts receive significantly more comments than others.
- **Social Behavior:** Relationships and interactions are not random.
  - Users are more likely to be friends with users from the same country.
  - Users are more likely to join groups related to their country and are influenced by the group's popularity.
  - Users are more likely to comment on posts made by their friends.
  - Users are more likely to tag their friends or fellow group members in posts.

5.2 Output Files

The generation process produces the following data files:

- 'users.csv' - Contains all user profile data.
- 'groups.csv' - Contains metadata for each group.
- 'posts.csv' - Contains metadata for each post, including optional links to activities and photos, and a list of tagged users.
- 'comments.csv' - Contains all comments made on posts.
- 'group_joins.csv' - A log of all user-group membership events.
- 'actual_friend.npy' - A NumPy adjacency matrix representing the user friendship graph.

5.3 Reproducibility

To regenerate the exact same dataset, first download the files from the following repository: `https://github.com/HartigFries/DBS2_project`. Then, install the required dependencies and run the main generator script. The script uses a fixed random seed and internal parameters to ensure that the output is identical each time it is executed.

Command to generate the dataset:

```
# First, ensure dependencies are installed:
pip install -r requirements.txt

# Then, run the main script. The following parameters
# are set within the parameters.py file to produce the dataset:
python main.py
```

```
# Key Parameters Used:
# SEED          150
# N_users       6000
# N_groups      500
# N_posts       10000
# START_YEAR    2020
# N_years       3
# mode = "expotential"
```