**User Nodes:**

CREATE (u1:User {user\_id: 1, name: 'Isaac', last\_name: 'Newton', age: 20});

CREATE (u2:User {user\_id: 2, name: 'Richard', last\_name: 'Feynman', age: 30});

CREATE (u3:User {user\_id: 3, name: 'Chandrasekhara', last\_name: 'Raman', age: 40});

CREATE (u4:User {user\_id: 4, name: 'Srinivasa', last\_name: 'Ramanujam', age: 25});

CREATE (u5:User {user\_id: 5, name: 'Galileo', last\_name: 'Galilei', age: 78});

CREATE (u6:User {user\_id: 6, name: 'Marie', last\_name: 'Curie', age: 35});

CREATE (u7:User {user\_id: 7, name: 'Albert', last\_name: 'Einstein', age: 100});

CREATE (u8:User {user\_id: 8, name: 'Oviyasri', last\_name: 'Aryabhata', age: 50});

CREATE (u9:User {user\_id: 9, name: 'Rabindranath', last\_name: 'Tagore', age: 90});

CREATE (u10:User {user\_id: 10, name: 'Paul', last\_name: 'Erdos', age: 75});

**Group Nodes:**

CREATE (g101:Group {group\_id: 101, name: 'Isotope Enthusiasts'});

CREATE (g102:Group {group\_id: 102, name: 'Food Lovers'});

CREATE (g103:Group {group\_id: 103, name: 'Tech Innovators'});

**Post Nodes:**

CREATE (p201:Post {post\_id: 201, title: "Exploring Radium"});

CREATE (p202:Post {post\_id: 202, title: "Best Pizza Places"});

CREATE (p203:Post {post\_id: 203, title: "AI Breakthroughs"});

**Comment Nodes:**

CREATE (c301:Comment {comment\_id: 301, text: "Awesome photos!"});

CREATE (c302:Comment {comment\_id: 302, text: "I want to visit the moon."});

CREATE (c303:Comment {comment\_id: 303, text: "That pizza looks delicious!"});

CREATE (c304:Comment {comment\_id: 304, text: "LoL!"});

CREATE (c305:Comment {comment\_id: 305, text: "That's hilarious"});

**Relationships:**

MATCH (u1:User {user\_id: 1}), (p201:Post {post\_id: 201})

CREATE (u1)-[:CREATED]->(p201);

MATCH (u2:User {user\_id: 2}), (p202:Post {post\_id: 202})

CREATE (u2)-[:CREATED]->(p202);

MATCH (u3:User {user\_id: 3}), (p203:Post {post\_id: 203})

CREATE (u3)-[:CREATED]->(p203);

MATCH (u1:User {user\_id: 1}), (p202:Post {post\_id: 202}), (c303:Comment {comment\_id: 303})

CREATE (u1)-[:COMMENTED\_ON]->(p202), (u1)-[:WROTE]->(c303), (c303)-[:ON\_POST]->(p202);

MATCH (u2:User {user\_id: 2}), (p201:Post {post\_id: 201}), (c301:Comment {comment\_id: 301})

CREATE (u2)-[:COMMENTED\_ON]->(p201), (u2)-[:WROTE]->(c301), (c301)-[:ON\_POST]->(p201);

MATCH (u2:User {user\_id: 2}), (p203:Post {post\_id: 203}), (c302:Comment {comment\_id: 302})

CREATE (u2)-[:COMMENTED\_ON]->(p203), (u2)-[:WROTE]->(c302), (c302)-[:ON\_POST]->(p203);

MATCH (u3:User {user\_id: 3}), (p201:Post {post\_id: 201}), (c305:Comment {comment\_id: 305})

CREATE (u3)-[:COMMENTED\_ON]->(p201), (u3)-[:WROTE]->(c305), (c305)-[:ON\_POST]->(p201);

MATCH (u3:User {user\_id: 3}), (p202:Post {post\_id: 202}), (c304:Comment {comment\_id: 304})

CREATE (u3)-[:COMMENTED\_ON]->(p202), (u3)-[:WROTE]->(c304), (c304)-[:ON\_POST]->(p202);

MATCH (u4:User {user\_id: 4}), (p201:Post {post\_id: 201})

CREATE (u4)-[:LIKED]->(p201);

MATCH (u4:User {user\_id: 4}), (c302:Comment {comment\_id: 302})

CREATE (u4)-[:LIKED]->(c302);

MATCH (u5:User {user\_id: 5}), (c301:Comment {comment\_id: 301})

CREATE (u5)-[:LIKED]->(c301);

MATCH (u5:User {user\_id: 5}), (c303:Comment {comment\_id: 303})

CREATE (u5)-[:LIKED]->(c303);

MATCH (u6:User {user\_id: 6}), (g101:Group {group\_id: 101})

CREATE (u6)-[:CREATED]->(g101);

MATCH (u7:User {user\_id: 7}), (g102:Group {group\_id: 102})

CREATE (u7)-[:CREATED]->(g102);

MATCH (u8:User {user\_id: 8}), (g103:Group {group\_id: 103})

CREATE (u8)-[:CREATED]->(g103);

MATCH (u1:User {user\_id: 1}), (g101:Group {group\_id: 101})

CREATE (u1)-[:MEMBER\_OF]->(g101);

MATCH (u2:User {user\_id: 2}), (g102:Group {group\_id: 102})

CREATE (u2)-[:MEMBER\_OF]->(g102);

MATCH (u3:User {user\_id: 3}), (g103:Group {group\_id: 103})

CREATE (u3)-[:MEMBER\_OF]->(g103);

MATCH (u10:User {user\_id: 10}), (g101:Group {group\_id: 101})

CREATE (u10)-[:MEMBER\_OF]->(g101);

MATCH (u9:User {user\_id: 9}), (g102:Group {group\_id: 102})

CREATE (u9)-[:MEMBER\_OF]->(g102);

A diagram of a network

Description automatically generated

A diagram of different colored circles

Description automatically generated

**Q1. Find all posts created by Isaac and list the users who liked those posts.**

MATCH (isaac:User {name: 'Isaac'})-[:CREATED]->(post:Post)<-[:LIKED]-(user:User)

RETURN post.title, user.name

A white rectangular object with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

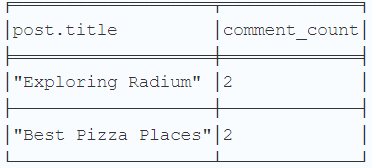
**Q2. Write a query to identify posts that received no less than two comments and display the number of comments for each of these posts.**

MATCH (comment:Comment)-[:ON\_POST]->(post:Post)

WITH post, count(comment) AS comment\_count

WHERE comment\_count >= 2

RETURN post.title,  comment\_count



A screenshot of a chat

Description automatically generated

**Q3. Calculate the user engagement score for each user, where the score is the sum of the likes received on their posts and comments. List users ranked by their engagement score.**

MATCH (user:User)

OPTIONAL MATCH (user)-[:CREATED]->(post:Post)<-[:LIKED]-(liker)

WITH user, count(liker) AS post\_likes

OPTIONAL MATCH (user)-[:WROTE]->(comment:Comment)<-[:LIKED]-(comment\_liker)

WITH user, post\_likes, count(comment\_liker) AS comment\_likes

RETURN user.user\_id, user.name, user.last\_name,

(post\_likes + comment\_likes) AS engagement\_score

A table of data with text

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

**Q7. Calculate the average number of comments on posts created by each user. List users with their average comment counts.**

MATCH (user:User)

OPTIONAL MATCH (user)-[:CREATED]->(post:Post)

OPTIONAL MATCH (post)<-[:ON\_POST]-(comment:Comment)

WITH user, count(comment) AS total\_comments, count(DISTINCT post) AS total\_posts

WITH user, CASE WHEN total\_posts > 0 THEN total\_comments \* 1.0 / total\_posts ELSE 0 END AS average\_comments

RETURN user.user\_id, user.name, user.last\_name, average\_comments

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Q10. Propose a real-world use case where this social network graph database could be applied effectively, considering user engagement, content creation, and interactions. Describe the scenario and how the database would be used.**

We can have a Cross Platform Game Community named Game Verse, which will be an extension to platforms like Steam, Xbox, PlayStation, Nintendo, and Epic Games. This is where players from different platforms can connect, discuss games, and form teams regardless of their device or service. Game Verse will provide a single social space where gamers across all platforms share strategies, find teammates, and participate in multi-platform events.

**Features:**

1. **Integrated Gaming Identity:** Game Verse will allow users to create a single common gaming profile by linking their accounts from various platforms such as Steam, Xbox, PlayStation, and Nintendo. This will be a common and centralized profile of each player showcasing a player's achievements, game library, and progress across all connected platforms.
2. **Universal Gaming Communities**: it will have game-specific groups and discussion forums. Each of these spaces will enable players from different platforms to come together and share strategies, exchange tips, and showcase their gaming content. This leads to a collaborative, inclusive and diverse game community/ecosystem.
3. **Smart Cross-Platform Matchmaking**: Game Verse will consist of an AI based matchmaking system that will connect players from different platforms. This will be based on their gaming preferences, skill levels, and activity patterns. For example, an Xbox player to team up seamlessly with a PC gamer on Steam, improving the multiplayer experience.
4. **Across-Platform Competitions**: We can have a single unified leaderboards and cross-platform events, allowing gamers to participate in tournaments regardless of their preferred gaming system. Leading to healthy competition promoting a sense of togetherness among gamers from different platform backgrounds.
5. **Personalized Cross-Platform Discovery**: Using user data and activity, the Game Verse will be able to recommend groups, games, and potential friends from various gaming platforms. This type of suggestion will help the gamers to discover new content, communities, and connections.

**How will the graph database be used?**

1. **User Profile Nodes and Relationships:**
   * The graph database will be used to link users to their individual gaming accounts, making cross-platform identification easier.
   * Profiles will be unified by relationships, allowing to display interconnected achievements and games across accounts.
2. **Cross-Platform Group Engagement:**
   * Groups will be linked to games, and users can be connected to multiple groups, making it easier to navigate between discussions and groups.
   * By tracking likes, comments, and shares in each group, we can identify highly interactive or trending conversations.
3. **Team Formation and Matchmaking:**

* Graph database can help analyse user skills, past playstyles, and active games, recommending teammates with complementary attributes.
* Allowing dynamic matchmaking across platforms

1. **Tracking Events and Competitions:**

* Users are linked to event participation data, allowing Game Verse to personalize invites to upcoming tournaments. It can track individual or team performances in events, using graph connections to rank them across platforms in real time.

1. **Content & Friend Recommendations:**

* Connections among users and groups allow Game Verse to recommend content and friends based on shared game preferences.
* Recommendations leverage relationship strength, engaging users by suggesting highly relevant content and friends across platforms.