

**Data Base Systems Mini Project**

A report submitted to the

Department of Electrical and Information Engineering

Faculty of Engineering

University of Ruhuna Sri Lanka

On 9th of April 2024

In completing an assignment for the module

EE4350 Data Base systems

By:

Group 28

|  |  |
| --- | --- |
| EG/2021/4549 | HERATH H.M.L.B. |
| EG/2021/4550 | HERATH H.M.L.T. |
| EG/2021/4556 | HEWAWASAM A.D.K. |

Contents

[Chapter 1 - Requirement Analysis 4](#_Toc163555433)

[1.1. Functional Requirements 4](#_Toc163555434)

[1.2. Data Requirements 4](#_Toc163555435)

[Chapter 2 - Conceptual Design 7](#_Toc163555436)

[Chapter 3 - Implementation 8](#_Toc163555437)

[3.1. Table implementation 8](#_Toc163555438)

[3.2. Creating Foreign key constraint 13](#_Toc163555439)

[3.3. Inserting data for tables 14](#_Toc163555440)

[3.4. Update and delete 19](#_Toc163555441)

[Chapter 4 - Transactions 24](#_Toc163555442)

[Chapter 5 - Tuning 35](#_Toc163555443)

**List of figures**

[Figure 2.1:ER diagram of the league management database management system 8](#_Toc163555550)

[Figure 3.1: Create League Table. 9](#_Toc163555551)

[Figure 3.2: Create Team Table. 9](#_Toc163555552)

[Figure 3.3: Create Table Player. 10](#_Toc163555553)

[Figure 3.4: Create Table Sponsor. 10](#_Toc163555554)

[Figure 3.5: Create Table League\_Sponsor. 11](#_Toc163555555)

[Figure 3.6: Create Table Team\_Sponsor. 11](#_Toc163555556)

[Figure 3.7: Create Table Referee. 11](#_Toc163555557)

[Figure 3.8: Create Table Venue. 12](#_Toc163555558)

[Figure 3.9: Create Table Game. 12](#_Toc163555559)

[Figure 3.10: Create Table Game\_Team. 12](#_Toc163555560)

[Figure 3.11: Create Table Game\_Referee. 13](#_Toc163555561)

[Figure 3.12: Create Table Player\_Stat. 13](#_Toc163555562)

[Figure 3.13: Create Table Team\_Stat. 13](#_Toc163555563)

[Figure 3.14: Create Foreign Key Constraints - I. 14](#_Toc163555564)

[Figure 3.15: Create Foreign Key Constraints - II. 14](#_Toc163555565)

[Figure 3.16: Tables in the League\_Management\_System. 15](#_Toc163555566)

[Figure 3.17: Inserting Values to League Table. 15](#_Toc163555567)

[Figure 3.18: Inserting Values to Team Table. 15](#_Toc163555568)

[Figure 3.19: Inserting Values to Player Table. 16](#_Toc163555569)

[Figure 3.20: Inserting Values to Sponsor Table. 16](#_Toc163555570)

[Figure 3.21: Inserting Values to League\_Sponsor Table. 16](#_Toc163555571)

[Figure 3.22: Inserting Values to Team\_Sponsor Table. 16](#_Toc163555572)

[Figure 3.23: Inserting Values to Referee Table. 17](#_Toc163555573)

[Figure 3.24: Inserting Values to Venue Table. 17](#_Toc163555574)

[Figure 3.25: Inserting Values to Game Table. 17](#_Toc163555575)

[Figure 3.26: Inserting Values to Game\_Referee Table. 18](#_Toc163555576)

[Figure 3.27: Inserting Values to Player\_Stat. 18](#_Toc163555577)

[Figure 3.28: Inserting Values to Team\_Stat Table. 19](#_Toc163555578)

[Figure 3.29: Updating League Table. 20](#_Toc163555579)

[Figure 3.30: Updating Team Table. 20](#_Toc163555580)

[Figure 3.31: Updating Player Table. 21](#_Toc163555581)

[Figure 3.32: Updating Sponsor Table. 21](#_Toc163555582)

[Figure 3.33: Updating Referee Table. 22](#_Toc163555583)

[Figure 3.34: Updating Venue Table. 22](#_Toc163555584)

[Figure 3.35: Updating Game Table. 23](#_Toc163555585)

[Figure 3.36: Updating Game\_Team Table. 23](#_Toc163555586)

[Figure 3.37: Updating Player\_Stat Table. 24](#_Toc163555587)

[Figure 3.38: Updating Team\_Stat Table. 24](#_Toc163555588)

[Figure 4.1: The SELECT Operation. 25](#_Toc163555589)

[Figure 4.2: The PROJECT Operation. 25](#_Toc163555590)

[Figure 4.3: The CARTESIAN PRODUCT - I. 26](#_Toc163555591)

[Figure 4.4: The CARTESIAN PRODUCT - II. 27](#_Toc163555592)

[Figure 4.5:The CARTESIAN PRODUCT - III. 27](#_Toc163555593)

[Figure 4.6: Creating USER VIEWS. 28](#_Toc163555594)

[Figure 4.7: Operation Renaming. 28](#_Toc163555595)

[Figure 4.8: Aggregation (Average). 28](#_Toc163555596)

[Figure 4.9: Using LIKE Keyword. 29](#_Toc163555597)

[Figure 4.10: Union. 30](#_Toc163555598)

[Figure 4.11: Intersection. 30](#_Toc163555599)

[Figure 4.12: Set Difference. 30](#_Toc163555600)

[Figure 4.13: Division. 31](#_Toc163555601)

[Figure 4.14: INNER JOIN. 31](#_Toc163555602)

[Figure 4.15: NATURAL JOIN. 32](#_Toc163555603)

[Figure 4.16: LEFT OUTER JOIN. 32](#_Toc163555604)

[Figure 4.17: RIGHT OUTER JOIN. 33](#_Toc163555605)

[Figure 4.18: FULL OUTER JOIN. 33](#_Toc163555606)

[Figure 4.19: OUTER UNION. 34](#_Toc163555607)

[Figure 4.20: Nested Query - I. 35](#_Toc163555608)

[Figure 4.21: Nested Query - II. 35](#_Toc163555609)

[Figure 4.22: Nested Query - III. 35](#_Toc163555610)

# Requirement Analysis

## Functional Requirements

All of the program's desired operations are incorporated step-by-step in accordance with this chapter. This database accomplishes the following things in order to do that.

• Data retrieval

• Ability to alter the database's content at any time

• Capable of completing potential data gaps or missing data

• Capability to edit data that has already been entered into the database

• The ability to add comments to data when the data is incomplete or ambiguous.

• The capacity to provide data with enhanced performance

This schema is created to represent the sport league and all the needed and maximum number of entities of this schema have been covered. They are league, team, player, sponsor, league sponsor, team sponsor, referee, venue, game, game team, game referee, player stat, team stat.. All the needed data for the schema shown in this database.

## Data Requirements

The attributes of each entity are shown below

League

* League\_ID
* League\_name
* Sport
* Start\_date
* End\_date
* Location

Team

* Team\_ID
* Team\_name
* League\_ID
* City

Player

* Player\_ID
* First\_name
* Last\_name
* Team\_ID
* Birthday
* Position
* Jersey\_number

Sponser:

* Sponser\_ID
* Company\_name
* Contact\_number
* Email
* Category

League\_Sponser:

* League\_ID
* Sponsor\_ID

Team\_Sponser:

* Team\_ID
* Sponsor\_ID

Referee:

* Referee\_ID
* First\_name
* Last\_name
* Contact\_number

Venue:

* Venue\_ID
* Venue\_name
* Capacity
* Country

Game:

* Game\_ID
* Venue\_ID
* Winning\_Team\_ID

Game\_Team:

* Game\_ID
* Team\_ID
* Winner\_flag

Game\_Refree:

* Game\_ID
* Refree\_ID

Player\_Stat:

* Match\_ID
* Player\_ID
* Goals
* Assists

Team\_Stat:

* Match\_ID
* Team\_ID
* Goals\_for
* Goals\_against

Also, we are included some important entities and relationships.

They are,

* Weak entities
* Recursive relationship
* Weak entities

There are two weak entities in our ER diagram shown, and we have shown them using a double-line rectangle. Player stat is one of them. It depends on the player entity. The player stat entity does not have a key either. Therefore, we have considered player stat as a weak entity. Another weak entity we have considered is the team stat entity. It does not have a key, and it also totally depends on the team entity. So, we consider team stat to be a weak entity.

* Recursive relationship

Here, we have considered the relationship between the team leader and the members as a recursive relationship as it exist between the same entity. A leader will lead the team, means that he alone will represent each and every one of the team. Therefore, this is a recursive relationship.

# Conceptual Design

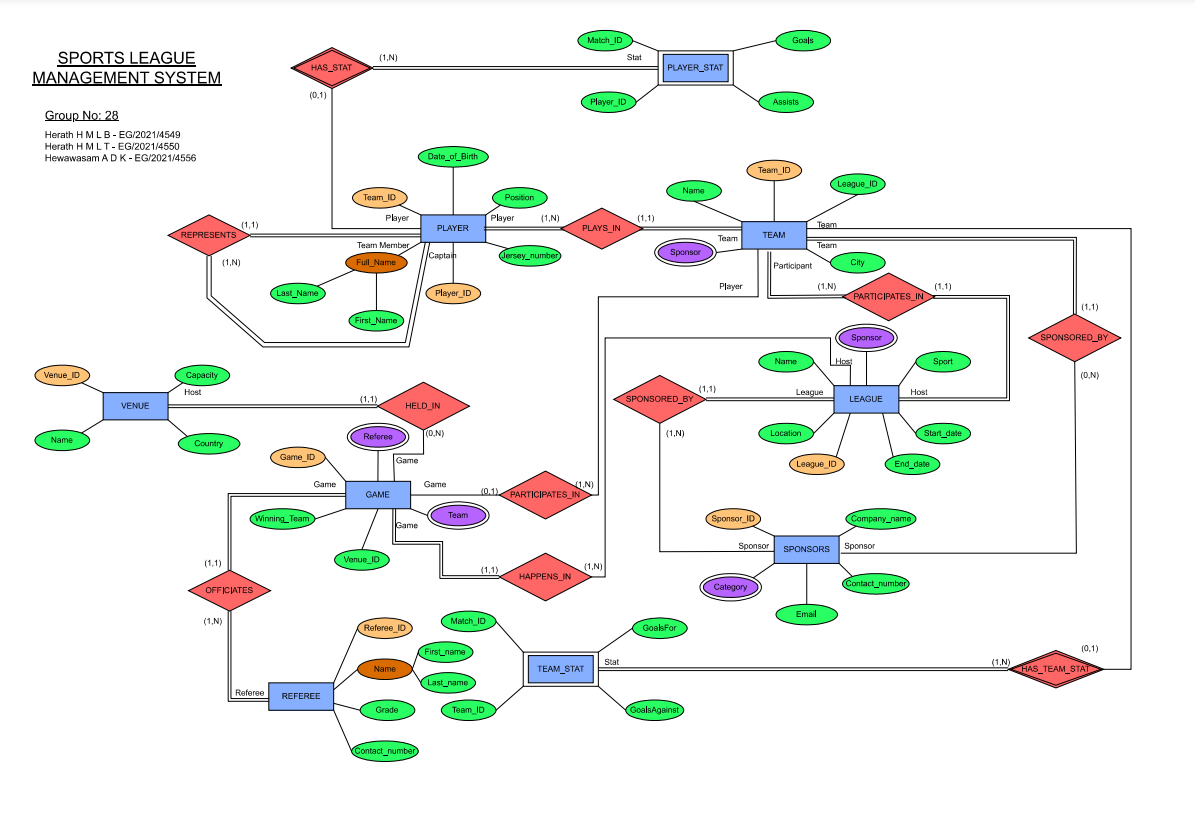


Figure 2.1:ER diagram of the league management database management system

# Implementation

In this chapter, all the screenshots of implemented database model are included.

## Table implementation

* Create table League

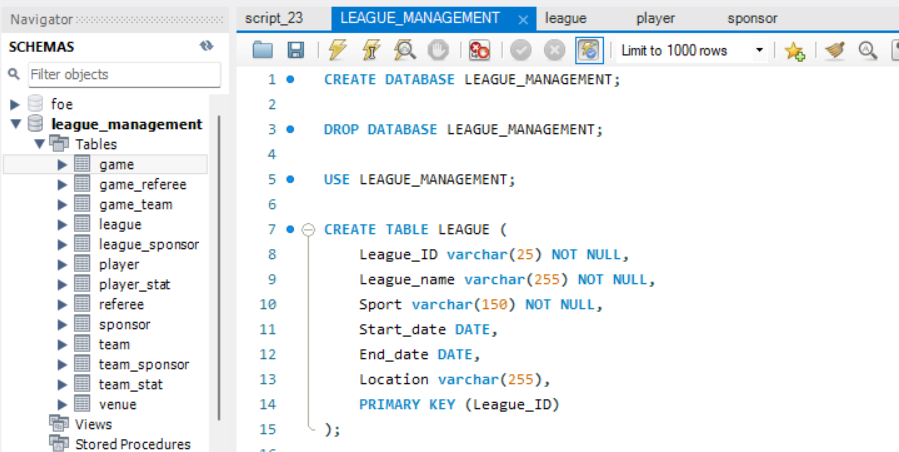


Figure 3.1: Create League Table.

* Create table Team

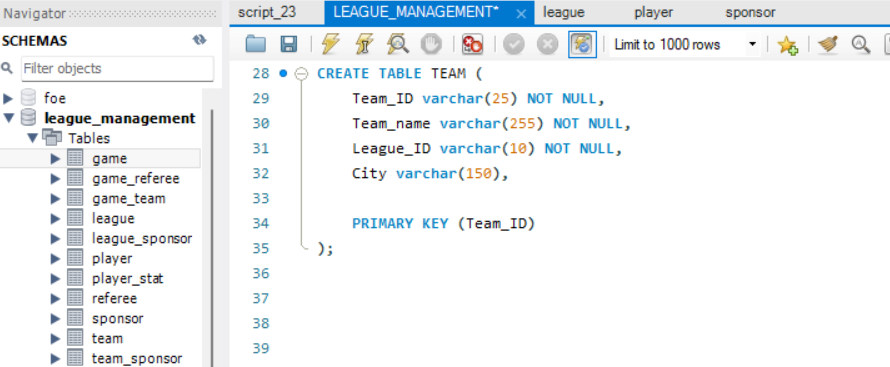


Figure 3.2: Create Team Table.

* Create table Player

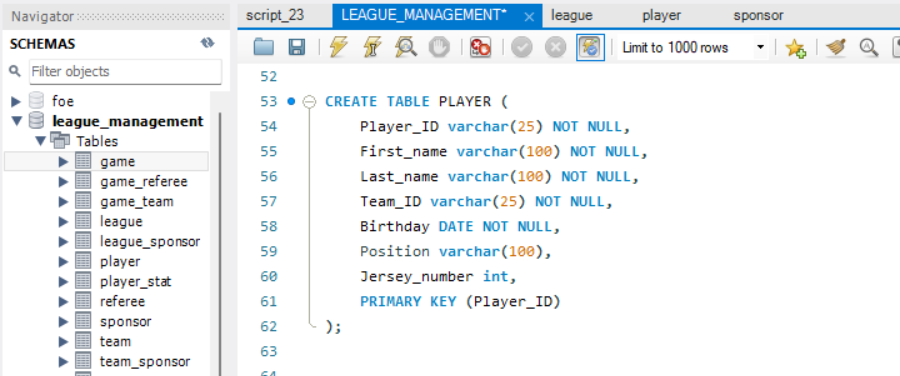


Figure 3.3: Create Table Player.

* Create table Sponsor

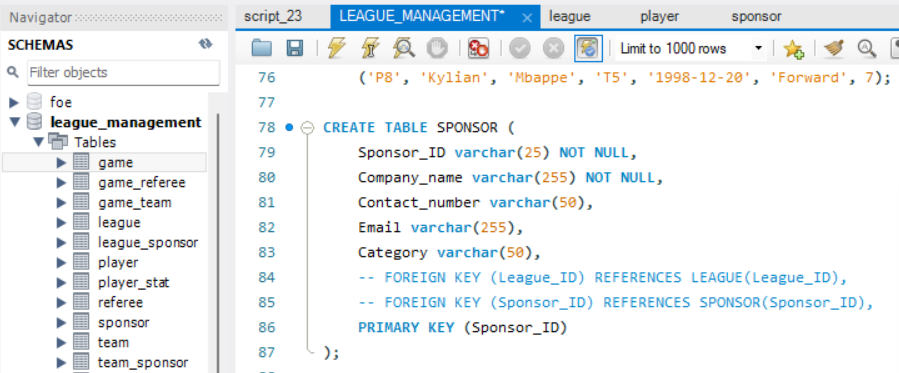


Figure 3.4: Create Table Sponsor.

* Create league\_sponser

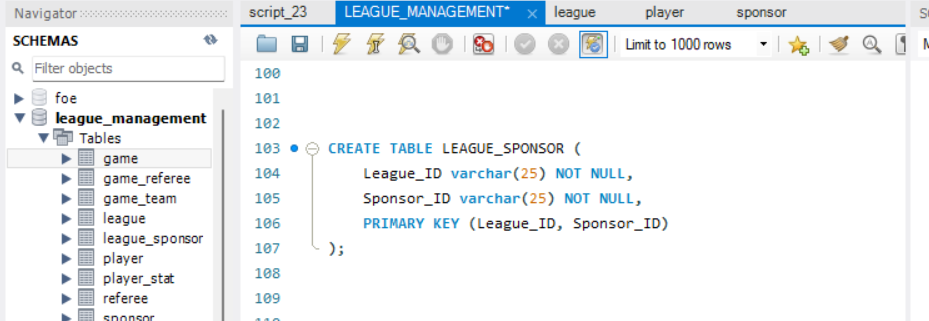


Figure 3.5: Create Table League\_Sponsor.

* Create table team\_sponser

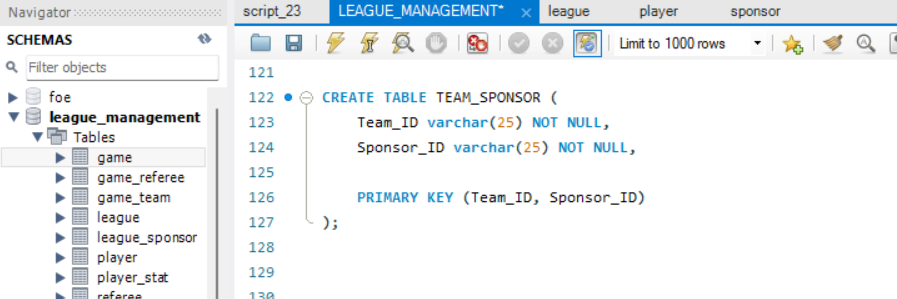


Figure 3.6: Create Table Team\_Sponsor.

* Create table referee

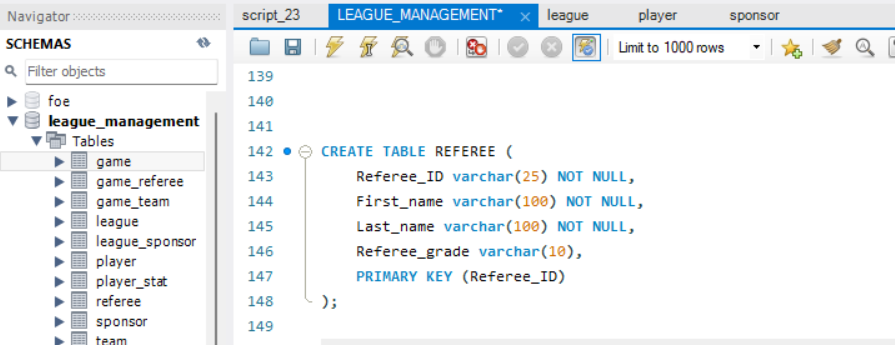


Figure 3.7: Create Table Referee.

* Create table venue

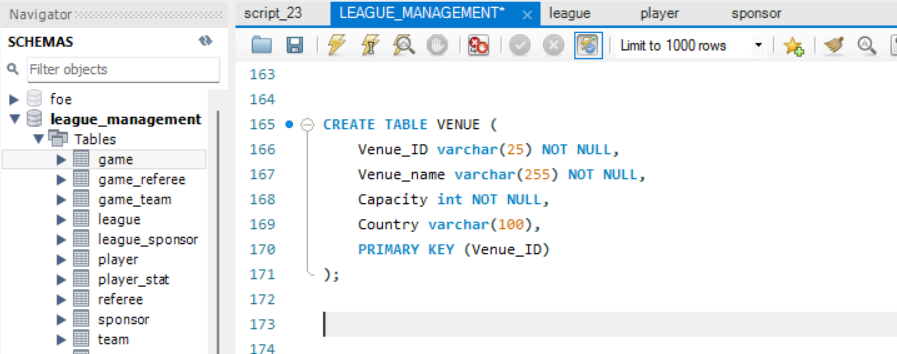


Figure 3.8: Create Table Venue.

* Create table game

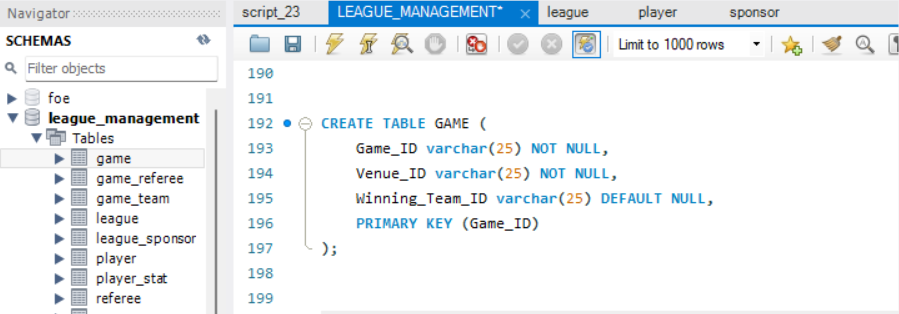


Figure 3.9: Create Table Game.

* Create table game\_team

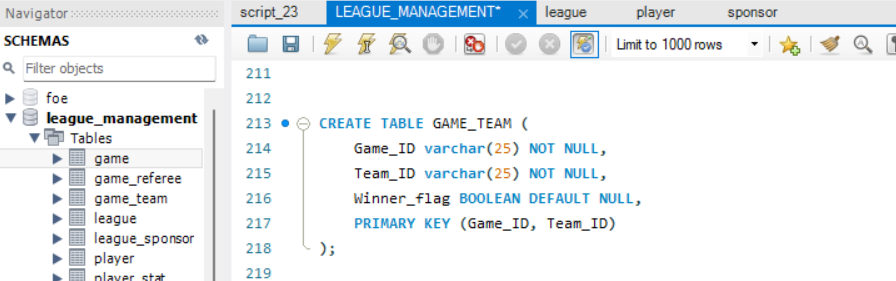


Figure 3.10: Create Table Game\_Team.

* Create table game\_refree

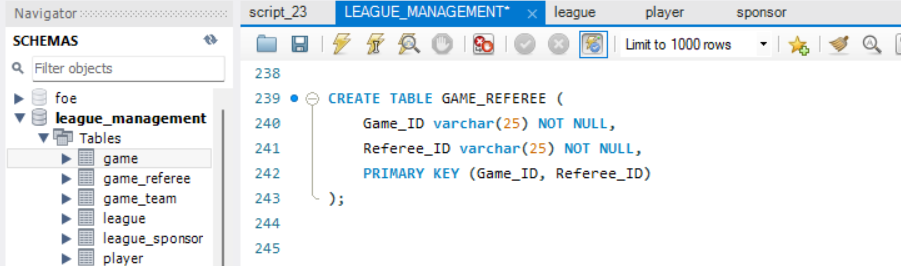


Figure 3.11: Create Table Game\_Referee.

* Create player\_stat

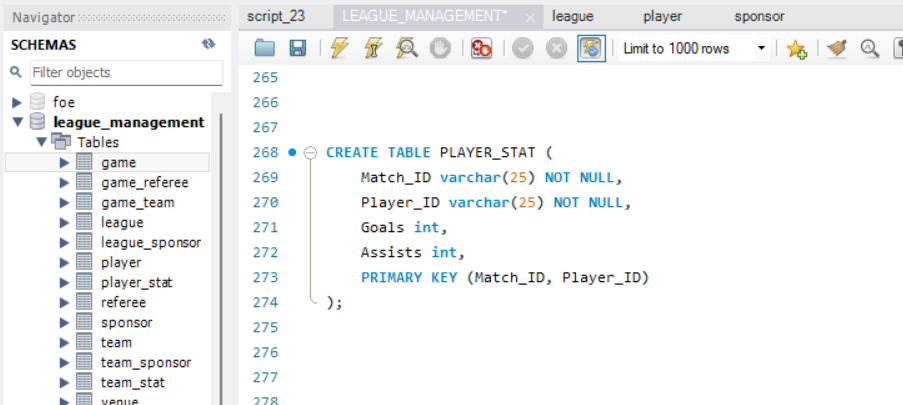


Figure 3.12: Create Table Player\_Stat.

* Create team\_stat

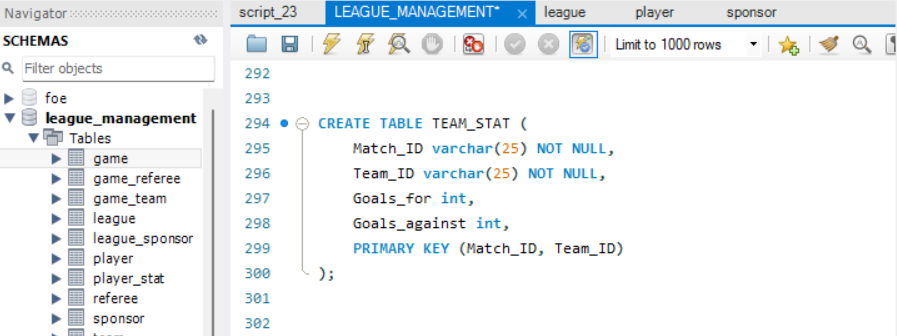


Figure 3.13: Create Table Team\_Stat.

## Creating Foreign key constraint

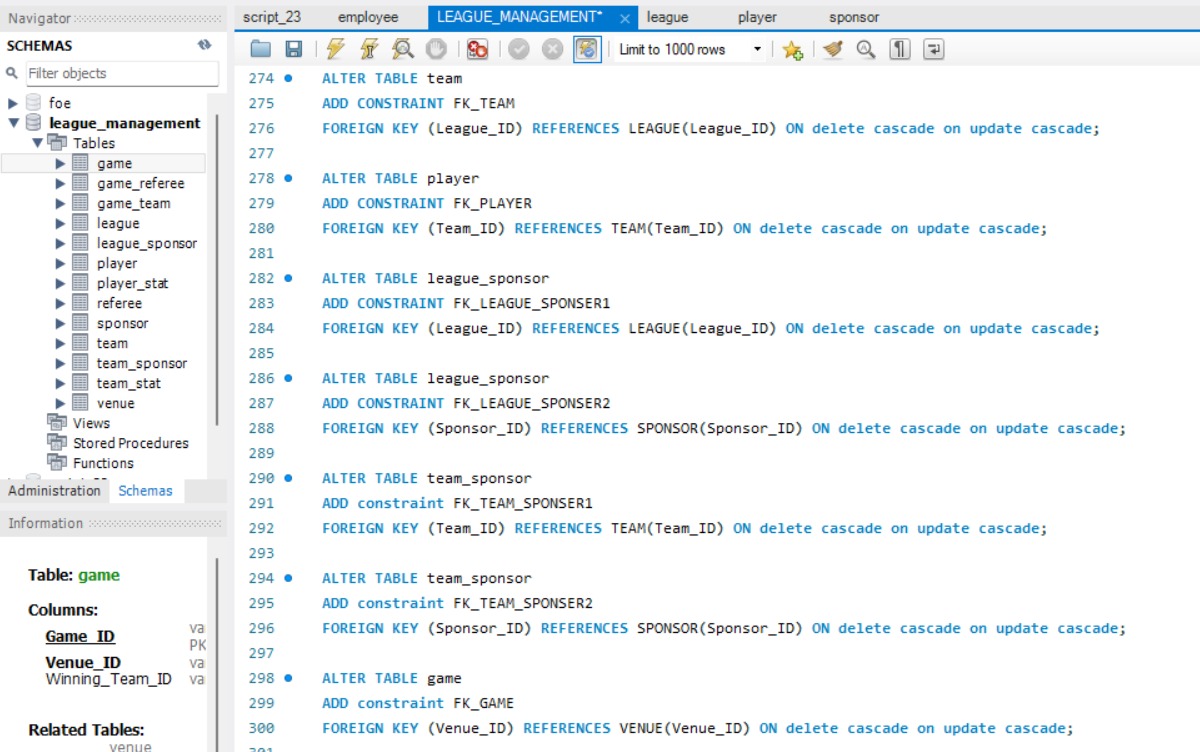


Figure 3.14: Create Foreign Key Constraints - I.

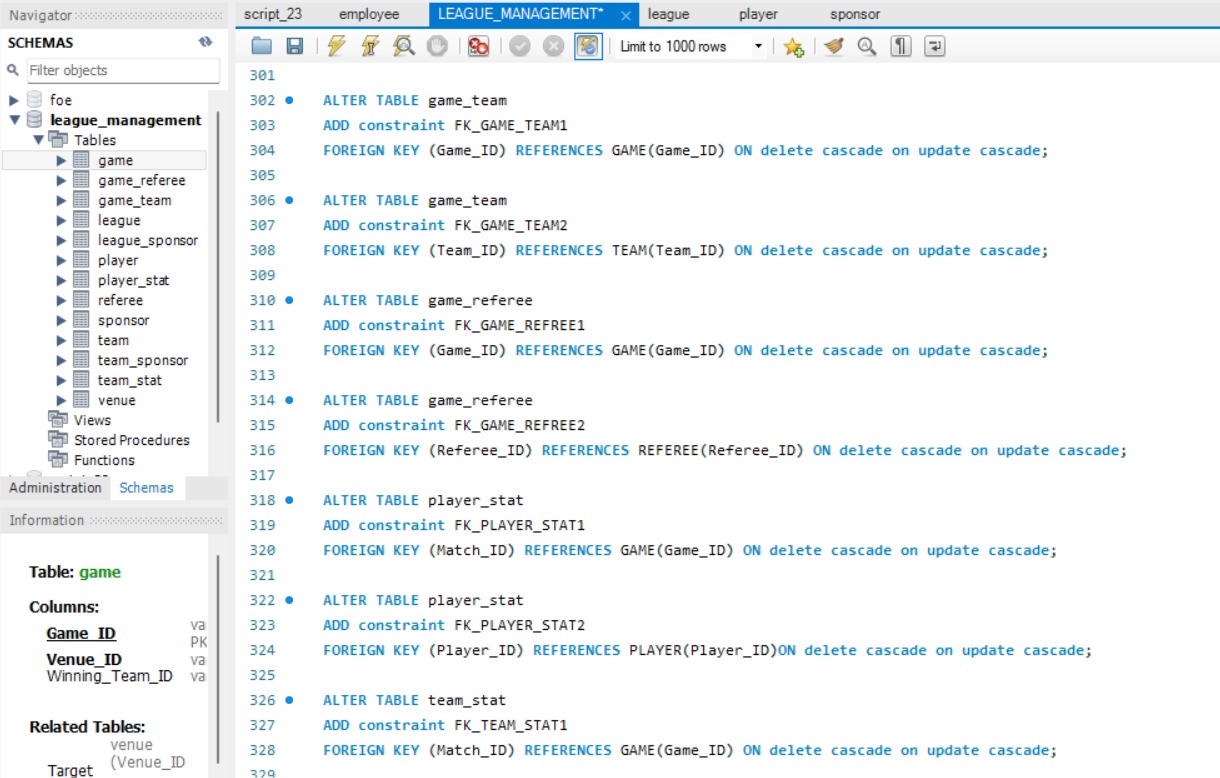


Figure 3.15: Create Foreign Key Constraints - II.

## Inserting data for tables



Figure 3.16: Tables in the League\_Management\_System.

* TABLE LEAGUE

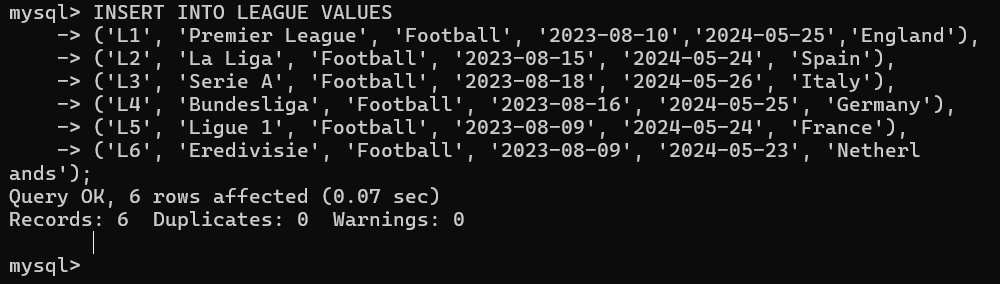


Figure 3.17: Inserting Values to League Table.

* TABLE TEAM



Figure 3.18: Inserting Values to Team Table.

* PLAYER

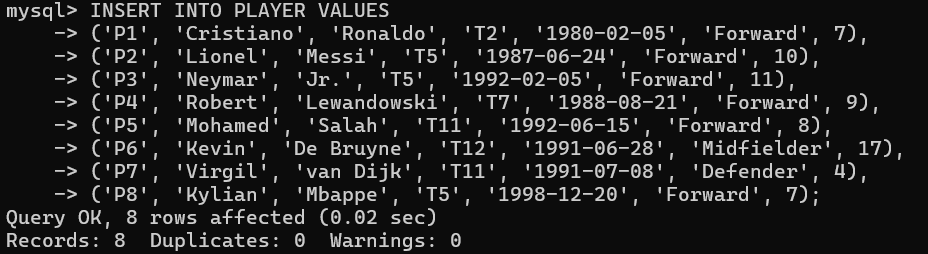


Figure 3.19: Inserting Values to Player Table.

* SPONSOR

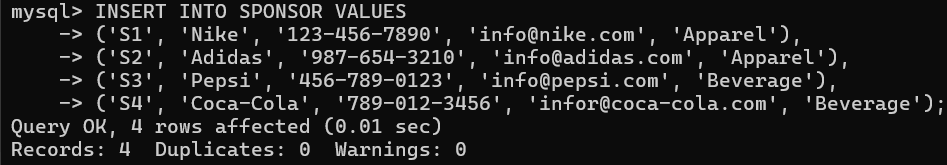


Figure 3.20: Inserting Values to Sponsor Table.

* LEAGUE\_SPONSOR

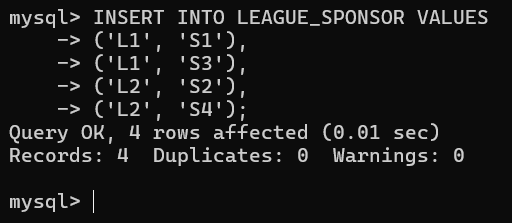


Figure 3.21: Inserting Values to League\_Sponsor Table.

* TEAM\_SPONSOR

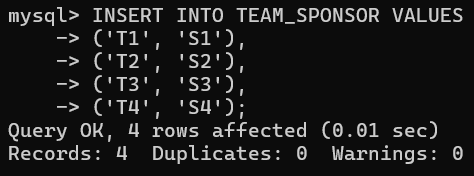


Figure 3.22: Inserting Values to Team\_Sponsor Table.

* REFREE

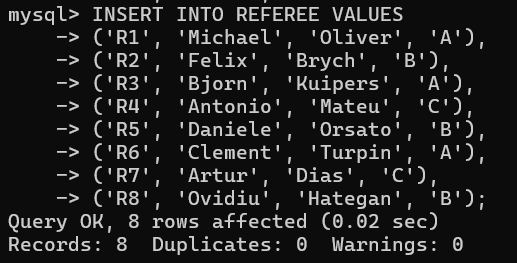


Figure 3.23: Inserting Values to Referee Table.

* VENUE

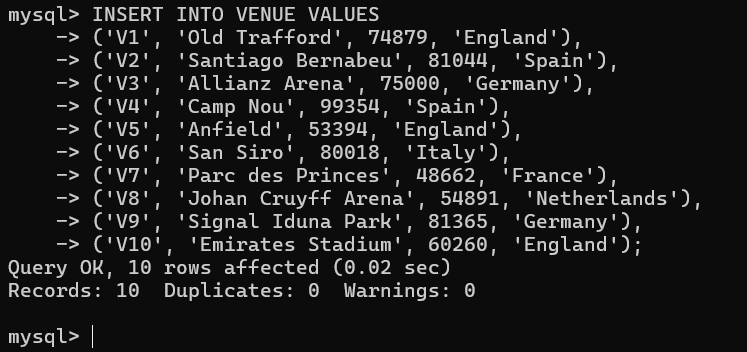


Figure 3.24: Inserting Values to Venue Table.

* GAME

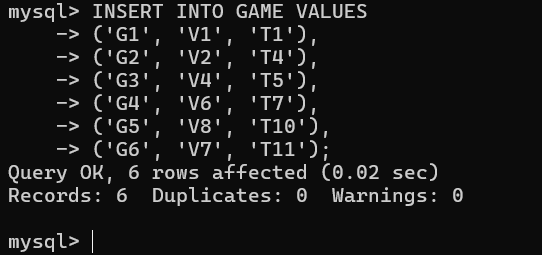


Figure 3.25: Inserting Values to Game Table.

GAME\_REFREE

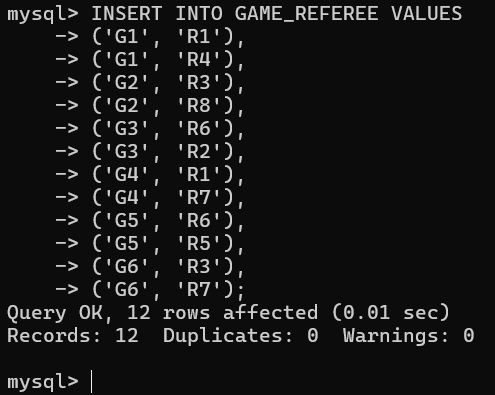
****

Figure 3.26: Inserting Values to Game\_Referee Table.

* PLAYER\_STAT

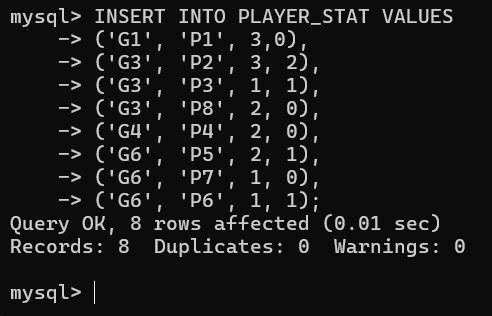


Figure 3.27: Inserting Values to Player\_Stat.

* TEAM\_STAT

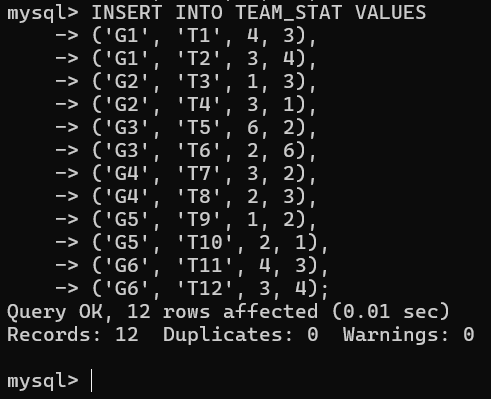


Figure 3.28: Inserting Values to Team\_Stat Table.

## Update and delete

* LEAGUE

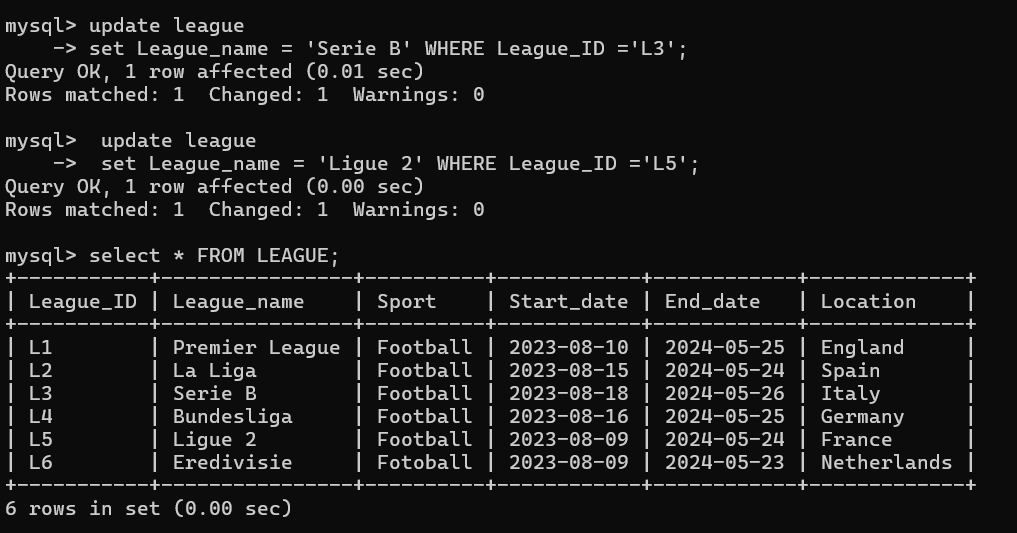


Figure 3.29: Updating League Table.

* TEAM

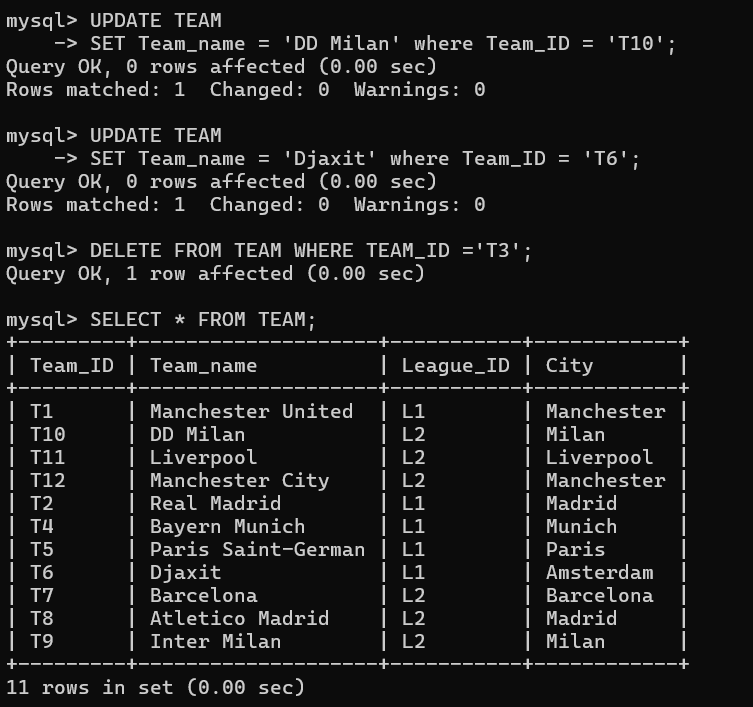


Figure 3.30: Updating Team Table.

* PLAYER



Figure 3.31: Updating Player Table.

* SPONSER

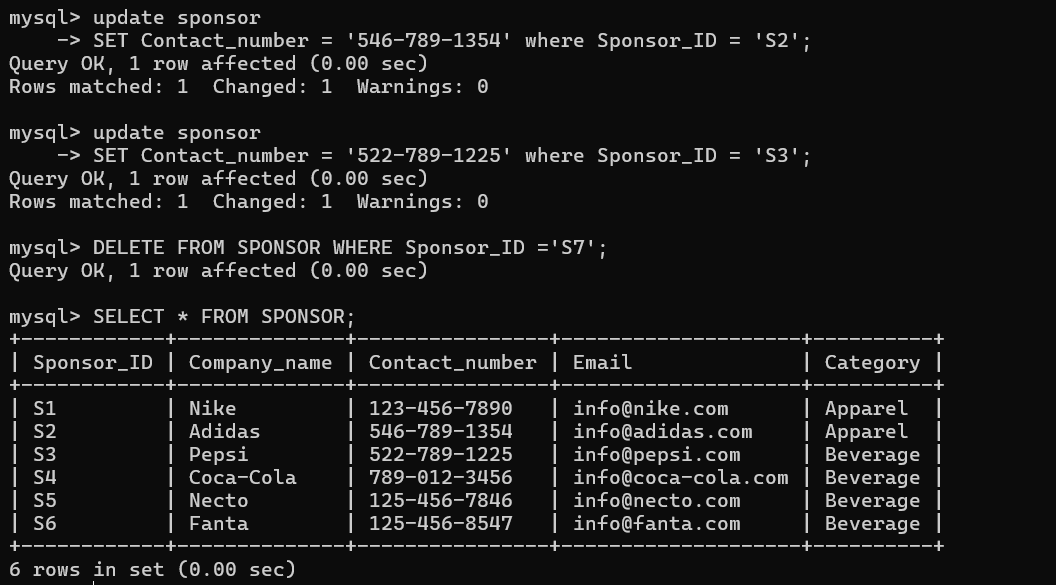


Figure 3.32: Updating Sponsor Table.

* REFEREE

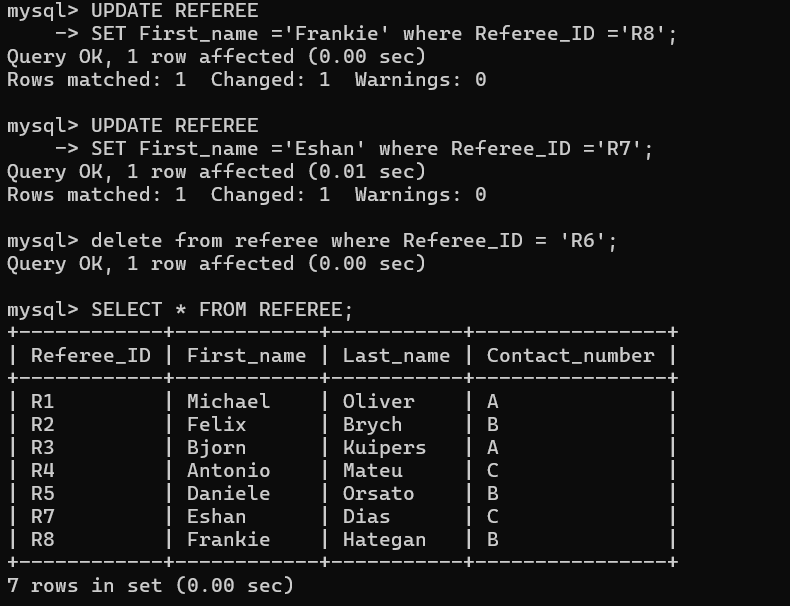


Figure 3.33: Updating Referee Table.

* VENUE



Figure 3.34: Updating Venue Table.

* GAME

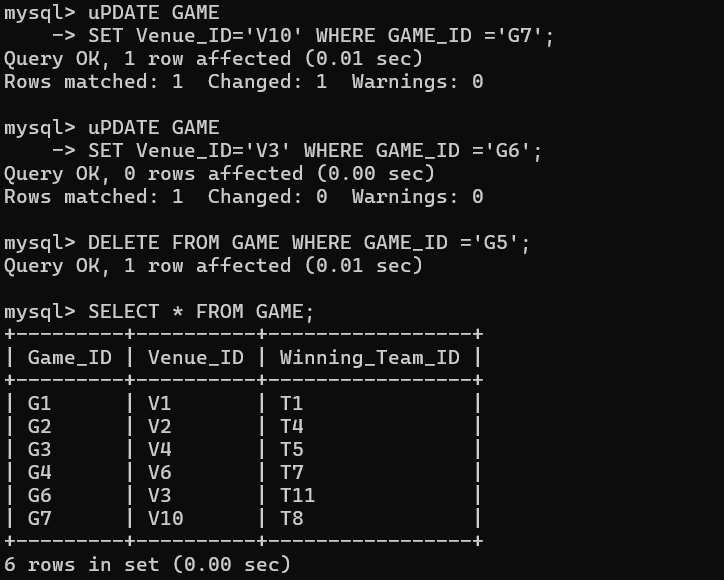


Figure 3.35: Updating Game Table.

* GAME\_TEAM

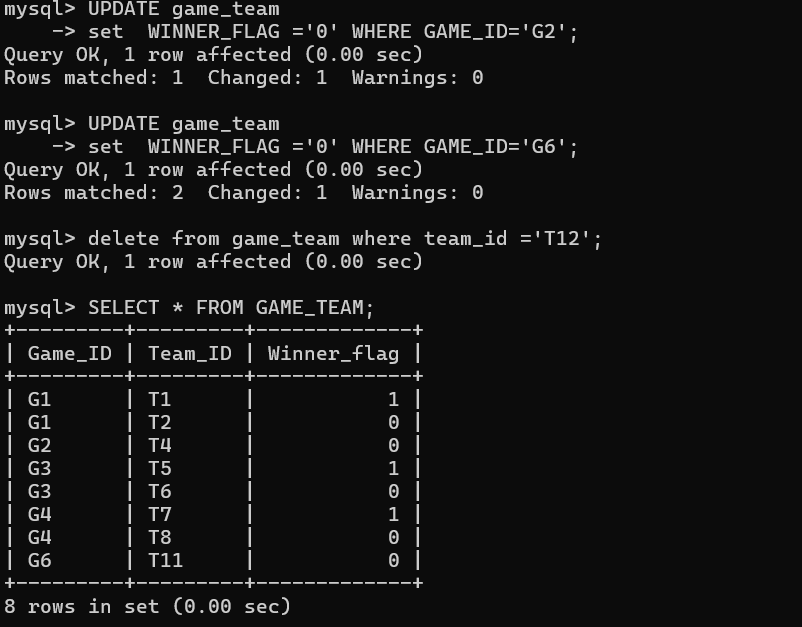


Figure 3.36: Updating Game\_Team Table.

* PLAYER\_STAT

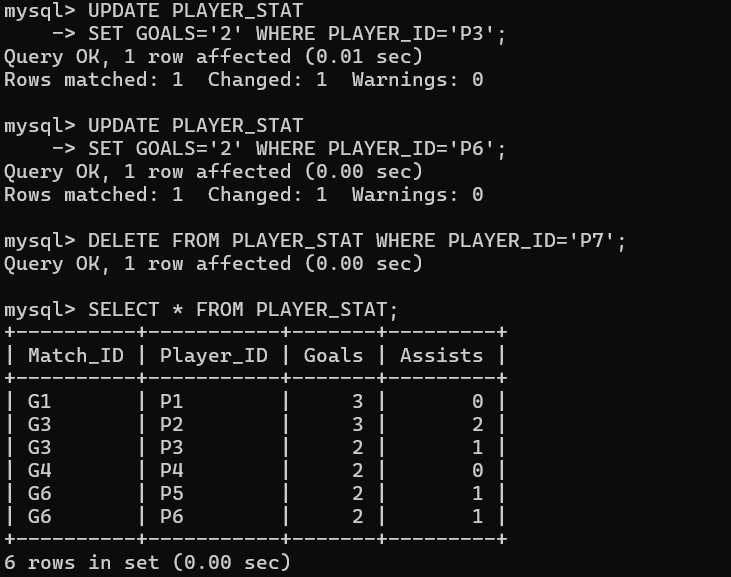


Figure 3.37: Updating Player\_Stat Table.

* TEAM\_STAT

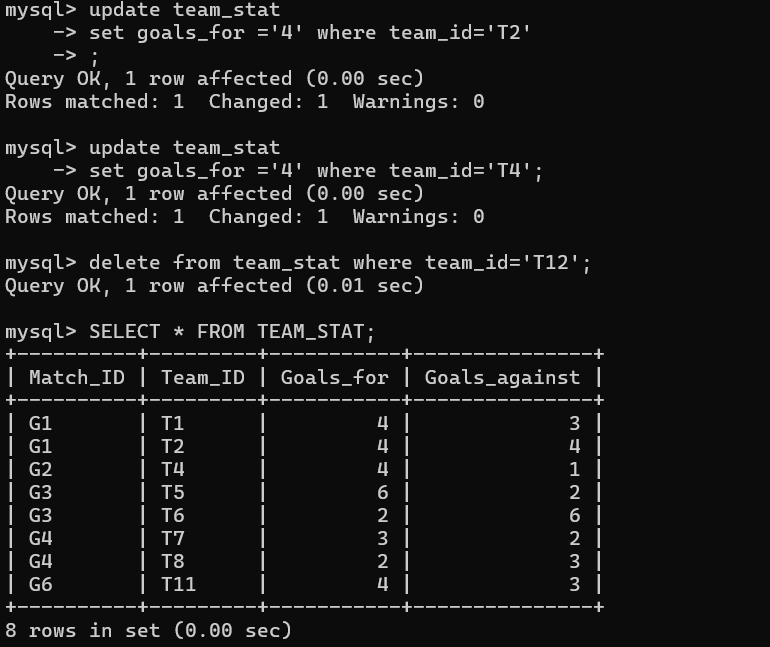


Figure 3.38: Updating Team\_Stat Table.

# Transactions

* Select Operation

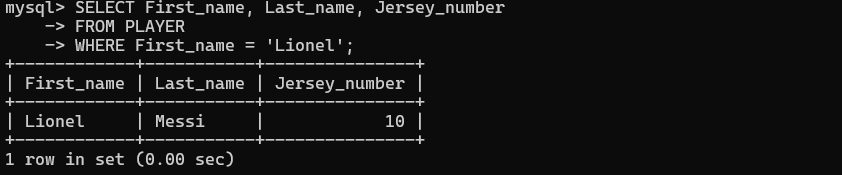


Figure 4.1: The SELECT Operation.

* Project Operation

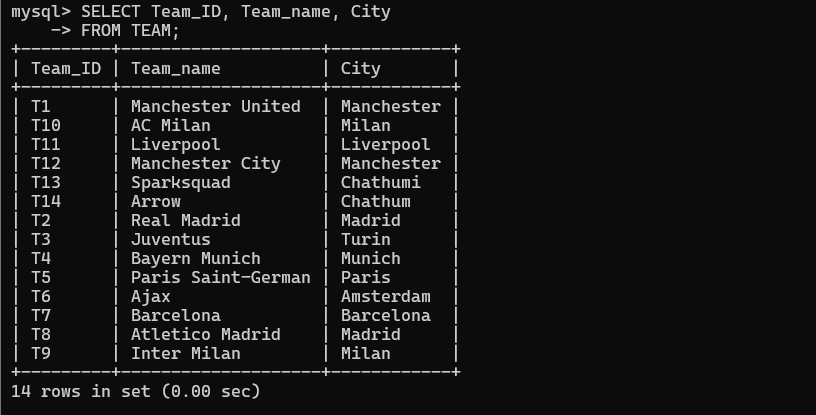


Figure 4.2: The PROJECT Operation.

* Cartesian product

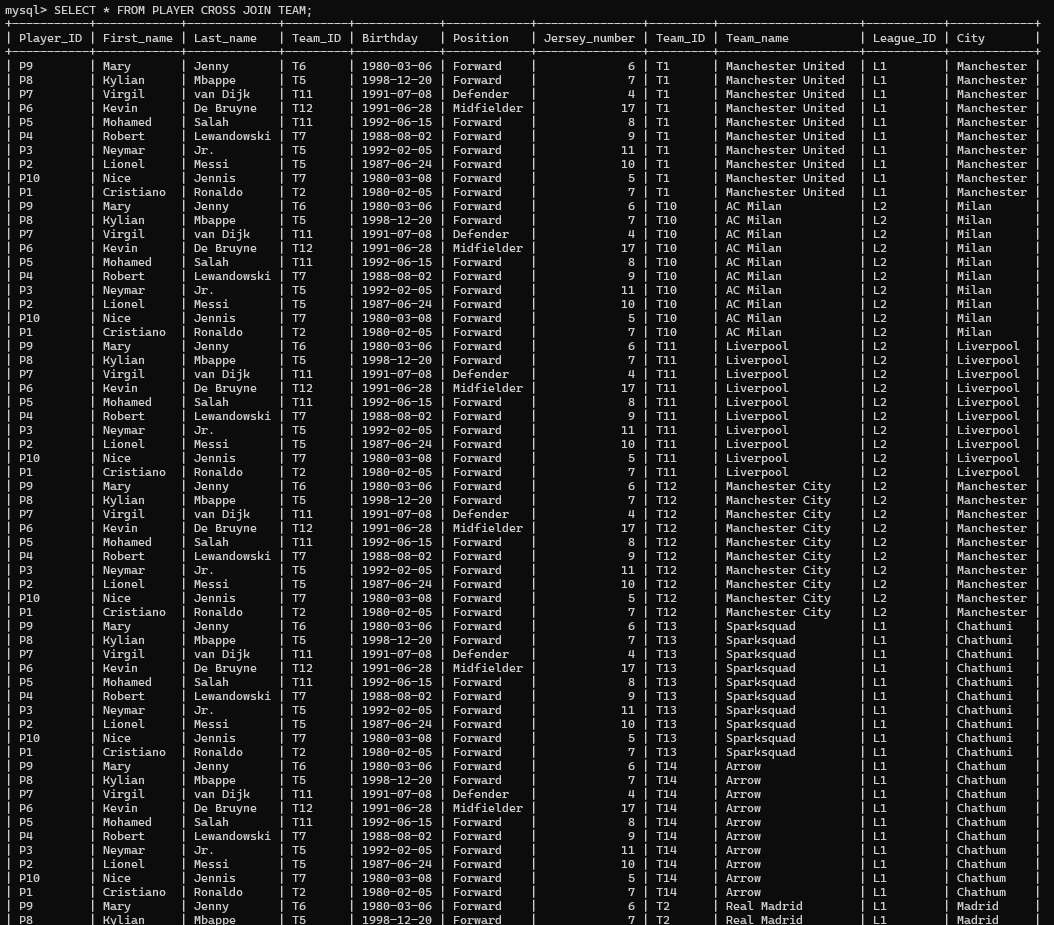


Figure 4.3: The CARTESIAN PRODUCT - I.

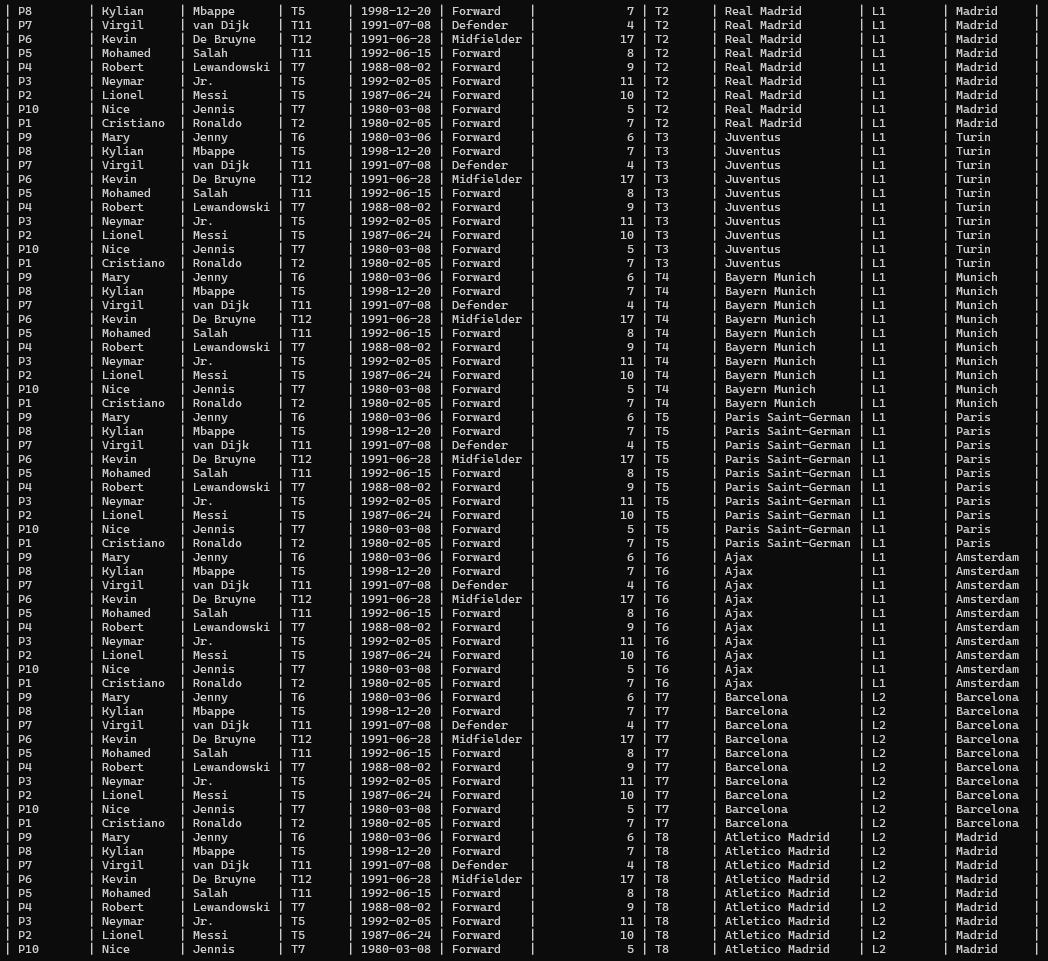


Figure 4.4: The CARTESIAN PRODUCT - II.

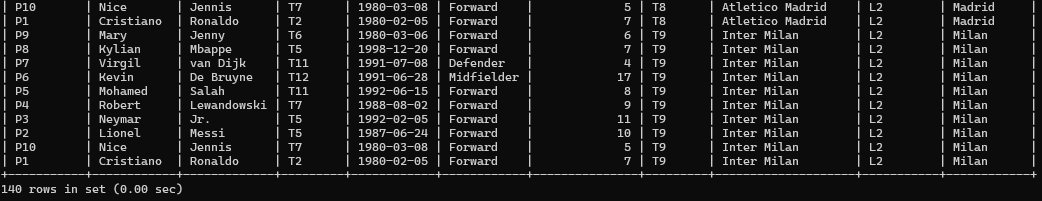


Figure 4.5:The CARTESIAN PRODUCT - III.

Creating a user view



Figure 4.6: Creating USER VIEWS.

* Renaming Operation

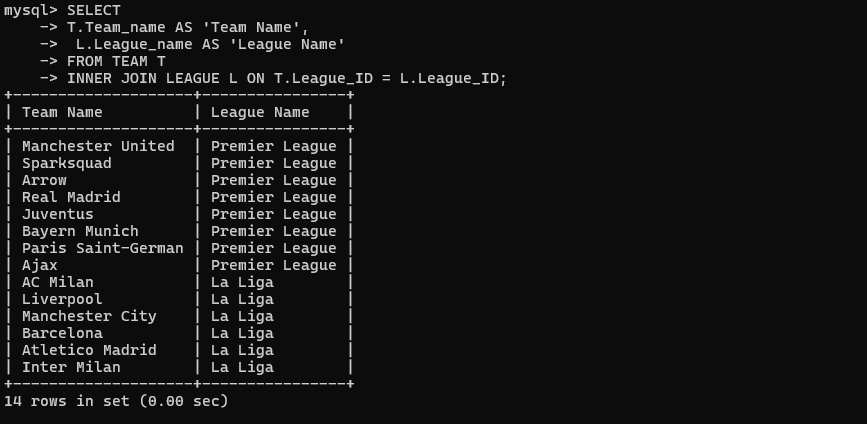


Figure 4.7: Operation Renaming.

* Aggregation (Average function)

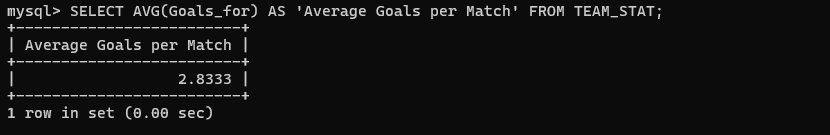


Figure 4.8: Aggregation (Average).

* Like key word

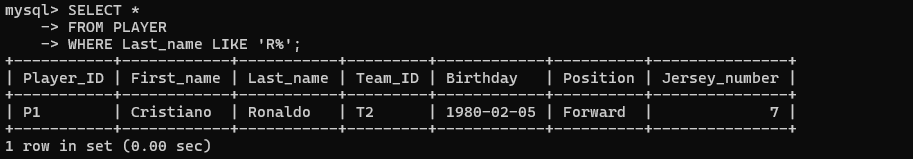


Figure 4.9: Using LIKE Keyword.

* **Complex Queries**
* Union



Figure 4.10: Union.

* Intersection

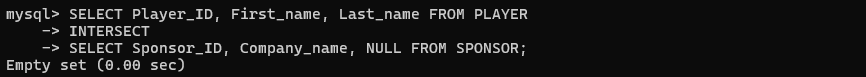


Figure 4.11: Intersection.

* Set Difference

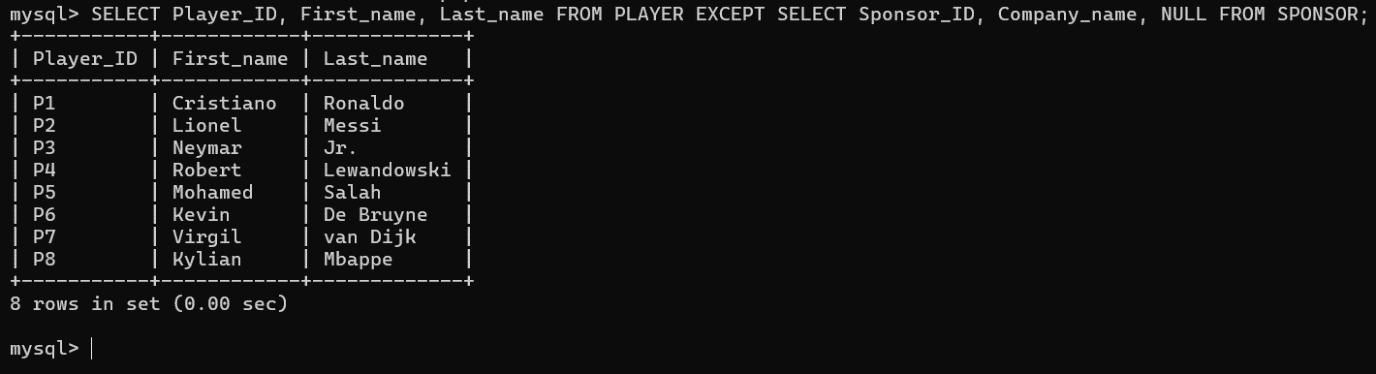


Figure 4.12: Set Difference.

* Division

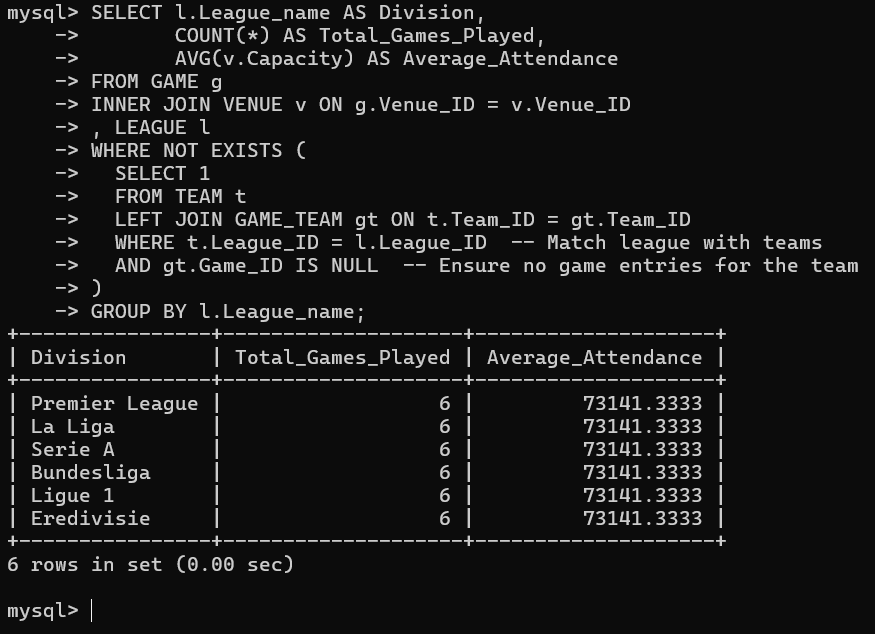


Figure 4.13: Division.

* Inner join

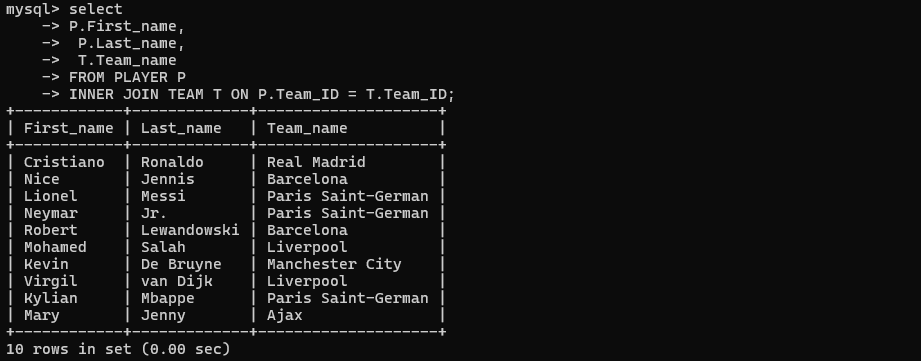


Figure 4.14: INNER JOIN.

* Natural join

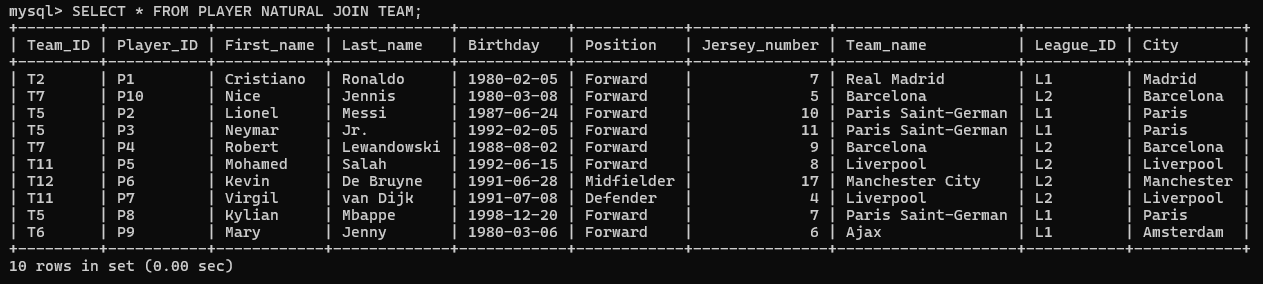


Figure 4.15: NATURAL JOIN.

* Left Outer Join

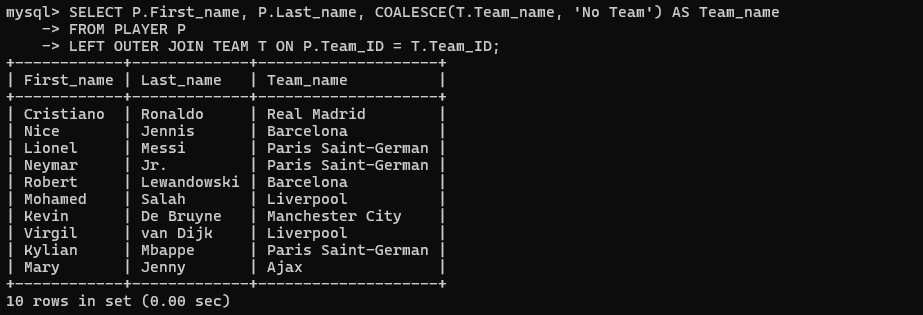


Figure 4.16: LEFT OUTER JOIN.

* Right outer join

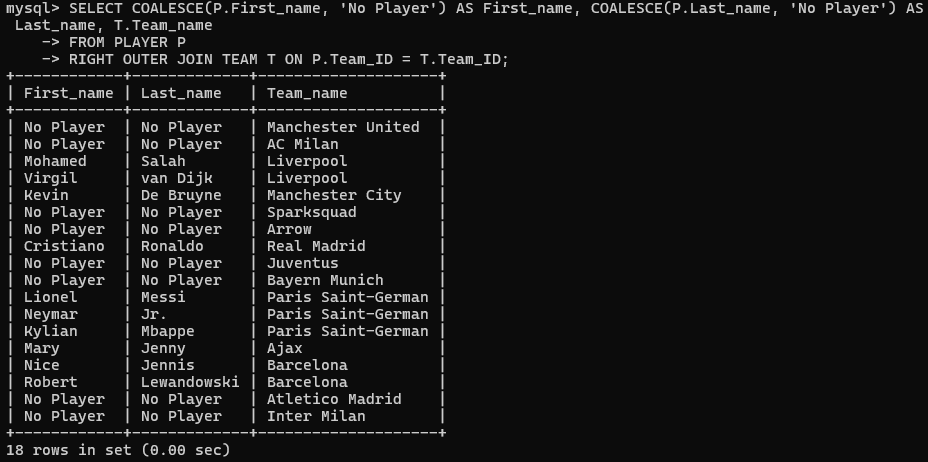


Figure 4.17: RIGHT OUTER JOIN.

* Full Outer join

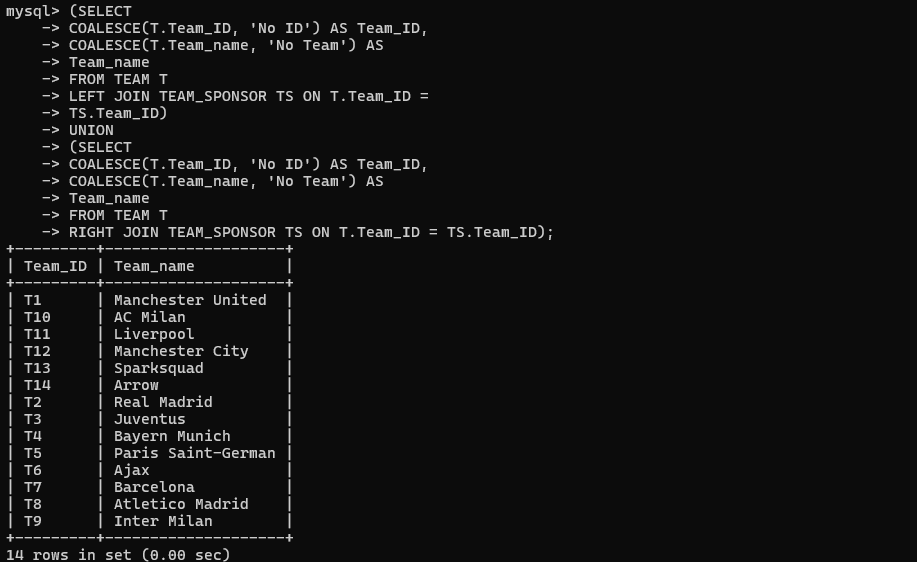


Figure 4.18: FULL OUTER JOIN.

* Outer union



Figure 4.19: OUTER UNION.

* Nested query -1

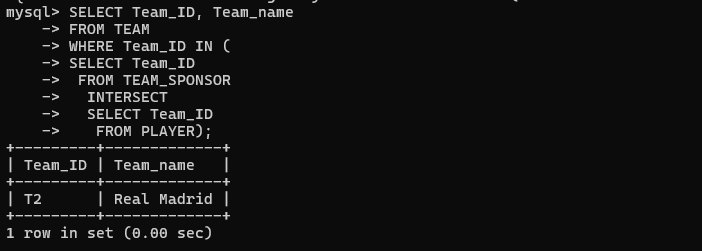


Figure 4.20: Nested Query - I.

* Nested query 2

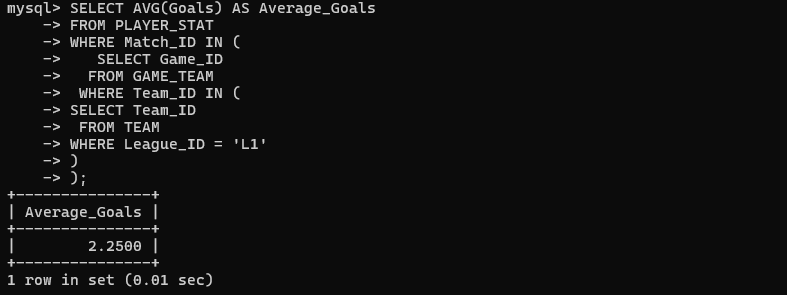


Figure 4.21: Nested Query - II.

* Nested query -3

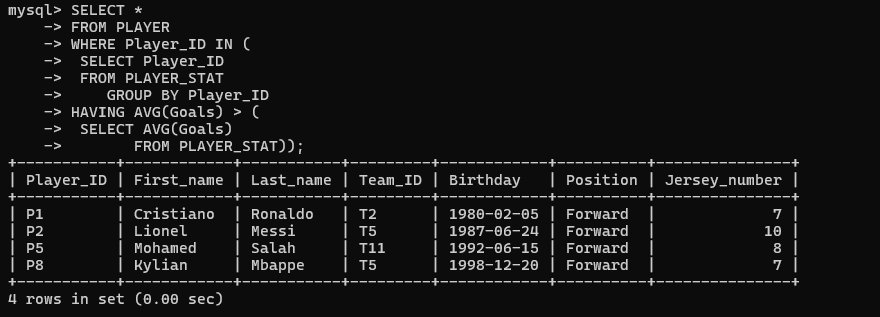
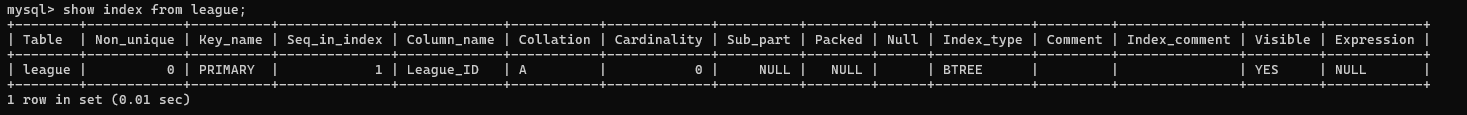


Figure 4.22: Nested Query - III.

# Tuning

The tuning of a query may be determined by comparing the amount of data accessed in the explanation table before and after an appropriate index was established. If the number of rows accessed decreases after an index is established, we may say that the query is tuned correctly. All of the screenshots in the list below satisfy that criterion.The following steps are utilized in query optimization for each difficult query to ensure a clear understanding.

* Remove any externally built indexes that are currently present in utilized tables. o
* After creating a proper index, display the number of accessed rows. (using EXPLAIN command)
* Display the number of rows accessed before creating a suitable index. (using EXPLAIN command)
* Display table indexes before creating a suitable index. (using SHOW INDEX command)
* Show table indexes after constructing an appropriate index. (using SHOW INDEX command)
* Union tuning



* Inner Join tuning
* Left outer join tuning
* Right outer join tuning
* Outer union tuning
* Division tuning
* Nested query 3 tuning
* Intersection tuning
* Set difference tuning
* Tuning nested query 1