|  |
| --- |
|  |
| Hockey League Database |
| Course Section: CS605.641.81  Summer, 2020 |
| Prepared by |
| **Brian Loughran** |
| **8/7/2020** |

|  |
| --- |
| Database Design Project Document |

**Table of Contents**

[1. Introduction 2](#_Toc47627557)

[1.1. Scope and Purpose of Document 3](#_Toc47627558)

[1.2. Project Objective 3](#_Toc47627559)

[2. System Requirements 4](#_Toc47627560)

[2.1 Hardware Requirements 4](#_Toc47627561)

[2.2 Software Requirements 4](#_Toc47627562)

[2.3 Functional Requirements 4](#_Toc47627563)

[2.4 Database Requirements 4](#_Toc47627564)

[3. Database Design Description 4](#_Toc47627565)

[E/R Model 8](#_Toc47627566)

[3.1.1 Entities 8](#_Toc47627567)

[3.1.2 Relationships 11](#_Toc47627568)

[3.1.3 E/R Diagram 13](#_Toc47627569)

[3.2 Relational Model 15](#_Toc47627570)

[3.2.1 Data Dictionary 15](#_Toc47627571)

[3.2.2 Integrity Rules 21](#_Toc47627572)

[3.2.3 Operational Rules 24](#_Toc47627573)

[3.2.4 Operations 24](#_Toc47627574)

[3.3 Security 24](#_Toc47627575)

[3.4 Database Backup and Recovery 25](#_Toc47627576)

[3.5 Using Database Design or CASE Tool 25](#_Toc47627577)

[3.6 Other Possible E/R Relationships 25](#_Toc47627578)

[4. Implementation Description 26](#_Toc47627579)

[4.1 Data Dictionary 26](#_Toc47627580)

[4.2 Advanced Features 30](#_Toc47627581)

[4.3 Queries 30](#_Toc47627582)

[4.3.1 Top Teams 30](#_Toc47627583)

[4.3.2 Top 5 Goal Scorers 31](#_Toc47627584)

[4.3.3 Top 5 Point Getters 31](#_Toc47627585)

[4.3.4 Active Schedule 32](#_Toc47627586)

[4.3.5 Next Game for Team 32](#_Toc47627587)

[4.3.6 Show Team Roster 33](#_Toc47627588)

[4.1.7 Show Team Roster with Goalies 33](#_Toc47627589)

[5. CRUD Matrix 34](#_Toc47627590)

[5.1 List of Entity Types 34](#_Toc47627591)

[5.2 List of Functions 34](#_Toc47627592)

[5.3 CRUD Matrix 35](#_Toc47627593)

[6. Concluding Remarks 35](#_Toc47627594)

[Appendices 37](#_Toc47627595)

# Introduction

My database project is designed as a database for a hockey league. Important league information is stored such as players, goalies, coaches, teams, games, schedule, referees, rinks, game statistics, and more. The information stored in the database is designed to be information that could be aggregated into a user-friendly website which a player, coach, team manager, etc. would be able to access and view the status of the league.

The reason I chose to create a hockey database is simply my love for the game of hockey. I have played hockey all my life, from recreational leagues which would have less data than is populated in this database, all the way through high school and college which would have more data than is populated in this database. With the NHL coming back from the shutdown of 2020, what better time is there to center a database project around something that I am passionate and knowledgeable about, and that is hockey.

## Scope and Purpose of Document

This document is intended to be a detailed description of the implemented hockey league database described in the introduction. High level items such as objectives and system requirements will be covered in sections 1-2. Section 3 gives a detailed look into the design of the database, including entities, relationships, conceptual and logical design for the database. Section 4 describes the implementation of the database, including advanced features and queries. Section 5 shows a CRUD matrix, and section 6 includes closing remarks. Included in appendices are further details into the implementation, including DDL statements for the whole project and state of the database after the DDL’s are administered.

The scope of the database is supposed to cover most if not all of the possible operations and queries for a hockey league. Not included in this document are any details about a front end service for the database. A front end could be created quite easily, whether using a GUI tool or web framework, however was not a part of development to this point.

## Project Objective

The objective of this project is to store and maintain all the information needed to summarize, maintain and document general league activity for a hockey league. This would be an invaluable tool to a league administrator and would allow the administrator to schedule game times, rinks, referees as well as to document game activities like scores, winners, losers, goal scorers, goalie statistics and more. Wrapping the database up into a website would allow the administrator easy access to read and update the data, as well as for players, coaches and goalies to view the status of their teams as well as others.

The objective for me is to design and implement a database using PostgreSQL in order to better learn how to do this task for similar database projects in the future. By capturing all the informational aspects of operations for a hockey league, I can prove to myself that I have the ability to apply those skills to capture all the informational aspects of a business case for a future employer (or myself, if I so choose).

# System Requirements

64Bit Operating System

RAID 1

Windows/Linux/Mc

## Hardware Requirements

1 GHz Processor

2 GB RAM

512 MB HDD

## Software Requirements

Windows 2016

## Functional Requirements

Data Types

Data Integrity

Concurrency, Performance

Reliability, Disaster Recovery

Security

Extensibility

Internationalization, Text Search  
See more at: <https://www.postgresql.org/about/#:~:text=PostgreSQL%20comes%20with%20many%20features,source%2C%20PostgreSQL%20is%20highly%20extensible.>

## Database Requirements

PostgreSQL version 12.3 64Bit

# Database Design Description

In the implemented hockey database there are 15 entities. Each entity describes a different part of hockey operations in a league. The entities are described briefly below:

* *Team*: Team describes a team in the league, and is a strong entity. The primary key for team is team\_name, and other values are wins, losses, ties and points. Team has a 1:1/M relationship with both player and goalie which is an identifying relationship. Team has a 1:0/M identifying relationship with captain, schedule and coach. Team also has a 1:0/M non-identifying relationship with goal, penalty and goalie stats.
* *Player*: Player describes a player on a team, and is a weak entity because it is a child of team. The primary keys for player are the player number and the team name, where team name is a foreign key from team. Other player keys include player name, number, goals, assists, PIM and emergency contact phone number. Player has a 0/1:1 identifying relationship with captain. Player has a 0/M:1 identifying relationship with player injury report. Player has a 1/M:1 identifying relationship with team. Player has a 1/M:0/M non-identifying relationship with goal. Finally, player has a 1:0/M non-identifying relationship with penalty.
* *Goalie*: Goalie describes a goalie on a team, and is a weak entity because it is a child of team. The primary keys for goalie are goalie number and team name, where team name is a foreign key from team. Other keys include goalie name, number, goals against, saves, and emergency contact phone number. Goalie has a 0/M:1 identifying relationship with goalie injury report. Goalie has a 1/M:1 identifying relationship with team. And goalie has a 1:0/M relationship with goalie stats.
* *Coach*: Coach describes the coach of a team and is a weak entity because it is the child of team. The primary keys for coach are coach id and team name, where team name is a foreign key from team. Coach also has key coach name. Coach has a 0/M:1 identifying relationship with team.
* *Player* *Injury* *Report*: Player injury report describes any recorded player injuries and is a weak entity because it is a child of player. Player injury report has primary keys injury id, player number, which is also a foreign key from player, and team name, which is a foreign key from team. Player injury report also has keys injury, injury date, and description. Player injury report has a 1:0/M identifying relationship with player.
* *Goalie* *Injury* *Report*: Goalie injury report describes any recorded goalie injuries and is a weak entity because it is a child of goalie. Goalie injury report has primary keys injury id, goalie number, which is also a foreign key from goalie, and team name, which is a foreign key from team. Goalie injury report also has keys injury, injury date, and description. Goalie injury report has a 1:0/M identifying relationship with goalie.
* *Captain*: Captain describes a captain or assistant captain on a team, and is a weak entity because it is a child of team and player. Captain has primary keys player number, which is a foreign key from player, and team name, which is a foreign key from team. Captain also has key captain role, which describes whether the captain is a captain or assistant. Captain has a 1:0/1 identifying relationship with player, and a 0/M:1 identifying relationship with team.
* *Rink*: Rink describes a rink where a game is played, and is a strong entity. Rink has primary key rink name. Rink also has keys rink location and rink size (designating Olympic or standard). Rink has a 1:0/M non-identifying relationship with game.
* *Referee*: Referee describes a single game official, and is a strong entity. Referee has primary key ref id. Referee also has keys ref name, hometown and certification level. Referee has a 1:M/0/M identifying relationship with game refs.
* *Game* *Refs*: Game refs describes a team of game officials for a game, and is a weak entity because it is a child of referee. Game refs has primary key ref crew id, and also has keys head ref id, ref2 id, linesman1 id and linesman2 id, all of which are foreign keys from referee. Game refs have a 0/M:1/M identifying relationship with referee and a 0/M:1 non-identifying relationship with game.
* *Game*: Game describes a game between two teams, and is a strong entity. Game has primary key game id, and also has keys rink name, which is a foreign key from rink, ref crew id, which is a foreign key from game refs, as well as the home team score and away team score. Game has a 1:0/M identifying relationship with schedule. Game has a 0/M:1 non-identifying relationship with rink. Game has a 1:0/M relationship with game refs. Finally, game has a 0/1:0/M relationship with each of goal, penalty and goalie stats.
* *Schedule*: Schedule is an entity to describe the M:N relationship between team and game, and is thus a weak entity. Schedule has primary keys home team name and away team name from team, both foreign keys from team, and primary key game id, which is a foreign key from game. Schedule also has key date. Schedule has 0/M:1 identifying relationships with both game and team to describe the M:N relationship between the two.
* *Penalty*: Penalty describes a player penalty, is one of three statistics entities, and is a weak entity because it is a child of game. Penalty has primary keys penalty number and game id, which is a foreign key from game. Penalty also has keys penalized team, a foreign key from team, penalized player, a foreign key from player, and PIM (penalty minutes). Penalty has a 0/M:0/1 relationship with game which is an identifying relationship. Penalty also has 0/M:1 relationships with both player and team.
* *Goal*: Goal describes a player goal, is one of three statistics entities, and is a weak entity because it is the child of game. Goal has primary keys goal number and game id, which is a foreign key from game. Goal also has keys scoring team, which is a foreign key from team. Goal also has keys scoring player, primary assisting player and secondary assisting player, all of which are foreign keys from player. Goal has a 0/M:0/1 identifying relationship with game, as well as a 0/M:1 non-identifying relationship with team. Finally, goal has a 0/M:1/M relationship with player which is a non-identifying relationship.
* *Goalie* *Stats*: Goalie stats describe the statistics for a goalie for a game, are one of three statistics entities, and are a weak entity because it is the child of game. Goalie stats have primary keys home away goalie and game id, which is a foreign key from game. Goalie stats also have keys goalie team name, a foreign key from team, goalie number, a foreign key from goalie, saves and goals against. Goalie stats have a 0:M/0/1 identifying relationship with game. Goalie stats also has a 0/M:1 relationship with both team and goalie which is non-identifying.

Note: Positive integer values denote any positive integer, 0 inclusive.

* 1. **Design Rationale**

While there are many good ways to build a database for a hockey league, the way that this particular database was implemented makes a lot of sense from the perspective of a league administrator to eventually convert into a website, and will give players, teams and other interested parties good access to data while maintaining good data integrity. Some rationale for how each entity was created. Some rationale for some of the decisions made are defended in this section, however this does not go over each of the possible ways that the database could be altered.

This section is not meant as an exhaustive description of every entity, key and relationship, since those items are discussed in subsequent sections. This section is meant as a high level overview of some of the decisions made for entities, relationships, and overall structure of the database, while more detailed discussion of keys and relationships are discussed in subsequent sections.

One decision made was to identify player and goalie by their team name. This is defendable because a league would not be overly concerned with players that are not included in the league. Since each player’s number must be unique on their team, the team and number combination is unique to the league allowing for a convenient primary key. This is better than using player name as a primary key, since multiple players may have the same name.

Player and goalie are separated into their own entities, rather than combining them. This is because goalies are just so different than players in terms of statistics. It would not make sense to track goalie goals assists and penalties, just as it would not make sense to track player goals against and saves. While the statistics tracked in this database are not all of the statistics that could possibly be tracked for either entity, it makes sense to separate player and goalie because they are so different. This forces player injury report and goalie injury report to be separate entities, even though they have almost all the same fields.

Coach, like player and goalie, is identified by the team name. Coach, however, is assigned a coach id, since coaches do not have a number and a coach name may not be unique. Other entities without convenient, unique primary keys are also assigned numeric id’s similar to coach.

Players and goalies have a 1/M:1 relationship with team, while captain and coach have a 0/M:1 relationship with team. This is because some leagues will not employ captains and coaches, however every league will need players and goalies. This relationship allows for the flexibility to not have a coach/captain if the team does not want one.

Game refs exist as a reference table to match the 1-4 referees to the league games. Ref crews often work many different games as their own team, thus to reduce data redundancy this table was created. Game could also include four keys, one for each referee, however since referees often work in pre-determined ref crews, this could result in a lot of redundant data. Game refs get their own id which can be used to identify the referees that are working a game.

For a given schedule, some games may be played and some games may not have been played yet. As a consequence of this, game scores may be null for games not played yet, and games that have not been played may not have statistics such as goals, penalties, and goalie stats. For the purposes of this database, games through 7/24/2020 have been played, while games after that date have not been played. This shows both cases while maintaining a manageable amount of test data for the database.

There are three entities used to describe statistics for the league, and these entities are goal, penalty and goalie stats. Each describes a different subset of statistics, and each has triggers to automatically update the statistics to the player or goalie profile. This allows for automatic updates to limit database operations and possible errors. Goals and penalties have no convenient primary keys since players can score or be penalized multiple times, thus have numeric event numbers as primary keys for each. Goalie stats can be identified by the home or away goalie and the game id, thus no event identifier is needed in this case.

## E/R Model

### Entities

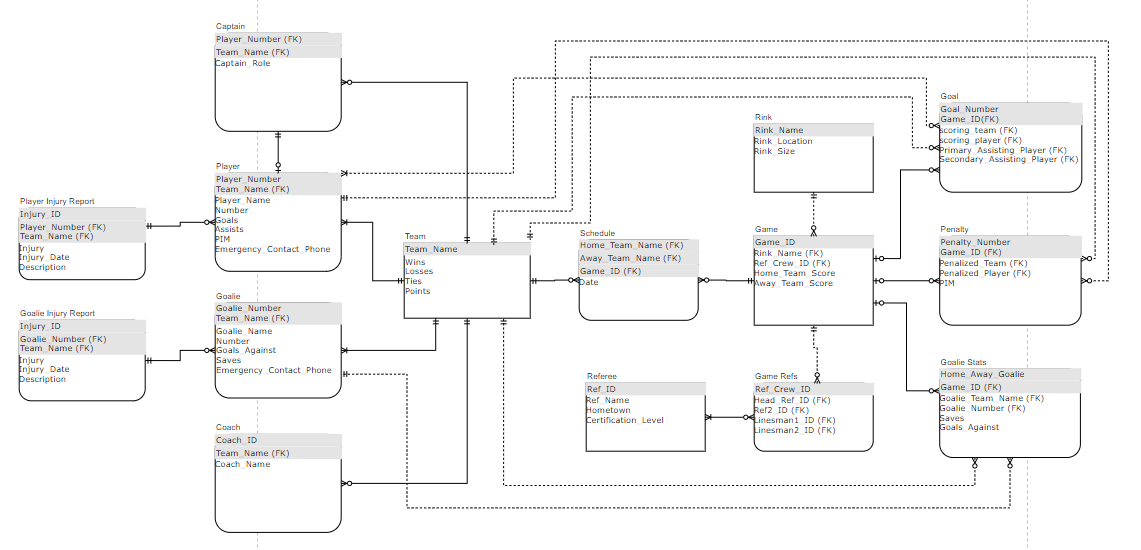
* *Team*: The primary key for team is team name, and other values are wins, losses, ties and points. This encompasses all the important values for a team and managing league standings. Team can exist independently of other entities, thus is a strong entity.
* *Player*: The primary keys for player are the player number and the team name, where team name is a foreign key from team. Player number and team can be used to uniquely identify player, thus is a useful primary key. Other player keys include player name, number, goals, assists, PIM and emergency contact phone number. Player goals assists and PIMs will be updated by goal and penalty entries. Player is a child of team, thus is a weak entity
* *Goalie*: The primary keys for goalie are goalie number and team name, where team name is a foreign key from team. Goalie number and team can be used to uniquely identify a goalie, thus is a useful primary key. Other keys include goalie name, number, goals against, saves, and emergency contact phone number. Goalie goals against and saves will be updated by the goalie statistics entity. Goalie is a child of team, thus is a weak entity.
* *Coach*: The primary keys for coach are coach id and team name, where team name is a foreign key from team. Coach needs a numeric coach id since coach name may not be a unique key. Coach is a child of team for the same reasons player and goalie are children of team, the league would not be concerned with coaches coaching outside of the league. Coach also has key coach name to identify who the coach is. Coach is a child of team, thus is a weak entity.
* *Player* *Injury* *Report*: Player injury report has primary keys injury id, player number, which is a foreign key from player, and team name, which is a foreign key from team. Player injury report needs the team name because as a child of player, the team name gets mapped as a foreign key. Player needs a numeric injury id because there are no other unique keys available. Player injury report also has keys injury, injury date, and description as ways to further describe the injury and severity. Player injury report is a child of player, thus is a weak entity.
* *Goalie* *Injury* *Report*: Goalie injury report has primary keys injury id, goalie number, which is also a foreign key from goalie, and team name, which is a foreign key from team. Goalie injury report needs the team name because as a child of goalie, the team name gets mapped as a foreign key. Goalie injury report needs a numeric injury id because there are no other unique keys available. Goalie injury report also has keys injury, injury date, and description which further describe the injury and severity. Goalie injury report is a child of goalie, thus is a weak entity.
* *Captain*: Captain has primary keys player number, which is a foreign key from player, and team name, which is a foreign key from team. Player number and team name uniquely identify a captain, thus are convenient primary keys. Captain also has key captain role, which describes whether the captain is a captain or assistant, which are the only two possible roles for a captain. Captain is a child of team and player, thus is a weak entity.
* *Rink*: Rink has primary key rink name, which is unique to each rink, thus is a convenient primary key. Rink also has keys rink location and rink size (designating Olympic or standard) as other helpful pieces of information for a rink. Rink has no parents, thus is a strong entity.
* *Referee*: Referee has primary key ref id. Ref name may not be unique, thus a numeric id is used as identification. Referee also has keys ref name, hometown and certification level as helpful pieces of information on referees. Referee is a strong entity because it has no parents.
* *Game* *Refs*: Game refs has primary key ref crew id. There is other convenient primary key, thus creating a numeric id is best. Game refs also has keys head ref id, ref2 id, linesman1 id and linesman2 id, all of which are foreign keys from referee. This allows game refs to encompass any combination of referees that will be used in a game. Game refs is a weak entity as a child of referee.
* *Game*: No convenient unique key was identified for game, thus game has primary key game id. Game also has keys rink name, which is a foreign key from rink, ref crew id, which is a foreign key from game refs, as well as the home team score and away team score. The foreign keys allow the game to be scheduled from a list of rinks and limit repeating data in game, and allows game to be scheduled with a single game refs entity rather than multiple referees. This is helpful since many times games will be officiated by the same group of referees. Home team score and away team score will only be recorded for games that have been played (in this case games after 7/24/2020).
* *Schedule*: Schedule is an entity to describe the M:N relationship between team and game, and is thus a weak entity. Schedule has primary keys home team name and away team name from team, both foreign keys from team, and primary key game id, which is a foreign key from game. This is standard for entities which describe M:N relationships, and since there are two teams to a game, there are two foreign keys for team. Schedule also has key date to specify when the game will be played.
* *Penalty*: Penalty has primary key penalty number, which is an ordered numeric value to track the order of penalties in a game. Penalty also features primary key game id, which is a foreign key from game. These primary keys provide a convenient way to uniquely identify the penalty. Penalty also has keys penalized team, a foreign key from team, penalized player, a foreign key from player, and PIM (penalty minutes). Using the penalized team, penalized player and PIM, a player entity can be updated with their new PIM statistic automatically. Penalty is a weak entity because it is a child of game.
* *Goal*: Goal has primary key goal number, which is an ordered numeric value to track the order of goals in a game. Goal also features primary key game id, which is a foreign key from game. These primary keys provide a convenient way to uniquely identify the goal. Goal also has keys scoring team, which is a foreign key from team. Goal also has keys scoring player, primary assisting player and secondary assisting player, all of which are foreign keys from player. Primary assisting player and secondary assisting player may be null if there was no assist on the goal. Using the scoring team, scoring player assisting players, a player entity can be updated with their new goals and assists statistics automatically. Goal is a weak entity because it is the child of game.
* *Goalie* *Stats*: Goalie stats have primary keys home away goalie and game id, which is a foreign key from game. Using game combined with home away goalie easily designates which goalie is playing, since there can be only one goalie for each team. Home away goalie can have values ‘Home’ and ‘Away’. Goalie stats also have keys goalie team name, a foreign key from team, goalie number, a foreign key from goalie, saves and goals against. Using the goalie team and goalie number, saves and goals against can be updated in the goalie entity automatically. Goalie stats are a weak entity because they are a child of game.

### Relationships

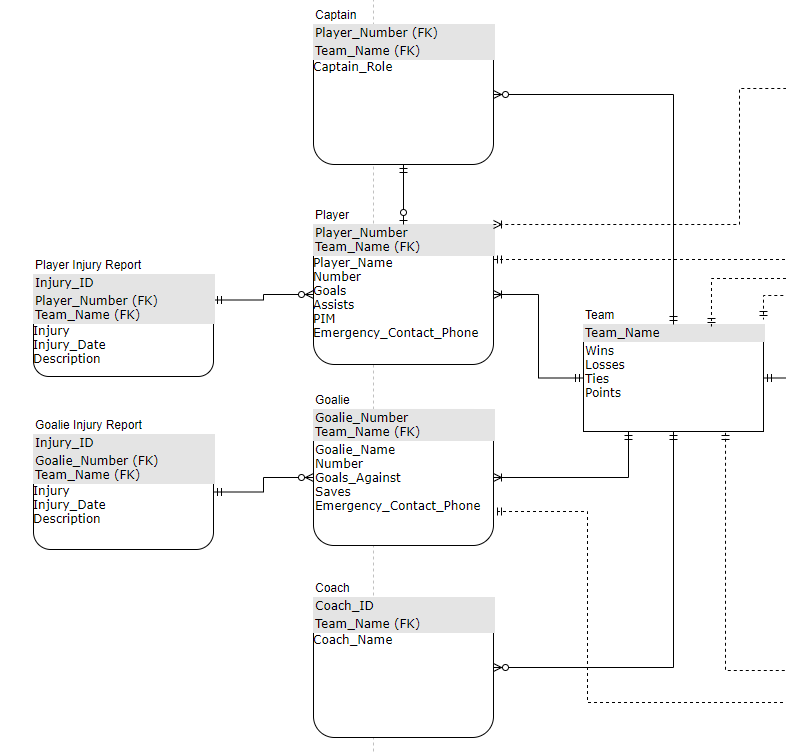
* *Team*: Team has a 1:1/M relationship with both player and goalie which is an identifying relationship. Player and goalie are identified by their team and number, thus the identifying relationship. There must be 1 or more players and goalies on a team, however there may be more. Team has a 1:0/M identifying relationship with captain, schedule and coach. Teams may have 0 or multiple captains and coaches. The relationship with schedule is because schedule manages the M:N relationship with team. All of these relationships are primary keys in the child entity, thus the identifying relationship. Team also has a 1:0/M non-identifying relationship with goal, penalty and goalie stats. Each team can have 0 relevant statistics if no games are played, or many if the team has been a part of many events. None of the statistics are identified by the team, thus the non-identifying relationship.
* *Player*: Player has a 0/1:1 identifying relationship with captain, since a player may or may not be a captain, but cannot be a captain twice. Player has a 0/M:1 identifying relationship with player injury report; each player may or may not be injured, and can be injured multiple times on different dates. Player has a 1/M:1 identifying relationship with team as discussed under team. Player has a 1/M:0/M non-identifying relationship with goal; multiple players may be involved in a goal if there are assists, however at least one player must score a goal, and a player can be involved in 0 or many goals. Finally, player has a 1:0/M non-identifying relationship with penalty because a player may be involved in 0 or many penalties, however there is only one player to a penalty.
* *Goalie*: Goalie has a 0/M:1 identifying relationship with goalie injury report. This is because a goalie may or not be injured, but can be injured multiple times on different dates. Goalie has a 1/M:1 identifying relationship with team as discussed under team. And goalie has a 1:0/M relationship with goalie stats because a goalie that has never played may have no statistics, however a goalie who has played many games may have many statistics.
* *Coach*: Coach has a 0/M:1 identifying relationship with team. The relationship is identifying because the coach must be a part of a team to be included in the league. A team may have 0 or many coaches.
* *Player* *Injury* *Report*: Player injury report has a 1:0/M identifying relationship with player. A player may never be injured, however may be injured multiple times. As a child of player, the relationship must be identifying.
* *Goalie* *Injury* *Report*: Goalie injury report has a 1:0/M identifying relationship with goalie. A goalie may never be injured, however may be injured multiple times. As a child of goalie, the relationship must be identifying.
* *Captain*: Captain has a 1:0/1 identifying relationship with player. This is because a player may or may not be a captain, but a player can only be one of assistant or captain. Captain also has a 0/M:1 identifying relationship with team, because a team may have 0 or many captains (typically no more than 3, but there can be up to 5 or more in some cases).
* *Rink*: Rink has a 1:0/M non-identifying relationship with game. A game will be scheduled at a 1 rink, and a rink may be schedule for 0 or many games. The relationship is non-identifying because game is not identified by the rink it is played at.
* *Referee*: Referee has a 1:M/0/M identifying relationship with game refs. There may be 1 to up to 4 referees as a crew for a game. And each referee may be on 0 or many game ref crews.
* *Game* *Refs*: Game refs have a 0/M:1/M identifying relationship with referee as discussed under referee. Game refs also have a 0/M:1 non-identifying relationship with game. A game will have 1 set of game refs, however each game ref crew can work 0 or many games. Games is not identified by the game refs, thus this is a non-identifying relationship.
* *Game*: Game has a 1:0/M identifying relationship with schedule which manages the M:N relationship between game and team. Game has a 0/M:1 non-identifying relationship with rink as discussed under rink. Game has a 1:0/M relationship with game refs as discussed under game refs. Finally, game has a 0/1:0/M relationship with each of goal, penalty and goalie stats. Games which have been played will have each of the statistics, however games that have not been played will not, hence the 0/1 for game. Each of goal, penalty and goalie stats are identified by game, thus the identifying relationship. Finally, each game may have 0 or many of each of goal, penalty and goalie stats depending on how many of each occur for a game and whether the game has been played yet.
* *Schedule*: Schedule has 0/M:1 identifying relationships with both game and team to describe the M:N relationship between the two. This is standard for entities used to map M:N relationships.
* *Penalty*: Penalty has a 0/M:0/1 relationship with game which is an identifying relationship, and is described under game. Penalty also has 0/M:1 relationships with both player and team, which are described in their respective sections.
* *Goal*: Goal has a 0/M:0/1 identifying relationship with game, as well as a 0/M:1 non-identifying relationship with team, described under team and game. Goal also has a 0/M:1/M relationship with player which is a non-identifying relationship described under player.
* *Goalie* *Stats*: Goalie stats have a 0:M/0/1 identifying relationship with game described in game. Goalie stats also has a 0/M:1 relationship with both team and goalie which is non-identifying and described under team and goalie respectively.

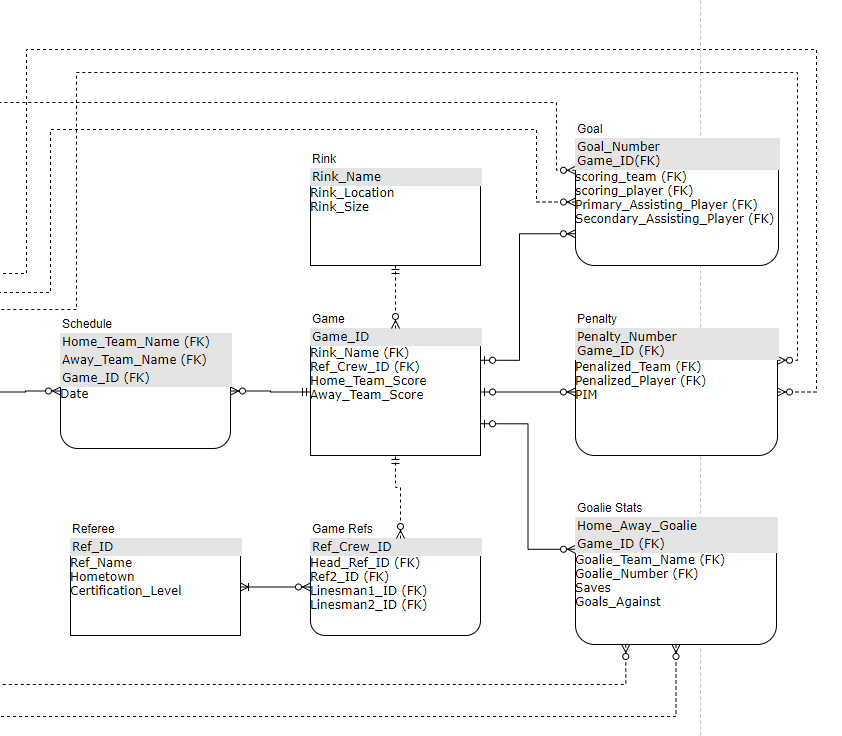
### E/R Diagram

The E/R diagram for the hockey database is shown below:



And each half of the E/R diagram is shown below for easier viewing of each entity:





## Relational Model

### Data Dictionary

*Team*: Describes a team in the league

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Team\_Name | The team name | Varchar | 30 | Primary Key | Y | String |
| Wins | The number of team wins | Int | - | - | y | Positive integers |
| Losses | The number of team losses | Int | - | - | y | Positive integers |
| Game\_Ties | The number of team ties | Int | - | - | y | Positive integers |
| Points | The number of standings points | Int | - | - | y | Positive integers |

*Player*: Describes a player on a team

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Player\_Number | The number on the player’s jersey | Int | - | Primary Key | Y | 1-99 |
| Team\_Name | The name of the player’s team | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Player\_Name | The name of the player | Varchar | 50 | - | y | String |
| Goals | The number of goals scored | Int | - | - | y | Positive integers |
| Assists | The number of standings points | Int | - | - | y | Positive integers |
| PIM | The number of player penalty minutes | Int | - | - | y | Positive integers |
| Emergency\_Contact\_Phone | The phone number of the EC | Varchar | 20 | - | n | Any phone number |

*Goalie*: Describes a goalie on a team

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Goalie\_Number | The number on the goalie’s jersey | Int | - | Primary Key | Y | 1-99 |
| Team\_Name | The name of the goalie’s team | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Goalie\_Name | The name of the goalie | Varchar | 50 | - | y | String |
| Goals\_Against | The number of goals scored agains the goalie | Int | - | - | y | Positive integers |
| Saves | The number of saves made by the goalie | Int | - | - | y | Positive integers |
| Emergency\_Contact\_Phone | The phone number of the EC | Varchar | 20 | - | n | Any phone number |

*Coach*: Describes the coach of a team

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Coach\_ID | The ID of the coach | Int | - | Primary Key | Y | Unique, Positive integers |
| Team\_Name | The name of the coach’s team | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Coach\_Name | The name of the coach | Varchar | 50 | - | y | String |

*Player* *Injury* *Report*: Describes any recorded player injuries

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Injury\_ID | An ID to track injuries | Int | - | Primary Key | Y | Unique, Positive integers |
| Player\_Number | The number of the injured player | Int | - | Primary Key / Foreign Key | y | 1-99 |
| Team\_Name | The team name for the injured player | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Injury | The name of the injury | Varchar | 30 | - | y | String |
| Injury\_Date | The date of the injury | Varchar | 50 | - | n | Any valid date (MM-DD\_YYYY) |
| Injury\_Description | A longer description of the injury | Varchar | 300 | - | n | String |

*Goalie* *Injury* *Report*: Describes any recorded goalie injuries

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Injury\_ID | An ID to track injuries | Int | - | Primary Key | Y | Unique, Positive integers |
| Goalie\_Number | The number of the injured goalie | Int | - | Primary Key / Foreign Key | y | 1-99 |
| Team\_Name | The team name for the injured player | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Injury | The name of the injury | Varchar | 30 | - | y | String |
| Injury\_Date | The date of the injury | Varchar | 50 | - | n | Any valid date (MM-DD\_YYYY) |
| Injury\_Description | A longer description of the injury | Varchar | 300 | - | n | String |

*Captain*: Describes a captain or assistant captain on a team

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Player\_Number | The number of the captain | Int | - | Primary Key / Foreign Key | Y | 1-99 |
| Team\_Name | The team name of the captain | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Captain\_Role | The role of the captain (Captain/Assistant) | Varchar | 9 | - | Y | IN [‘Captain’, ‘Assistant’] |

*Rink*: Describes a rink where a game is played

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Rink\_Name | The name of the rink | Varchar | 50 | Primary Key | Y | String |
| Rink\_Location | The location of the rink | Varchar | 30 | - | y | String |
| Rink\_Size | Standard or Olympic | Varchar | 9 | - | Y | IN [‘Standard’, ‘Olympic’] |

*Referee*: Describes a single game official

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Ref\_ID | Identifier for referee | Int | - | Primary Key | Y | Positive Integer |
| Ref\_Name | The name of the referee | Varchar | 30 | - | y | String |
| Hometown | The referee hometown | Varchar | 50 | - | n | String |
| Certification\_Level | The referee certification level | Varchar | 30 | - | y | String |

*Game* *Refs*: Describes a team of game officials for a game

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Ref\_Crew\_ID | Identifier for referee team | Int | - | Primary Key | Y | Positive Integer |
| Head\_Ref\_ID | The ID of the head referee | Int | - | Foreign Key | y | Positive Integer |
| Ref2\_ID | The ID of the second referee | Int | - | Foreign Key | n | Positive Integer |
| Linesman1\_ID | The ID of the first linesman | Int | - | Foreign Key | n | Positive Integer |
| Linesman2\_ID | The ID of the second linesman | Int | - | Foreign Key | n | Positive Integer |

*Game*: Describes a game between two teams

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Game\_ID | The numeric ID for a game | Int | - | Primary Key | Y | Positive Integer |
| Rink\_Name | The name of the rink to play the game | Varchar | 50 | Foreign Key | y | String |
| Ref\_Crew\_ID | The ID of the officiating crew | Int | - | Foreign Key | y | Positive Integer |
| Home\_Team\_Score | The number of goals scored by the home team | Int | - | - | n | Positive Integer |
| Away\_Team\_Score | The number of goals scored by the away team | Int | - | - | n | Positive Integer |

*Schedule*: Entity to describe M:N relationship between team and game

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Home\_Team\_Name | The name of the home team | Varchar | 30 | Primary Key / Foreign Key | Y | String |
| Away\_Team\_Name | The name of the away team | Varchar | 30 | Primary Key / Foreign Key | y | String |
| Game\_ID | The numeric ID of the game | Int | - | Primary Key / Foreign Key | y | Positive Integer |
| Date | The date of the game | Varchar | 30 | - | y | Any valid date (MM-DD\_YYYY) |

*Penalty*: Describes a player penalty

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Penalty\_Number | The numeric ID of the game penalty | Int | - | Primary Key | Y | Positive Integer |
| Game\_ID | The numeric game ID | Int | - | Primary Key / Foreign Key | y | Positive Integer |
| Penalized Team | The team that took the penalty | Varchar | 30 | Foreign Key | y | String |
| Penalized\_Player | The player number that took the penalty | Int | - | Foreign Key | y | 1-99 |
| PIM | The number of penalty minutes | Int | - | - | y | IN [2, 5, 10] |

*Goal*: Describes a player goal

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Goal\_Number | The numeric ID of the goal for the game | Int | - | Primary Key | Y | Positive Integer |
| Game\_ID | The numeric game ID | Int | - | Primary Key / Foreign Key | y | Positive Integer |
| Scoring\_Team | The team that scored | Varchar | 30 | Foreign Key | y | String |
| Scoring\_Player | The player that scored | Int | - | Foreign Key | y | 1-99 |
| Primary\_Assisting\_Player | The player that got the primary assist | Int | - | Foreign Key | n | 1-99 |
| Secondary\_Assisting\_Player | The player that got the secondary assist | Int | - | Foreign Key | n | 1-99 |

*Goalie* *Stats*: Describes a goalie’s statistics for a game

| Column Name | Description | Data Type | Size | Constraint Type | Not Null? | Valid Values |
| --- | --- | --- | --- | --- | --- | --- |
| Home\_Away\_Goalie | Determines if the goalie played home or away | Varchar | 4 | Primary Key | Y | IN[‘Home’, ‘Away’] |
| Game\_ID | The numeric game ID | Int | - | Primary Key / Foreign Key | y | Positive Integer |
| Goalie\_Team\_Name | The name of the goalie’s team | Varchar | 30 | Foreign Key | y | String |
| Saves | The number of saves the goalie made | Int | - | - | y | Positive Integer |
| Goals\_Against | The number of goals the goalie allowed | Int | - | - | y | Positive Integer |

### Integrity Rules

Each of the entities described have different integrity rules managing mandatory/non-mandatory fields, data format, and valid values. Other integrity rules involve foreign keys describing relationships between tables. Since referential integrity is discussed in detail in section 3.1.2 Relationships, they will not be discussed again here, and information regarding relationships should be referenced in section 3.1.2 Relationships.

* *Team*: Team name is the primary key for team, and thus must be a unique positive string. Positive integer values must be included for wins, losses, game ties and points, with each of those values being updated as the team plays games.
* *Player*: Primary keys player number and team name are mandatory, and player number must be an integer 1-99 and must be unique for that team. Player name is a mandatory field which may be any string. Goals, assists and pim are also mandatory and may be any positive integer. Emergency contact phone number is formatted as (xxx) xxx-xxxx but is not considered mandatory.
* *Goalie*: Primary keys goalie number and team name are mandatory, and goalie number must be an integer 1-99 and must be unique for that team. Goalie name is a mandatory field which may be any string. Goals against and saves are mandatory as well and can be any positive integer. Emergency contact phone number is formatted as (xxx) xxx-xxxx but is not considered mandatory.
* *Coach*: Coach has primary key coach id, which is an ordered integer value and must be unique. Coach also has foreign key team name which is mandatory and must match a valid team name, and a coach name, which is mandatory and can be any string.
* *Player* *Injury* *Report*: Player injury report has primary keys injury id, player number, and team name. Injury id is an ordered integer value similar to coach id, thus must be unique. Player number and team name also must match a player number and team name as foreign keys. Injury is a mandatory field and must be a short string, while description is non-mandatory and can be longer. Injury date is non-mandatory, and is formatted DD-MM-YYYY.
* *Goalie* *Injury* *Report*: Goalie injury report has primary keys injury id, goalie number, and team name. Injury id is an ordered integer value similar to coach id, thus must be unique. Goalie number and team name also must match a goalie number and team name as foreign keys. Injury is a mandatory field and must be a short string, while description is non-mandatory and can be longer. Injury date is non-mandatory, and is formatted DD-MM-YYYY.
* Captain: Captain has primary keys player number and team name, each of which are mandatory and must match a player number and team name in player. Captain role must have values either ‘Captain’ or ‘Assistant’, matching the player role on the team.
* *Rink*: Rink has primary key rink name, which must be unique but can be any string. Rink also has mandatory fields rink location and rink size. Rink size must be either ‘Standard’ or ‘Olympic’; however rink location can be any string.
* *Referee*: Referee has primary key ref id, which must be a unique integer. Referee also has mandatory string fields ref name and certification level which are strings, and non-mandatory field hometown, which is also a string.
* *Game* *Refs*: Game refs has primary key ref crew id, which must be a unique integer. Game refs also have fields head ref id, ref2 id, linesman1 id, and linesman2 id, each of which is a foreign key and must match an id from referee. Only ref crew id and head ref id are mandatory, since not all games will include more than 1 referee.
* *Game*: Game has primary key game id, which is a unique, ordered integer. Game also has keys rink name and ref crew id, each of which are foreign keys, thus mandatory, and must match a rink name and ref crew id respectively. Non-mandatory fields are home team score and away team score, since if a game has not yet been played there will be no values there.
* *Schedule*: Schedule has primary keys home team name and away team name, both of which match with a team name from team, and a game id, which must match with a game id from game. Game also must have a date, formatted DD-MM-YYYY.
* *Penalty*: Penalty has primary key penalty number, an ordered integer denoting each penalty in a game, which is a mandatory and unique key. Penalty also has primary key game id which must match with game id from game. All primary keys are mandatory. Penalty has foreign, mandatory key penalized team and penalized player, each of which are foreign keys and must match a player number and team from player. Penalty also has mandatory field PIM, specifying the number of penalty minutes, and can have values of either 2, 5, or 10.
* *Goal*: Goal has primary key Goal number, an ordered integer denoting each Goal in a game, which is a mandatory and unique key. Goal also has primary key game id which must match with game id from game. All primary keys are mandatory. Goal has foreign, mandatory key scoring team and scoring player, each of which are foreign keys and must match a player number and team from player. Goal also has non-mandatory field primary assisting player and secondary assisting player, which must match a player number from player if entered.
* *Goalie* *Stats*: Goalie stats has primary key home away goalie, denoting either ‘Home’ or ‘Away’ and is mandatory. Goalie also has mandatory primary key game id matching a valid game id from game. Goalie stats have foreign, mandatory key goalie team name and goalie number, each of which are foreign keys and must match a goalie number and team from goalie. Also mandatory are saves and goals against, which must be positive integers.

### Operational Rules

League administrators will have full read/write access to every part of a database to be able to manage any operations that will occur. However, not everyone should have such operational power. For example, players, referees and goalies will not have write access to any part of the database, however they should have read access to everything. Coaches should have write access to the players on their team to determine the rosters, players, and player numbers, but not write access to player or goalie statistics. Finally, scorekeepers should have access to game scores and statistics (goal, penalty, goalie stats) to upload the events of the game, however should not have write access to player, goalie, coach, referee, rink, or schedule.

### Operations

Some basic operations for the hockey database include updating goal, penalty and goalie stats with statistics information as well as updating the game scores and team values of wins, losses and ties. These operations will be discussed in this section, with more operations being discussed in section 5 pertaining to the CRUD matrix.

One operation is inserting a penalty into entity penalty. This will create a record in penalty, which requires insert/retrieve/update permissions. This will also update the corresponding record in player, which requires retrieve/update permissions.

Another operation is inserting a goal into entity goal. This will create a record in goal, which requires insert/retrieve/update permissions. This will also update the corresponding record in player, which requires retrieve/update permissions.

Yet another operation is inserting a record into entity goalie stats. This will create a record in goalie stats, which requires insert/retrieve/update permissions. This will also update the corresponding record in goalie, which requires retrieve/update permissions.

Another operation is updating the game scores in game. This can be done after a game is complete. This requires retrieve/update permissions with game.

One final operation is updating the wins and losses record under team. This can also be done after a game is complete. This requires retrieve/update permissions with team.

More operations will be discussed in more detail within the CRUD matrix in section 5.

## Security

PostgreSQL has many security features which are baked into the implementation. User authentication is performed when a new user logs into the database. Host based access control allows control over which clients are allowed which operations, which can be referenced under operational rules. A secure TCP/IP connection is used to transfer data from the input machine to centralized PosgreSQL database.

## Database Backup and Recovery

PostgreSQL provides several ways to perform database backup and recovery. Some methods for backup and recovery include an SQL dump, which dumps all the relevant SQL code to a file which can be easily read back into the database to restore data. Another method is file system level backup, which stores the database as files in a file location of your choice, which can also be easily restored. Another method is continuous archiving using the postgreSQL write ahead log, which records every change made to the database files and stores the changes in the pg\_xlog/ subdirectory of the cluster’s data directory. In the case the system crashes, the database can be restored using the archive.

## Using Database Design or CASE Tool

CASE tools were not used for this project. To create the ERD, draw.io was used, which while not integrated with postgreSQL directly, did provide an easy interface with which to create the ERD. In hindsight, a nice tool to use to produce the ERD from postrgreSQL would have been DBeaver, which integrates nicely with postgreSQL, however the ERD was already created for this project before scripting began on the database. In the future, I would likely use DBeaver to create the ERD to ensure data consistency.

## Other Possible E/R Relationships

While the structure for this database was generally very good, there were some decisions made which could have been changed to produce a still very good database.

One possible change is to create id’s for both players and goalies rather than using the player number and team name to describe each. This would allow player and goalie to be strong entities, and would likely make it much easier to facilitate trades between teams for players. For pro leagues, this would be much preferred, however pro leagues were not a likely customer for this low-detail database, thus the decision to identify players and goalies by number and team is defendable.

Another possible change is to include each of the four referees directly as a part of game. This would eliminate the need for the game refs entity. This would reduce the number of entities in the table, but would also likely increase the amount of redundant data in the system. This decision could go either way.

Another possible change could introduce the concept of a “game sheet”. This game sheet would represent the report generated by the timekeepers, and would contain the goals, penalties, goalie stats, scores, +/- statistics, and times for events. This would provide a nice interface between many entities, thus reducing the number of relationships in the table, but would add a rather complex entity to the table. This decision could also go either way.

Clearly there are many ways to design a hockey league database. Some ways are worse than others. And some ideas balance detail with complexity to create the best product possible for league administrators.

# Implementation Description

Implementation was done using PostgreSQL version 12.3 which is the most recent version at the time of writing. Because some other tools like MySQL, Aurora, etc. have slightly different syntax for DDL statements. Shown in Appendix B are the exact DDL statements for the implementation.Thus for implementation with the given DDL statements, it is prudent to use PostgreSQL version 12.3 or to convert the SQL for another tool.

## Data Dictionary

PostgreSQL, unfortunately does not have a DESCRIBE operation for their tables. To describe a table, the following SQL code is used:

SELECT

column\_name,

data\_type,

character\_maximum\_length,

is\_nullable

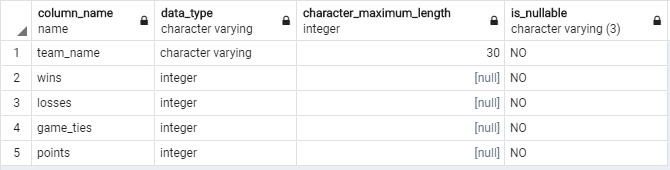
FROM

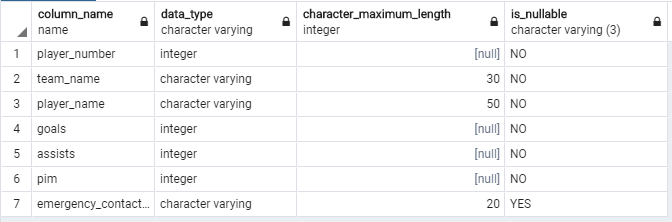
information\_schema.columns

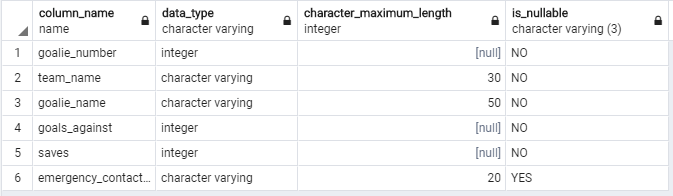
WHERE

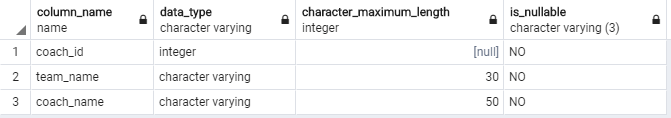
table\_name = ‘<table\_name>’;

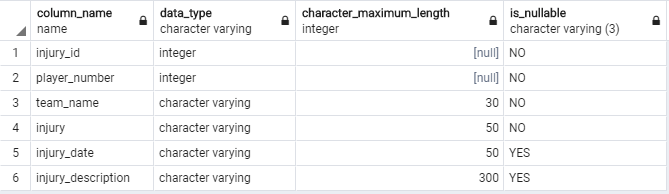
Data is copied directly from PostgreSQL PGadmin tool.

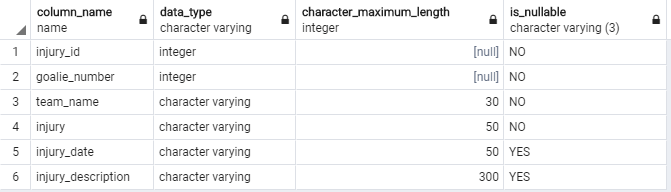
*Team: *

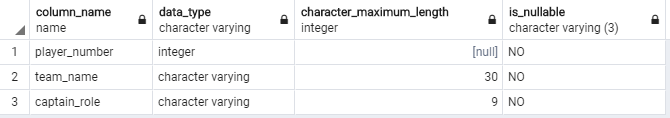
*Player: *

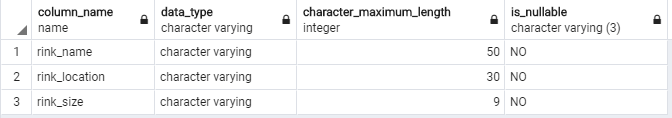
*Goalie: *

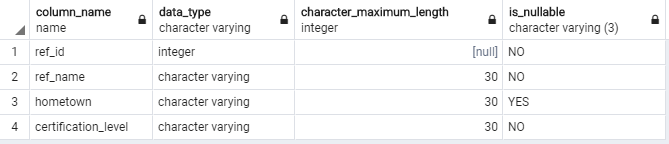
*Coach: *

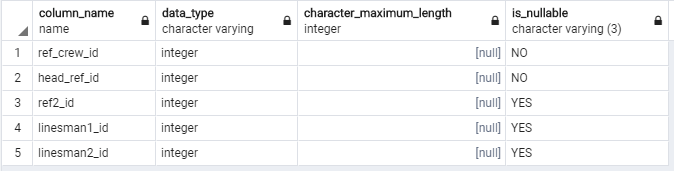
*Player Injury Report: *

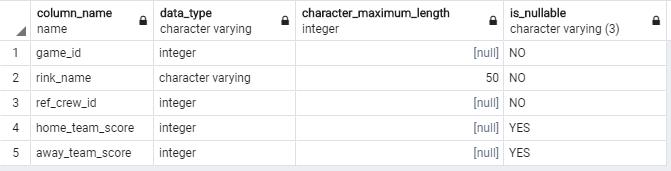
*Goalie Injury Report: *

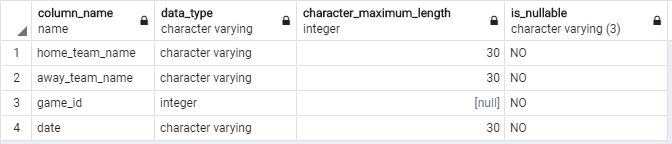
*Captain: *

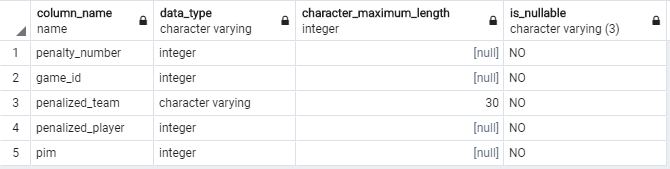
*Rink: *

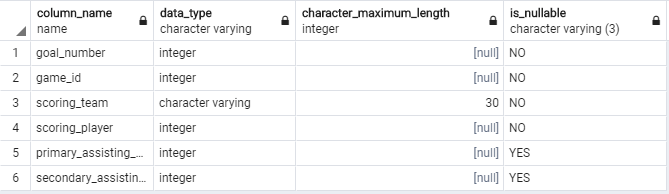
*Referee: *

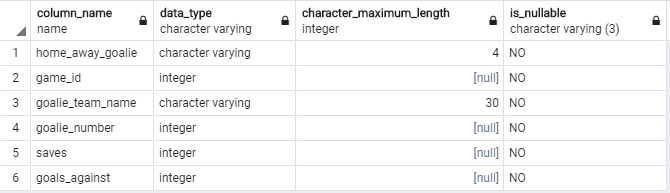
*Game Refs: *

*Game: *

*Schedule: *

*Penalty: *

*Goal: *

*Goalie Stats: *

## Advanced Features

Advanced features including triggers and functions are used to ensure that statistics entered under penalty, goal and goalie stats are shown under the player and goalie relations. After insert for a penalty, the player pim value for the accompanying player will be automatically updated by a trigger which calls a function to find the player and add the penalty to the player total. The same happens for goal to update player goals and assists (if the assisting player is not null). And the same happens for goalie stats; after insert a goalie will have their goals against and saves added to their season statistics.

Exact implementation details will not be included in this section since they are included in appendix A, which includes all DLL’s for the project. The table below shows for each of penalty, goal and goalie stats the name of the trigger and function called:

|  |  |  |
| --- | --- | --- |
|  | Trigger Name | Function Name |
| Penalty | add\_penalty\_trigger | update\_player\_pim() |
| Goal | update\_goal\_trigger | update\_player\_on\_goal() |
| Goalie Stats | update\_goalie\_stats\_trigger | update\_goalie\_on\_stats() |

## Queries

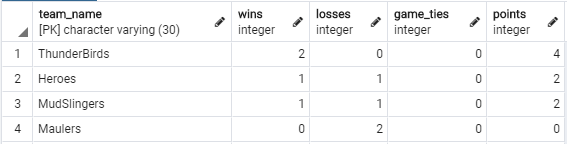
Below are some possible queries for the database. These queries may be useful to display scoring leaderboards on a website or update the database after games.

### Top Teams

This query will show the top teams in terms of points for the current season. This is similar to goal scorers again, except points are a result of wins and ties for teams. This would be useful on a website if we wanted to show the league standings.

select \* from team

order by points desc

Result: 

### Top 5 Goal Scorers

This query will show the top goal scorers for the current season. This would be useful on a website if we wanted to show the goal scorer’s leader board.

select player\_number, team\_name, player\_name, goals, assists from player

order by goals desc

limit 5

Result:



### Top 5 Point Getters

This query will show the top point scorers for the current season. This is similar to goal scorers, except points are a summation of goals and assists This would be useful on a website if we wanted to show the points leader board.

select player\_number, team\_name, player\_name, goals, assists from player

order by (goals + assists) desc

limit 5

Result:



### Active Schedule

The goal of the active schedule query is to show the current season progress. The information will contain the playing teams, the date they are playing, the rink they are playing at, and the score if the game has been completed. This is useful information for players trying to find their next game, coaches for scouting other team’s games, and league administrators to track the progress of the league. This information could be directly pasted into a website table.

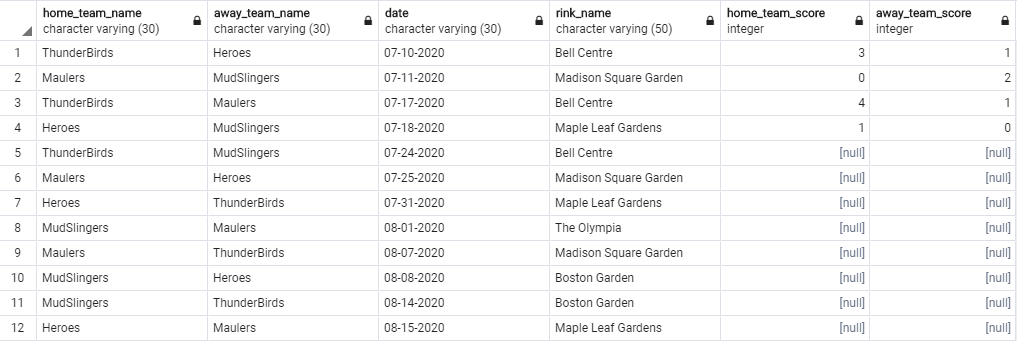
select home\_team\_name, away\_team\_name, date, rink\_name, home\_team\_score, away\_team\_score

from schedule

join game on

(schedule.game\_id = game.game\_id)

Result:



### Next Game for Team

This query will find the next game on the schedule for a given team. This would allow for a quick search for the next game for a player that does not have the time to consult the whole league table and figure out which game is next. The team name has to be included in this query, and for this instance, I was looking for the next game for the ‘Maulers’. Any team could be chosen, and an external website could change the query with a different team name to display for a different team. This could be formatted prominently on a team webpage for players and coaches to see their next game.

(select game.rink\_name, schedule.date, schedule.home\_team\_name, schedule.away\_team\_name from game

join schedule on

(game.game\_id = schedule.game\_id)

where (home\_team\_score is null) and

(home\_team\_name = 'Maulers' or away\_team\_name = 'Maulers'))

order by game.game\_id asc

limit 1

Result: 

### Show Team Roster

This query is a simple query to filter some extra information to show the player roster for a given team. This can be used on a team webpage to show information for the players. The team name needs to be included in the query, so following previous convention, this shows the ‘Maulers’ roster.

select player\_number, player\_name, goals, assists, pim from player

where team\_name = 'Maulers'

Result:



### Show Team Roster with Goalies

In the interest of including the goalies as well with the rest of the team, a query is shown below to list out all of the players on an active roster for a given team. We choose the ‘Maulers’ once again for this query.

select team\_name, player\_name from player

where team\_name = 'Maulers'

union

select team\_name, goalie\_name from goalie

where team\_name = 'Maulers'

Result:



# CRUD Matrix

## List of Entity Types

* Team
* Player
* Goalie
* Player Injury Report
* Goalie Injury Report
* Captain
* Rink
* Referee
* Game Refs
* Game
* Schedule
* Penalty
* Goal
* Goalie Stats

## List of Functions

* Add\_Player() – Adds a player to a team
* Remove\_Player() – Removes a player from a team
* Trade\_Player() – Moves a player from one team to another
* Add\_Goalie() – Adds a goalie to a team
* Remove\_Goalie() – Removes a goalie from a team
* Trade\_Goalie() – Moves a goalie from one team to another
* Fire\_Coach() – Removes a coach from a team
* Hire\_Coach() – Adds a coach to a team
* Add\_Goal() – Insert values to goal
* Add\_Penalty() – Insert values to penalty
* Add\_Goalie\_Stats() – Insert values to goalie stats
* Insert\_Game\_Score() – Add home and away team scores to game and updates the team wins, losses, ties and points
* Add\_Team() – Add a new team to the league
* Remove\_Team() – Remove a team from the league
* Create\_Schedule() – Create the season schedule

## CRUD Matrix



# Concluding Remarks

Lessons learned and strengths and weaknesses, what you may add to the database project if you have more time.

Throughout the database project I learned a lot about database implementation in SQL, which is exactly what I set out to do in this course, so for that I am thankful. In terms of strengths, I think I have a very good grasp of how to design a database in a way that makes sense with foreign keys and primary keys and relationships. One possible weakness still is with more complicated SQL queries, for example the query from HW4 was difficult for me. However, as I practice more I am getting better, and some of the queries included in this report were relatively complex.

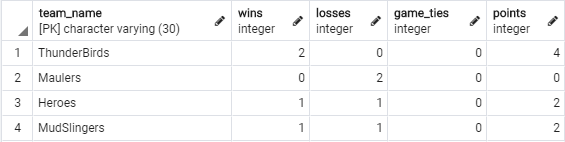
Some items I may add to the database with more time include more automated ways to update team and game statistics with scores, wins and points. In the current implementation those are entered manually, however with the proposed game sheet entity this could potentially be done automatically with triggers and functions. Another item I may add would be more statistics entities like player shots, corsi, hits, etc. This could be fun to track more statistics, and eventually do some analytics on the data the way they do in the NHL. However this can be added to the existing implementation rather easily. Another thing that could be added to the database is some way to manage playoffs, as there is no current way to do that. Some brainstorming would have to be done to accomplish this.

Appendices

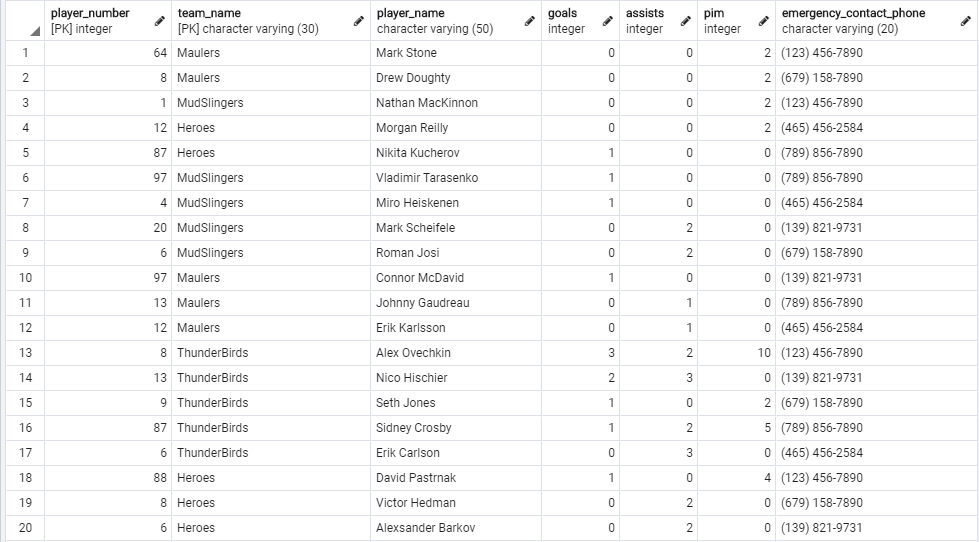
Appendix A – Table Contents by Entity

The contents of this section are taken directly with the SQL command  
SELECT \* FROM <entity>  
Where entity is specified above the table values.

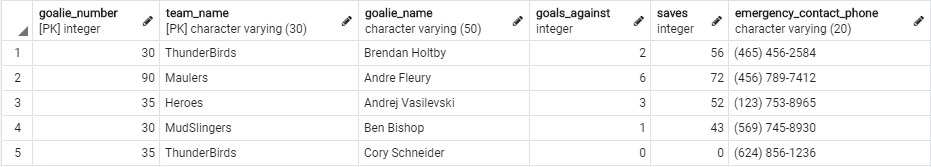
*Team:*

**

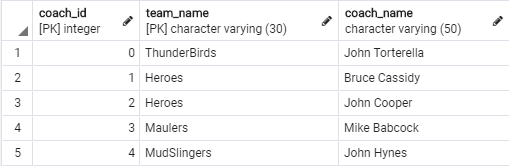
*Player:*

**

*Goalie:*

**

*Coach:*

**

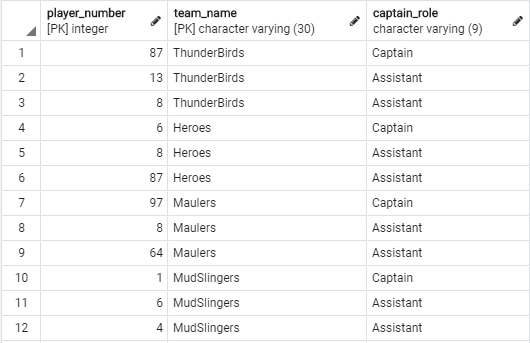
*Player Injury Report:*

**

*Goalie Injury Report:*

**

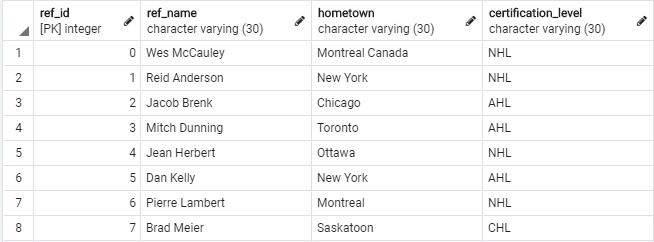
*Captain:*

**

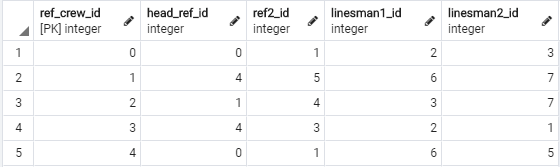
*Rink:*

**

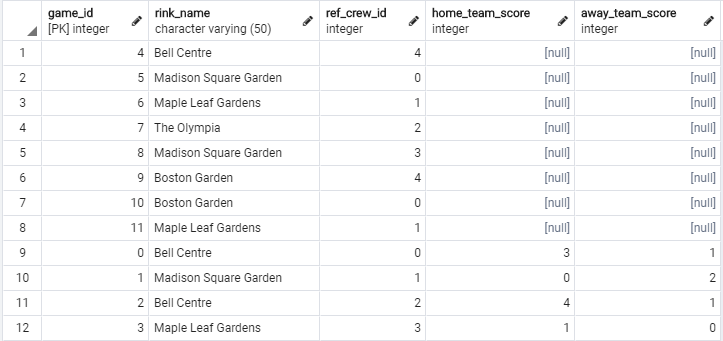
*Referee:*

**

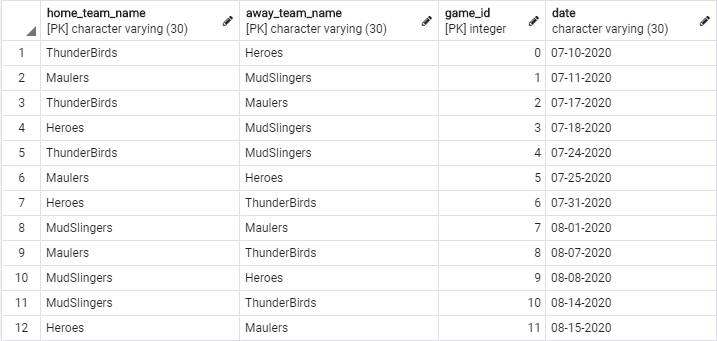
*Game Refs:*

**

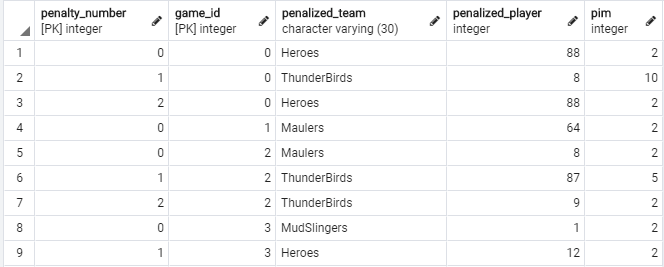
*Game:*

**

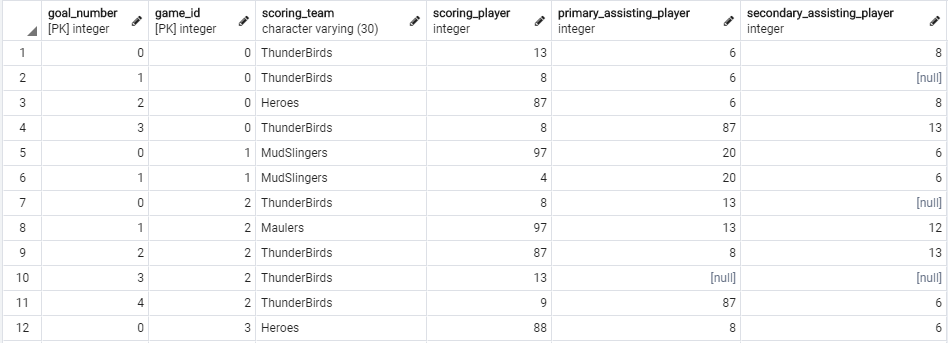
*Schedule:*

**

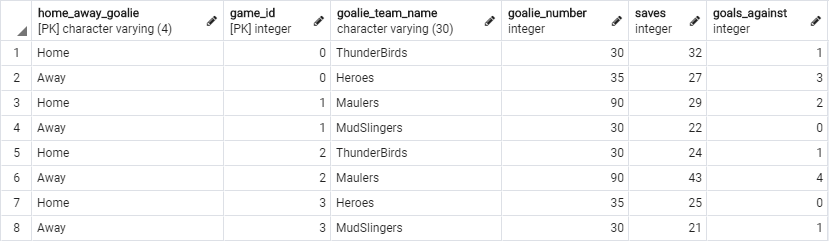
*Penalty:*

**

*Goal:*

**

*Goalie Stats:*

**

Appendix B - DDL, INSERT, SELECT, etc. Statements

CREATE TABLE team (

team\_name VARCHAR(30) NOT NULL,

wins INT NOT NULL,

losses INT NOT NULL,

game\_ties INT NOT NULL,

points INT NOT NULL,

CONSTRAINT team\_name\_pk

PRIMARY KEY (team\_name)

);

insert into team(team\_name, wins, losses, game\_ties, points) values

('ThunderBirds', 0, 0, 0, 0);

insert into team(team\_name, wins, losses, game\_ties, points) values

('Heroes', 0, 0, 0, 0);

insert into team(team\_name, wins, losses, game\_ties, points) values

('Maulers', 0, 0, 0, 0);

insert into team(team\_name, wins, losses, game\_ties, points) values

('MudSlingers', 0, 0, 0, 0);

CREATE TABLE player (

player\_number INT NOT NULL,

team\_name VARCHAR(30) NOT NULL,

player\_name VARCHAR(50) NOT NULL,

goals INT NOT NULL,

assists INT NOT NULL,

pim INT NOT NULL,

emergency\_contact\_phone VARCHAR(20),

CONSTRAINT player\_pk

PRIMARY KEY (player\_number, team\_name),

CONSTRAINT team\_fk

FOREIGN KEY (team\_name) REFERENCES team(team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT player\_number\_vals

CHECK (0 < player\_number),

CHECK (player\_number < 100)

);

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(8, 'ThunderBirds', 'Alex Ovechkin', 0, 0, 0, '(123) 456-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(87, 'ThunderBirds', 'Sidney Crosby', 0, 0, 0, '(789) 856-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(13, 'ThunderBirds', 'Nico Hischier', 0, 0, 0, '(139) 821-9731');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(9, 'ThunderBirds', 'Seth Jones', 0, 0, 0, '(679) 158-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(6, 'ThunderBirds', 'Erik Carlson', 0, 0, 0, '(465) 456-2584');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(88, 'Heroes', 'David Pastrnak', 0, 0, 0, '(123) 456-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(87, 'Heroes', 'Nikita Kucherov', 0, 0, 0, '(789) 856-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(6, 'Heroes', 'Alexsander Barkov', 0, 0, 0, '(139) 821-9731');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(8, 'Heroes', 'Victor Hedman', 0, 0, 0, '(679) 158-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(12, 'Heroes', 'Morgan Reilly', 0, 0, 0, '(465) 456-2584');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(64, 'Maulers', 'Mark Stone', 0, 0, 0, '(123) 456-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(13, 'Maulers', 'Johnny Gaudreau', 0, 0, 0, '(789) 856-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(97, 'Maulers', 'Connor McDavid', 0, 0, 0, '(139) 821-9731');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(8, 'Maulers', 'Drew Doughty', 0, 0, 0, '(679) 158-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(12, 'Maulers', 'Erik Karlsson', 0, 0, 0, '(465) 456-2584');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(1, 'MudSlingers', 'Nathan MacKinnon', 0, 0, 0, '(123) 456-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(97, 'MudSlingers', 'Vladimir Tarasenko', 0, 0, 0, '(789) 856-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(20, 'MudSlingers', 'Mark Scheifele', 0, 0, 0, '(139) 821-9731');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(6, 'MudSlingers', 'Roman Josi', 0, 0, 0, '(679) 158-7890');

INSERT into player(player\_number, team\_name, player\_name, goals, assists, pim, emergency\_contact\_phone) values

(4, 'MudSlingers', 'Miro Heiskenen', 0, 0, 0, '(465) 456-2584');

CREATE TABLE goalie (

goalie\_number INT NOT NULL,

team\_name VARCHAR(30) NOT NULL,

goalie\_name VARCHAR(50) NOT NULL,

goals\_against INT NOT NULL,

saves INT NOT NULL,

emergency\_contact\_phone VARCHAR(20),

CONSTRAINT goalie\_pk

PRIMARY KEY (goalie\_number, team\_name),

CONSTRAINT team\_fk

FOREIGN KEY (team\_name) REFERENCES team(team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT goalie\_number\_vals

CHECK (0 < goalie\_number),

CHECK (goalie\_number < 100)

);

INSERT into goalie(goalie\_number, team\_name, goalie\_name, goals\_against, saves, emergency\_contact\_phone) values

(30, 'ThunderBirds', 'Brendan Holtby', 0, 0, '(465) 456-2584');

INSERT into goalie(goalie\_number, team\_name, goalie\_name, goals\_against, saves, emergency\_contact\_phone) values

(35, 'ThunderBirds', 'Cory Schneider', 0, 0, '(624) 856-1236');

INSERT into goalie(goalie\_number, team\_name, goalie\_name, goals\_against, saves, emergency\_contact\_phone) values

(35, 'Heroes', 'Andrej Vasilevski', 0, 0, '(123) 753-8965');

INSERT into goalie(goalie\_number, team\_name, goalie\_name, goals\_against, saves, emergency\_contact\_phone) values

(90, 'Maulers', 'Andre Fleury', 0, 0, '(456) 789-7412');

INSERT into goalie(goalie\_number, team\_name, goalie\_name, goals\_against, saves, emergency\_contact\_phone) values

(30, 'MudSlingers', 'Ben Bishop', 0, 0, '(569) 745-8930');

CREATE TABLE coach (

coach\_id INT NOT NULL,

team\_name VARCHAR(30) NOT NULL,

coach\_name VARCHAR(50) NOT NULL,

CONSTRAINT coach\_pk

PRIMARY KEY (coach\_id, team\_name),

CONSTRAINT team\_fk

FOREIGN KEY (team\_name) REFERENCES team(team\_name)

ON DELETE SET NULL ON UPDATE CASCADE

);

INSERT into coach(coach\_id, team\_name, coach\_name) values

(0, 'ThunderBirds', 'John Torterella');

INSERT into coach(coach\_id, team\_name, coach\_name) values

(1, 'Heroes', 'Bruce Cassidy');

INSERT into coach(coach\_id, team\_name, coach\_name) values

(2, 'Heroes', 'John Cooper');

INSERT into coach(coach\_id, team\_name, coach\_name) values

(3, 'Maulers', 'Mike Babcock');

INSERT into coach(coach\_id, team\_name, coach\_name) values

(4, 'MudSlingers', 'John Hynes');

CREATE TABLE player\_injury\_report (

injury\_id INT NOT NULL,

player\_number INT NOT NULL,

team\_name VARCHAR(30) NOT NULL,

injury VARCHAR(50) NOT NULL,

injury\_date VARCHAR(50),

injury\_description VARCHAR(300),

CONSTRAINT injury\_pk

PRIMARY KEY (injury\_id, player\_number, team\_name),

CONSTRAINT player\_team\_fk

FOREIGN KEY (team\_name, player\_number) REFERENCES player(team\_name, player\_number)

ON DELETE SET NULL ON UPDATE CASCADE

);

INSERT into player\_injury\_report(injury\_id, player\_number, team\_name, injury, injury\_date, injury\_description) values

(0, 87, 'ThunderBirds', 'Concussion', '07-12-2012', 'Mild concussion');

INSERT into player\_injury\_report(injury\_id, player\_number, team\_name, injury, injury\_date, injury\_description) values

(1, 6, 'Heroes', 'Pulled Hamstring', '09-14-2014', 'Severe hamstring tear with minor ligament damage');

INSERT into player\_injury\_report(injury\_id, player\_number, team\_name, injury, injury\_date, injury\_description) values

(2, 87, 'ThunderBirds', 'Concussion', '12-19-2015', 'Severe concussion');

INSERT into player\_injury\_report(injury\_id, player\_number, team\_name, injury, injury\_date, injury\_description) values

(3, 20, 'MudSlingers', 'Broken Ankle', '02-14-2016', 'Broken ankle from blocking a shot. No surgery needed');

INSERT into player\_injury\_report(injury\_id, player\_number, team\_name, injury, injury\_date, injury\_description) values

(4, 13, 'Maulers', 'Broken Wrist', '02-20-2016', 'Broken wrist from slash. Minor surgery needed to repair ligaments');

INSERT into player\_injury\_report(injury\_id, player\_number, team\_name, injury, injury\_date, injury\_description) values

(5, 87, 'ThunderBirds', 'Concussion', '12-19-2019', 'Moderate concussion');

CREATE TABLE goalie\_injury\_report (

injury\_id INT NOT NULL,

goalie\_number INT NOT NULL,

team\_name VARCHAR(30) NOT NULL,

injury VARCHAR(50) NOT NULL,

injury\_date VARCHAR(50),

injury\_description VARCHAR(300),

CONSTRAINT goalie\_injury\_pk

PRIMARY KEY (injury\_id, goalie\_number, team\_name),

CONSTRAINT goalie\_team\_fk

FOREIGN KEY (team\_name, goalie\_number) REFERENCES goalie(team\_name, goalie\_number)

ON DELETE SET NULL ON UPDATE CASCADE

);

INSERT into goalie\_injury\_report(injury\_id, goalie\_number, team\_name, injury, injury\_date, injury\_description) values

(0, 30, 'ThunderBirds', 'Hipflexer Strain', '12-15-2014', 'Mildly strained hipflexer');

INSERT into goalie\_injury\_report(injury\_id, goalie\_number, team\_name, injury, injury\_date, injury\_description) values

(1, 90, 'Maulers', 'Concussion', '12-04-2017', 'Mild Concussion');

INSERT into goalie\_injury\_report(injury\_id, goalie\_number, team\_name, injury, injury\_date, injury\_description) values

(2, 35, 'Heroes', 'Broken Pinkie', '02-02-2018', 'Impact broken pinkie. No surgery needed. Ice reguarly');

INSERT into goalie\_injury\_report(injury\_id, goalie\_number, team\_name, injury, injury\_date, injury\_description) values

(3, 30, 'MudSlingers', 'Broken Clavicle', '12-15-2019', 'Clavicle broken on shot. Surgery needed');

CREATE TABLE captain (

player\_number INT NOT NULL,

team\_name VARCHAR(30) NOT NULL,

captain\_role VARCHAR(9) NOT NULL,

CONSTRAINT captain\_pk

PRIMARY KEY (player\_number, team\_name),

CONSTRAINT captain\_fk

FOREIGN KEY (player\_number, team\_name) REFERENCES player(player\_number, team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT captain\_role\_values

CHECK (captain\_role IN ('Captain', 'Assistant'))

);

INSERT into captain(player\_number, team\_name, captain\_role) values

(87, 'ThunderBirds', 'Captain');

INSERT into captain(player\_number, team\_name, captain\_role) values

(13, 'ThunderBirds', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(8, 'ThunderBirds', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(6, 'Heroes', 'Captain');

INSERT into captain(player\_number, team\_name, captain\_role) values

(8, 'Heroes', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(87, 'Heroes', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(97, 'Maulers', 'Captain');

INSERT into captain(player\_number, team\_name, captain\_role) values

(8, 'Maulers', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(64, 'Maulers', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(1, 'MudSlingers', 'Captain');

INSERT into captain(player\_number, team\_name, captain\_role) values

(6, 'MudSlingers', 'Assistant');

INSERT into captain(player\_number, team\_name, captain\_role) values

(4, 'MudSlingers', 'Assistant');

CREATE TABLE rink (

rink\_name VARCHAR(50) NOT NULL,

rink\_location VARCHAR(30) NOT NULL,

rink\_size VARCHAR(9) NOT NULL,

CONSTRAINT rink\_pk

PRIMARY KEY (rink\_name),

CONSTRAINT rink\_size\_values

CHECK (rink\_size IN ('Olympic', 'Standard'))

);

INSERT into rink(rink\_name, rink\_location, rink\_size) values

('Bell Centre', 'Montreal', 'Standard');

INSERT into rink(rink\_name, rink\_location, rink\_size) values

('Madison Square Garden', 'New York', 'Standard');

INSERT into rink(rink\_name, rink\_location, rink\_size) values

('Maple Leaf Gardens', 'Toronto', 'Standard');

INSERT into rink(rink\_name, rink\_location, rink\_size) values

('The Olympia', 'Detroit', 'Olympic');

INSERT into rink(rink\_name, rink\_location, rink\_size) values

('Boston Garden', 'Boston', 'Standard');

CREATE TABLE referee (

ref\_id INT NOT NULL,

ref\_name VARCHAR(30) NOT NULL,

hometown VARCHAR(50),

certification\_level VARCHAR(30) NOT NULL,

CONSTRAINT ref\_pk

PRIMARY KEY (ref\_id)

);

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(0, 'Wes McCauley', 'Montreal', 'NHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(1, 'Reid Anderson', 'New York', 'NHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(2, 'Jacob Brenk', 'Chicago', 'AHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(3, 'Mitch Dunning', 'Toronto', 'AHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(4, 'Jean Herbert', 'Ottawa', 'NHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(5, 'Dan Kelly', 'New York', 'AHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(6, 'Pierre Lambert', 'Montreal', 'NHL');

INSERT into referee(ref\_id, ref\_name, hometown, certification\_level) values

(7, 'Brad Meier', 'Saskatoon', 'CHL');

CREATE TABLE game\_refs (

ref\_crew\_id INT NOT NULL,

head\_ref\_id INT NOT NULL,

ref2\_id INT,

linesman1\_id INT,

linesman2\_id INT,

CONSTRAINT game\_refs\_pk

PRIMARY KEY (ref\_crew\_id),

CONSTRAINT head\_ref\_fk

FOREIGN KEY (head\_ref\_id) REFERENCES referee(ref\_id)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT ref2\_FK

FOREIGN KEY (ref2\_id) REFERENCES referee(ref\_id)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT linesman1\_fk

FOREIGN KEY (linesman1\_id) REFERENCES referee(ref\_id)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT linesman2\_fk

FOREIGN KEY (linesman2\_id) REFERENCES referee(ref\_id)

ON DELETE SET NULL ON UPDATE CASCADE

);

INSERT into game\_refs(ref\_crew\_id, head\_ref\_id, ref2\_id, linesman1\_id, linesman2\_id) values

(0, 0, 1, 2, 3);

INSERT into game\_refs(ref\_crew\_id, head\_ref\_id, ref2\_id, linesman1\_id, linesman2\_id) values

(1, 4, 5, 6, 7);

INSERT into game\_refs(ref\_crew\_id, head\_ref\_id, ref2\_id, linesman1\_id, linesman2\_id) values

(2, 1, 4, 3, 7);

INSERT into game\_refs(ref\_crew\_id, head\_ref\_id, ref2\_id, linesman1\_id, linesman2\_id) values

(3, 4, 3, 2, 1);

INSERT into game\_refs(ref\_crew\_id, head\_ref\_id, ref2\_id, linesman1\_id, linesman2\_id) values

(4, 0, 1, 6, 5);

CREATE TABLE game (

game\_id INT NOT NULL,

rink\_name VARCHAR(50) NOT NULL,

ref\_crew\_id INT NOT NULL,

home\_team\_score INT,

away\_team\_score INT,

CONSTRAINT game\_pk

PRIMARY KEY (game\_id),

CONSTRAINT rink\_name\_fk

FOREIGN KEY (rink\_name) REFERENCES rink(rink\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT ref\_crew\_id\_fk

FOREIGN KEY (ref\_crew\_id) REFERENCES game\_refs(ref\_crew\_id)

ON DELETE SET NULL ON UPDATE CASCADE

);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(0, 'Bell Centre', 0);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(1, 'Madison Square Garden', 1);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(2, 'Bell Centre', 2);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(3, 'Maple Leaf Gardens', 3);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(4, 'Bell Centre', 4);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(5, 'Madison Square Garden', 0);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(6, 'Maple Leaf Gardens', 1);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(7, 'The Olympia', 2);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(8, 'Madison Square Garden', 3);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(9, 'Boston Garden', 4);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(10, 'Boston Garden', 0);

INSERT into game(game\_id, rink\_name, ref\_crew\_id) values

(11, 'Maple Leaf Gardens', 1);

CREATE TABLE schedule (

home\_team\_name VARCHAR(30) NOT NULL,

away\_team\_name VARCHAR(30) NOT NULL,

game\_id INT NOT NULL,

date VARCHAR(30) NOT NULL,

CONSTRAINT games\_pk

PRIMARY KEY (home\_team\_name, away\_team\_name, game\_id),

CONSTRAINT home\_team\_name\_fk

FOREIGN KEY (home\_team\_name) REFERENCES team(team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT away\_team\_name\_fk

FOREIGN KEY (away\_team\_name) REFERENCES team(team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT game\_id\_fk

FOREIGN KEY (game\_id) REFERENCES game(game\_id)

ON DELETE SET NULL ON UPDATE CASCADE

);

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('ThunderBirds', 'Heroes', 0, '07-10-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('Maulers', 'MudSlingers', 1, '07-11-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('ThunderBirds', 'Maulers', 2, '07-17-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('Heroes', 'MudSlingers', 3, '07-18-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('ThunderBirds', 'MudSlingers', 4, '07-24-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('Maulers', 'Heroes', 5, '07-25-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('Heroes', 'ThunderBirds', 6, '07-31-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('MudSlingers', 'Maulers', 7, '08-01-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('Maulers', 'ThunderBirds', 8, '08-07-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('MudSlingers', 'Heroes', 9, '08-08-2020');

INSERT into schedule(home\_team\_name, away\_team\_name, game\_id, date) values

('MudSlingers', 'ThunderBirds', 10, '08-14-2020');

INSERT into schedule(home\_team\_name, away\_tfeam\_name, game\_id, date) values

('Heroes', 'Maulers', 11, '08-15-2020');

CREATE TABLE penalty (

penalty\_number INT NOT NULL,

game\_id INT NOT NULL,

penalized\_team VARCHAR(30) NOT NULL,

penalized\_player INT NOT NULL,

pim INT NOT NULL,

CONSTRAINT penalty\_pk

PRIMARY KEY (penalty\_number, game\_id),

CONSTRAINT game\_id\_fk

FOREIGN KEY (game\_id) REFERENCES game(game\_id)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT player\_id\_fk

FOREIGN KEY (penalized\_player, penalized\_team) REFERENCES player(player\_number, team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT pim\_valid\_values

CHECK (pim IN (2, 5, 10))

);

CREATE OR REPLACE FUNCTION update\_player\_pim() RETURNS TRIGGER AS

$BODY$

BEGIN

UPDATE player

SET pim = pim + NEW.pim

WHERE player\_number = NEW.penalized\_player and

team\_name = NEW.penalized\_team;

RETURN NEW;

END;

$BODY$

language plpgsql;

CREATE TRIGGER add\_penalty\_trigger

AFTER INSERT ON penalty

FOR EACH ROW

EXECUTE PROCEDURE update\_player\_pim();

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(0, 0, 'Heroes', 88, 2);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(1, 0, 'ThunderBirds', 8, 10);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(2, 0, 'Heroes', 88, 2);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(0, 1, 'Maulers', 64, 2);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(0, 2, 'Maulers', 8, 2);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(1, 2, 'ThunderBirds', 87, 5);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(2, 2, 'ThunderBirds', 9, 2);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(0, 3, 'MudSlingers', 1, 2);

INSERT into penalty(penalty\_number, game\_id, penalized\_team, penalized\_player, pim) values

(1, 3, 'Heroes', 12, 2);

CREATE TABLE goal (

goal\_number INT NOT NULL,

game\_id INT NOT NULL,

scoring\_team VARCHAR(30) NOT NULL,

scoring\_player INT NOT NULL,

primary\_assisting\_player INT,

secondary\_assisting\_player INT,

CONSTRAINT goal\_pk

PRIMARY KEY (goal\_number, game\_id),

CONSTRAINT game\_id\_fk

FOREIGN KEY (game\_id) REFERENCES game(game\_id)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT scoring\_player\_fk

FOREIGN KEY (scoring\_player, scoring\_team) REFERENCES player(player\_number, team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT p\_assisting\_player\_fk

FOREIGN KEY (primary\_assisting\_player, scoring\_team) REFERENCES player(player\_number, team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT s\_assisting\_player\_fk

FOREIGN KEY (secondary\_assisting\_player, scoring\_team) REFERENCES player(player\_number, team\_name)

ON DELETE SET NULL ON UPDATE CASCADE

);

CREATE OR REPLACE FUNCTION update\_player\_on\_goal() RETURNS TRIGGER AS

$BODY$

BEGIN

UPDATE player

SET goals = goals + 1

WHERE player\_number = NEW.scoring\_player and

team\_name = NEW.scoring\_team;

UPDATE player

SET assists = assists + 1

WHERE

player\_number = NEW.primary\_assisting\_player and

team\_name = NEW.scoring\_team and

NEW.primary\_assisting\_player IS NOT NULL;

UPDATE player

SET assists = assists + 1

WHERE

player\_number = NEW.secondary\_assisting\_player and

team\_name = NEW.scoring\_team and

NEW.secondary\_assisting\_player IS NOT NULL;

RETURN NEW;

END;

$BODY$

language plpgsql;

CREATE TRIGGER update\_goal\_trigger

AFTER INSERT ON goal

FOR EACH ROW

EXECUTE PROCEDURE update\_player\_on\_goal();

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(0, 0, 'ThunderBirds', 13, 6, 8);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(1, 0, 'ThunderBirds', 8, 6, NULL);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(2, 0, 'Heroes', 87, 6, 8);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(3, 0, 'ThunderBirds', 8, 87, 13);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(0, 1, 'MudSlingers', 97, 20, 6);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(1, 1, 'MudSlingers', 4, 20, 6);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(0, 2, 'ThunderBirds', 8, 13, NULL);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(1, 2, 'Maulers', 97, 13, 12);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(2, 2, 'ThunderBirds', 87, 8, 13);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(3, 2, 'ThunderBirds', 13, NULL, NULL);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(4, 2, 'ThunderBirds', 9, 87, 6);

INSERT into goal(goal\_number, game\_id, scoring\_team, scoring\_player, primary\_assisting\_player, secondary\_assisting\_player) values

(0, 3, 'Heroes', 88, 8, 6);

CREATE TABLE goalie\_stats (

home\_away\_goalie VARCHAR(4) NOT NULL,

game\_id INT NOT NULL,

goalie\_team\_name VARCHAR(30) NOT NULL,

goalie\_number INT NOT NULL,

saves INT NOT NULL,

goals\_against INT NOT NULL

CONSTRAINT goalie\_stats\_pk

PRIMARY KEY (home\_away\_goalie, game\_id),

CONSTRAINT goalie\_game\_id\_fk

FOREIGN KEY (game\_id) REFERENCES game(game\_id)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT goalie\_fk

FOREIGN KEY (goalie\_number, goalie\_team\_name) REFERENCES goalie(goalie\_number, team\_name)

ON DELETE SET NULL ON UPDATE CASCADE,

CONSTRAINT home\_away\_goalie\_values

CHECK (home\_away\_goalie IN ('Home', 'Away'))

);

CREATE OR REPLACE FUNCTION update\_goalie\_on\_stats() RETURNS TRIGGER AS

$BODY$

BEGIN

UPDATE goalie

SET goals\_against = goals\_against + NEW.goals\_against

WHERE goalie\_number = NEW.goalie\_number and

team\_name = NEW.goalie\_team\_name;

UPDATE goalie

SET saves = saves + NEW.saves

WHERE goalie\_number = NEW.goalie\_number and

team\_name = NEW.goalie\_team\_name;

RETURN NEW;

END;

$BODY$

language plpgsql;

CREATE TRIGGER update\_goalie\_stats\_trigger

AFTER INSERT ON goalie\_stats

FOR EACH ROW

EXECUTE PROCEDURE update\_goalie\_on\_stats();

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Home', 0, 'ThunderBirds', 30, 32, 1);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Away', 0, 'Heroes', 35, 27, 3);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Home', 1, 'Maulers', 90, 29, 2);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Away', 1, 'MudSlingers', 30, 22, 0);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Home', 2, 'ThunderBirds', 30, 24, 1);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Away', 2, 'Maulers', 90, 43, 4);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Home', 3, 'Heroes', 35, 25, 0);

INSERT into goalie\_stats(home\_away\_goalie, game\_id, goalie\_team\_name, goalie\_number, saves, goals\_against) values

('Away', 3, 'MudSlingers', 30, 21, 1);

UPDATE game

SET home\_team\_score = 3,

away\_team\_score = 1

WHERE game\_id = 0;

UPDATE game

SET home\_team\_score = 0,

away\_team\_score = 2

WHERE game\_id = 1;

UPDATE game

SET home\_team\_score = 4,

away\_team\_score = 1

WHERE game\_id = 2;

UPDATE game

SET home\_team\_score = 1,

away\_team\_score = 0

WHERE game\_id = 3;

UPDATE team

SET wins = wins + 1,

points = points + 2

WHERE team\_name = 'ThunderBirds';

UPDATE team

SET losses = losses + 1

WHERE team\_name = 'Heroes';

UPDATE team

SET wins = wins + 1,

points = points + 2

WHERE team\_name = 'MudSlingers';

UPDATE team

SET losses = losses + 1

WHERE team\_name = 'Maulers';

UPDATE team

SET wins = wins + 1,

points = points + 2

WHERE team\_name = 'ThunderBirds';

UPDATE team

SET losses = losses + 1

WHERE team\_name = 'Maulers';

UPDATE team

SET wins = wins + 1,

points = points + 2

WHERE team\_name = 'Heroes';

UPDATE team

SET losses = losses + 1

WHERE team\_name = 'MudSlingers';