PROJECT PHASE - 1

MiniWorld Topic - CPstalk

TEAM NUMBER: 12

TEAM MEMBERS:

1. Sriteja Reddy Pashya (2021111019)
2. Keval Jain (2021111030)
3. Romica Raisinghani (2021101053)

Introduction to the mini-world

Everyone who does Competitive Programming, has wasted time juggling between different coding platforms in finding problems, keeping track of contests, or even stalking your friends!

We provide you a comprehensive solution for all this, by introducing you to CPstalker. It provides the below features across ALL popular coding sites

- To analyze your progress and compare with your friends
- Remind for future contests
- Recommend problems according to the tags and difficulty

Users of this database are anyone who does Competitive Programming, as this mini-world would help you keep track of your progress, and keep you motivated by stalking your friends!

Database Requirements

a) Assumptions

1. Users can practice as well as participate in contests
2. Teams can only participate in contests.
3. The Problem IDs are labeled A,B,C etc.…in a particular contest.

b) Strong Entity types

1. **Users**
    1. Username for CPstalk (Primary Key)
    2. Name (Compound Attribute)
        1. First name
        2. Middle name
        3. Last name
    3. Year of Graduation
    4. Institution
    5. Profession
    6. Country

2. **Platform**
    1. Name (Primary Key Attribute)
    2. Users (Compound Attribute)
        1. Active Users
        2. Total Users
    3. Languages (Multivalued)
    4. Headquarters
    5. Founders
    6. Launched Year

3. **Problem (Weak Entity)**
    1. Problem ID (Partial Key)
    2. Problem Rating
    3. Problem Author (Multivalued)

    4. Number of solves
    5. Total Attempts
    6. Accuracy (Derived Attribute)
    7. Tags (Multivalued)

4. **Contest (Weak Entity)**
    1. Contest ID (Partial Key)
    2. Style
    3. Start Time
    4. Duration
    5. Number of participants

5. **Teams**
    1. Team ID (Primary Key)
    2. Team name
    3. Team rating
    4. Date of Creation
    5. Team size

d) Relationship types

1. **Attempts:** (Quaternary Relationship) : Team consists of users (or individual users) attempting problems in a contest
    1. **Descriptive attributes:**
        - i. Submission ID
        - ii. Time of Submission
        - iii. Language
        - iv. Verdict
        - v. Execution Time
        - vi. Memory
    2. Degree: 4
    3. Participating Entity Types: Users, Teams, Problems, Contests
    4. Cardinality Ratio, Users : Teams : Problems : Contests - N:M:A:B

2. **Register on:** Users register on platforms, and teams register on platforms
    1. **Descriptive attributes**:
        i. Username/Teamname
        ii. Rating/Team Rating
        iii. Rank
        iv. Percentile (derived attribute)
    2. Degree: 2
    3. Participating Entity Types: Users - Platform, Teams - Platforms
    4. Cardinality Ratio, Users : Platform = N:Mn, Teams : Platform = N:M

3. **Practice:** Users practice problems
    1. **Descriptive attributes:**
        i. Submission ID
        ii. Time of Submission
        iii. Language
        iv. Verdict
        v. Execution Time
        vi. Memory
    2. Degree: 2
    3. Participating Entity Types: Users, Problems
    4. Cardinality Ratio, Users : Problems = N:M

4. **Friends: (Recursive relationship)** Users are friends with users
    1. Degree: 1
    2. Participating Entity Types: Users - Users
    3. Cardinality Ratio: Users : Users = N:M

5. **Host (Identifying relationship):** Platform hosts problems
    1. Degree: 2
    2. Participating Entity Types: Platform - Problems
    3. Cardinality Ratio: Platform : Problems = 1 : N

Functional Requirements

Modifications

1. **Insert:**
    1. **insert_user**: inserts a new user on CPstalk
    2. **insert_platform_info_for_user:** inserts the data for a particular user of CPstalk on a particular platform such as platform rating, username on the platform, etc.
    3. **insert_problem**: inserts a problem into the database in a particular platform
    4. **insert_contest**: inserts information about a new contest in a particular platform
    5. **insert_team**: inserts a new team with the required information
    6. **insert_submission:** adds a submission in a particular platform and calls update_problem_info

2. Delete:

1. **delete_user**: delete the entire data corresponding to a particular user from CPstalk
2. **delete_team**: delete the entire data corresponding to a particular team

3. Update:

1. **update_user_info:** modifies the information of a particular user of CPstalk such as username, institution, country etc
2. **update_platform_info_for_user:** updates the data for a particular user of CPstalk on a particular platform such as platform rating, platform username etc
3. **update_problem_info:** updates the information for a particular problem after each submission such as number of solves, total attempts and accuracy
4. **update_platform_info:** updates information such as number of active users, total users etc

5. **update_team:** updates the number of people in a team
6. **update_contest_info:** updates the info for a particular contest such as number of registered users, start time, etc.

Retrievals

1. Selection:

- **Country:** Give details of all people who are from a certain country
- **Institution:** Give details of all people who are from a certain institution
- **Platform:** Give details of all users on a certain platform
- **Contests:** Give details of all contest on a certain platform
- **Team:** Give all detail of users in a certain team

2. Projection:

- **Rating range:** Give the usernames of all users within a certain rating range on a particular platform
- **Problem_contest:** Give the problem name and tags of all problems in a contest
- **Problem_platform:** Give the problem rating and name of all problems above/below a certain accuracy on a platform
- **Problem_rating:** Give the problem name and id of all problems in certain rating range on a certain platform
- **Problem_tag:** Give the problem name and rating of all problems with a certain tag

3. Aggregate:

- **Max_rating_user:** Gives the details of the highest rated problem of a certain user
- **Count_Problems:** Gives number of problems above a certain rating, or with a particular tag on a platform, or solved by a particular user, in a platform given

4. Search:

- **Search_user:** Search for usernames on a certain platform partially matching a certain string.
- **Search_problem:** Search for problems on a certain platform partially matching a certain string.

5. Analysis:

- **Strength of people who use CPstalk:** For each user on CPstalk, calculate their average percentile across the platforms they are registered on, and then get the average of each users percentile which will depict the approximate strength of users who use CPstalk
- **Deeper analysis of a user:** For each user on CPstalk, we can analyze the performance of a particular user by observing their rating changes, frequency of solving problems with particular attributes (such as tag, range) etc