



**AMERICAN
UNIVERSITY OF BEIRUT**

**MAROUN SEMAAN FACULTY OF
ENGINEERING & ARCHITECTURE**

American University of Beirut

School of Engineering and Architecture

Department of Electrical and Computer Engineering



A database design for the Dakowdas Competitive Programming Website

By

Dana Kossaybati (Group Leader)

(dak39@mail.aub.edu)

Mohammad Khaled Charaf

(mmc51@mail.aub.edu)

Mohammad Ayman Charaf

(mmc50@mail.aub.edu)

A REPORT

submitted to Dr. Hussein Bakri in partial fulfillment of the requirements of phase 1
of the database project for the course EECE433 – Database Systems

September 2023

Table of Contents

1 – Introduction
2 – References/Copyright
3 – Tool(s) used to draw the ER
4 – System Description & Requirements
5 - Legend of ER diagram symbols
6 – Complete Old ER Diagram for the Dakowdas database
7 – New Complete Amended ER Diagram for the Dakowdas database
7.1 Entity Types & Their Attributes
7.2 Relationships and their explanations
8 – ER to Relational Mapping:
8.1 – Step 1 – Mapping Strong Entity Types
8.2 – Step 2 – Mapping of Weak Entity Types
8.3 – Step 3 – Mapping of Binary 1:1 Relationship Types
8.4 – Step 4 – Mapping of Binary 1:N Relationship Types
8.5 – Step 5 – Mapping of Binary M:N Relationship Types
8.6 – Step 6 – Mapping of multivalued attributes
8.7 – Step 7 – Mapping of N-ary attributes
8.8 – Step 8 Mapping aggregation, specialization relationships
9 – Final Display – all tables
10 – Tables' states
11 – Conclusion
12 – Instructor Feedback

1- Introduction

The exciting and dynamic world of competitive programming presents a knowledge-filled environment for creative minds with a passion for problem solving. Competitive programming has gained immense popularity over the years, attracting participants from around the globe. This popularity brings with it a substantial amount of data, including user profiles, contest results, problem sets, and discussions. Managing this data effectively is essential not only for the website's performance but also for ensuring a seamless user experience. To streamline the processes, ensure fair competition, and provide an enriching experience for programmers and coding enthusiasts, our database project for the competitive programming website **Dakowdas** aims to satisfy the needs and wants of programmers.

It's quintessential to take into consideration all the parties involved while maintaining CIA: Confidentiality, Integrity, and Availability. As owners of the website, we'd want a secure and logical database with minimal errors. As users of the website, we'd want fast performance and a fair experience. Our project is mainly based on the popular competitive programming website **Codeforces** but with a thrilling added twist: *Battles*.

This report describes the **Dakodas** main database design using an Entity-Relationship diagram. Section 3 mentions the tool(s) that were utilized to draw the ER. After that section 4 describes the requirements of the system. Following, section 5 is the legend of ER diagram symbols which aids in clarifying the ER. Furthermore, sections 6 and 7 encompass the entity types and relationships in detail. After that, section 8 goes step by step into the mapping algorithm, leading to section 9 which displays all tables. Section 9 illustrates how the tables work by providing table states. Finally, the report wraps up with a conclusion. By the end of this project, we aim to deliver a powerful and reliable database solution that will serve as the backbone of our competitive programming website, ensuring that programmers can focus on what they do best: solving complex problems and pushing the boundaries of their coding abilities.

2-References/Copyright

Legend Reference:

R. Elmasri and S.B. Navathe, "Fundamentals of Database Systems." Pearson, 2020, pp. 83. Figure 3.14: Summary of the notation for ER diagrams.

3- Tool(s) Used to Draw ER

Draw.io was the only tool used to draw the ER.

4- System Description & Requirements:

- A user chooses a unique username to be identified by, a password and a unique email used to login to the website. He fills in his first name, last name, and his country of residence, which can be selected from a predefined list. For each user, we must also record the registration date, the number of friends, and the number of problems solved successfully.
- Users can message each other at a given timestamp. Each message has some content.
- A user may also befriend another user at a certain time.
- A user can make an announcement. Each announcement is given a unique ID. It has some content which conveys some news or updates. It also has the language that it was written in (Eg: English, Russian,...), timestamp recording the time at which the announcement was made, and the time elapsed since the announcement.
- The competitive programming website also has a blog entry feature. A user may choose to post a blog entry which is displayed on his page. A distinctive ID is given to each and every blog entry,

as well as a topic rating (vote from other users), a title, the timestamp at which it was posted, and content of the entry. A blog entry can also include some hashtags that help other users explore blog entries in accordance with their preferences.

- A user comments on a blog entry to reflect his view on the topic discussed. The comment has content, a comment timestamp, and the vote it receives by viewers. Many users can comment on a blog entry.
- An admin is a user identified by an AdminID. Admins are responsible for maintaining standards and guidelines, contest coordination, user support, policy enforcement, etc. Such responsibilities of the admin must also be recorded, along with the contribution level of the admin. Furthermore, an admin has a role and can be a super admin, a moderator, or a support admin.
- The competitive programming website also has programming problems. Each problem is assigned a problem number and a difficulty level (A, B, C, D, E or F). The problem ID, composed of the problem number and difficulty level, sets a problem apart from others. A problem has a title, description, memory limit, time limit. Tags such as dynamic programming, bitmasks, and combinatorics are added to a problem to help categorize it and give insight into solving it.
- A solution entity is identified by the solution's ID and the specific problem which the solution solves. Each problem must have one and only one solution. The solution has source code.
- Each programming problem is tested by at least one test case. A test case is distinguished by the combination of the test case number and the problem ID. Each test case has input and expected output according to the problem specifications.
- Submissions are differentiated by the submission ID. For each submission, the programming language, source code, instance of submission (timestamp) and verdict (Accepted, Time Limit Exceeded on Test X, Memory Limit Exceeded on Test X, Wrong Answer on Test X) are recorded.
- A user may submit as many submissions to as many problems, but a specific submission is always submitted by one user only.
- The number of correct submissions must be recorded for every problem.
- The website also holds contests. A contest is characterized by a room number. A contest is also given a name, length, start timestamp, time before start, division (1,2,3, or 4), and short description to introduce people to the competition.
- A contest contains at least one programming problem.
- A programming problem used in one contest cannot be used by any other contest.
- A user may either be a contestant or a contest creator.

- No 2 contest creators share the same. Each contest creator organizes at least one contest, and every contest is organized by at least one contest creator.
- Each contestant is given an individualized ID. A contestant has a rating which reflects his competence in problem solving. Based on this rating, each contestant is assigned a bracket.
- A contestant must compete in at least one contest. For each contest the contestant takes part in, he gets a score, a rank (relative to other participating contestants), and the timestamp of registration. A contest is held even if no contestants compete in it.
- A contest creator may choose to notify a contestant regarding a programming problem. The notification content will be displayed and must be stored.
- A contestant can also hack a submission if he believes that it's incorrect and is invalid for a certain input. Each hack is assigned a distinguishing hack ID and has a date, defender(user who submitted the submission), verdict (Invalid Input, Unsuccessful Hacking Attempt, or Successful Hacking Attempt), test (for which the hacker believes the submission's code outputs a wrong answer).
- Another exciting feature is the battle. A battle represents a friendly one-to-one contest between 2 users. Only 2 users are allowed to join a battle, and the winner must be noted. A battle id is specified for every battle held, and each battle is composed of 2-3 programming problems.

5 - Legend of ER diagram symbols










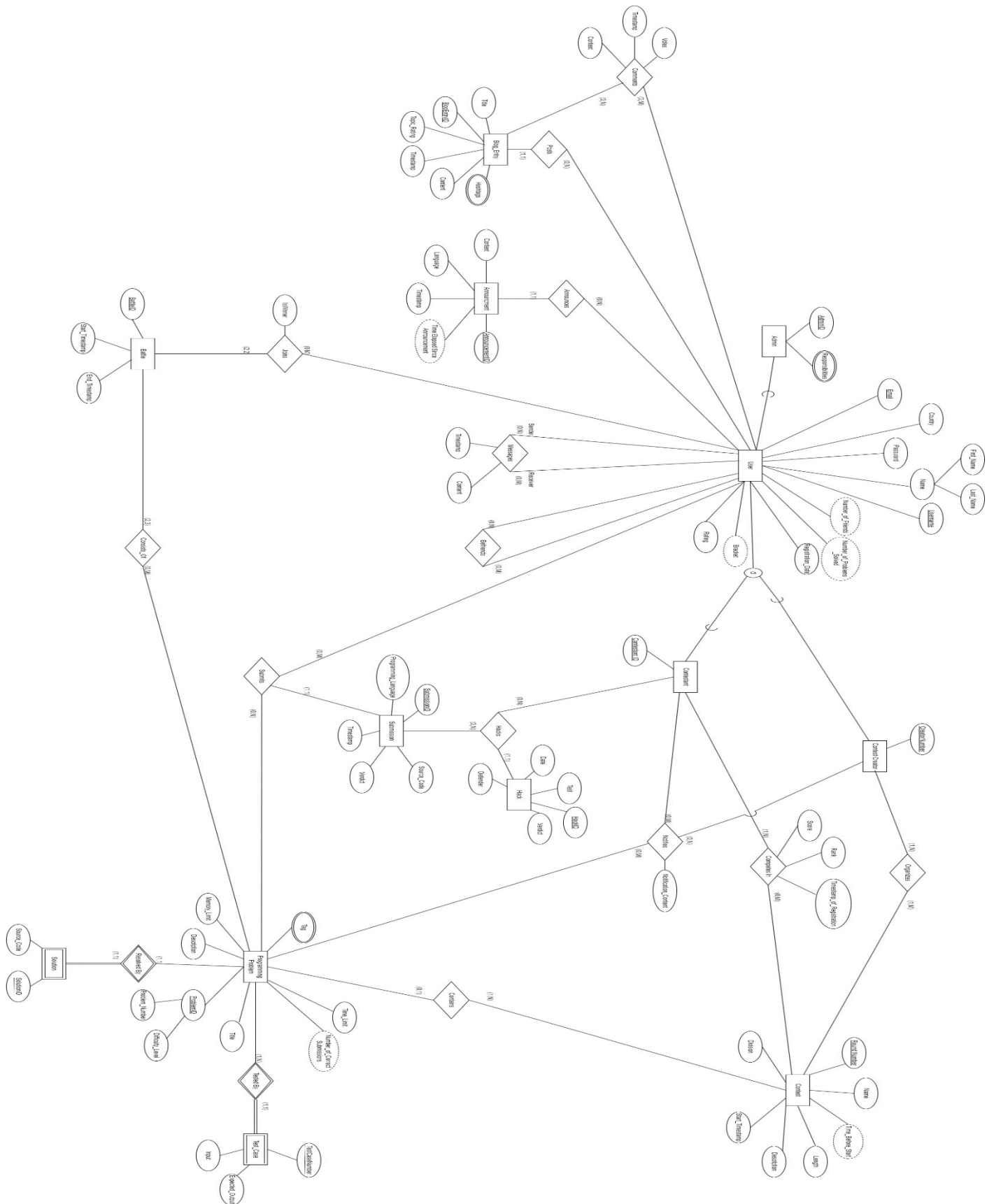
Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute

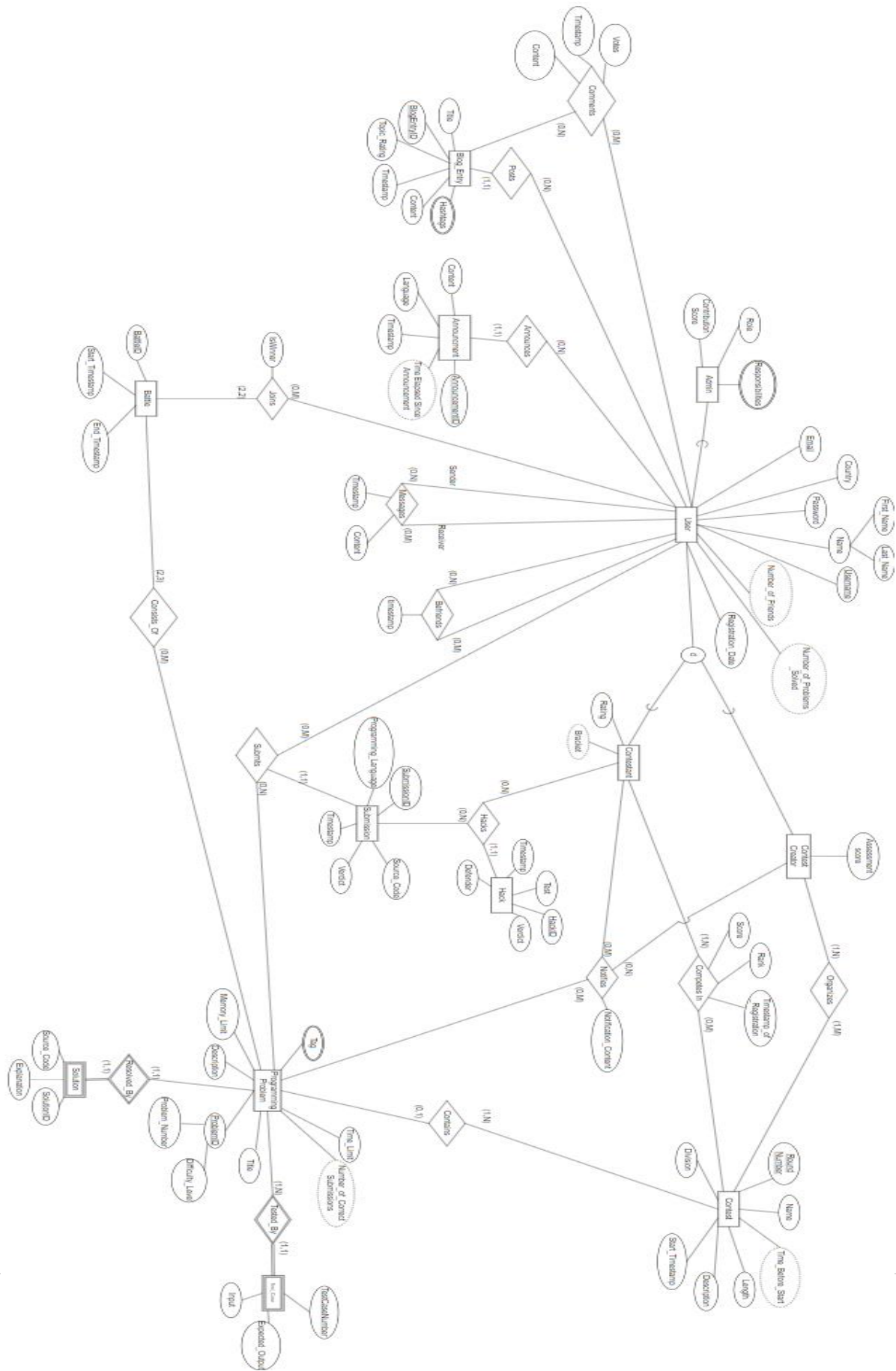
Figure 3.14
Summary of the notation for ER diagrams.



6 – Complete Old ER Diagram for the Dakowdas database

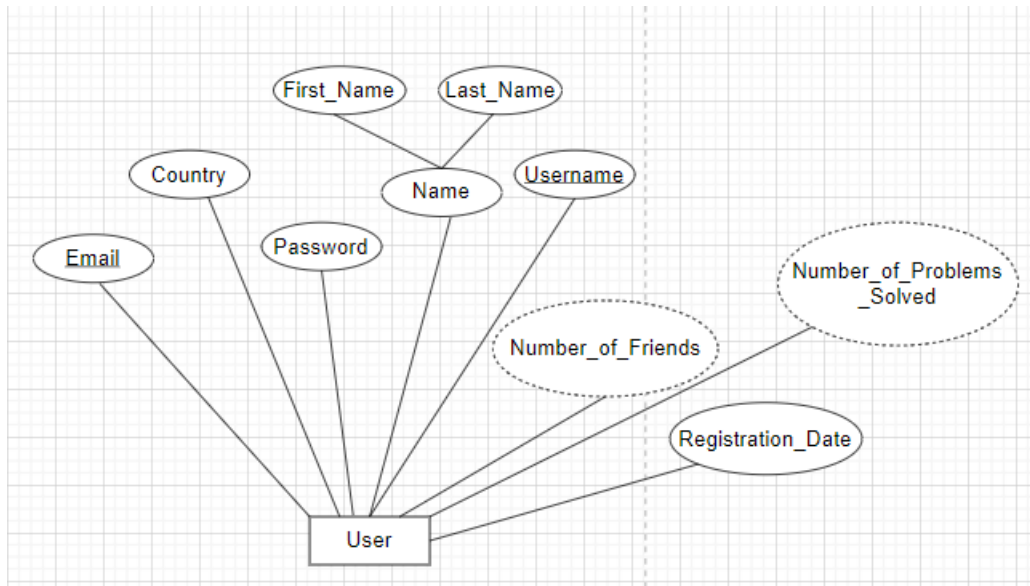


7 – Complete New ER Diagram for the Dakowdas database`



7.1- Entity Types & Their Attributes

7.1.1- User:



-Entity Name: User

This entity represents all the people that use the competitive programming website. It includes the information displayed on the user's profile as well as the user's confidential information.

-Entity Type: strong

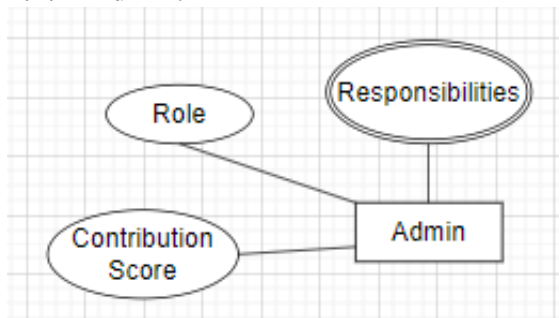
-Key Attribute(s): Username

>Reasoning Behind Choice of Key Attribute: A username may only be attributed to 1 user. Upon registration, the website prohibits the new user from using a username that's already taken

-Other Attributes:

- 1)Name: **composite** attribute. It is composed of the user's first name and the last name.
- 2)Email: used as contact information for the user. It can be used to notify the user of upcoming contests, rating updates, etc.
- 3>Password: used to validate that the party trying to register to the account is actually the alleged user.
- 4)Country: country of residence of the user. It's used to rank users in the same region.
- 5)Registration Date
- 6)Number of Friends: **derived** attribute. It's derived from the Befriends relationship between 2 users.
- 7)Number of Problems Solved: **derived** attribute.It's derived from the number of correct submissions that the user submits to the programming problems.

7.1.2- Admin:



-Entity Name: Admin

Admins are those who run and manage the website, ensuring its overall functionality and adherence to a high standard.

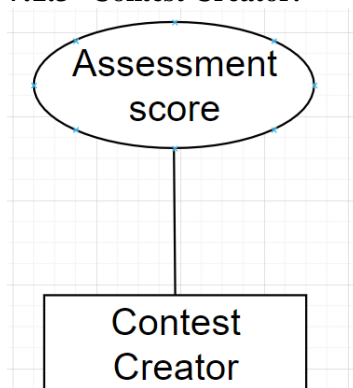
-Entity Type: strong

-Key Attribute(s): AdminID

-Other Attributes:

- 1) Responsibilities: **multivalued** attribute. It's used to store the jobs assigned to an admin such as: user support, rating system, community guidelines, or technical maintenance. Since each admin may have several responsibilities, it's a **multivalued** attribute.
- 2) Admin Role: An attribute indicating the specific role or level of admin privileges (e.g., super admin, moderator, support, etc.). This can determine the scope of their authority.
- 3) Contribution Score: A custom score or metric to measure the admin's overall contributions to the Codeforces community.

7.1.3- Contest Creator:



-Entity Name: Contest Creator

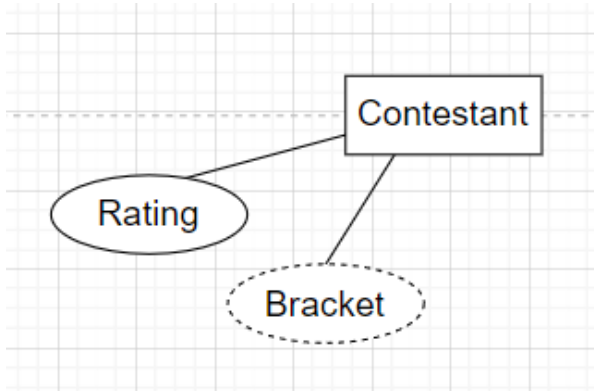
Contests are organized by contest creators. This distinguishes a normal user from a contest creator.

-Entity Type: strong

-Key Attribute(s): Creator Number

Each contest creator is given a unique creator number to distinguish him from other contest creators.

7.1.4- Contestant:



-Entity Name: Contestant

This entity represents those who compete in contests.

-Entity Type: strong

-Key Attribute(s): Contestant ID

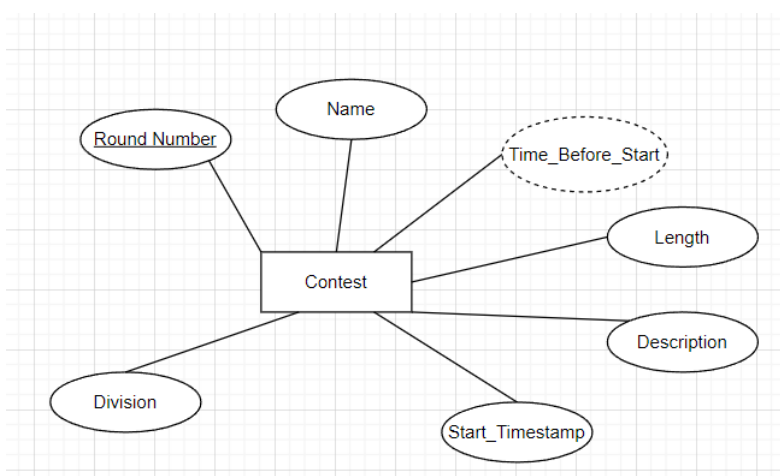
Each contestant is assigned a unique contestant ID which helps in distinguishing contestants.

-Other attributes: 1) Rating: demonstrates the prowess of the contestant.

2) Bracket: **derived** attribute. Each bracket corresponds to a range of rating values.

Brackets range from Newbie, Pupil, Specialist, Expert, Candidate Master, Grandmaster, etc. It's derived from the current rating of the contestant.

7.1.5- Contest:



-Entity Name: Contest

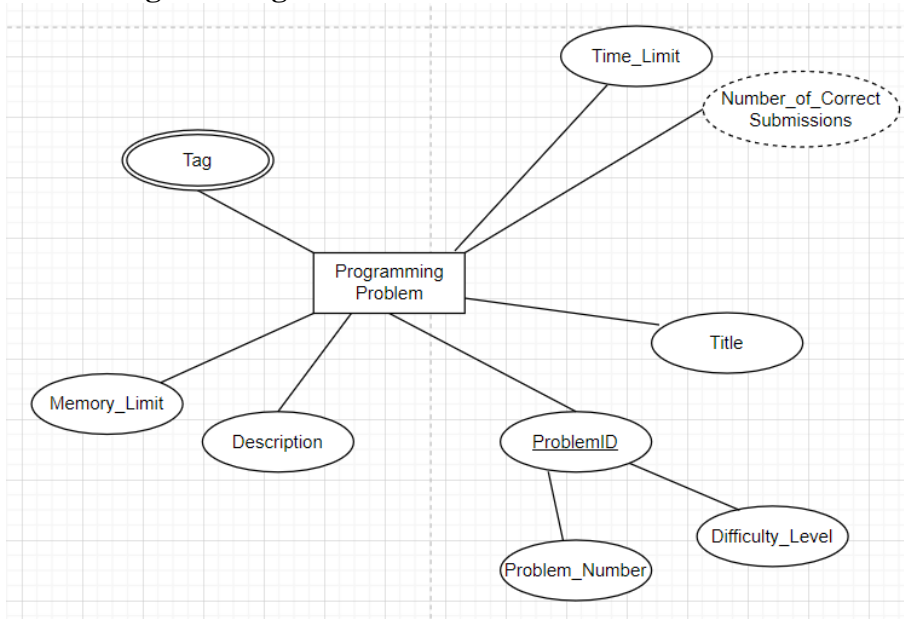
-Entity Type: strong

-Key Attribute(s): Round Number

-Other Attributes:

- 1)Name: considered as the title of the contest, often related to a theme, a sponsor, or a specific occasion.
- 2)Length: duration which the participants have to complete the contest, usually in hours and minutes.
- 3)Start Timestamp: the date and time when the contest will begin. This is typically provided in Coordinated Universal Time (UTC).
- 4)Description: short introduction to the contest. This may include the inspiration or the min goals.
- 5)Division: this reflects the level of difficulty and expertise required of the competition. For example, division 1 is for more proficient and higher-rated users.
- 6)Time Before Start: *derived* attribute. It's derived from the current time and the start time of the competition. It helps contestants better prepare for the contest and prevents delayed entry.

7.1.6- Programming Problem:



-Entity Name: Problem

-Entity Type: strong

-Key Attribute(s): Problem ID

Problem ID is a **composite** entity. It is composed of

- >Problem Number: index of the problem with respect to other problems with the same difficulty level.
- >Difficulty Level: ex: A, B, C, D, E, F.

The combination of the problem number and the difficulty level uniquely identifies the problem.

-Other Attributes:

1)Title

2)Description: introduction to the problem. It may contain a story to help elucidate the problem's logic.

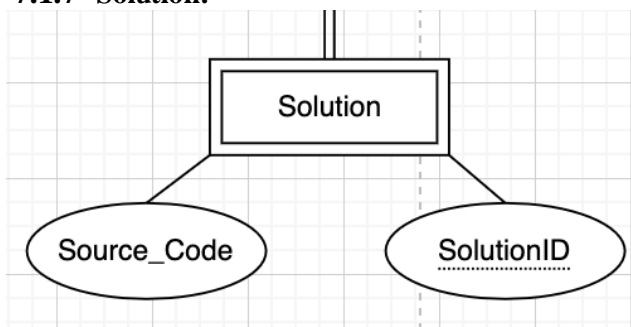
3)Memory Limit: maximum memory that a submission may use. If a submission exceeds the memory limit, then it is rejected. It's usually given in gigabytes.

4)Time Limit: maximum running time that a submission may use. If a submission exceeds the time limit, then it is rejected. It's usually given in milliseconds.

5)Tag: **multivalued** attribute. Tags reflect the topics that the problem covers and that are required for a successful submission. These include: maths, greedy, dynamic programming, etc. It's **multivalued** since a single problem may cover several topics.

6)Number of Correct Submissions: derived attribute. It's derived from the number of submits relationship with submissions with an "Accepted" Verdict for the problem.

7.1.7- Solution:



-Entity Name: Solution

-Entity Type: weak. Its identifying entity is Programming Problem.

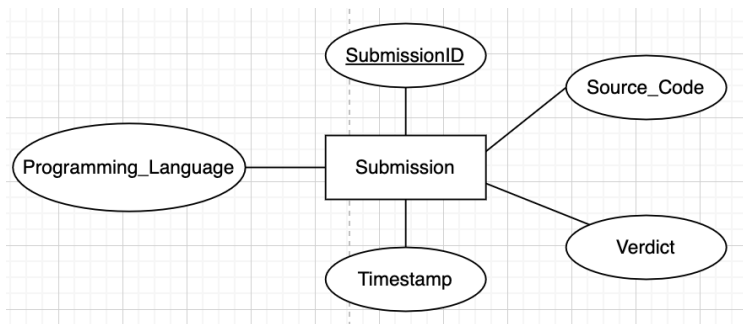
-Partial Key: Solution ID.

A solution is uniquely identified by the combination of the solution ID and the identifying programming problem's ID.

-Other Attributes:

1)Source Code: actual code implementation of the solution to the problem.

7.1.8- Submission:



-Entity Name: Submission

-Entity Type: strong

-Key Attribute(s): Submission ID

-Other Attributes:

- 1)Source Code: actual code implementation of the submission proposed by the user
- 2)Programming Language: the programming language used in the source code. This can be C++, Java, Python, or some other language.
- 3)Timestamp: exact instance of submission.
- 4)Verdict:this shows whether the submission was accepted or not.

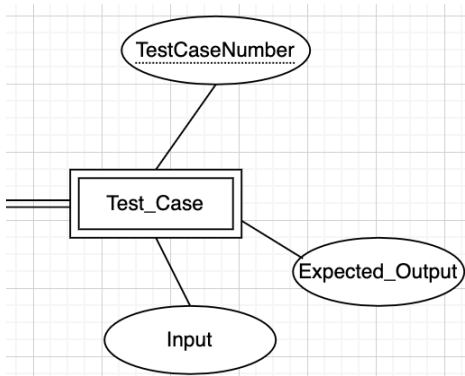
Verdicts are: -Accepted:source code ran correctly on all test cases.

-Wrong Answer on Test X: source code gave an output not equal to the expected output on test X.

-Time Limit Exceeded on Test X: source code exceeded the time limit when run on test X.

-Memory Limit Exceeded on Test X:source code exceeded the memory limit when run on test X.

7.1.9- Test Case:



-Entity Name: Test Case

-Entity Type: weak. Its identifying entity is Programming Problem.

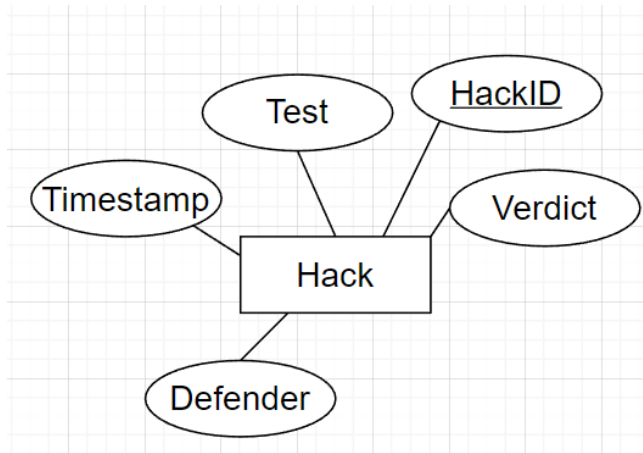
-Partial Key: TestCaseNumber.

A solution is uniquely identified by the combination of the TestCaseNumeber and the identifying programming problem's ID.

-Other Attributes:

- 1)Input: input to the code
- 2)Expected output: correct output based on the input and problem specifications.

7.1.10- Hack:



-Entity Name: Hack

A contestant can hack another contestant's submission if he believes that the submission is faulty and gives a wrong output on a certain Test. If the hack is successful, the hacker is awarded some points. However, if the hack is unsuccessful, he will be penalised.

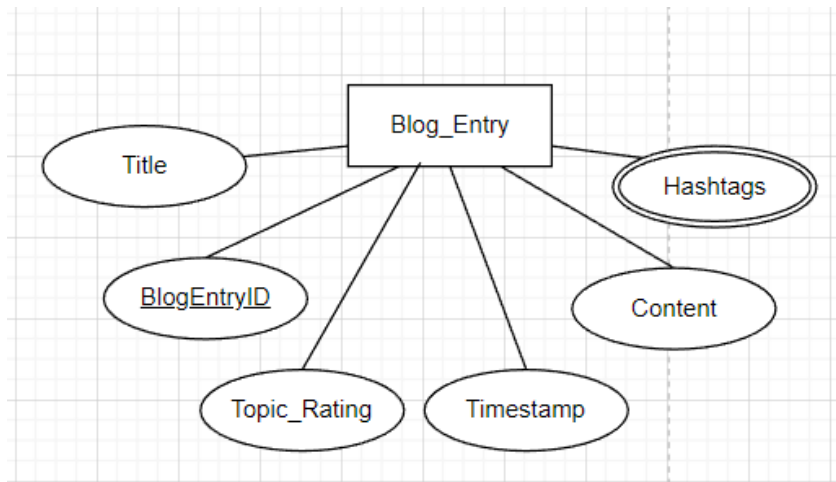
-Entity Type: strong

-Key Attribute(s): Hack ID

-Other Attributes:

- 1)Defender: contestant who submitted the submission in question.
- 2)Test: test that the hacker claims the submission fails on.
- 3)Verdict:
 - Invalid Input: input provided by the hacker doesn't meet the problem's input requirements.
 - Unsuccessful Hacking Attempt: submission didn't fail on test.
 - Successful Hacking Attempt submission failed on test.
- 4)Timestamp: date on which the hacker submitted the hack.

7.1.11- Blog Entry:



-Entity Name: Blog Entry

Blog Entry is used to give users a voice and allow them to share their concerns and ideas as well as interact with each other.

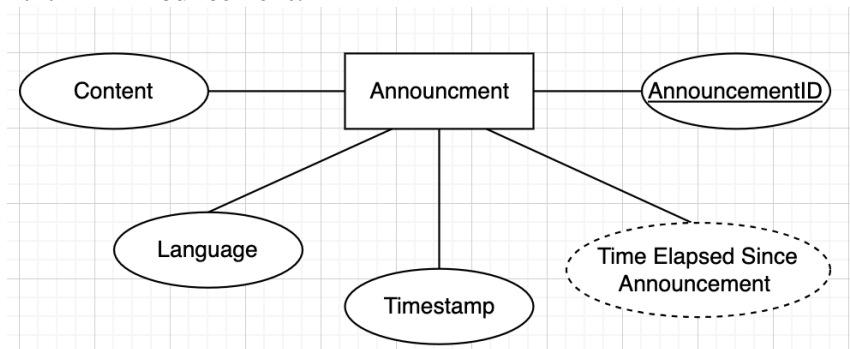
-Entity Type: strong

-Key Attribute(s): BlogEntryID

-Other Attributes:

- 1)Title: reflects the topic discussed in the entry.
- 2)Content: actual text of the blog entry
- 3)Topic Rating: vote given by other users to assess the relevance of the topic discussed.
- 4)Hashtags: **multivalued** attribute. Hashtags are used for categorization and discoverability of the blog entries posted by the users. They usually convey the theme of the blog. It's a **multivalued** attribute since a single blog entry may discuss several subject matters and hence have several hashtags.

7.1.12- Announcement:



-Entity Name: Announcement

Announcements are made by users to advertise contests.

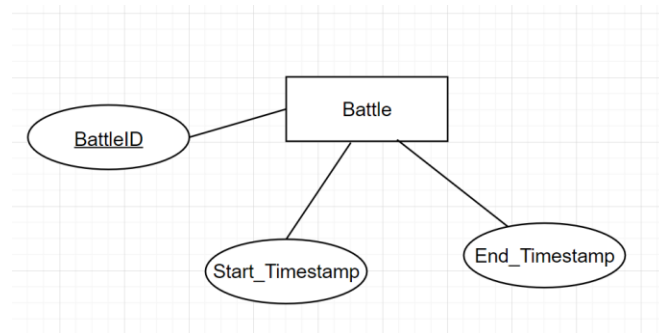
-Entity Type: strong

-Key Attribute(s):AnnouncementID

-Other Attributes:

- 1)Content: actual text of the announcement
- 2)Language: language used in the announcement (for example: English, French, Russian)
- 3)Timestamp: time at which the announcement was made.
- 4)Time Elapsed Since Announcement: *derived* attribute. It shows how long it has been since the announcement was made. It's derived from the current time and the timestamp of the announcement.

7.1.13- Battle:



-Entity Name: Battle

Battles are similar to contests but on a more private level. They involve 2 users going head-to-head while solving 2-3 programming problems. The first to finish all problems successfully is crowned as the winner and is awarded extra points to his rating.

-Entity Type: strong

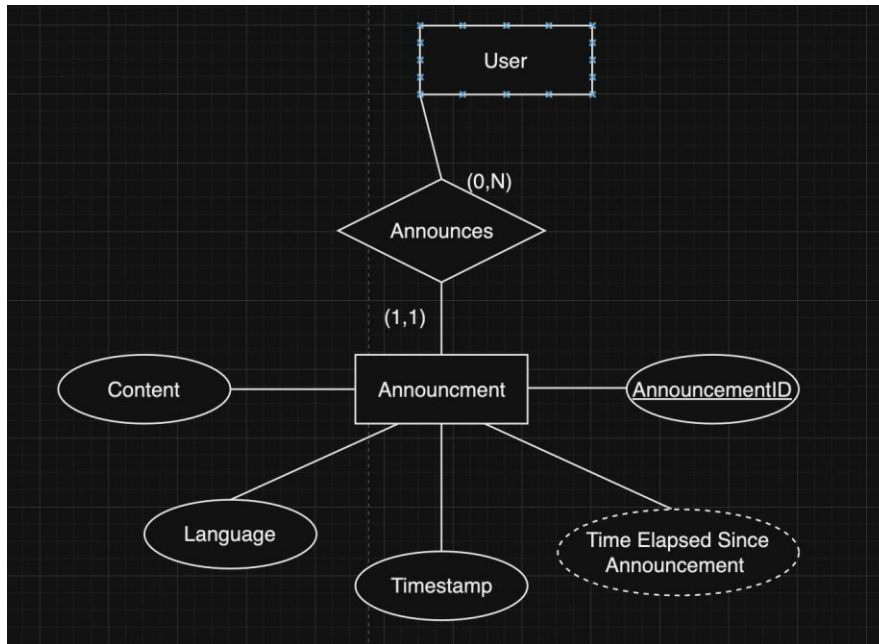
-Key Attribute(s):BattleID

-Other Attributes:

- 1)Start Timestamp: time at which the battle started.
- 2)End Timestamp: time at which the battle ended. Note that a battle ends when the winner finishes all problems.

7.2- Relationships and Their Explanations

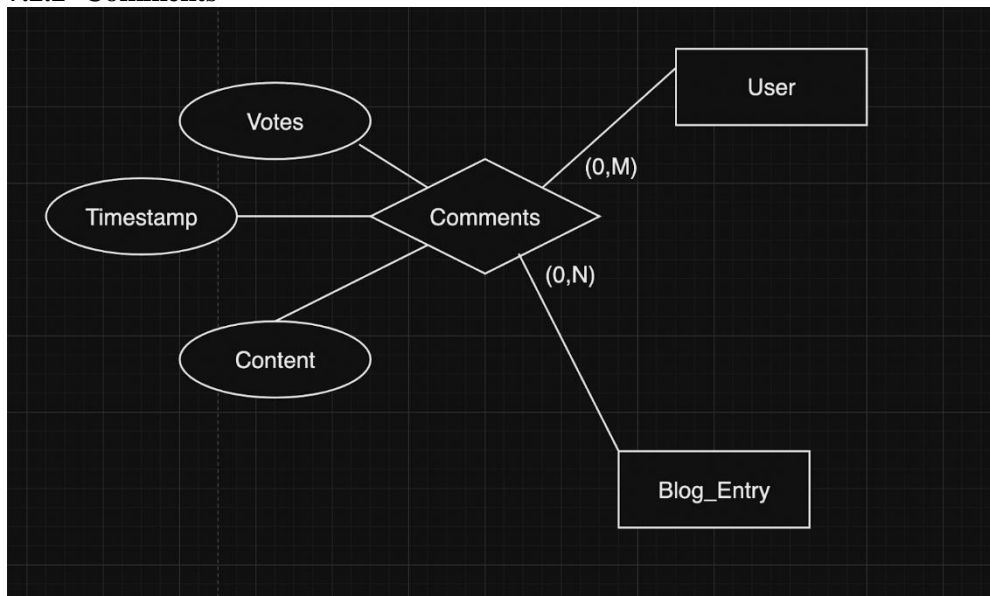
7.2.1- Announces



Relationship Type: Binary

A user can announce many announcements. However, an announcement can be announced by only one user.

7.2.2- Comments



Relationship Type: Binary

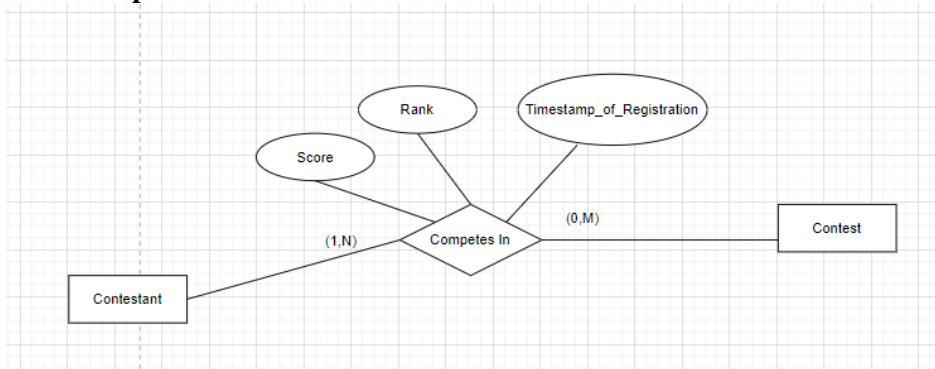
A user can comment in a Blog Entry.

A Blog Entry can have many comments.

For each comment, we store:

- Content: information shared in the comment.
- Timestamp: time at which the comment was made
- Votes: other users' rating of the comment, which reflects their approval or dissatisfaction with the comment.

7.2.3 Competes



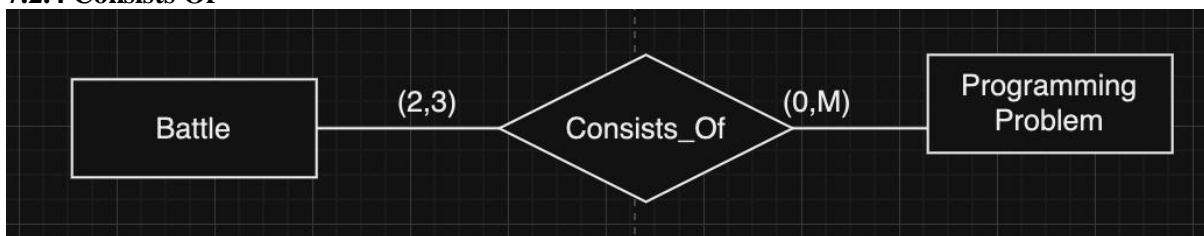
Relationship Type :Binary

A contestant must compete in at least one contest . A contest can have many contestants competing in it.

A contest will start even if no contestants compete in it.

The score, rank and timestamp of registration are recorded for each contestant in the competition.

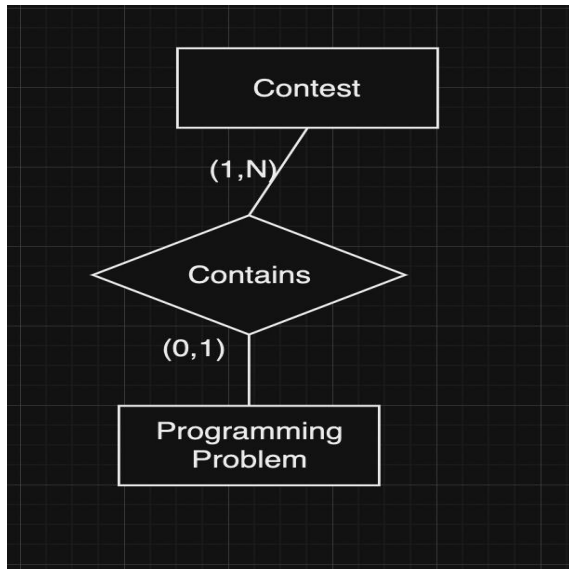
7.2.4 Consists Of



Relationship Type: Binary

A Battle consists of at least 2 and at most 3 problems. A programming problem can be part of many battles.

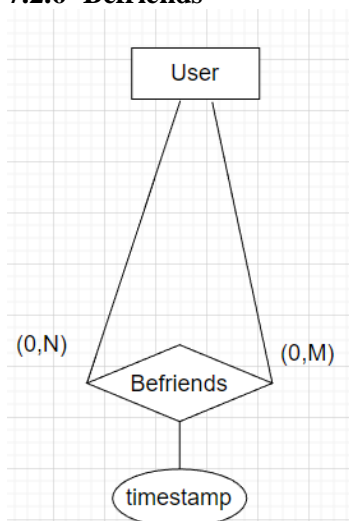
7.2.5- Consists Of



Relationship Type: Binary

A Contest contains at least one problem. A programming problem belongs to utmost one contest.

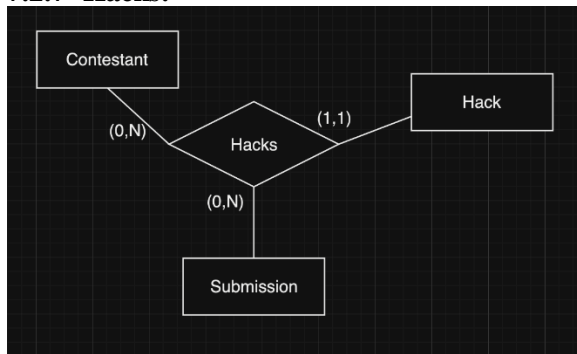
7.2.6- Befriends



Relationship Type: Recursive

A user can befriend many users. Timestamp of when the 2 users became friends is recorded.

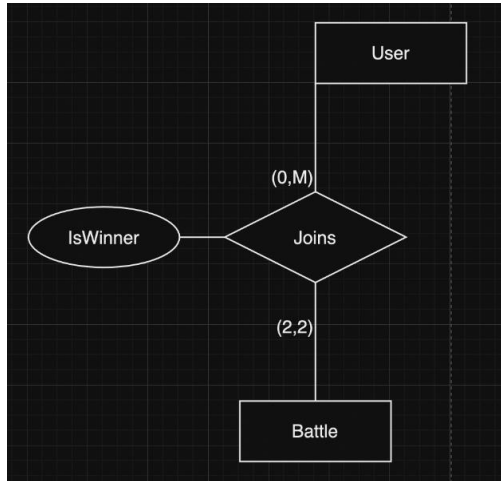
7.2.7- Hacks:



Relationship Type: Ternary

A contestant can hack many submissions with a hack. Any failed hacks are stored to penalise the hacker for an unsuccessful submission. Otherwise, the hacker is rewarded.

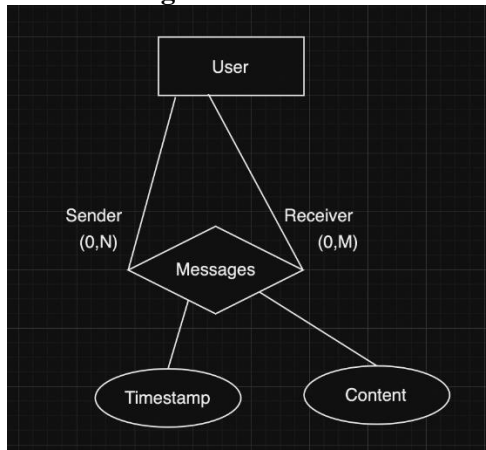
7.2.8- Joins:



Relationship Type: Binary

A user can join many battles. A battle consists of only 2 users.

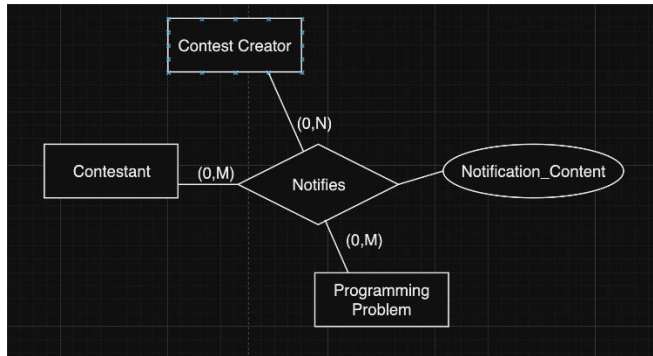
7.2.9- Messages:



Relationship Type: Recursive

A sender sends many messages. A receiver receives many messages. The content and timestamp of each message sent is recorded.

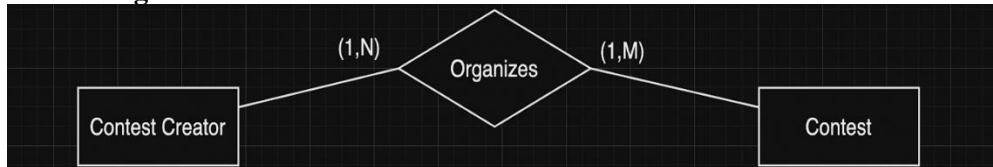
7.2.10- Notifies:



Relationship Type: Ternary

A contestant creator(s) can notify all contestants about any missing details for a programming problem. The content of the notification is recorded.

7.2.11- Organizes:

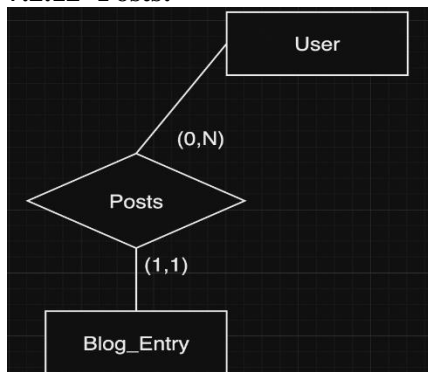


Relationship Type: Binary

A contest creator organizes at least one contest.

A contest must be organized by at least one contest creator.

7.2.12- Posts:

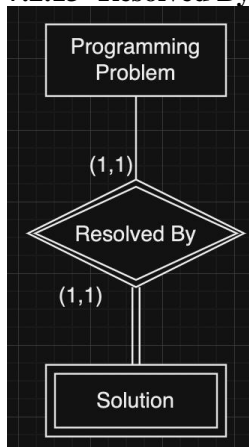


Relationship Type: Binary

A user can post many blog entries.

A blog entry must be posted by only one user.

7.2.13- Resolved By:

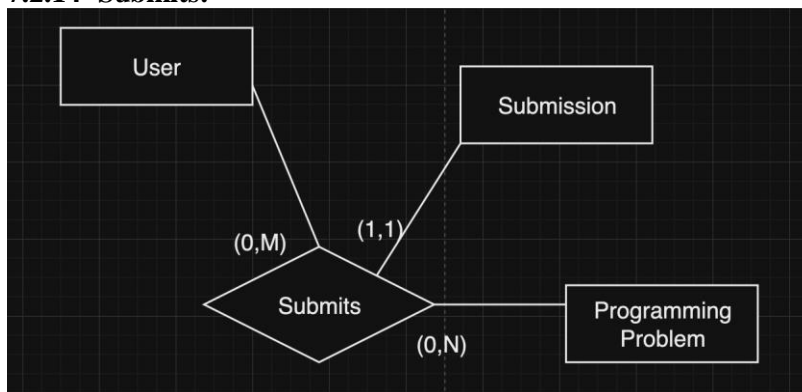


Relationship Type: Binary - Identifying Relationship

A programming problem is resolved by exactly one solution.

A solution resolves exactly one programming problem.

7.2.14- Submits:

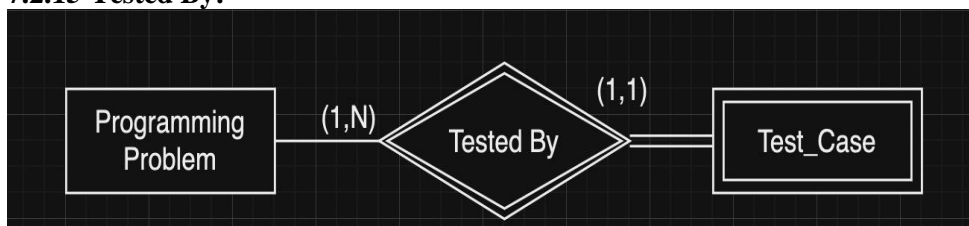


Relationship Type: Ternary

A user submits many submissions for many programming problems.

A submission is submitted by exactly one user.

7.2.15-Tested By:

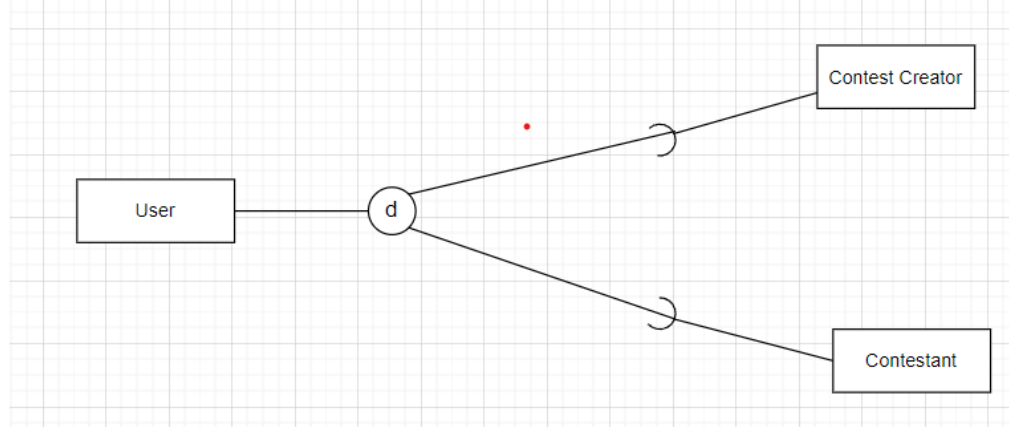


Relationship Type: Binary - Identifying Relationship

A Programming problem is tested by at least one test case.

A test case tests only one programming problem.

7.2.16-Inheritance 1:

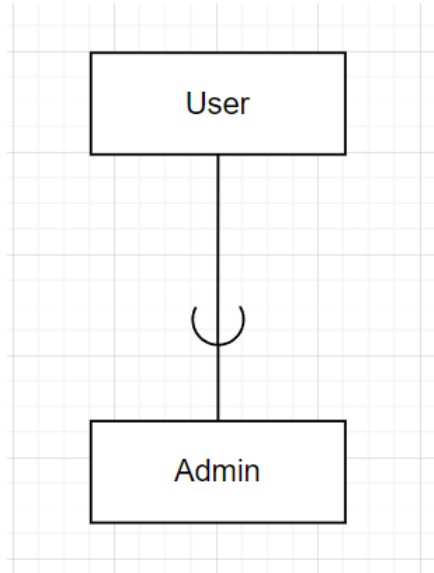


Parent: User

Children: Contestant & Contest Creator

Inheritance Type: Disjoint. A user can either be a contest creator or a contestant, but not both.

7.2.17-Inheritance 2:



Parent: User

Children: Admin

An admin is a user with extra authority and responsibilities.

8 – ER to Relational Mapping:

8.1- Step 1 – Mapping Strong Entity Types

8.1.1-User:

User						
<u>username</u>	email	password	firstName	lastName	Country	registrationDate

Primary Key: username

Foreign Keys: None

8.1.2-Contest:

Contest

<u>roundNumber</u>	name	division	startTimestamp	length	description
--------------------	------	----------	----------------	--------	-------------

Primary Key: roundNumber

Foreign Keys: None

8.1.3-Programming Problem:

ProgrammingProblem

<u>problemID</u>	difficultyLevel	title	description	timeLimit	memoryLimit	<i>roundNumber</i>
------------------	-----------------	-------	-------------	-----------	-------------	--------------------

Primary Key: problemID

Foreign Keys: -roundNumber foreign key references roundNumber primary key in Contest relation.

8.1.4-Submission:

Submission

<u>submissionID</u>	programmingLanguge	sourceCode	verdict	timestamp
---------------------	--------------------	------------	---------	-----------

Primary Key: submissionID

Foreign Keys: -None

8.1.4-Hack:

Hack

<u>hackID</u>	test	<i>defender</i>	verdict	timestamp
---------------	------	-----------------	---------	-----------

Primary Key: hackID

Foreign Keys: -defender foreign key references contestantUsername primary key in Contestant relation.

8.1.5-Blog Entry:

BlogEntry

<u>blogEntryID</u>	<i>username</i>	conetent	timestamp	votes
--------------------	-----------------	----------	-----------	-------

Primary Key: blogEntryID

Foreign Keys: -username foreign key references username primary key in User relation.

8.1.5-Announcement:

Announcement

<u>announcementID</u>	<i>username</i>	language	content	timestamp
-----------------------	-----------------	----------	---------	-----------

Primary Key: announcementID

Foreign Keys: -username foreign key references username primary key in User relation.

8.1.6-Battle:

Battle

<u>battleID</u>	startTimestamp	endTimestamp
-----------------	----------------	--------------

Primary Key: battleID

Foreign Keys: -None

8.2 – Step 2 – Mapping of Weak Entity Types

8.2.1-Solution:

Solution

<u>problemID</u>	<u>solutionID</u>	sourceCode	explanation
------------------	-------------------	------------	-------------

Primary Key: combination of solutionID (partial key of the weak entity type Solution) and problemID primary of the owner entity Programming Problem)

Foreign Keys: - problemID foreign key references problemID primary key in Programming Problem relation.

8.2.2-Test Case:

TestCase

<u>problemID</u>	<u>testCaseNumber</u>	input	expectedOutput
------------------	-----------------------	-------	----------------

Primary Key: combination of testCaseNumber (partial key of the weak entity type TestCase) and problemID (primary of the owner entity Programming Problem)

Foreign Keys: - problemID foreign key references problemID primary key in Programming Problem relation.

8.3 – Step 3 – Mapping of Binary 1:1 Relationship Types

8.3.1-Resolved_By:

Solution

<u>problemID</u>	<u>solutionID</u>	sourceCode	explanation
------------------	-------------------	------------	-------------

1:1 Relationship between Programming Problem and Solution entity types.

Include primary key problemID of Programming Problem relation as a foreign key in Solution relation.

8.4 – Step 4 – Mapping of Binary 1:N Relationship Types

8.4.1-Tested_By:

TestCase

<u>problemID</u>	<u>testCaseNumber</u>	input	expectedOutput
------------------	-----------------------	-------	----------------

1:N Relationship between Programming Problem and Test_Case entity types.

TestCase relation represents Test_Case entity type which participates at the N-side of the Tested_By relationship.

Include primary key problemID of ProgrammingProblem relation as a foreign key in TestCase relation.

8.4.2-Contains:

ProgrammingProblem

<u>problemID</u>	difficultyLevel	title	description	timeLimit	memoryLimit	<i>roundNumber</i>
------------------	-----------------	-------	-------------	-----------	-------------	--------------------

1:N Relationship between Programming Problem and Contest entity types.

ProgrammingProblem relation represents Programming Problem entity type which participates at the N-side of the Tested_By relationship.

Include primary key roundNumber of Contest relation as a foreign key in ProgrammingProblem relation.

8.4.3-Posts:

BlogEntry

<u>blogEntryID</u>	<i>username</i>	conetent	timestamp	votes
--------------------	-----------------	----------	-----------	-------

1:N Relationship between User and Blog_Entry entity types.

BlogEntry relation represents Blog_Entry entity type which participates at the N-side of the Tested_By relationship.

Include primary key username of User relation as a foreign key in BlogEntry relation.

8.4.4-Announces:

Announcement

<u>announcementID</u>	<i>username</i>	<i>language</i>	<i>content</i>	<i>timestamp</i>
-----------------------	-----------------	-----------------	----------------	------------------

1:N Relationship between User and Announcement entity types.

Announcement relation represents Announcement entity type which participates at the N-side of the Tested_By relationship.

Include primary key username of User relation as a foreign key in Announcement relation.

8.5- Step 5 – Mapping of Binary M:N Relationship Types

8.5.1-Befriends:

Befriends

<u><i>friend1Username</i></u>	<u><i>friend2Username</i></u>	<i>timestamp</i>
-------------------------------	-------------------------------	------------------

M:N Recursive Relationship between 2 Users

Relation Befriends is created with the following:

Primary Key: combination of friend1username and friend2username, the 2 usernames (primary keys) of the 2 User entities participating in the Befriends relationship.

Foreign Keys: -friend1Username references username primary key in User relation.

-friend2Username references username primary key in User relation.

Timestamp of befriends relationship is also recorded.

8.5.2-Messages:

Messages

<u><i>senderUsername</i></u>	<u><i>receiverUsername</i></u>	<i>content</i>	<i>timestamp</i>
------------------------------	--------------------------------	----------------	------------------

M:N Recursive Relationship between 2 Users

Relation Messages is created with the following:

Primary Key: combination of senderUsername and receiverUsername, the 2 usernames (primary keys) of the 2 User entities participating in the Messages relationship.

Foreign Keys: - senderUsername references username primary key in User relation.

- receiverUsername references username primary key in User relation.

Timestamp and content of message relationship are also recorded.

8.5.3-Comments:

Comments				
<u>blogEntryID</u>	<u>username</u>	content	timestamp	votes

M:N Relationship between User and BlogEntry

Relation Comments is created with the following:

Primary Key: combination of blogEntryID and username, the primary keys of the BlogEntry and User entities participating in Comments relationship.

Foreign Keys: - blogEntryID references blogEntryID primary key in BlogEntry relation.

- username references username primary key in User relation.

Content, timestamp and votes of the comment relationship are also recorded.

8.5.4-Joins:

Joins		
<u>username</u>	<u>battleID</u>	isWinner

M:N Relationship between User and Battle

Relation Joins is created with the following:

Primary Key: combination of battleID and username, the primary keys of the Battle and User entities participating in Joins relationship.

Foreign Keys: - battleID references battleID primary key in Battle relation.

- username references username primary key in User relation.

isWinner boolean statues of the joins relationship is also recorded.

8.5.5-ConsistsOf:

ConsistsOf	
<u>battleID</u>	<u>problemID</u>

M:N Relationship between Programming Problem and Battle

Relation ConsistsOf is created with the following:

Primary Key: combination of battleID and problemID, the primary keys of the Battle and Programming Problem entities participating in ConsistsOf relationship.

Foreign Keys: - battleID references battleID primary key in Battle relation.

- problemID references problemID primary key in Programming Problem relation.

8.5.6-Organizes:

Organizes

<u>creatorUsername</u>	<u>roundNumber</u>
------------------------	--------------------

M:N Relationship between ContestCreator and Contest

Relation Organizes is created with the following:

Primary Key: combination of creatorUsername and roundNumber, the primary keys of the ContestCreator and Contest entities participating in Organizes relationship.

Foreign Keys: - creatorUsername references creatorUsername primary key in ContestCreator relation.

- roundNumber references roundNumber primary key in Contest relation.

8.5.7- CompetesIn:

CompetesIn

<u>contestantUsername</u>	<u>roundNumber</u>	score	rank	timestampOfRegistration
---------------------------	--------------------	-------	------	-------------------------

M:N Relationship between Contestant and Contest

Relation CompetesIn is created with the following:

Primary Key: combination of contestantUsername and roundNumber, the primary keys of the Contestant and Contest entities participating in CompetesIn relationship.

Foreign Keys: - contestantUsername references contestantUsername primary key in Contestant relation.

- roundNumber references roundNumber primary key in Contest relation.

8.6- Step 6 – Mapping of multivalued attributes

8.6.1- Tag:

Tag

<u>problemID</u>	<u>tag</u>
------------------	------------

Multivalued attribute of Programming Problem entity.

Relation Tag is created with the following:

Primary Key: combination of problemID which is the primary key of the Programming Problem relation and tag.

Foreign Keys: - problemID references problemID primary key in Programming Problem relation.

8.6.2- Responsibilities:

Responsibilities	
<u>adminUsername</u>	<u>responsibility</u>

Multivalued attribute of Admin entity.

Relation Responsibilities is created with the following:

Primary Key: combination of adminUsername which is the primary key of the Admin relation and responsibility.

Foreign Keys: - adminUsername references adminUsername primary key in Admin relation.

8.6.3- Hashtags:

Hashtags	
<u>blogEntryID</u>	<u>tag</u>

Multivalued attribute of BlogEntry entity.

Relation Hashtags is created with the following:

Primary Key: combination of blogEntryID which is the primary key of the BlogEntry relation and tag.

Foreign Keys: - blogEntryID references blogEntryID primary key in BlogEntry relation.

8.7- Step 7 – Mapping of N-ary attributes

8.7.1- Hacks:

Hacks		
<u>contestantUsername</u>	<u>submissionID</u>	<u>hackID</u>

Ternary relationship between Contestant, Submission, & Hack.

Relation Hacks is created as follows:

Primary Key: combination of contestantUsername which is the primary key of the Contestant relation, submissionId which is the primary key of the Submission relation, and hackID which is the primary key of the Hack relation.

Foreign Keys: - contestantUsername references contestantUsername primary key in Contestant relation.
- submissionId references submissionId primary key in Submission relation.
- hackID references hackID primary key in Hack relation.

8.7.2- Submits:

Submits

<u>username</u>	<u>submissionID</u>	<u>problemID</u>
-----------------	---------------------	------------------

Ternary relationship between User, Submission, & Programming Problem.

Relation Submits is created as follows:

Primary Key: combination of username which is the primary key of the User relation, submissionId which is the primary key of the Submission relation, and problemID which is the primary key of the ProgrammingProblem relation.

Foreign Keys: - username references username primary key in User relation.
- submissionId references submissionId primary key in Submission relation.
- problemID references problemID primary key in ProgrammingProblem relation.

8.7.3- Notifies:

Notifies

<u>creatorUsername</u>	<u>contestantUsername</u>	<u>problemID</u>	notificationContent
------------------------	---------------------------	------------------	---------------------

Ternary relationship between ContestCreator, Contestant, & Programming Problem.

Relation Submits is created as follows:

Primary Key: combination of creatorUsername which is the primary key of the ContestCreator relation, contestantUsername which is the primary key of the Contestant relation, and problemID which is the primary key of the ProgrammingProblem relation.

Foreign Keys: - creatorUsername references creatorUsername primary key in ContestCreator relation.
- contestantUsername references contestantUsername primary key in Contestant relation.
- problemID references problemID primary key in ProgrammingProblem relation.

The notification content is also recorded.

8.8- Step 8 – Mapping aggregation, specialization/generalization relationships

8.8.1- Admin:

Admin		
<u>username</u>	role	contributionScore

Admin inherits from User.

Primary Key: -username (inherited username from User)

Foreign Keys: - username references username primary key in User relation.

The role and contribution score are also recorded.

8.8.1- ContestCreator and Contestants:

Disjoint inheritance from User.

8.8.1.1. ContestCreator

ContestCreator	
<u>creatorUsername</u>	assessmentScore

Primary Key: -creatorUsername (inherited username from User)

Foreign Keys: - creatorUsername references username primary key in User relation.

The assessment score is also recorded.

8.8.1.1. Contestant

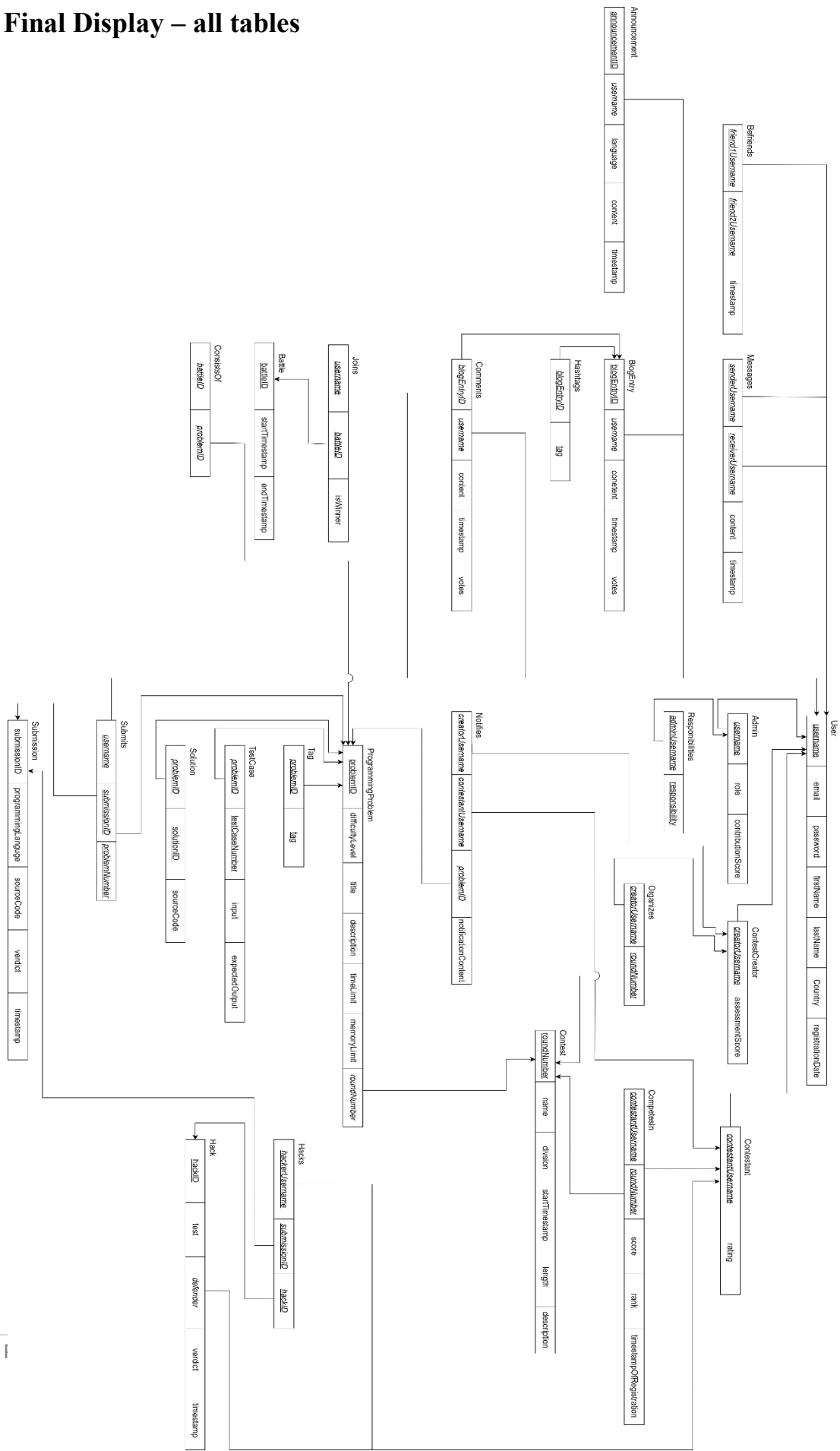
Contestant	
<u>contestantUsername</u>	rating

Primary Key: -contestantUsername (inherited username from User)

Foreign Keys: - contestantUsername references username primary key in User relation.

The rating is also recorded.

9 – Final Display – all tables



10-Tabls' States

User						
Username	email	password	firstName	lastName	Country	registrationDate
Khaled47	Khaled47@gmail.com	DamanWizDaPlan	Khaled	Charaf	Lebanon	1/1/2024
ay_sharaf	ay_sharaf@gmail.com	ay_s098	Ayman	Charaf	Lebanon	1/2/2024
AliFarhat	AliFarhat@gmail.com	Alifa987	Ali	Farhat	Lebanon	1/3/2024
Dankosay	Dankosay@gmail.com	Dana789	Dana	Kossaybati	Lebanon	1/4/2024
Brokie	Brokie@gmail.com	Brokie654	Joseph	Aoun	Lebanon	1/5/2024
Tourist	Tourist@gmail.com	Tourist456	Gennady	Korotkevich	Belarus	1/6/2024
jiangly	jiangly@gmail.com	jiangly321	Lingyu	Jiang	China	1/7/2024
Benq	Benq@gmail.com	Benq123	Benjamin	Qi	United States	1/8/2024
HussBak	HussBak@gmail.com	DakowdezTheBest	Hussein	Bakri	Lebanon	1/9/2024
ImanGha	ImanGha@gmail.com	KhaledIsMyFavorite	Iman	Ghalayini	Lebanon	1/10/2024

Contest					
roundNumber	Division	name	description	startTimestamp	length
1	3	PinelyRounds	In pinely, we believe. Yes you know it, winner winner chicken dinner.	2023-10-18 09:23:45	2:00:00
2	3	HarbourSpace	Don't you just love HarbourSpace university. Earn a free scholarship by placing top 1 on our competition	2023-08-15 14:37:22	2:30:00
3	1	AUB CUP	DA BEST university in the region, You know it. Earn a free scholarship by placing top 10 on our coding competition	2023-05-27 19:51:10	2:00:00
4	1	Face CUP	Get a free job interver at facebook by placing top 50 on our hacker cup	2023-04-02 08:12:30	2:00:00
5	1	MIT CUP	The legendary umass competition is up. Place top one to get accepted at the best university in the world	2023-11-09 17:29:55	2:30:00
6	3	Educational Round	Educational Round on Dakowdez website	2023-06-30 11:45:18	2:30:00
7	3	Educational Round	Educational Round on Dakowdez website	2023-02-14 22:03:40	3:30:00
8	2	Hussein Bakri CUP	The best database teacher here. Place top 5 on my coding contest and you will get 2500\$ in my database company !!! Capiche ?	2023-07-20 05:55:08	3:30:00
9	1	Iman Ghalayini CUP	Place top 10 in my hacker cup and you will receive a free skip lab offer on Bakri's next 433 database class	2023-09-03 13:18:57	4:00:00
10	1	KhaledTheBest CUP	I mean there are no prizes in my competition because we all know Khaled is top 1 :) !	2023-12-25 00:00:00	3:30:00

Befriends		
friend1Username	friend2Username	TimeStam
Khaled47	AliFarhat	2023-03-12 16:40:25
HussBak	ImanGha	2023-06-19 03:27:15
Khaled47	HussBak	2023-10-05 12:08:38
Khaled47	ImanGha	2023-04-30 08:56:50
ay_sharaf	Dankosay	2023-07-14 20:17:09
Brokie	Tourist	2023-09-28 06:45:33
HussBak	Khaled47	2023-01-21 14:59:42
ImanGha	Khaled47	2023-08-08 10:30:07
HussBak	ay_sharaf	2023-11-11 19:22:55
HussBak	Dankosay	2023-05-07 02:33:11

Message			
senderUsername	recieverUsername	Content	TimeStamp
Khaled47	AliFarhat	Yo Bestie Wassup	2023-08-21 17:12:04
HussBak	ImanGha	This student Khaled of mine is t	2023-02-03 09:47:32
Khaled47	HussBak	Hello Professor Bakri	2023-06-13 22:30:55
Khaled47	ImanGha	Hello Miss Iman	2023-10-29 14:25:19
ay_sharaf	Dankosay	yo wassup	2023-04-15 06:18:40
Brokie	Tourist	OMG Tourist wassup	2023-12-07 11:55:03
HussBak	Khaled47	Good job on that 1000 rating ma	2023-03-26 19:37:28
ImanGha	Khaled47	Hello Khaled, Congrats on your n	2023-09-17 03:03:11
HussBak	ay_sharaf	Congrats on getting that 1 point r	2023-07-05 15:49:29
HussBak	Dankosay	Comeon Dana, you should partic	2023-01-01 00:00:00

Submission						
SubmissionID	programmingLangua	sourceCode	verdict	timeStamp	Username	problemNumber
1	C++	#include <iostream>	accepted	2023-04-20 08:14:52	Khaled47	1
2	C++	#include <iostream>	Wrong Answer	2023-07-11 17:26:39	ay_sharaf	2
3	C++	#include <iostream>	Compilation Error	2023-11-28 03:59:18	AliFarhat	3
4	Python	#!/usr/bin/env python3	Time Limit Exceed	2023-05-14 12:33:27	Dankosay	4
5	C++	#include <iostream>	accepted	2023-09-05 21:45:01	Brokie	5
6	C++	#include <iostream>	accepted	2023-01-19 06:02:45	Tourist	6
7	C++	#include <iostream>	Memory Limit Exceeded	2023-06-07 14:57:10	jiangly	7
8	C++	#include <iostream>	Wrong Answer	2023-10-02 00:25:33	Benq	8
9	Java	public class AddOneToTwo {	Wrong Answer	2023-03-23 10:09:47	HussBak	9
10	Python	import random	accepted	2023-08-16 19:41:54	ImanGha	10
11	C++	#include <iostream>	accepted	2023-03-23 10:09:47	Khaled47	11

Problems							
problemNumber	Tag	difficultyLevel	title	description	timeLimit	memoryLimit	roundNumber
1	A	800	Sum of Three	Monocarp has an	1	256	1
2	B	1400	Three Threadlets	Problem Statement	1	512	1
3	C	2000	Decreasing String	Problem Statement	2	128	1
4	A	1000	Rigged!	Problem Statement	3	256	2
5	B	1500	Aleksa and Stack	Problem Statement	2	512	2
6	C	2200	Jellyfish and EVA	Problem Statement	1	128	2
7	A	500	Short Sort	Problem Statement	2	256	3
8	B	1000	Good Kid	Slavic is preparing a	3	512	3
9	C	1500	Non-coprime Split	Problem Statement	4	128	3
10	A	1200	MEXanized Array	Problem Statement	3	256	4
11	B	1600	Friendly Arrays	Problem Statement	1	512	4
12	C	2200	Salyg1n and the MEX Game	Problem Statement	1	128	4
13	A	1600	Ambitious Kid	Problem Statement	2	256	5
14	B	2000	XOR Palindromes	Problem Statement	3	512	5
15	C	2400	Word on the Paper	Problem Statement	2	128	5
16	A	750	To My Critics	Problem Statement	1	256	6
17	B	1250	Ten Words of Wisdom	Problem Statement	2	512	6
18	C	1700	Tiles Comeback	Problem Statement	3	128	6
19	A	750	Morning Sandwich	Problem Statement	4	256	7
20	B	1200	Come Together	Problem Statement	3	512	7
21	C	1600	Tenzing and Balls	Problem Statement	1	128	7
22	A	950	Unit Array	Problem Statement	1	256	8
23	B	1350	Maximum Strength	Problem Statement	2	512	8
24	C	1700	Travel Plan	Problem Statement	3	128	8
25	A	1850	Musical Puzzle	Problem Statement	2	256	9
26	B	2000	Rudolph and Tic-Tac-Toe	Problem Statement	1	512	9
27	C	2600	Rudolf and the Another Co	Problem Statement	2	128	9
28	A	2100	Ian Visits Mary	Problem Statement	3	256	10
29	B	2500	Grid Reconstruction	Problem Statement	4	512	10
30	C	3000	Similar Polynomials	Problem Statement	3	128	10

Solution

problemID	SolutionID	Explanation	SourceCode
1	1	Greedy	#include <iostream>
2	2	Dynamic Programming	#include <iostream>
3	3	Two Pointers	#include <iostream>
4	4	Divide and Conquer	#!/usr/bin/env python3
5	5	Heap Data Structure	#include <iostream>
6	6	Set Data Structure	#include <iostream>
7	7	queue Data Structure	#include <iostream>
8	8	Graph Traversal	#include <iostream>
9	9	Fast Fourier Transform	public class AddOneToTwo {
10	10	Combinatorics	import random
11	11	Greedy	#include <iostream>
12	12	Dynamic Programming	#include <iostream>
13	13	Two Pointers	#include <iostream>
14	14	Divide and Conquer	#include <iostream>
15	15	Heap Data Structure	#!/usr/bin/env python3
16	16	Set Data Structure	#include <iostream>
17	17	queue Data Structure	#include <iostream>
18	18	Graph Traversal	#include <iostream>
19	19	Fast Fourier Transform	#include <iostream>
20	20	Combinatorics	public class AddOneToTwo {
21	21	Greedy	import random
22	22	Dynamic Programming	#include <iostream>
23	23	Two Pointers	#include <iostream>
24	24	Divide and Conquer	#include <iostream>
25	25	Heap Data Structure	#include <iostream>
26	26	Set Data Structure	#include <iostream>
27	27	queue Data Structure	#!/usr/bin/env python3
28	28	Graph Traversal	#include <iostream>
29	29	Fast Fourier Transform	#include <iostream>
30	30	Combinatorics	#include <iostream>

Battle		
BattleID	startTimeStamp	endTimeStamp
1	2023-12-12 19:48:26	2023-02-28 16:45:20
2	2023-06-05 11:09:33	2023-09-11 10:18:35
3	2023-03-28 08:37:14	2023-06-24 03:54:42
4	2023-09-19 16:20:57	2023-12-05 21:09:57
5	2023-01-15 04:52:42	2023-04-16 14:27:10
6	2023-07-22 14:14:30	2023-10-20 06:33:25
7	2023-10-31 00:00:00	2023-07-09 19:46:30
8	2023-04-08 05:26:03	2023-03-13 12:01:45

Joins		
Username	battleID	isWinner
Khaled47	1	TRUE
Dankosay	1	FALSE
Khaled47	2	TRUE
ay_sharaf	2	FALSE
Khaled47	3	FALSE
HussBak	3	TRUE
HussBak	4	TRUE
ImanGha	4	FALSE
ay_sharaf	5	FALSE
Dankosay	5	TRUE

Consists Of	
battleID	problemID
1	1
1	2
2	3
2	4
3	5
3	6
4	7
4	8
5	9
5	10

Hashtags	
blogEntryID	Hashtags
1	pro
2	coding
3	AliFarhat
4	HIIII
5	Dynamic
6	Competitive
7	Dakowdez
8	Dakowdez
9	Khaled47
9	The Best
10	Dakowdez

Test Cases			
problemID	testCaseNumber	input	expectedOutput
1	1	10	13
2	1	11	14
3	1	12	15
4	1	13	16
5	1	14	17
6	1	15	18
7	1	16	19
8	1	17	20
9	1	18	21
10	1	19	22
11	1	20	23
12	1	21	24
13	1	22	25
14	1	23	26
15	1	24	27
16	1	25	28
17	1	26	29
18	1	27	30
19	1	28	31
20	1	29	32
21	1	30	33
22	1	31	34
23	1	32	35
24	1	33	36
25	1	34	37
26	1	35	38
27	1	36	39
28	1	37	40
29	1	38	41
30	1	39	42

Tag	
problemID	tag
1	Math
1	DP
2	Math
2	DP
2	Data Structures
6	Math
7	DP
8	Data Structures
9	Bitwise XOR
9	Bitwise XOR
10	DakoDPwdez

Contestant	
Username	Rating
Tourist	2123
jiangly	2000
Benq	1750
ay_sharaf	1500
AliFarhat	1250
Khaled47	1000
Dankosay	0

Announcements				
announcementID	username	language	content	timestamp
1	Khaled47	English	Hello Dacowders...	2023-03-02 08:15:30
2	ay_sharaf	English	Hello Dacowders...	2023-07-19 14:46:42
3	AliFarhat	English	Hello Dacowders...	2023-11-10 21:28:55
4	Dankosay	English	Hello Dacowders...	2023-05-28 03:57:10
5	Brokie	English	Hello Dacowders...	2023-09-15 10:33:22
6	Tourist	English	Hello Dacowders...	2023-02-07 17:12:41
7	jiangly	English	Hello Dacowders...	2023-06-14 00:00:00
8	Benq	English	Hello Dacowders...	2023-10-01 06:45:18
9	HussBak	English	Hello Dacowders...	2023-04-25 13:22:29
10	ImanGha	English	Hello Dacowders...	2023-08-31 19:59:54

BlogEntry				
blogEntryID	username	content	timestamp	votes
1	Khaled47	How to prevent hack	2023-10-18 08:45:23	5
2	ay_sharaf	Plan to get 2000 rating	2023-09-25 16:30:10	4
3	AliFarhat	I am a pro player	2023-08-12 12:15:47	6
4	Dankosay	HIIIIIIII Wassuppp W	2023-07-03 19:55:33	2
5	Brokie	Eyo wassup today w	2022-06-14 05:20:55	1
6	Tourist	Welcome to my blog	2023-05-29 14:10:38	5
7	jiangly	Welcome to my blog	2023-04-22 03:05:12	4
8	Benq	Welcome to my blog	2023-03-17 09:28:40	3
9	HussBak	This is the best code	2023-02-09 18:40:25	3
10	ImanGha	This is the best code	2023-01-04 20:11:17	2

Comments				
blogEntryID	username	content	timeStamp	votes
1	Khaled47	So just don't hack	2023-12-15 14:22:30	3
2	ay_sharaf	Practice !!!	2023-11-20 10:55:45	6
3	AliFarhat	Yeah so don't try	2023-10-07 17:30:18	5
4	Dankosay	HIII	2023-09-03 22:40:55	2
5	Brokie	so we gonna talk abt	2023-08-28 08:15:27	3
6	Tourist	Welcome I Say	2023-07-13 19:48:10	5
7	jiangly	Welcome I Say	2023-06-08 03:10:56	5
8	Benq	Welcome I Say	2023-05-11 12:37:42	6
9	HussBak	you guys did an am	2023-04-04 06:25:33	3
10	ImanGha	you guys did an am	2023-03-01 23:50:20	2

Admin		
username	role	contributionScore
HussBak	superAdmin	2394
ImanGha	moderator	1001
Khaled47	supportAdmin	1000

Responsibilities	
adminUsername	Responsibility
HussBak	Assesses Contest Creators
ImanGha	User Management
Khaled47	Technical Support

Notifies				
creatorUsername	contestantUsername	problemID	notificationContent	
HussBak	Khaled47	2	There is an error. We meant that exactly one ball should be taken out not atleast	
HussBak	AliFarhat	2	There is an error. We meant that exactly one ball should be taken out not atleast	
HussBak	ay_sharaf	2	There is an error. We meant that exactly one ball should be taken out not atleast	
ImanGha	Dankosay	5	Read the problem statement well. Atmost one potato should be eaten	
ImanGha	Tourist	5	Read the problem statement well. Atmost one potato should be eaten	

ContestCreator		Organizes	
creatorUsername	assessmentScore	creatorUsername	roundNumber
HussBak	2445	HussBak	1
ImanGha	2543	ImanGha	2
		HussBak	3
		ImanGha	4
		HussBak	5
		ImanGha	6
		HussBak	7
		ImanGha	8
		HussBak	9
		ImanGha	10
Hacks			
hackerUsername	submissionID	hackID	
ay_sharaf	1	1	
Khaled47	2	2	
Dankosay	3	3	

CompetesIn				
contestantUsername	roundNumber	score	rank	timestampOfRegistration
Tourist	2	1000	56	1/1/2024-1:00:00
jiangly	3	800	34	1/2/2024-1:00:00
Benq	3	900	23	1/3/2024-1:00:00
ay_sharaf	1	700	4355	1/4/2024-1:00:00
AliFarhat	1	800	1232	1/5/2024-1:00:00
Khaled47	1	800	4532	1/6/2024-1:00:00
Khaled47	4	500	2342	1/7/2024-1:00:00
Dankosay	2	0	8532	1/8/2024-1:00:00
hack				
hackID	test	defender	verdict	timestamp
1	65	Khaled47	fail	1/1/2024-5:00:00
2	35	ay_sharaf	fail	1/2/2024-6:00:00
3	24	AliFarhat	fail	1/3/2024-6:30:00

Submits			
usernames	submissionID	problemID	
Khaled47	1	2	
Dankosay	1	4	
Khaled47	2	8	
ay_sharaf	2	1	
Khaled47	3	2	
HussBak	3	4	
HussBak	4	6	

11-Conclusion

After conducting a comprehensive analysis of our database structure for Dakowdas.com, we take pride in presenting our initial Entity-Relationship (ER) draft as well as its mapping. While we acknowledge the need for some future refinements and adjustments, we are pleased with the organized and coherent framework that our first draft represents. This draft lays a strong foundation for our coding website's database, reflecting our commitment to creating a robust and scalable platform. Our mission is to scale this website to become a front-tier competitive programming platform, and this initial draft is a significant stride towards achieving that goal. As we progress in our development journey, we will continue to refine and optimize the database to ensure it aligns seamlessly with our evolving needs and objectives, bolstering our vision to establish Dakowdas.com as a leading force in the competitive programming arena.

12- Instructor Feedback