**BANA 570**

**Final Report**

Group 1

Jackson Newton

Soo Hyuk Chang

Ekaterina Dunovskaia

Nolan Hamilton Thoroughgood

Muhammad Mahin

# 

[**Executive Summary 2**](#_1ulnaekewrc9)

[**Introduction 3**](#_4wofm8dgsr3p)

[**Information Requirement 4**](#_9o21yyu8xinj)

[**Conceptual Design 6**](#_db05uhwf6pkq)

[Initial ERD Design 6](#_xg8msaqhfugs)

[Business Rules 7](#_e7mhb73922wx)

[**Implementation 8**](#_fzs3k0civvxk)

[Final ERD 8](#_qqu0owly83qn)

[Design Cardinalities 8](#_2uzrkhy6byrn)

[Metadata 10](#_o96hl7z6a72w)

[Tables and Sample data 11](#_ibpc2lgxl3ua)

[**Business Reports 14**](#_i602472vl317)

[Business Report #1: The Bad Referee 14](#_uyhd668y15s7)

[Business Report #2: Team Rankings 15](#_jeboavrrucge)

[Business Report 3: The Emergency 16](#_jipqjvezss4x)

[**Further Improvement 17**](#_q7ka1ioidate)

[**Conclusion 20**](#_xvb4yk6i1lzq)

[**Works Cited 21**](#_9qd87cgdc2fc)

# 

# 

# Executive Summary

We are a small data management consulting company called Sports Data Consulting (SDC), and we have been approached by Cascade School District to create a database system for their high school soccer league (Elite Pitch High School League) scores, standings, and individual team statistics. The league currently consists of six teams, or high schools, and has only played as a league for the last five years. The league has been a significant success in the community, helping young high school students better their social and team-building skills by playing a fun, competitive game in a structured environment.

As the league has been so successful, they are looking to see if they can start looking for additional high schools to join their league. Still, they understand that some parts of the league's infrastructure are lacking and must be updated before any teams can be added. For starters, The league is currently using an Excel spreadsheet to store and manage its data, but as the league grows, so will the sheer amount of data and needs from that data, so league administrators are starting to worry about the consistency, reliability, longevity, and ease of use of their data storage at the moment. While using excel, the administrators found that the reliability of the data often depended on the person recording it. While many of the people recording games are volunteers, they are not always detail oriented. On occasion, they have found that team wins are not correctly recorded because of simple spelling errors. This has become an issue where teams are not properly ranked heading into playoffs. Additionally, making adjustments to things such as emergency contacts has been a problem in the past. It takes time to go through and update each place the emergency contact is recorded. In order to address these data management concerns, the league administrators have set out a budget of $10,000 to develop and implement a new data storage system that can fit the league's data growth needs for at least 25 years into the future.

# Introduction

To develop a new database system for the Elite Pitch High School League in Cascade School District, Sports Data Consulting has a lot of essential considerations to make based on the background information that has been provided. This involves pinpointing data storage needs like individual team statistics, league standings, and rankings. Understanding the league's structure, rules, and ensuring easy access to information from the database are essential steps. Verifying and updating the current composition of six teams should also be straightforward with the new system.

As the league expands, understanding the potential challenges related to increasing data volume becomes vital. Being aware of the types of data gathered, including scores, rankings, and team statistics, is essential for preparing for future demands. This awareness ensures that the system remains efficient and effective as the league's growth persists.

# 

# Information Requirement

The data we will be collecting will be both team and individual soccer statistics. On the team side, we will collect the statistics: minutes played as well as the outcomes of each match (win, loss, or tie). On the individual side, we will collect the statistics: goals and assists. With the use of foreign and primary keys, we will be able to provide all of these statistics in one place.

Currently, the data is collected by volunteers and inputted into the Excel document by hand. We will continue to have volunteers collect the data, but instead, it will be inputted directly into the database.

Here are further considerations for data processing and management:

* Team Statistics: To manage individual team statistics, our initial step involves collecting diverse data points such as team names, player statistics, and match outcomes. This data is currently scattered across various Excel spreadsheets. We will centralize this data into a unified database. Each team will have a dedicated record, linked to individual player profiles and match records, ensuring a cohesive and comprehensive view of each team's performance.
* League Standings: League standings are dynamic and require constant updates. Currently, this information is manually updated in Excel, which is prone to errors and delays. Our proposed system automatically updates league standings based on the latest match results. This automation ensures real-time accuracy and eliminates manual errors. Standings will include points, wins, losses, and other relevant statistics.
* Tracking and Reporting of Match Scores: Match scores are a crucial element of the database, requiring precise and timely updates. Presently, scores are entered manually into the spreadsheet after each game. In the new system, scores will be input directly into the database post-match. This data will then interlink with team statistics and league standings, allowing for instantaneous updates across the system.

Our last consideration is metadata, which, by its definition, means data about data. For example, if we have a variable “minutes played” for each player during each game, the metadata would be the total possible range of acceptable inputs for this variable or from 0 minutes and 0 seconds to the longest possible game the league allows (130 minutes). Another example of metadata in our database would be the defined primary keys of student ID and game ID. While talking about metadata, it is also important to talk about the relationships between tables and how we will need transactional tables to support our many-to-many relationships; for example, “Game Played” will merge multiple other tables together to create a ledger of games played. Lastly, one final example of metadata in our database will be “allowed game times,” as these game times will need to be preset to avoid students being scheduled for a game during their scheduled class times.

# Conceptual Design

## Initial ERD Design

# 

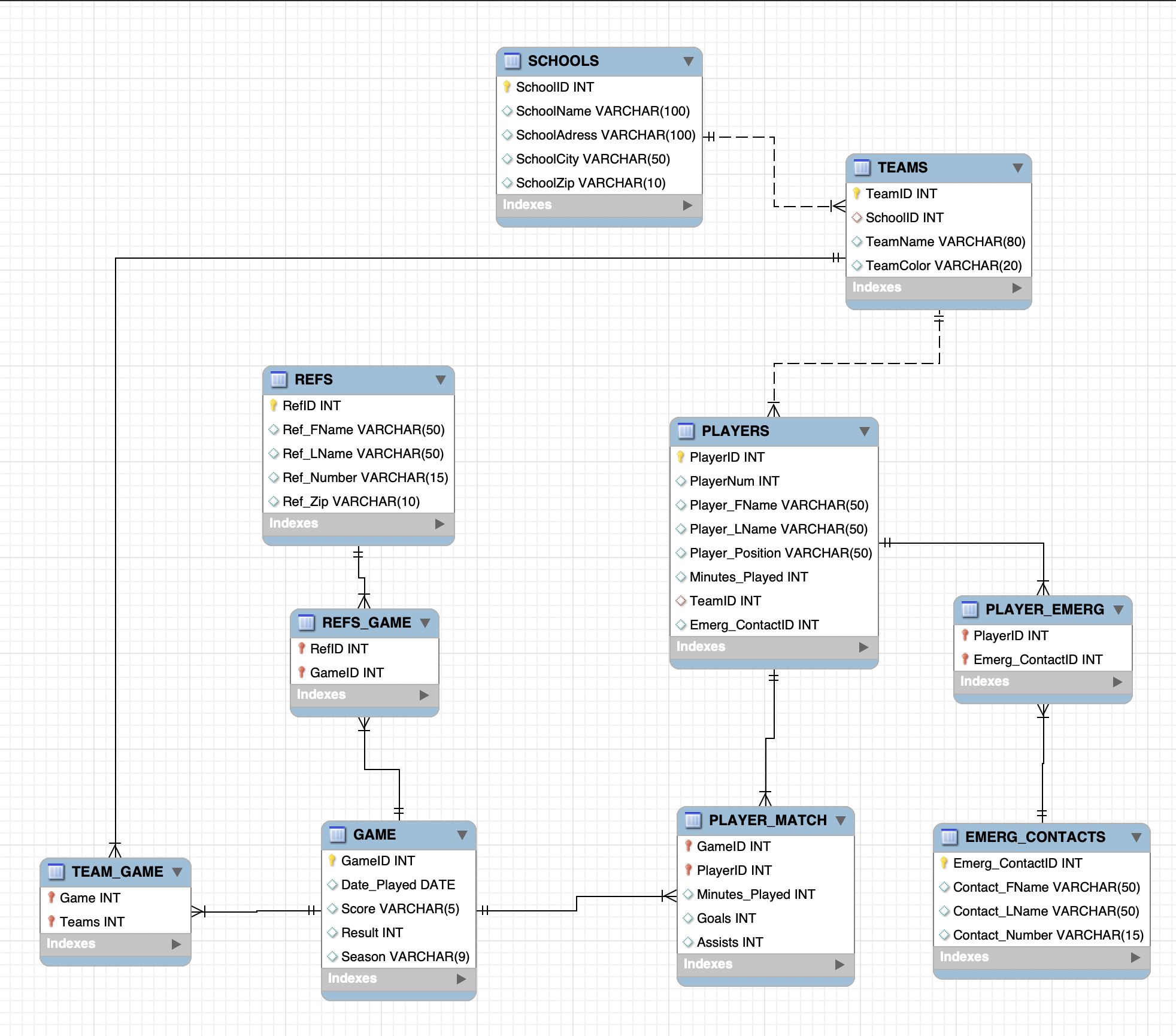
## Business Rules

The business rules include:

* Each school can have zero or more teams (a school might exist without a team for soccer).
* Each team must be associated with exactly one school.
* Each team can have zero or more players (a team might be newly formed or disbanded and thus have no current players).
* Each player must be associated with exactly one team.
* Each referee can officiate zero or more games (a referee may be new or temporarily inactive and thus not currently assigned to any games).
* Each game must have one or more referees (a game cannot be held without referees).
* Each team can participate in one or more games.
* Each game must involve two or more teams.
* Each player can have one or more emergency contacts.
* Each emergency contact can be associated with one or more players.
* The database records specific statistics for players for each game they play, including minutes played, goals, and assists.
* Each game is recorded with a unique identifier and includes information about the date played, the score, the result, and the season.

# Implementation

## Final ERD



## Design Cardinalities

The Entity-Relationship Diagram (ERD) illustrates the logical structure of a database that is used to manage information about schools, teams, players, games, and referees. The cardinalities, denoted by the line ends connecting entities, express the nature and strength of the relationships between these entities.

* The SCHOOLS table relates to the TEAMS table with a one-to-many relationship, indicated by a line ending in a crow's foot near the TEAMS table. This signifies that one school can have multiple teams, but each team is associated with only one school.
* The PLAYERS table is connected to the TEAMS table by a many-to-one relationship, also represented by a crow's foot near the PLAYERS table. This suggests that while a team comprises many players, each player belongs to exactly one team.
* The REFS (referees) table has a many-to-many relationship with the GAME table, facilitated by the REFS\_GAME bridge table. This implies that a game can have multiple referees, and a referee can officiate multiple games.
* GAMES are associated with TEAMS through the TEAM\_GAME bridge table, suggesting a many-to-many relationship. A team can participate in many games, and each game involves multiple teams.
* The PLAYER\_MATCH table serves as a bridge between PLAYERS and GAMES, which also reflects a many-to-many relationship where players participate in many games, and each game includes many players.
* The EMERG\_CONTACTS table, which holds emergency contact information, relates to the PLAYERS table through a many-to-many relationship with a PLAYER\_EMERG as a bridge entity. This implies that each player can have multiple emergency contacts and each contact can be associated with multiple players.

These cardinalities are fundamental for database operations, as they ensure referential integrity and the meaningful organization of data within the database. The relationships help in querying complex data, updating records, and maintaining consistency across the database.

## Metadata

The GAMES and PLAYERS tables could be considered the two most important in our database due to their central roles in representing the core activities (games) and participants (players).

GAMES Table Metadata:

* GameID (INT): Serves as the primary key to uniquely identify each game record. The integer data type is efficient for indexing and fast lookups.
* Date\_Played (DATE): Stores the date when the game was played. The DATE data type is appropriate for storing dates, allowing for easy sorting and filtering.
* Score (VARCHAR(5)): Holds the game's score, in the following format - "3-1". VARCHAR is used instead of a numerical type to accommodate non-numeric characters like the dash.
* Result (INT): Represents the outcome of the game (win, lose, draw). The integer corresponds to a set of predefined values (e.g., 1 for win, 2 for draw, 3 for lose).
* Season (VARCHAR(9)): Contains the season during which the game was played. VARCHAR is used to accommodate various formats, such as "2019-2020" or "Spring 2020".

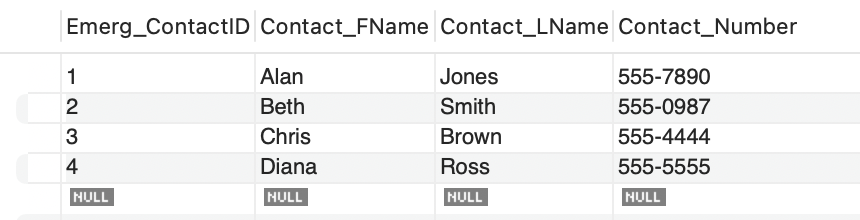
PLAYERS Table Metadata:

* PlayerID (INT): Acts as the primary key for the table, providing a unique identifier for each player.
* PlayerNum (INT): Represent the player's jersey number, which is an integer.
* Player\_FName (VARCHAR(50)) / Player\_LName (VARCHAR(50)): Store the first and last names of the players. VARCHAR(50) allows for a broad range of name lengths.
* Player\_Position (VARCHAR(50)): Holds the player's position on the team. VARCHAR is chosen to accommodate text values like "goalkeeper" or "midfielder".
* Minutes\_Played (INT): Records the number of minutes the player played, which is a numeric value.
* TeamID (INT): A foreign key that references the TEAMS table, denoting which team the player is on.
* Emerg\_ContactID (INT): A foreign key that links to the EMERG\_CONTACTS table for the player's emergency contact information.

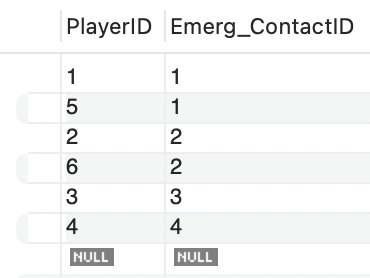
These metadata choices reflect the need to capture both numeric and textual information about games and players. They allow for effective storage, retrieval, and management of data pertaining to the operational and logistical aspects of sporting events.

## Tables and Sample data

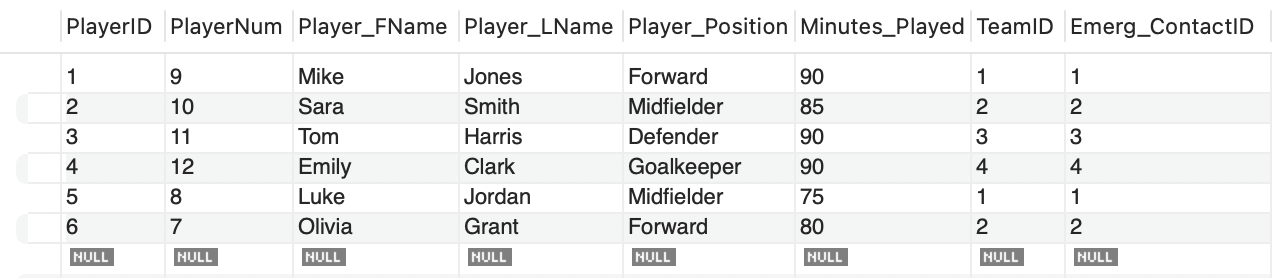
EMERG\_CONTACTS



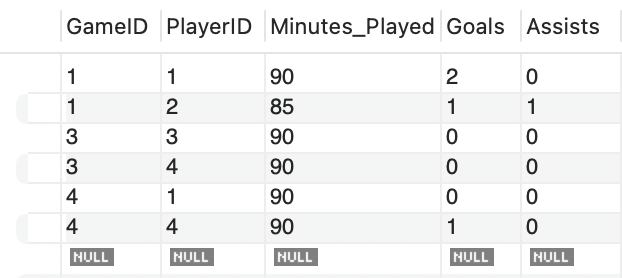
PLAYER\_EMERG



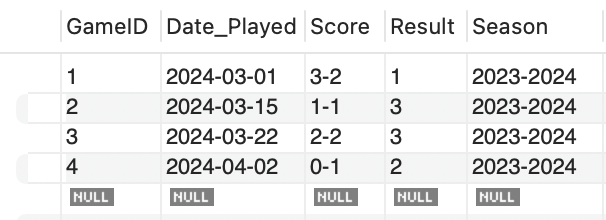
PLAYERS



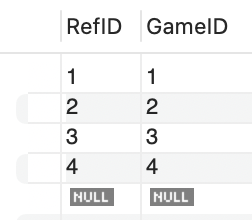
PLAYER\_MATCH



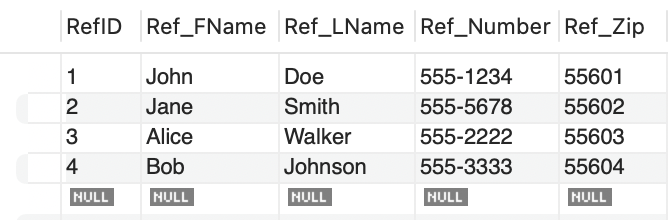
GAME



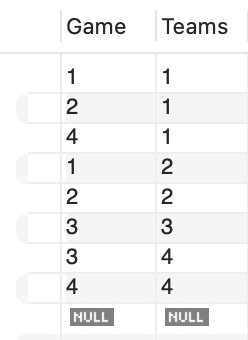
REFS\_GAME



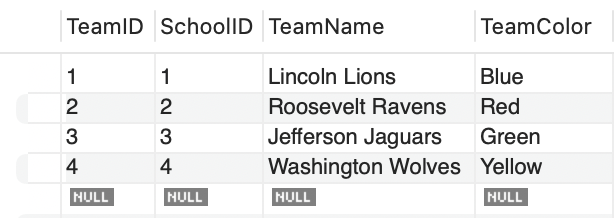
REFS



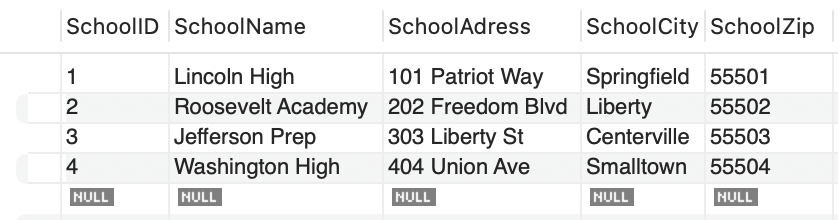
TEAM\_GAME



TEAMS



SCHOOLS



# Business Reports

## Business Report #1: The Bad Referee

*Table 1. PlayerName, SchoolName, SchoolAdress, and SchoolCity by Referee*

**The Situation:**

Sadly, the league has received complaints about one of the league officials being paid off and rigging games. The league administration would like to investigate this problem and, to facilitate this investigation, interview all the players who have played games officiated by the suspected offender.

This report was generated from the league's database, showing the names and schools, along with the addresses, of all the players who played a game officiated by the suspected official. This will allow league administrators to further investigate the suspected official by interviewing the players who played in the games in question.

**SQL Query:**

This query joins the REFS, REFS\_GAME, GAME, PLAYER\_MATCH, PLAYERS, TEAMS, and SCHOOLS tables to facilitate the generation of a results table with the player's first and last name as one and the player's school along with its address to allow league administrators' to contact and interview the identified students. The students are identified using there name or player number through their connection with the suspected official in the REFS\_GAME table, through the GAME table.

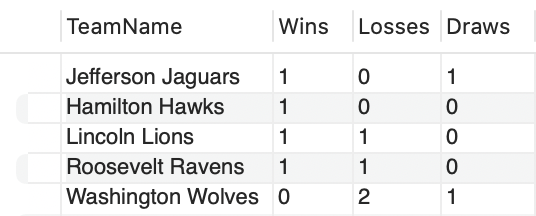
***SELECT*** *CONCAT(PLAYERS.Player\_FName, ' ', PLAYERS.Player\_LName) as 'Player Name', SchoolName, SchoolAdress, SchoolCity*

***FROM*** *REFS, REFS\_GAME, GAME, PLAYER\_MATCH, PLAYERS, TEAMS, SCHOOLS*

***WHERE*** *REFS.Ref\_LName = 'Walker' and REFS.RefID = REFS\_GAME.RefID and REFS\_GAME.GameID = GAME.GameID and GAME.GameID = PLAYER\_MATCH.GameID and PLAYER\_MATCH.PlayerID = PLAYERS.PlayerID and PLAYERS.TeamID = TEAMS.TeamID and TEAMS.SchoolID = SCHOOLS.SchoolID*

## Business Report #2: Team Rankings

*Table #2. Teams Ordered By Wins*



**The Situation:**

The league administration would like to know each team's win, loss, and draw totals and rank the teams by wins and draws in descending order.

**SQL Query:**

This query joins the TEAMS, TEAM\_GAME, and GAME tables, grouping by TEAM\_NAME and performing the aggregate function “sum” on each team’s win, loss, and tie columns. It then orders by Wins and Draws in descending order to get the league team rankings in descending order, with the top team being ranked on top of the league!

***SELECT*** *TEAMS.TeamName,*

*SUM(CASE WHEN GAME.Result = 1 THEN 1 ELSE 0 END) as Wins,*

*SUM(CASE WHEN GAME.Result = 2 THEN 1 ELSE 0 END) as Losses,*

*SUM(CASE WHEN GAME.Result = 3 THEN 1 ELSE 0 END) as Draws*

***FROM*** *TEAMS, TEAM\_GAME, GAME*

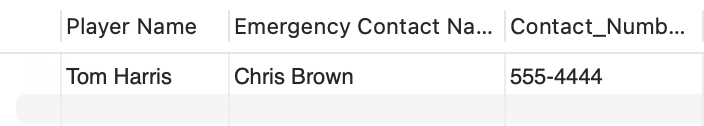
***WHERE*** *TEAMS.TeamID = TEAM\_GAME.Teams and TEAM\_GAME.Game = GAME.GameID*

***GROUP BY*** *TEAMS.TeamName*

***ORDER BY*** *Wins DESC, Draws DESC, Losses*

## Business Report 3: The Emergency

*Table #3. Emergency Contact Full Name and Phone Number*



**The Situation:**

Quick! We need the emergency contact information for the player “Tom Harris” he has broken his arm and needs someone to meet him at the hospital!

This report can quickly retrieve all the necessary information on any player, whether it's a real emergency or not. This can be important to help players get a ride home when they are sick, or facilitate league communication with legal guardians about topics such as away tournaments.

**SQL Query:**

This query joins the required tables to quickly retrieve emergency contact information. The PLAYER\_EMERG table acts as a linking table between the PLAYERS and the EMERG\_CONTACTS, which allows us to identify the player's designated emergency contact.

***SELECT*** *CONCAT(PLAYERS.Player\_FName, ' ', PLAYERS.Player\_LName) as 'Player Name', CONCAT(EMERG\_CONTACTS.Contact\_FName, ' ', EMERG\_CONTACTS.Contact\_LName) as 'Emergency Contact Name', Contact\_Number*

***FROM*** *EMERG\_CONTACTS, PLAYER\_EMERG, PLAYERS*

**WHERE** PLAYERS.Player\_LName = 'Harris' and PLAYERS.PlayerID = PLAYER\_EMERG.PlayerID and PLAYER\_EMERG.Emerg\_ContactID = EMERG\_CONTACTS.Emerg\_ContactID

# Further Improvement

Understanding database design lifecycle principles is of vital importance when building information systems. There are many phases needed for database design, some of which have already been implemented in the project. A database design also allows an evaluation and revision to be done iteratively within the database lifecycle framework.

Database lifecycle frameworks typically include three phases, which are conceptual design, logical design and physical design.

**Conceptual design:** During this phase, the structure and organization of a database are defined on a high level. The conceptual design is often represented by Entity-Relationship models (ER), which focus on entities, attributes, and relationships (Coronel and Morris 450).

**Logical design:** In this phase, the conceptual model is translated into a logical version that can be implemented with a database management system. The logical model includes the definition of tables, specifying keys and data types (Coronel and Morris 451). To eliminate redundant data, normalization is commonly performed.

**Physical design:** This phase is focused on the details of the database implementation on the selected hardware and software platforms. This phase involves making decisions regarding storage structures, indexing and performance optimization. The goal is translating the logical design to an efficient and effective structure for a database (Coronel and Morris 452).

These three phases work together to create a database that is well-structured, functional and meets the needs of an organization. Our project demonstrated our proficiency in completing these initial phases which are conceptual design and logical design. However, stages such as physical design implementation and maintenance are still outstanding and need to be implemented.

**Physical Design Phase**

Physical design is key to turning abstract conceptual designs into tangible database structures with optimized performance. Here, the focus shifts away from data presentation and toward actual implementation - including decisions regarding indexing strategies, partitioning techniques, and storage mechanisms. Our project has an established conceptual basis; now, however, we must optimize its database architecture to achieve peak performance. Identified the need for a new database system to handle the league's scores, standings, and statistics due to the limitations of the current Excel-based system.

**Implementation Phase**

Implementation marks the transition between design theory and practical application. Our chosen database management software will use its translation feature to convert our logical model of a database into physical form, and from here, our actual database is constructed based on its finalized model to ensure its compliance with conceptual designs. As with any project with conceptual designs, implementation remains key during its entirety. Outlined the types of data to be stored (e.g., team statistics, league standings, player statistics) and the relationships between them.

**Maintenance Phase**

Maintenance is an ongoing phase of the database lifecycle that adapts to the ever-evolving needs of the system. It entails constant monitoring, performance optimization and responding to changes in technology or requirements that arise over time. Even at this early stage of our project's development, an effective maintenance plan must be put in place to ensure long-term success. For example, maintenance tasks will typically include updating security protocols and adding enhancements based on user feedback and technological advances.

**Evaluation and Revision Iteratively**

Iterative design is at the heart of successful database lifecycles. Evaluation and revision processes should take place throughout their entirety to ensure our project remains flexible, responsive, and long-lasting. Iterative evaluation is the process of examining a project against predefined criteria. Examining query execution time, identifying bottlenecks, and measuring user satisfaction are all useful measures of assessment that may yield valuable insight (Coronel and Morris 462). Conducting iterative evaluation in our project could enable us to quickly recognise issues early and address them before they become more severe.

Revision in relation to the database lifecycle is both minor and significant; examples for our project could include improving data relationships in response to user feedback. Adapting database structure to accommodate new functionality or adding improved security after conducting a comprehensive vulnerability assessment (William et al.). Iterative revisions ensure the project remains agile while meeting industry standards and evolving requirements.

In conclusion, aligning our project to the principles of the Database Life Cycle provides a systematic approach to database design. Noticing unfinished phases as important, using iterative evaluations and revisions and prioritizing resilience with adaptive information systems by drawing from this comprehensive framework of DBLC will significantly enhance project efficiency and sustainability.

# Conclusion

The relational database we've developed for the Elite Pitch High School League not only addresses their prior challenges with Excel but also lays down a scalable foundation for growth. It ensures data's referential integrity, preventing errors due to disorganized information. Beyond minimizing mistakes, our database streamlines decision-making. With customized queries, league administrators can swiftly access vital information, from precise rankings and official records to reliable emergency contacts, enhancing efficiency significantly.

Our solution marks a significant starting point for the league, offering a robust platform for future enhancements. Should their data requirements evolve, the database's structure will simplify data storage and retrieval. While Excel was adequate for years, our relational database introduces a superior, faster, and more effective method for managing the league's data needs as they move forward.

# Works Cited

Coronel C & Morris, S “Database systems, design, implementation and management.” 13thed. *Cengage Learning.*

William, P., et al. "Framework for design and implementation of chat support system using natural language processing." *2023 4th International Conference on Intelligent Engineering and Management (ICIEM)*. IEEE, 2023.