GUNJAN GUPTA - 2019111035
NIKHIL BISHNOI - 2019114021
MUSTAFA SIDDIQUI – 2018101128

PROJECT PHASE #1

CAPTURE THE FLAG

Introduction:

This is an SRS Documentation for a DBMS which keeps track of a mini-world for any Attack-Defend Style CTF event, handling most of the major functions.

Purpose:

The following document describes a database requirements for an Attack-Defend CTF Event. As an overview it allows the creation, maintenance and modifying of details about the team and players along with the Rounds played over the course of the event.

Users:

Spectators or enthusiastic coders of the event can refer to the DB for current stats and team coaches can gauge the performance of individual members of their teams as well as opponents they need to be wary of.

What users do with DBMS:

This DBMS allows efficient and reliable retrieval of information by the users. Finding out round statistics and analysing the performance of different coders. This can be helpful to book tickets in advance for future match-ups by looking up the day they are going to be played and the venue. Also the user can follow his favourite coders' performances in the Event. It also helps to readily obtain the best performers and performances of the event.

Entities:

1. *Team:*

- Team_ID *(KEY ATTIBUTE)*
- Name *(KEY ATTIBUTE)*
- Region_ID *(FOREIGN KEY)*
- Coach *(COMPOSITE ATTRIBUTE)*
- Members *(MULTIVALUED ATTRIBUTE)*

2. *Coder:*

There are 2 sub-classes depending on coder type:

- Attacker
- Defender

- Username *(KEY ATTIBUTE)*
- Team_ID *(FOREIGN KEY)*
- Name(first + last) *(COMPOSITE ATTRIBUTE)*
- Captain_Username
- Date of Birth(date + month + year) *(COMPOSITE ATTRIBUTE)*
- Age *(DERIVED ATTRIBUTE)*

3. *Match:*

- Team1_ID *(FOREIGN KEY)*
- Team2_ID *(FOREIGN KEY)*
- Day_number
- Won_By
- MVP
- Venue_ID *(FOREIGN KEY)*

4. *Attacker:*

- Attacker_ID *(KEY ATTIBUTE)*
- No_participated *(CAN BE A NULL ATTIBUTE)*
- Flags_captured

- Traps_Triggered
- ATT_Success_Rate

5. *Defender:*
   - Defender_ID *(KEY ATTIBUTE)*
   - No_participated *(CAN BE A NULL ATTIBUTE)*
   - Succesful_Traps
   - Unsuccesful_Traps *(DERIVED ATTRIBUTE)*
   - DEFF_Success_Rate
6. *Region:*
   - Region_ID *(KEY ATTIBUTE)*
   - Region_Name
7. *Venue:*
   - Venue_ID *(KEY ATTIBUTE)*
   - Venue_Name *(COMPOSITE ATTRIBUTE)*
8. *Coder type:*
   - Coder_Username *(FOREIGN KEY)*
   - Type *(MULTIVALUED ATTRIBUTE)*
9. *Schedule:*
   - Team1_ID *(FOREIGN KEY)*
   - Team2_ID *(FOREIGN KEY)*
   - Day_number

WEAK-ENTITY:

- Schedule: PARTIAL KEY(Team1_ID + Team2_ID + Day_number)
- Coder Type: PARTIAL KEY(Username + Type)
- Match: PARTIAL KEY(Team1_ID + Team2_ID + Day_number)

Relationships:

1. Team REPRESENTS Region : 1-1 Relationship

(Binary Relationship)

➔ A Team will represent a particular region only.

2. Coder PLAYS FOR Team : N-1 Relationship

   (Binary Relationship)

   ➔ *A Coder can only play for a single team.*

3. Coder CAPTION OF Coder : 1-N Relationship

   (Unary or Recursive Relationship)

   ➔ *A Coder can become caption of a n number of Coders*

4. Team PLAYS Match : N-M Relationship

   (Binary Relationship)

   ➔ Any Team can play any number of matches.

5. Match PLAYED ON Venue : (0,3)-(1,1)Relationship

   (Binary Relationship)

   ➔ *Minimum number of matches can be played at particular Venue is 0 and maximum can be 3.*

n>3 Relationships:

1. At a Venue A, a certain Match B is played with one of the playing Teams being C, represting a Region D, having a coder E.

   [Venue]> [Match]-> [Teams] -> [Region]-> [Coder]

Functional Requirements:

1. *Modifications:*

- Insert:
  - Insertion of a New coder if he is selected into the team
  - Insertion of a New Team if it qualifies for the Event
  - Insertion of a New Attacker record if a Defender starts Attacking
  - Insertion of a New Defender record if a Attacker starts Defending
  - Insertion of a New Venue if a new stadium is constructed

- Inserting the MVP entry after a match up is over
- Inserting the Won By entry after the match is over

- Delete:
  - Deleting a Venue record if it is demolished.
  - Deleting a Team if it disqualifies.

- Update:
  - Updating the Records of a Defender after every match
  - Updating the Records of an Attacker after every match
  - Update the Age of a Player to -1 if he dies due to unforeseen circumstances

2. *Retrievals:*

- *Selection:*
  - Select Coders by Team.
  - Select Defenders and Attackers by no of participations(no_participated).
- *Projection:*
  - Able to search Attackers and Defenders with more than 50% success rate.
- *Aggregate:*
  - Calculate Defender success rate using succesful_traps and unsuccesful traps.
  - Calculate Attacker success rate using flags captured and traps triggered.
- *Search:*
  - Search up coders and team by their name.
- *Analysis:*
  - Finding the Best Attackers of the tournament: Sort the Flags_Captured attribute in the Attacker entity and take the first entry
  - Finding the Best Defender of the tournament: Sort the Succesful_Traps attribute in the Defender entity and take the first entry
  - Finding the Number of Coders who are All Rounders, i.e, both Attackers and Defenders: In the entity Coder_Type we calculate the number of entries which are multi-valued, i.e, have both the types Attacker and Defender assigned to them.
  - Calculating the Number of WIns for each team: We can do this by adding all the times a particular team's id appears in the attribute (Won by) in the Match up enti*ty.*