**T.C.**

**MARMARA UNIVERSITY**

**FACULTY of ENGINEERING**

**COMPUTER ENGINEERING DEPARTMENT**

CSE 3055 – Database Systems Logical Database Design Report  
   
  
Project Name: Football Tournament

**Enes Ülker - 150117502**  
 **Cem Güleç - 150117828**

*01 January 2021*

**Project Description:**

- Aim of this project is to create a simulation of a championship league. In this simulation, there will be a group of football teams and the simulation will show match results and the final table.

- Rules associated with this league is that there will be teams and groups (team and group numbers are tentative). In each group, each team should play with each other, and the first two teams in the group should go to the next round. Every match will be in elimination manner.

- After first step of initial matches resulted, the group matches will be played randomly and there should be a group result table which consisting of number of matches played, scores, goals, and averages.

- At final, for every match played in every step will be showed with their score values.

**Project Scope:**

- By the time we finish this project, we aim to handle all the requirements of database implementation. Beside that, we plan to handle back-end and front-end development.

- For the back-end development part, we will be creating our own API to be communicating with the web interface. This part will process match tables, scores and all the necessary information to the front-end part. Technologies to be used here is: Node.js

- For the front-end development part, as I mentioned above, gathered information from the back-end will be displayed. Technologies to be used here is: Angular, Typescript, HTML and CSS.

**Business process and its definitions:  
Step1:** Determination of groups By the time the new season is opened, draw is being made by the responsible organization. Depending on the results each team is assigned to corresponding group.

**Step2**: Group matches By the rules defined in the section 1.3., each team plays matches for the whole seasons against the teams that are in the same group. Succeeding teams will have the right to play further matches in quarter finals.

**Step3**: Elimination matches This stage starts with quarter finals which contains teams that succeeded in group stage. Every group leader matches randomly with a team that finished the group stage second. After quarter finals remaining teams match in semifinals. At last, remaining two teams match in the finals and tournament concludes. Teams plays two times with each other in every round and loser is out of the tournament.

**Business rules:**

• Number of teams inside the database is determined as 16.

• Number of groups inside the database is determined as 4.

• In each group teams must play with each other at least twice.

• First two teams in each group will be able to go next round.

• After group matches handled next will be in elimination manner.

• Determination of each team’s opponent is done randomly.

• If the points of the two teams are equal, the higher average will be placed higher.

• Match scores can range from 0 to 10

• Average is calculated by goal scored – conceded goal   
  
  
**Other functional and non-functional requirements:**

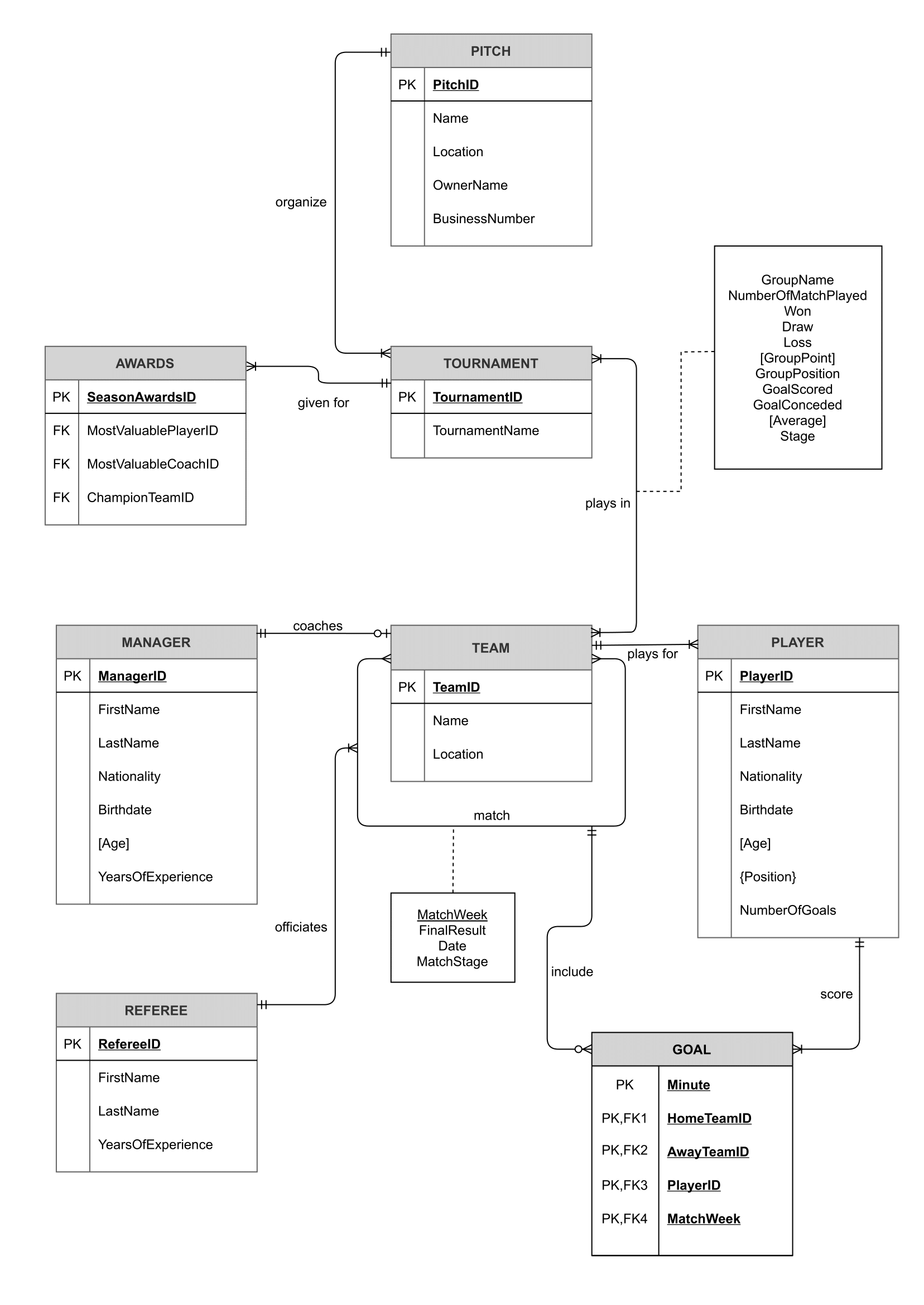
• In each group the two teams with most point is going to advance to elimination round

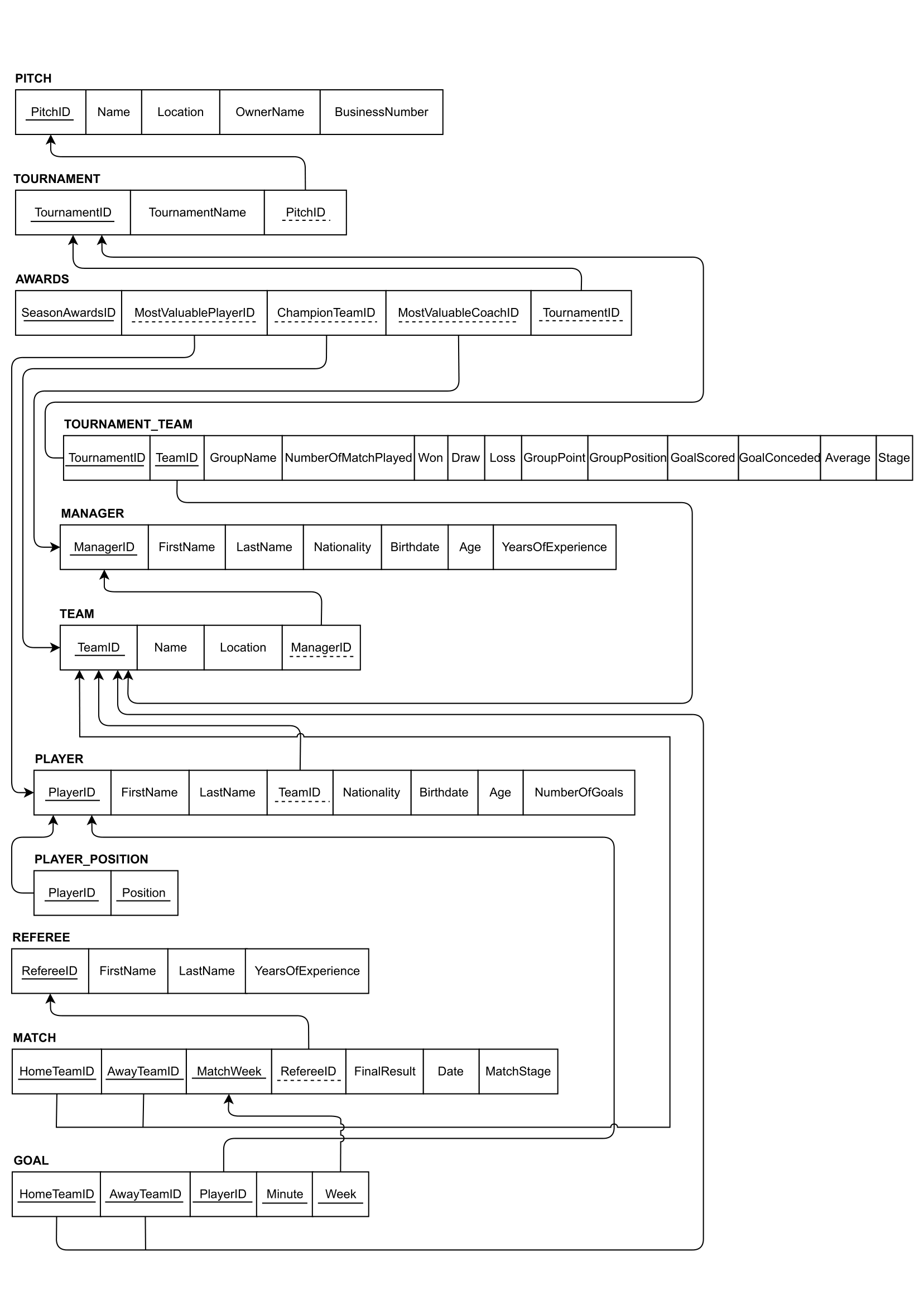
• Elimination rounds have a future of first overtime then penalty shootout in case of total draw with equal scores for each team.

• An API is required for handling communication between backend and frontend

• In the first step group matches will be fetched from the backend via API service.

**B) E-R Diagram of whole datebase**

  
  
  
**Relational Mapping of the Database**



**C) TABLES**

**TABLE AWARDS**

**i)** SeasonAwardsID, ChampionTeamID, MostValuablePlayerID, MostValuableCoachID

and TournamentID

**ii)** This table includes some data about awards given at the end of a tournament.

**iii)** SeasonAwardsID-->**tinyint**

ChampionTeamID-->**tinyint**

MostValuablePlayerID-->**tinyint**

MostValuableCoachID-->**tinyint**

TournamentID-->**tinyint**

**iv)** There is no index for table AWARDS.

SeasonAwardsID-->**Primay Key**

ChampionTeamID-->**Foreign Key** (from table TEAM)

MostValuablePlayerID--> **Foreign Key** (from table PLAYER)

MostValuableCoachID--> **Foreign Key** (from table MANAGER)

TournamentID--> **Foreign Key** (from table TOURNAMENT)

**v)** There is no unique, identity, check constraint, default nor computed columns for table AWARDS

**vi)** No trigger applied.

**TABLE GOAL**

**i)** HomeTeamID, AwayTeamID, PlayerID, Minute, Week

**ii)** This table keeps the records of all goals scored in matches.

**iii)** HomeTeamID-->tinyint

AwayTeamID-->tinyint

PlayerID-->tinyint

Minute-->tinyint

Week-->char(2)

**iv)** There is a index for table GOAL named goal\_information.

HomeTeamID-->Primary Key and Foreign Key (from table TEAM)

AwayTeamID-->Primary Key and Foreign Key (from table TEAM)

PlayerID-->PrimaryKey and Foreign Key (from table PLAYER)

Minute-->Primary Key and Foreign Key (from table MATCH)

Week-->Primary Key and Foreign Key (from table MATCH)

**v)** There is no unique, identity, check constraint, default nor computed columns for table GOAL

**vi)** There is one trigger attached to GOAL table. It is triggered when some row is updated or newly goal information is entered.

**TABLE MANAGER**

**i)** ManagerID, FirstName, LastName, Nationality, Birthdate, Age, YearsOfExperience

**ii)** This table keeps information about managers.

**iii)** ManagerID-->tinyint

FirstName-->nvarchar(20)

LastName-->nvarchar(20)

Nationality-->char(7)

Birthdate-->date

Age--> COMPUTED COLUMN (from Birthdate)

YearsOfExperience--> tinyint

**iv)** There is no index for table MANAGER

ManagerID-->Primary Key

**v)** There is a default value in column NATIONALITY which is ‘Turkish’ and also a computed column AGE (computed from birthdate). There is no uniques, identities, and check constrains in this table

**vi)** No trigger applied.

**TABLE MATCH**

**i)** HomeTeamID, AwayTeamID, Week, RefereeID, Date, FinalResult, MatchStage

**ii)** This table keeps information about matches played in a tournament

**iii)** HomeTeamID-->tinyint

AwayTeamID-->tinyint

Week-->char(2)

RefereeID-->tinyint

Date-->date

FinalResult-->varchar(5)

MatchStage-->varchar(15)

**iv)** There is no index for table MATCH

HomeTeamID-->Primary Key and Foreign Key (from table TEAM)

AwayTeamID-->Primary Key and Foreign Key (from table TEAM)

Week-->Primary Key

**v)** There is no unique, identity, check constraint, default nor computed columns for table MATCH

**vi)** No trigger applied.

**TABLE PITCH**

**i)** PitchID, Name, Location, OwnerName, BusinessNumber

**ii)** This table keeps information about the pitch which organize football tournament.

**iii)** PitchID-->tinyint

Name-->nvarchar(20)

Location-->nvarchar(20)

OwnerName-->nvarchar(25)

BusinessNumber-->char(10)

**iv)** There is no index for table PITCH

PitchID-->Primary Key

**v)** There is a unique in column BUSINESSNUMBER. There is no identity, check constraint, default, computed column for table PITCH

**vi)** No trigger applied.

**TABLE PLAYER**

**i)** PlayerID, FirstName, LastName, TeamID, Nationality, Birthdate, Age, NumberOfGoals

**ii)** This table keeps information about all the players.

**iii)** PlayerID-->tinyint

FirstName-->nvarchar(20)

LastName-->nvarchar(20)

TeamID-->tinyint

Nationality-->varchar(25)

Birthdate-->date

Age-->COMPUTED COLUMN

NumberOfGoals-->tinyint

**iv)** There is a index for table PLAYER named player\_firstname\_lastname.

PlayerID-->Primary Key

TeamID-->Foreign Key

**v)** There is no unique, identity, check constraint, default nor computed columns for table PLAYER.

**vi)** After trigger is triggered from the GOAL table “NumberOfGoals” attribute is updated accordingly to the scenario.

**TABLE PLAYER\_POSITION**

**i)** PlayerID, Position

**ii)** This table keeps information about positions of players played.

**iii)** PlayerID-->tinyint

Position-->varchar(15)

**iv)** There is no index for table PLAYER\_POSITION

PlayerID-->Primary Key and Foreign Key (from table PLAYER)

Position-->Primary Key

**v)** There is no unique, identity, check constraint, default nor computed columns for table PLAYER\_POSITION.

**vi)** No trigger applied.

**TABLE REFEREE**

**i)** RefereeID, FirstName, LastName, YearsOfExperience

**ii)** This table keeps information about referees

**iii)** RefereeID-->tinyint

FirstName-->nvarchar(20)

LastName-->nvarchar(20)

YearsOfExperience-->tinyint

**iv)** There is no index for table REFEREE.

RefereeID-->Primary Key

**v)** There is a check constrain for table REFEREE named Chk\_REFEREE\_YearsOfExperience which determines that YearsOfExperience must be higher than 0. There is no unique, identity, default nor computed column in table REFEREE

**vi)** No trigger applied.

**TABLE TEAM**

**i)** TeamID, Name, Location, ManagerID

**ii)** This table keeps information about teams.

**iii)** TeamID-->tinyint

Name-->nvarchar(20)

Location-->nvarchar(20)

ManagerID-->tinyint

**iv)** There is no index for table TEAM

TeamID-->Primary Key

ManagerID-->Foreign Key

**v)** There is no unique, identity, check constraint, default nor computed columns for table TEAM.

**vi)** No trigger applied.

**TABLE TOURNAMENT**

**i)** TournamentID, TournamentName, PitchID

**ii)** This table keeps information about tournament.

**iii)** TournamentID-->tinyint

TournamentName-->nvarchar(75)

PitchID-->tinyint

**iv)** There is no index for table TOURNAMENT.

TournamentID-->Primary Key

PitchID-->Foreign Key

**v)** There is no unique, identity, check constraint, default nor computed columns for table TOURNAMENT.  
  
**vi)** No trigger applied.

**TABLE TOURNAMENT\_TEAM**

**i)** TournamentID, TeamID, GroupName, NumberOfMatchPlayed, Won, Draw, Loss, GroupPoint, GoalScored, GoalConceded, Average, GroupPosition, Stage

**ii)** This table keeps information for some team statistics in a tournament.

**iii)** TournamentID-->tinyint

TeamID-->tinyint

GroupName-->char(1)

NumberOfMatchPlayed-->tinyint

Won-->tinyint

Draw-->tinyint

Loss-->tinyint

GroupPoint-->COMPUTED COLUMN (from Won\*3 + Draw)

GoalScored-->smallint

GoalConceded-->smallint

Average-->COMPUTED COLUMN (from GoalScored – GoalConceded)

GroupPosition-->char(1)

Stage-->varchar(15)

**iv)** There is no index for table TOURNAMENT\_TEAM

TournamentID-->Primary Key and Foreign Key (from table TOURNAMENT)

TeamID-->Primary Key and Foreign Key (from table TEAM)

**v)** There is two computed colums in table TOURNAMENT\_TEAM which are GroupPoint and Average. There is no unique, identity, check constraint nor defaults is table TOURNAMENT\_TEAM.

**vi)** No trigger applied.

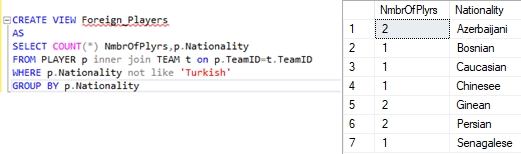
**D) VIEWS**

**1-Foreign Players View**

**i)** Foreign\_Players

**ii)** This view shows number of foreign players for each nation.

**iii)**

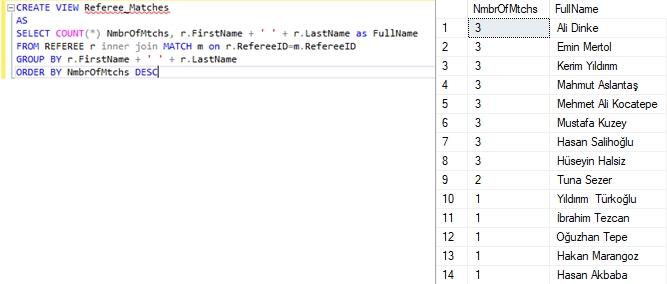


**2-Match number of each referee View**

**i)** Referee\_Matches

**ii)** This view shows the number of match officiated for each referee

**iii)**



**3- Second half top scorers View**

**i)** SecondHalf\_TopScorers

**ii)** This view shows the top five second half scorer and how much goal he scored

**iii)** 

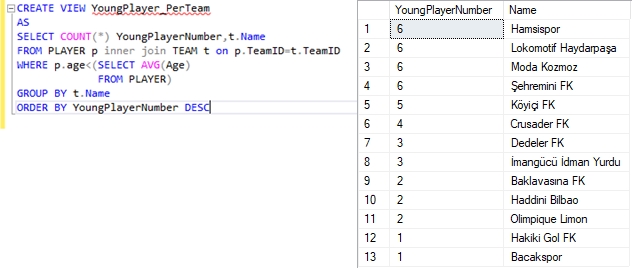
**4- Young Player per Team View**

**i)** YoungPlayer\_PerTeam

**ii)** This view shows how many young players plays for each team. If a player has a age that

is smaller than average age of all players then he is considered as young player.

**iii)**

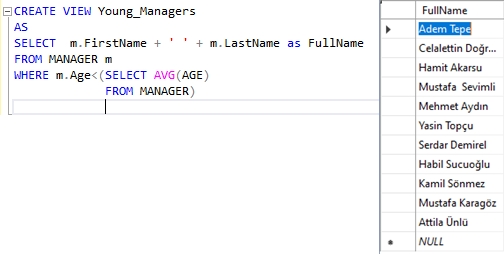


**5-Young Managers View**

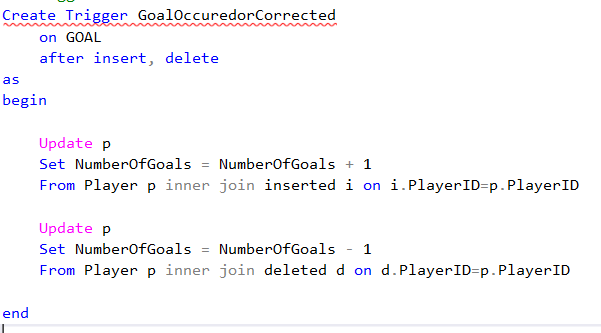
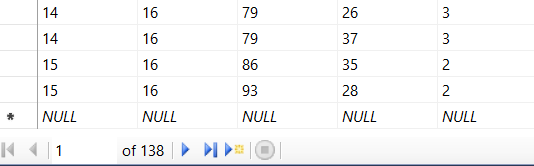
**i)** Young\_Managers

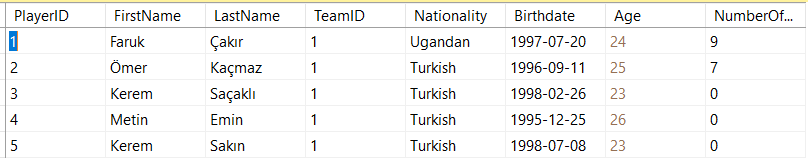
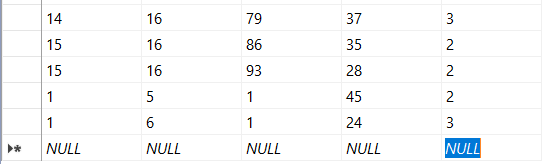
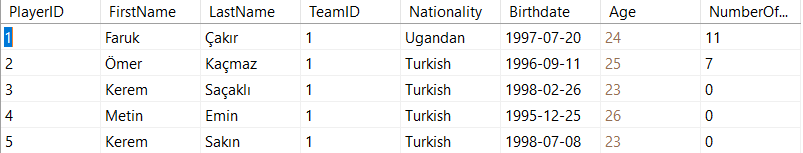
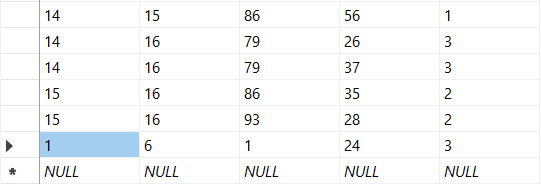
**ii)** This view show the young managers. If a manager has a age that is smaller than average age of all managers then he is considered as young manager

**iii)**



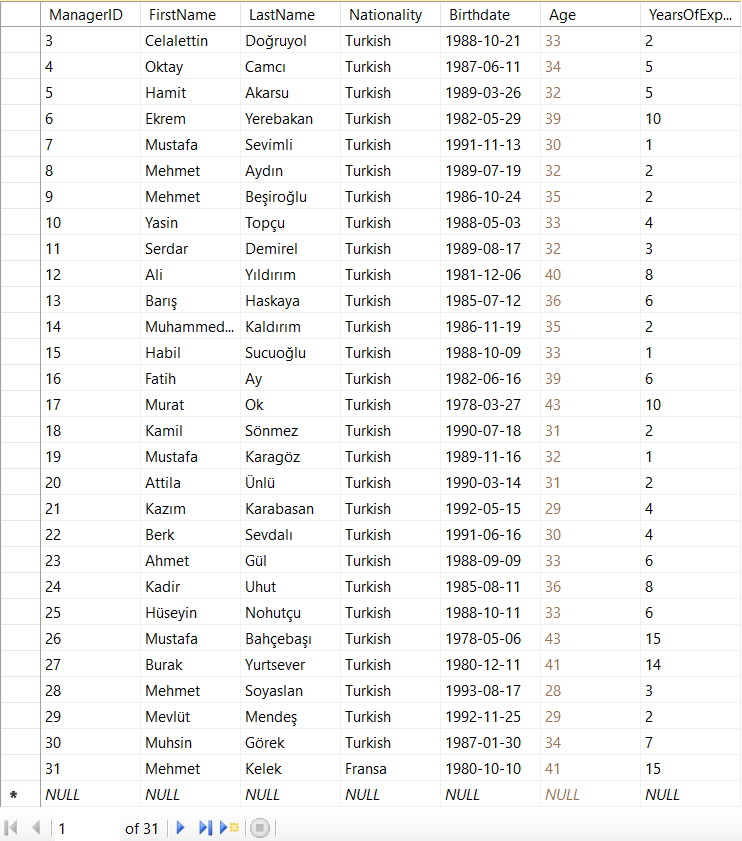
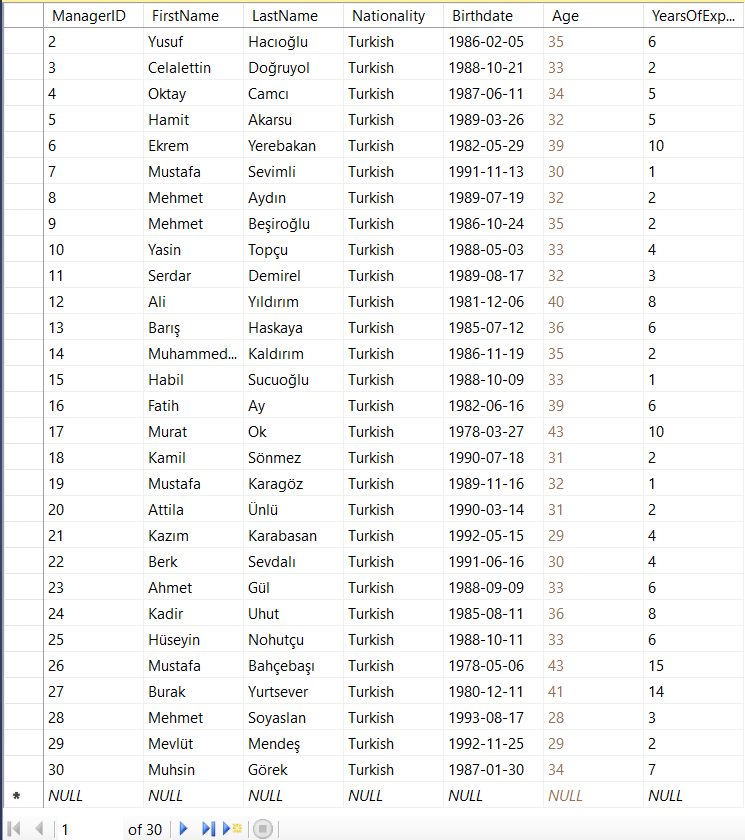
**E) TRIGGERS**

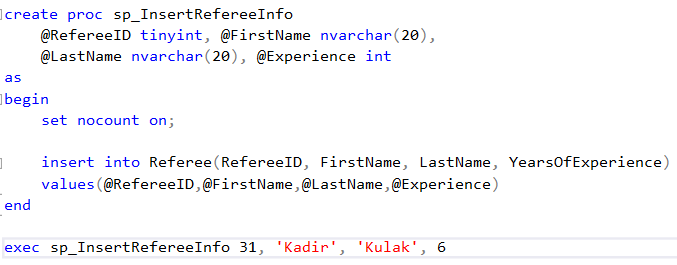
**GoalOccuredorCorrected**Explanation: This trigger works between “Goal” and “Player” tables. It works when a newly goal information is inserted or some of them deleted from the table GOAL. When it is triggered “NumberOfGoals” attribute from the PLAYER table is updated.  
  
Code part:  
  
  
Initials States:  


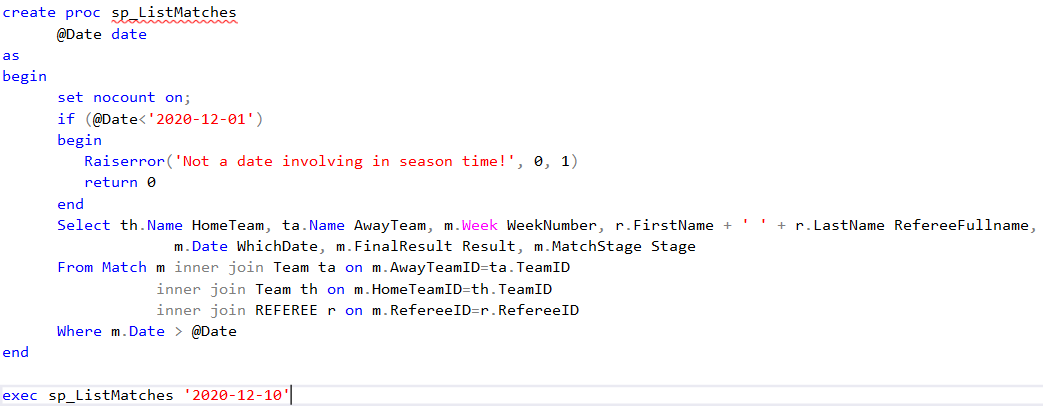
  
  
After inserting two goals information to the GOAL table:  
  
   
  
  
  
After deleting one of the goals newly added from the GOAL table:  
  

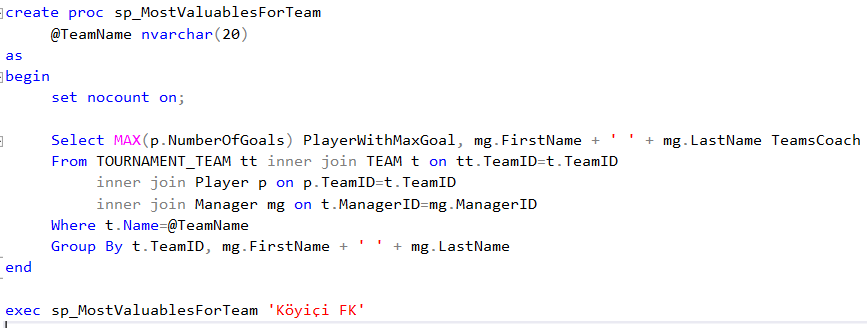

**F) STORED PROCEDURES**

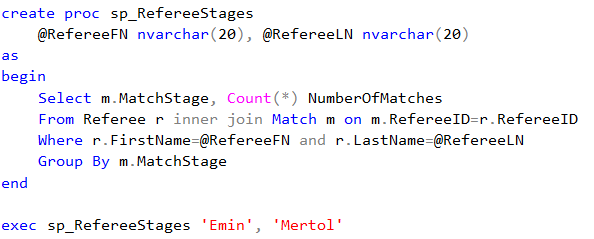
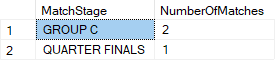
**1 - sp\_InsertManagerInfo**

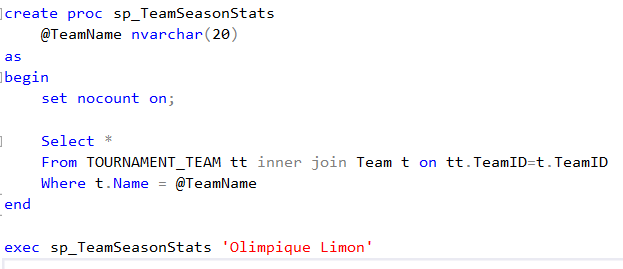
Explanation: A new manager can be created with using this stored procedure.  
  
Coding part:  
Before and After:  
  
  
  
 **2- sp\_InsertRefereeInfo**

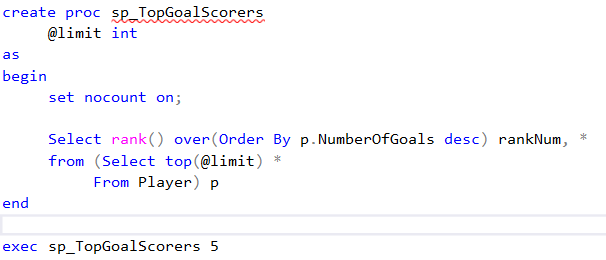
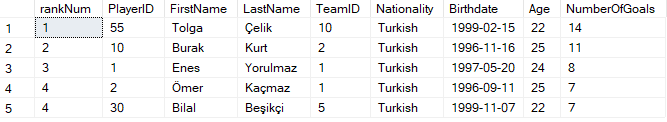
Explanation: A new referee can be created with using this stored procedure.  
  
Coding part:  
  
Before and After:  

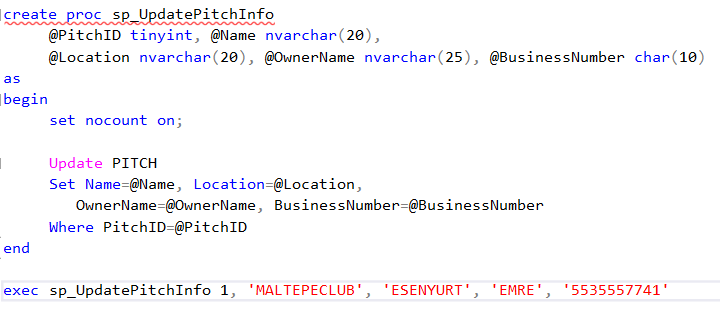
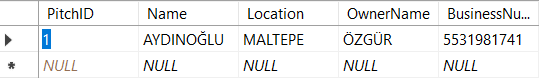
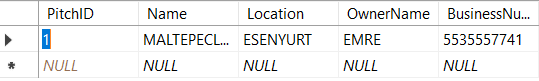

**3- sp\_ListMatches**  
  
Explanation: This stored procedure takes a date as input value and lists the matches played between date and current date. If the entered date time is not within the season time, it raises an error informing the database user.  
  
Code part:  
  
Execution:  

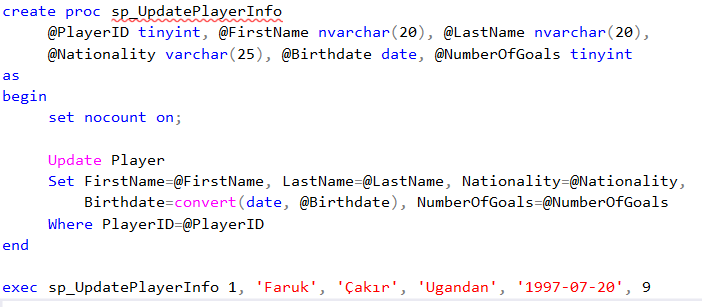
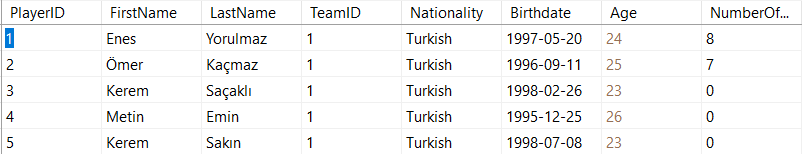
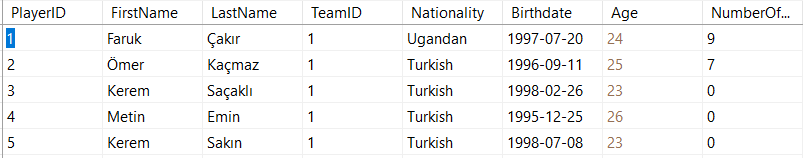

**4- sp\_MostValuablesForTeam**  
  
Explanation: This stored procedure takes a team name as input parameter and finds out max goals achieved by a player during the whole season and also team’s coach informations.  
  
Code part:  
  
Execution:  

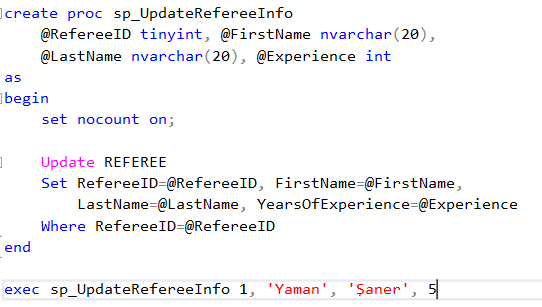

**5- sp\_RefereeStages**  
  
Explanation: This stored procedure first name and last name information of the referee’s as input parameters and prints out how many matches officiated by the referee at which stages.  
  
Code part:  
  
Execution:  


**6- sp\_TeamSeasonStats**  
  
Explanation: This stored procedure takes a team name as the input parameter and prints out every detail of the statistics about the team.  
  
Code part:  
  
Execution:  


**7- sp\_TopGoalScorers**  
  
Explanation: This stored procedure takes a limit number as the input parameter and prints out top x players having most goals scored.  
  
Code part:  
  
Execution:

**8- sp\_UpdatePitchInfo**  
  
Explanation: Considering there might be a typo occurred during storage of a row entered, this stored procedure allows to update table PITCH.  
  
Code part:  
  
Before and After:  
  


**9- sp\_UpdatePlayerInfo**  
  
Explanation: Considering there might be a typo occurred during storage of a row entered, this stored procedure allows to update table PLAYER.  
  
Code part:  
Before and After:  
  
  


**10- sp\_UpdateRefereeInfo:**  
  
Explanation: Considering there might be a typo occurred during storage of a row entered, this stored procedure allows to update table REFREE.  
  
Code part:  
  
  
Before and After:  
  
  
