American International University-Bangladesh
Department of Computer Science and Engineering
Dhaka, Bangladesh

00892 ADVANCE DATABASE MANAGEMENT SYSTEM
PROJECT Report

[B][SUMMER 22-23]

E-Sports Management System

Group 5

Submitted by

Names of Students	ID
AJRAN HOSSAIN	19-39334-1
SHARIF HADI MAHATAB	20-43625-2
MD. SARAFAT ALI ADIR	20-41926-1
MEDHA CHOWDHURY	20-41930-1

Date of Submission
August 27, 2023

Submitted to
JUENA AHMED NOSHIN
Assistant Professor, Faculty
Department of Computer Science and Engineering
American International University-Bangladesh

Contribution

	AJRAN HOSSAIN	SHARIF HADI MAHATAB	MD. SARAFAT ALI ADIR	MEDHA CHOWDHURY	Contribution (%)
	19-39334-1	*20-43625-2*	*20-41926-1*	*20-41930-1*	
Diagram	60%	20%	10%	10%	100(%)
UI Design	100%	0%	0%	0%	100(%)
Normalization	50%	50%	0%	0%	100(%)
SQL Query	85%	5%	5%	5%	100(%)
Relational Algebra	15%	0%	85%	0%	100(%)
Report Writing	55%	35%	5%	5%	100(%)

Contents

Introduction

Esports Management System is an innovative platform that will revolutionize the management and organization of esports teams, participants, tournaments, and sponsors. This system seeks to provide users with an efficient and user-friendly way to search for their preferred professional esports players.

A user-friendly interface is at the core of the Esports Management System, allowing users to seamlessly navigate and explore the realm of professional esports. With only a few clicks, users can search for potential professional athletes and teams, as well as access valuable information such as their winning records and accomplishments. This enables fans and enthusiasts to remain up-to-date on their preferred players and teams, nurturing a stronger connection within the esports community.

The Esports Management System's ability to facilitate sponsorships is a crucial feature. Numerous organizations and businesses can engage in sponsorship activities, whether for the purpose of supporting tournaments or individual athletes. The system serves as a centralized repository where the information and details of these sponsors can be efficiently stored and managed. This facilitates the sponsorship process and ensures that sponsors and the esports industry collaborate effectively.

There are specialized administrators within the Esports Management System who play crucial roles in managing and enhancing the overall experience. The social media manager is among these supervisors; he or she supervises the organization's online presence and engagement on various social media platforms. In addition, the content creator/VFX/GFX team assures the creation of visually stunning and captivating content that enhances the overall esports experience.

Dynamic features and functionalities make the Esports Management System an indispensable instrument for the esports industry. It makes it easier for fans to discover and connect with professional esports players, allowing them to remain informed and engaged. It enhances collaboration between organizations and the esports community by providing a centralized platform for sponsorship management. In addition, the system enables administrators to enhance the organization's online presence and develop visually appealing content, ensuring that all stakeholders have an engaging experience.

In the following sections, we will delve deeper into the features, functionalities, and innovative aspects of the Esports Management System, demonstrating its potential to revolutionize the management and celebration of esports teams, players, tournaments, and sponsors.

1.1 Project Proposal

This proposal for the development and implementation of an Esports Management System is presented with pleasure. This revolutionary platform seeks to transform the management and organization of esports teams, players, tournaments, and sponsors. The Esports Management System will improve the user experience, encourage community engagement, and expedite operations within the esports industry by leveraging advanced technology and comprehensive functionalities.

1.1.1 Purposes

- Create an intuitive web-based platform that serves as the central hub for esports administration, catering to the requirements of teams, players, tournament organizers, and sponsors.
- Implement a sophisticated matching algorithm to facilitate the search and discovery of favored professional esports players, thereby enhancing the fan experience and fostering esports community connections.
- Provide efficient sponsorship administration capabilities, enabling organizations and businesses to support tournaments or individual athletes through sponsorship activities.
- Enhance the organization's online presence by supervising social media platforms and having the content creator/VFX/GFX team produce visually spectacular and engaging content.

1.1.2 Methodology

System Development:

- Conduct exhaustive investigation on the necessary requirements and features of an effective Esports Management System.
- Utilize industry-standard programming languages and technologies to create a scalable and secure web-based platform.
- Implement a user-friendly interface with intuitive navigation in order to provide a seamless and enjoyable user experience.

Matching Algorithm

- Collaboration with data scientists and psychologists to create a matching algorithm based on personality traits, values, and beliefs.
- Integrate the matching algorithm into the system to recommend professional esports players compatible with the user's preferences.

Sponsorship Management

- Create an all-encompassing sponsorship management module to facilitate collaborations between organizations and the esports industry.
- Provide a centralized repository for sponsor information to facilitate communication and sponsorship efficiency.

Online Presence Enhancement

- Appoint a social media manager to supervise the organization's online presence and interact with the esports community.
- Appoint a social media manager to supervise the organization's online presence and interact with the esports community.

1.2 Project Scenario

A company called EsportsFTW oversees a number of gaming industry teams and competitions.This company is run by an administrator who is in charge of everything.A unique ID is used to save the admin profile in the system. It was also necessary to register other data in the system, such as an email, photo, password, and name.

There are many Managers that report to the Admin and are in charge of several specialized departments. The system keeps track of each Manager's information, including their name, email address, pay, and date of hiring.The system produced a unique Department and Manager id for them. The Finance division is exclusively managed by one Manager, who also oversees the organization's finances and ensures financial stability. Data from the finances, including account numbers and balances, are also maintained in the system. Also created by the system is an ID.

Each Manager oversees a certain team in addition to the money. The Teams profile contains information about the team, including its name, country of origin, winning streaks, and total prize money. A different Manager is given to each team, guaranteeing effective structure and collaboration. The system-generated id serves as a connection between the Manager and the Teams object.

There are a number of Players who play for each squad and represent the organization in different games. Name, photo, income, winning prize money, total hours played, phone number, and address (including nation, city, zip code, and road number) are just a few of the details that may be found in a player's profile. The URLs to the athletes' social media accounts on websites like Facebook, Instagram, Twitter, and YouTube are also available.

The organization organizes competitions that bring together teams from various video games. Data from the Tournament profile includes the name, price pool, start date, and finish date. Games like Valorant, Mobile Legends: Bang Bang (MLBB), and Rainbow Six Siege are all included in each event. Name, release date, genre, game image, publisher, platform, and pricing pool are only a few examples of the game data. Different games might be linked to each event via the game id.

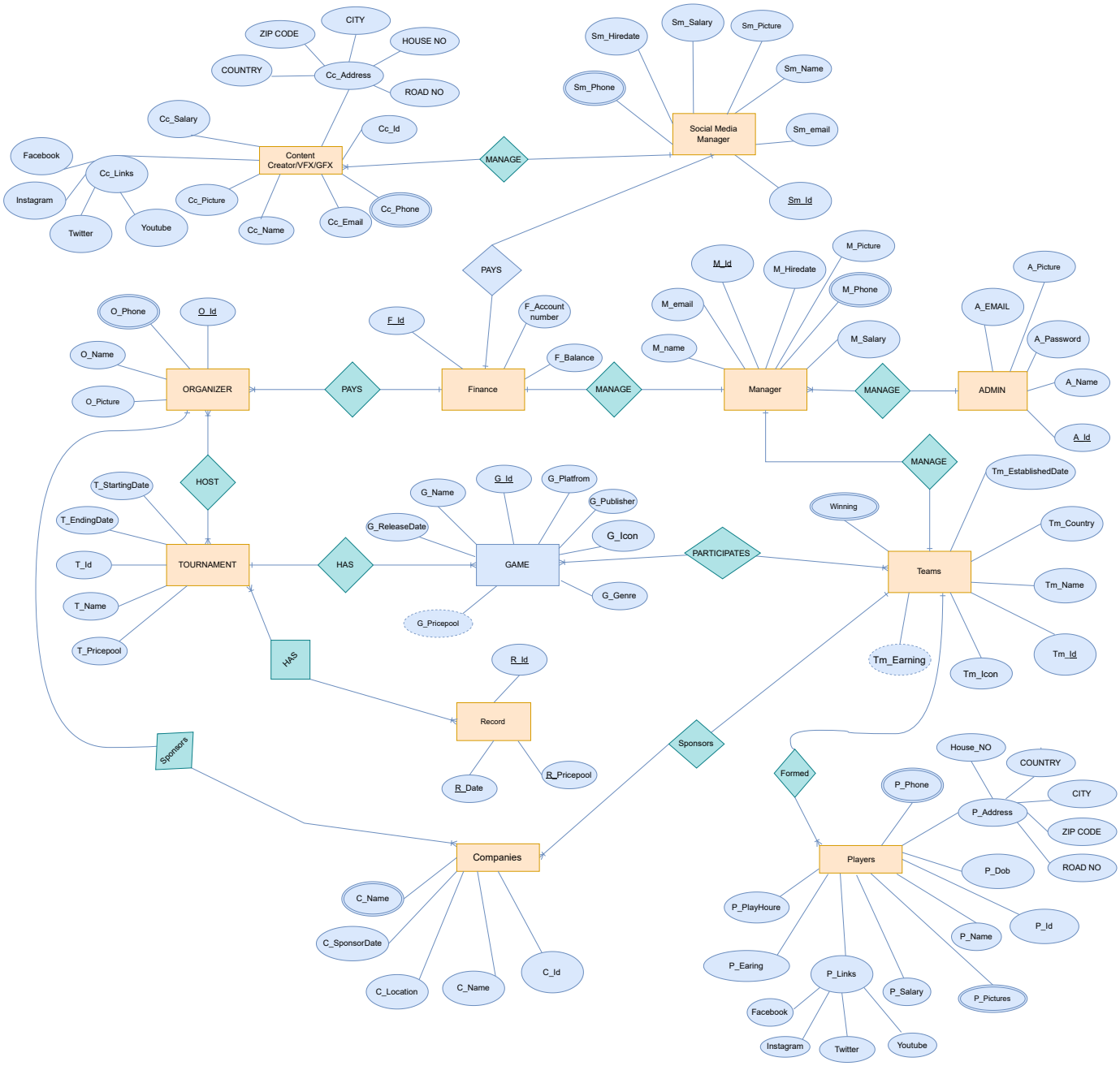
EsportsFTW looks for corporate sponsorships to help pay for the teams and competitions. The company's information, including name, address, sponsoring date, and phone number. Both teams and tournaments may have several corporate sponsors, creating a many-to-many link between the data for the corporations, teams, and tournaments via the system-generated id.

A Social Media Manager is also employed by EsportsFTW to oversee the company's online presence. Information like name, image, email, date of hiring, salary, phone number, and social media links (Facebook, Instagram, Twitter, and YouTube) are all included in the profile for the Social Media Manager. The VFX/GFX and material Creator teams are under the Social Media Manager's control, and both teams are responsible for producing interesting material. Data like name, image, email, phone number, income, and address (country, city, zip code, and road number) are all included in the VFX/GFX and Content Creator profiles.

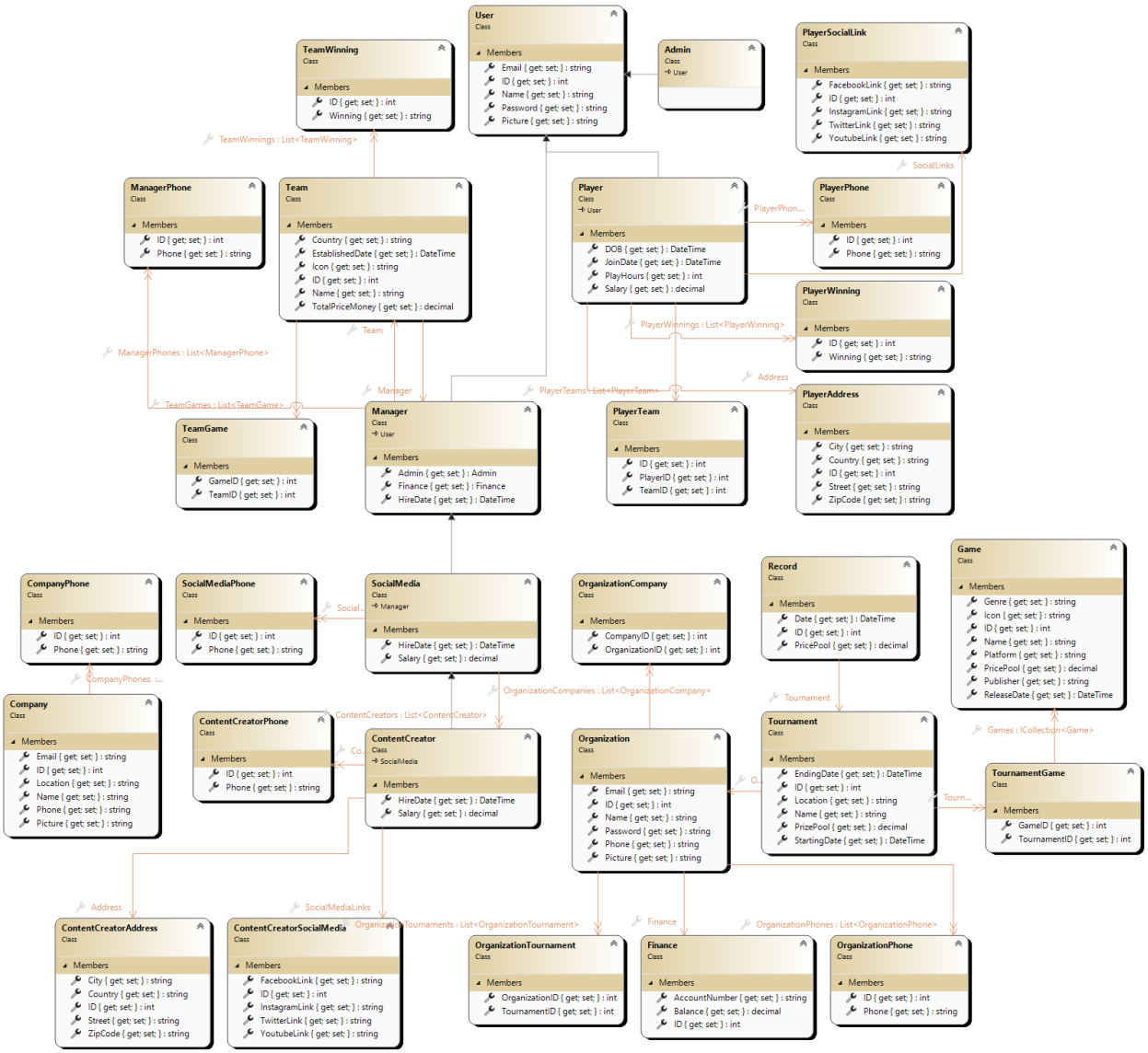
The whole organization of EsportsFTW is described in this scenario, including its administration, management structure, teams, players, coaches, competitions, sponsorships, and social media management. The database offers a thorough way to arrange and monitor all parts of the business' operations, guaranteeing efficient administration and achievement in the cutthroat world of eSports.

Diagrams

2.1 ER Diagram

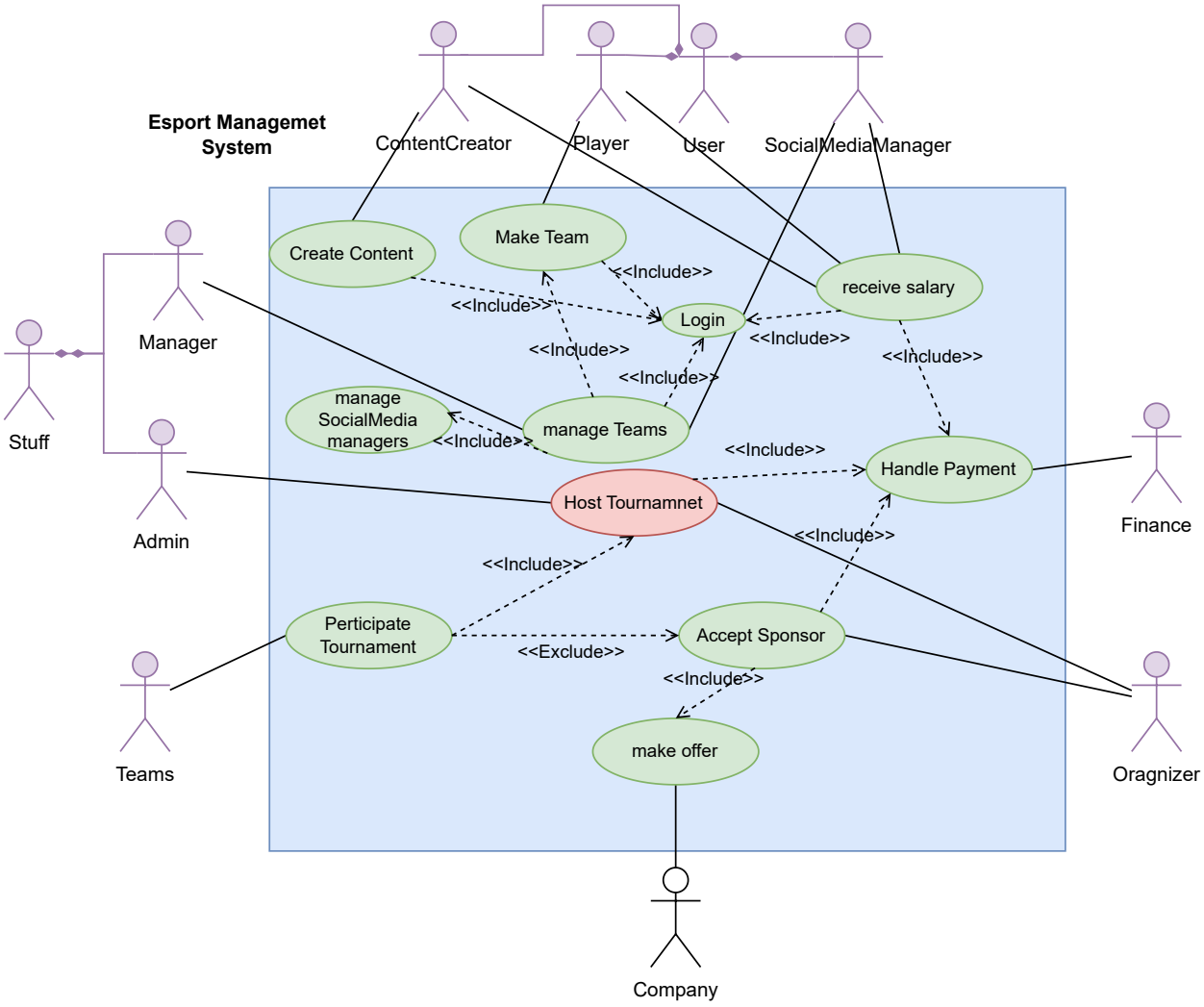


2.2 Class Diagram

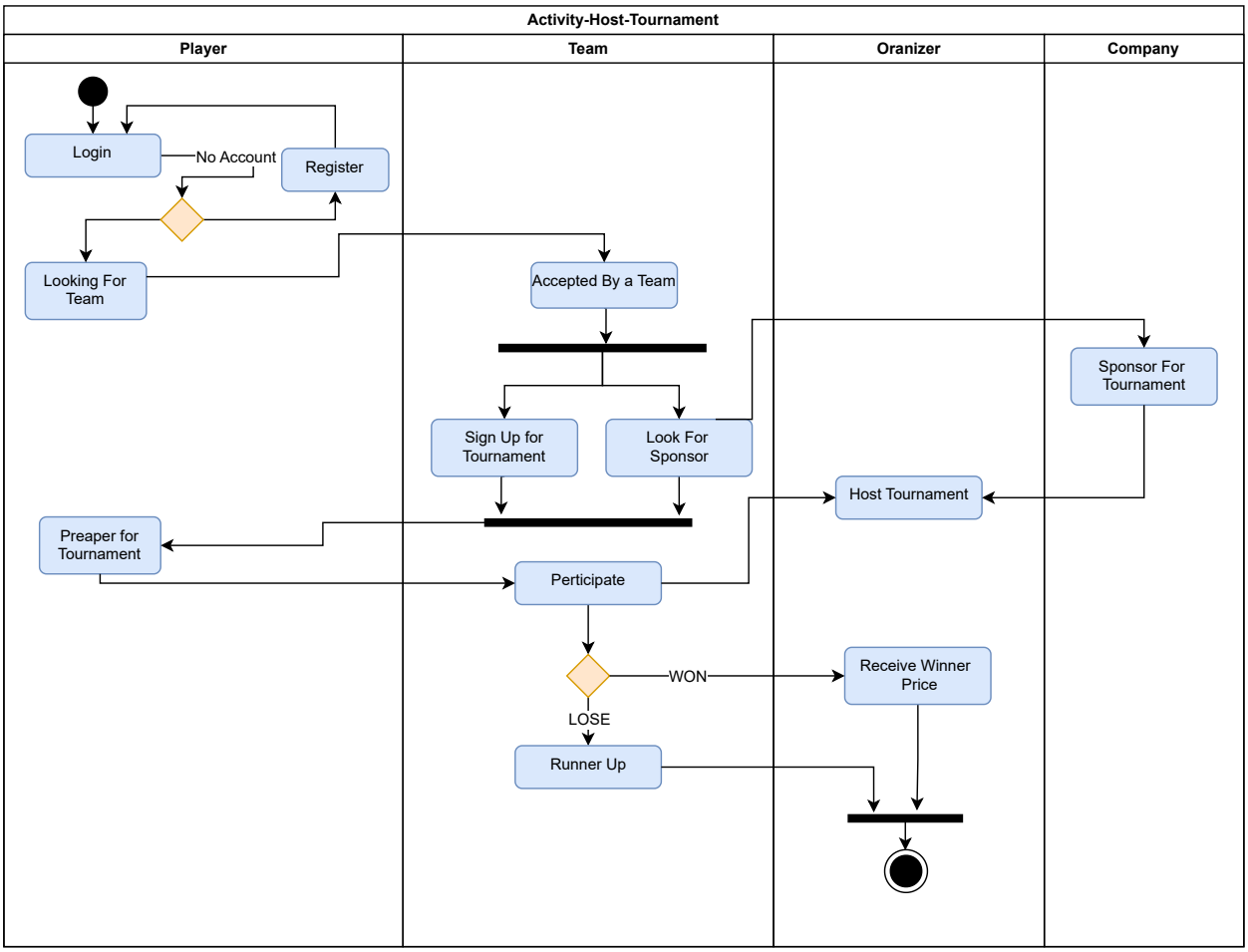


Class Diagram

2.3 Use Case Diagram



2.4 Activity Diagram

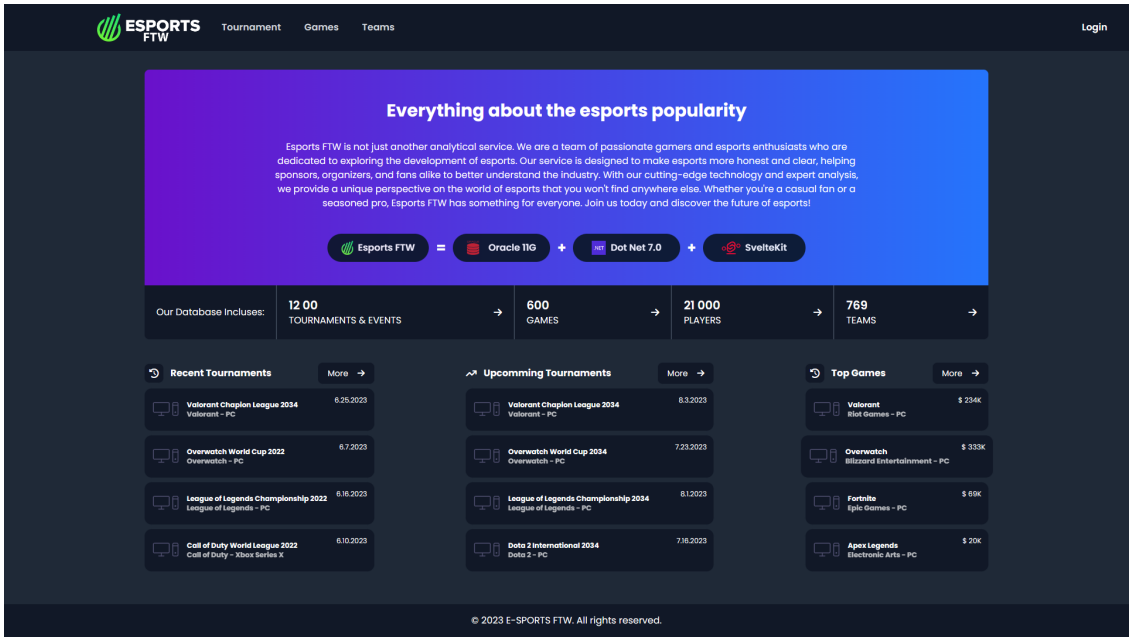


User Interface

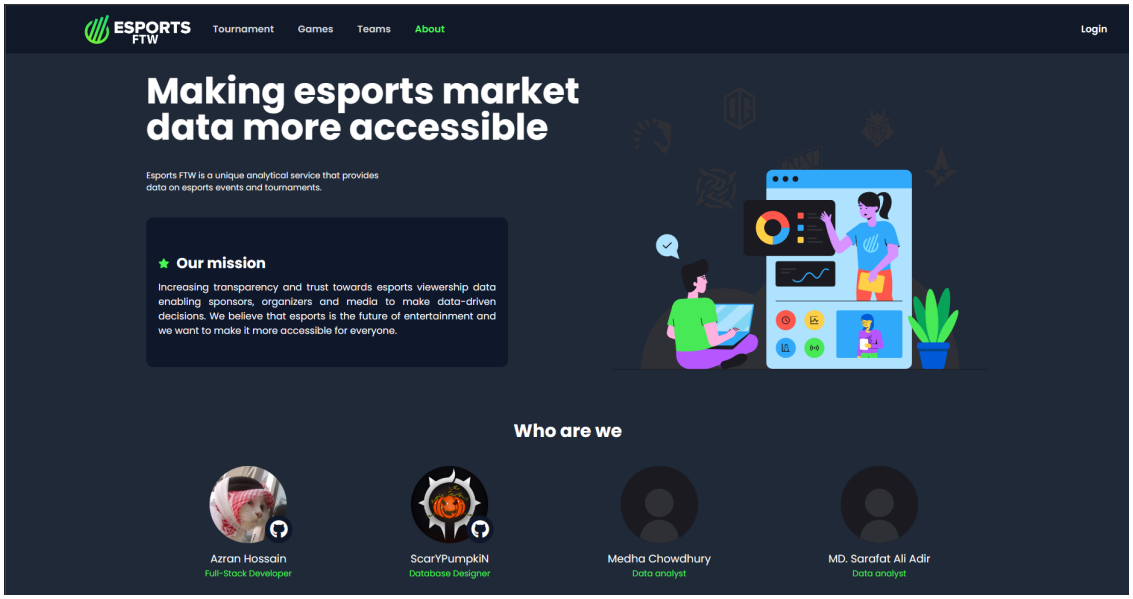
3.1 Technologies Used

- SvelteKit
- Tailwind CSS

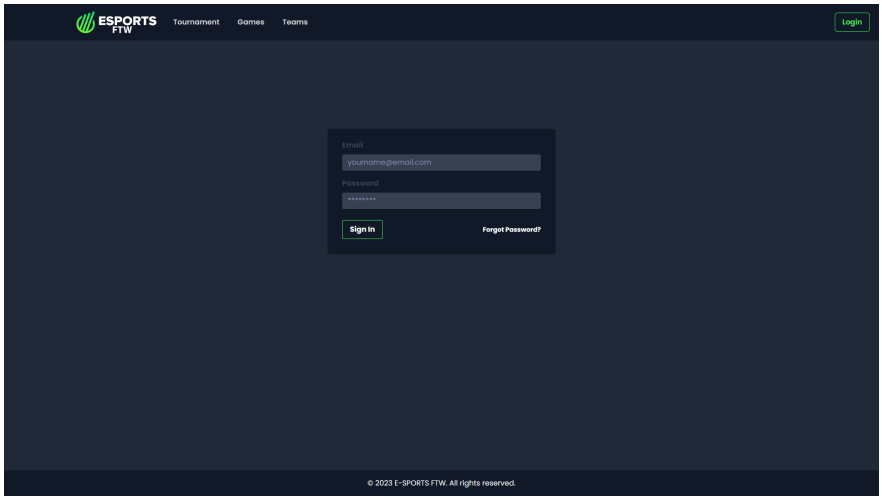
3.1.1 Home Page



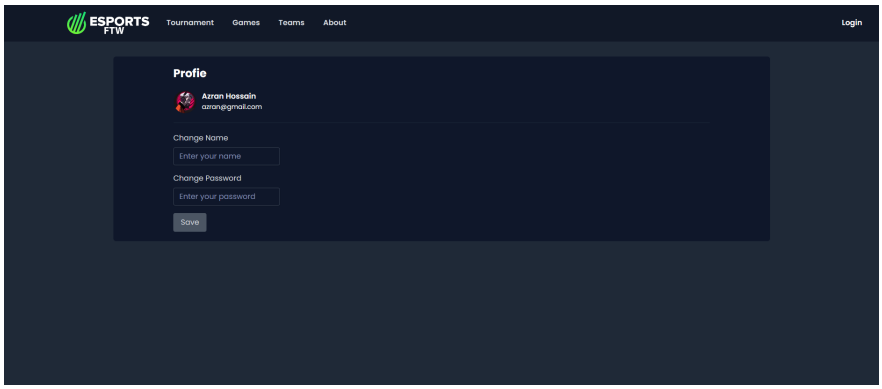
3.1.2 About Page



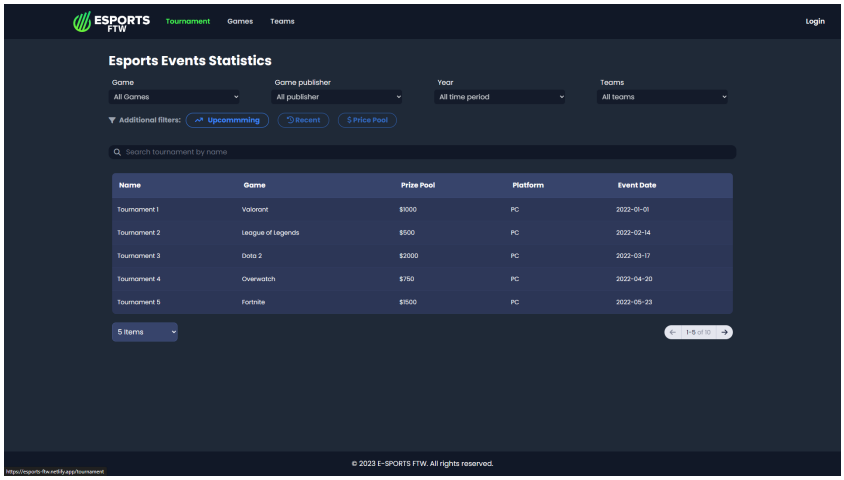
3.1.3 Login Page



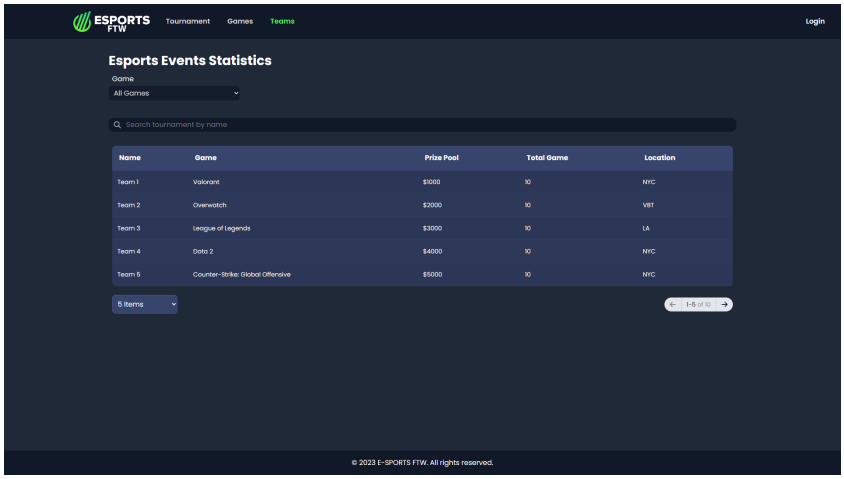
3.1.4 Profile Page



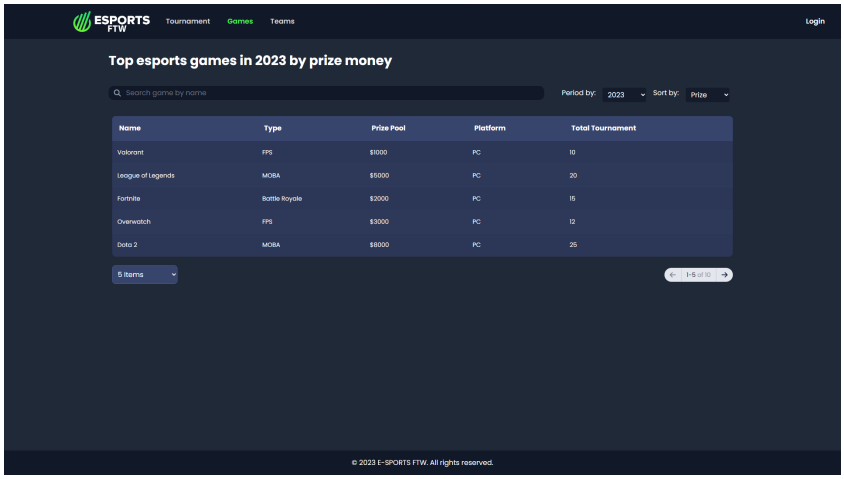
3.1.5 Tournament Page



3.1.6 Team Page



3.1.7 Game Page



Normalization and Schema Design

4.1 Manage branch

4.1.1 (Admin → Manager)

UNF

(Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture, Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone)

1NF

Phone is multi-valued attribute.

1. (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture, Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone)

2NF

1. Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture
2. Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate, Manager_Phone

3NF

No transitive dependencies found.Same as 2NF

1. Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture
2. Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate, Manager_Phone

Table after Normalization

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate,**Admin_ID**)
2. (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
3. (Mp_ID, **Manager_ID**,Manager_Phone)

4.1.2 (Manager → Finance)

UNF

(Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone, Finance_ID, Finance_Account_Number, Finance_Balance)

1NF

Phone is multi-valued attribute.

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone, Finance_ID, Finance_Account_Number, Finance_Balance)

2NF

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone)
2. (Finance_ID, Finance_Account_Number, Finance_Balance)

3NF

No transitive dependency found.Same as 2NF.

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone)
2. (Finance_ID, Finance_Account_Number, Finance_Balance)

Table after Normalization

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate)
2. (Finance_ID, Finance_Account_Number, Finance_Balance, **Manager_ID**)
3. (Mp_ID, **Manager_ID**,Manager_Phone)

4.1.3 (Manager → Teams)

UNF

(Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone, Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Price_Money,Team_Winnig)

1NF

Winning & Phone are multi-valued attribute.

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone, Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Team_Total_Price_Money,Team_Winnig)

2NF

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone)
2. (Team_ID, Team_Name,Team_Icon, Team_established_date,Total_Price_Money,Team_country)

3NF

No transitive dependency found.Same as 2NF.

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password,Manager_Picture, Manager_Hiredate, Manager_Phone)
2. (Team_ID, Team_Name,Team_Icon, Team_established_date,Total_Price_Money,Team_country)

Table after Normalization

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate)
2. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country, Total_Price_Money, **Manager_ID**
3. (Tw_ID, **Team_ID**, Team_Winnig)
4. (Mp_ID, **Manager_ID**,Manager_Phone)

4.1.4 (SocialMedia → ContentCreator)

UNF

(SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary, ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture, ContentCreator_Phone , ContentCreator_Hiredate , ContentCreator_Salary, ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_Instagram_Link, ContentCreator_Youtube_Link , ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)

1NF

Phone is multi-valued attribute.

1. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary, ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture, ContentCreator_Phone , ContentCreator_Hiredate , ContentCreator_Salary, ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_
   Instagram_Link, ContentCreator_Youtube_Link , ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)

2NF

1. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary)

2. (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture, ContentCreator_Phone , ContentCreator_Hiredate , ContentCreator_Salary, ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_
   Instagram_Link, ContentCreator_Youtube_Link , ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)

3NF

1. SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture , SocialMedia_Hiredate , SocialMedia_Salary)

2. ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture , ContentCreator_Hiredate , ContentCreator_Salary)

3. (**ContentCreator_ID**, **SocialMedia_ID** , ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_
   Instagram_Link, ContentCreator_Youtube_Link)

4. (**ContentCreator_ID**, ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)

Table after Normalization

1. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture , SocialMedia_Hiredate , SocialMedia_Salary)

2. (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture , ContentCreator_Hiredate , ContentCreator_Salary)

3. (Ccs_ID,**ContentCreator_ID** , ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_
   Instagram_Link, ContentCreator_Youtube_Link)

4. (Cca_ID, **ContentCreator_ID**, ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)

5. (Ccp_ID, **ContentCreator_ID**, ContentCreator_Phone)

6. (Smp_ID, **SocialMedia_ID**, SocialMedia_Phone)

7. (Ccp_ID, **ContentCreator_ID**, ContentCreator_Phone)

4.2 Pay branch

4.2.1 (Finance → SocialMedia)

UNF

(Finance_ID, Finance_Account_Number, Finance_Balance,
SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary)

1NF

Phone is multi-valued attribute.

1. (Finance_ID, Finance_Account_Number, Finance_Balance,
   SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary)

2NF

1. (Finance_ID, Finance_Account_Number, Finance_Balance)
2. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary)

3NF

No transitive dependency found.Same as 2NF.

1. (Finance_ID, Finance_Account_Number, Finance_Balance)
2. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture, SocialMedia_Phone , SocialMedia_Hiredate , SocialMedia_Salary)

Table after Normalization

1. (Finance_ID, Finance_Account_Number, Finance_Balance)
2. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture , SocialMedia_Hiredate , SocialMedia_Salary, **Finance_ID**)
3. (Smp_ID, **SocialMedia_ID**,SocialMedia_Phone)

4.2.2 (Finance → organization)

UNF

(Finance_ID, Finance_Account_Number, Finance_Balance,
Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

1NF

Phone is multi-valued attribute.

1. (Finance_ID, Finance_Account_Number, Finance_Balance,
   Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

2NF

1. (Finance_ID, Finance_Account_Number, Finance_Balance)
2. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

3NF

No transitive dependency found.Same as 2NF.

1. (Finance_ID, Finance_Account_Number, Finance_Balance)
2. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

Table after Normalization

1. (Finance_ID, Finance_Account_Number, Finance_Balance)
2. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone, **Finance_ID**)
3. (Op_ID, **Organization_ID**,Organization_Phone)

4.3 Formed branch

4.3.1 (Teams → Player)

UNF

(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning,

Team_Winnig, Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Phone, Player_Salary, Player_Winnig_Money, Player_Play_Hours, Player_country, Player_City, Player_Street, Player_Zip_Code, Player_DOB, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Youtube_Link)

1NF

Phone & Wining number are multi-valued attribute.

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig, Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Phone, Player_Salary, Player_Winnig_Money, Player_Play_Hours, Player_country, Player_City, Player_Street, Player_Zip_Code, Player_DOB, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Youtube_Link)

2NF

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)
2. (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Salary, Player_Winnig_Money, Player_Play_Hours, Player_country, Player_City, Player_Street, Player_Zip_Code, Player_DOB, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Youtube_Link)

3NF

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)
2. (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Salary, Player_Winnig_Money, Player_Play_Hours, Player_DOB)
3. (Pa_ID, **Player_ID**, Player_country, Player_City, Player_Street, Player_Zip_Code)
4. (Psl_ID, **Player_ID**, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Yo

Table after Normalization

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)
2. (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
3. (Pa_ID, **Player_ID**, Player_country, Player_City, Player_Street, Player_Zip_Code)
4. (Psl_ID, **Player_ID**, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Yo
5. (Pp_ID, **Player_ID**, Player_Phone)
6. (Pw_ID, **Player_ID**, Player_Winnig)
7. (Pt_ID, **Player_ID**, **Team_ID**)

4.4 Has branch

4.4.1 (Record → Tournament)

UNF

(Record_ID, Record_Date, Record_Price_Pool, Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

1NF

1. (Record_ID, Record_Date, Record_Price_Pool, Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

2NF

1. (Record_ID, Record_Date, Record_Price_Pool, **Tournament_ID**)
2. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

3NF

No transitive dependency found.Same as 2NF.

1. (Record_ID, Record_Date, Record_Price_Pool, **Tournament_ID**)
2. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

Table after Normalization

1. (Record_ID, Record_Date, Record_Price_Pool, **Tournament_ID**)
2. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

4.4.2 (Tournament → Game)

UNF

(Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

1NF

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

2NF

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

2. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

3NF

No transitive dependency found.Same as 2NF.

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

2. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

Table after Normalization

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

2. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

3. (Tournament_ID, Game_ID)

4.5 Participate branch

4.5.1 (Teams → Game)

UNF

(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig, Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool Game_Genre, Game_Publisher))

1NF

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig, Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

2NF

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)
2. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

3NF

No transitive dependency found.Same as 2NF.

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)
2. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

Table after Normalization

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)
2. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)
3. (Team_ID, Game_ID)

4.6 Host branch

4.6.1 (Organizer → Tournament)

UNF

(Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

1NF

Phone is a multi-vale attribute.

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

2NF

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Organization_ID)
2. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

3NF

No transitive dependency found.Same as 2NF.

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Organization_ID)
2. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone)

Table after Normalization

1. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Organization_ID)
2. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture)
3. (Organization_ID, Organization_Phone)
4. (Organization_ID, Tournament_ID)

4.7 Sponsor branch

4.7.1 (Companies → Organization)

UNF

(Organization_ID, Organization_Name, Organization_Email, Organization_Picture, Organization_Phone, Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)

1NF

phone is a multi-value attribute.

1. (Organization_ID, Organization_Name, Organization_Email, Organization_Picture, Organization_Phone, Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)

2NF

1. (Organization_ID, Organization_Name, Organization_Email, Organization_Picture, Organization_Phone, Company_ID)
2. (Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)

3NF

No transitive dependency found.Same as 2NF.

1. (Organization_ID, Organization_Name, Organization_Email, Organization_Picture, Organization_Phone, Company_ID)
2. (Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)

Table after Normalization

1. (Organization_ID, Organization_Name, Organization_Email, Organization_Picture, Company_ID)
2. **Organization_Phone**(Organization_ID, Organization_Phone)
3. (Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)
4. (Company_ID, Company_Phone)
5. (Organization_ID, Company_ID)

4.7.2 (Companies → Teams)

UNF

(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig, Company_ID, Company_Name, Company_Email, Company_Password, Company_Picture, Company_Phone, Company_location)

1NF

Phone is multi-value attribute.

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig, Company_ID, Company_Name, Company_Email, Company_Password, Company_Picture, Company_Phone, Company_location)

2NF

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig

2. (Company_ID, Company_Name, Company_Email, Company_Password, Company_Picture, Company_Phone, Company_location)

3NF

No transitive dependency found.Same as 2NF.

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig

2. (Company_ID, Company_Name, Company_Email, Company_Password, Company_Picture, Company_Phone, Company_location)

Table after Normalization

1. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig

2. (Company_ID, Company_Name, Company_Email, Company_Password, Company_Picture, Company_Phone, Company_location)

3. (Team_ID, Company_ID)

4. (Company_ID, Company_Phone)

4.8 Temporary Tables

---

1. (Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate,**Admin_ID**)
2. (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
3. (Mp_ID, **Manager_ID**,Manager_Phone)
4. ~~(Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate)~~
5. (Finance_ID, Finance_Account_Number, Finance_Balance, **Manager_ID**)
6. ~~(Mp_ID, **Manager_ID**,Manager_Phone)~~
7. ~~(Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate)~~
8. (Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country, Total_Price_Money, **Manager_ID**
9. (Tw_ID, **Team_ID**, Team_Winnig)
10. ~~(Mp_ID, **Manager_ID**,Manager_Phone)~~
11. (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture , SocialMedia_Hiredate , SocialMedia_Salary)
12. (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture , ContentCreator_Hiredate , ContentCreator_Salary)
13. (Ccs_ID,**ContentCreator_ID** , ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_
Instagram_Link, ContentCreator_Youtube_Link)
14. (Cca_ID, **ContentCreator_ID**, ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
15. (Ccp_ID, **ContentCreator_ID**, ContentCreator_Phone)
16. (Smp_ID, **SocialMedia_ID**, SocialMedia_Phone)
17. ~~(Finance_ID, Finance_Account_Number, Finance_Balance)~~
18. (Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone, **Finance_ID**)
19. (Op_ID, **Organization_ID**,Organization_Phone)
20. ~~(Finance_ID, Finance_Account_Number, Finance_Balance)~~
21. ~~(SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Pictu , SocialMedia_Hiredate , SocialMedia_Salary, **Finance_ID**)~~
22. ~~(Smp_ID, **SocialMedia_ID**,SocialMedia_Phone)~~

23. ~~(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)~~

24. (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)

25. (Pa_ID, **Player_ID**, Player_country, Player_City, Player_Street, Player_Zip_Code)

26. (Psl_ID, **Player_ID**, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Yo

27. (Pp_ID, **Player_ID**, Player_Phone)

28. (Pw_ID, **Player_ID**, Player_Winnig)

29. (Pt_ID, **Player_ID**, **Team_ID**)

30. (Record_ID, Record_Date, Record_Price_Pool, **Tournament_ID**)

31. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

32. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

33. (Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

34. (Tournament_ID, Game_ID)

35. (Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool, Organization_ID)

36. ~~(Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture)~~

37. ~~(Organization_ID, Organization_Phone)~~

38. (Organization_ID, Tournament_ID)

39. ~~(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig)~~

40. ~~(Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)~~

41. (Team_ID, Game_ID)

42. ~~(Organization_ID, Organization_Name, Organization_Email, Organization_Picture, Company_ID)~~

43. ~~(Organization_ID, Organization_Phone)~~

44. (Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)

45. (Company_ID, Company_Phone)

46. (Organization_ID, Company_ID)

47. ~~(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country,Total_Earning, Team_Winnig~~

48. ~~(Company_ID, Company_Name, Company_Email, Company_Password, Company_Picture, Company_Phone, Company_location)~~

49. (Team_ID, Company_ID)

50. ~~(Company_ID, Company_Phone)~~

4.9 Final Tables

---

1. **Manager** (Manager_ID, Manager_Name, Manager_Email, Manager_Password, Manager_Picture, Manager_Hiredate,**Admin_ID**)
2. **Admin** (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
3. **Manager_Phone** (Mp_ID, **Manager_ID**,Manager_Phone)
4. **Finance**(Finance_ID, Finance_Account_Number, Finance_Balance, **Manager_ID**)
5. **Team**(Team_ID, Team_Name,Team_Icon, Team_established_date, Team_country, Total_Price_Money, **Manager_ID**
6. **Team_Winnig**(Tw_ID, **Team_ID**, Team_Winnig)
7. **SocialMedia** (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email, SocialMedia_Password, SocialMedia_Picture , SocialMedia_Hiredate , SocialMedia_Salary)
8. **ContentCreator** (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email, ContentCreator_Password, ContentCreator_Picture , ContentCreator_Hiredate , ContentCreator_Salary)
9. **ContentCreator_SocialMedia** (Ccs_ID,**ContentCreator_ID** , ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_ Instagram_Link, ContentCreator_Youtube_Link)
10. **ContentCreator_Address** (Cca_ID, **ContentCreator_ID**, ContentCreator_Country, ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
11. **ContentCreator_Phone** (Ccp_ID, **ContentCreator_ID**, ContentCreator_Phone)
12. **SocialMedia_Phone** (Smp_ID, **SocialMedia_ID**, SocialMedia_Phone)
13. **Organization**(Organization_ID, Organization_Name, Organization_Email, Organization_Password, Organization_Picture, Organization_Phone, **Finance_ID**)
14. **Organization_Phone** (Op_ID, **Organization_ID**,Organization_Phone)

15. **Player**(Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture, Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)

16. **Player_Address** (Pa_ID, **Player_ID**, Player_country, Player_City, Player_Street, Player_Zip_Code)

17. **Player_Social_Link** (Psl_ID, **Player_ID**, Player_Facebook_Link, Player_Instagram_Link, Player_Twitter_Link,Player_Youtube_Link)

18. **Player_Phone** (Pp_ID, **Player_ID**, Player_Phone)

19. **Player_Wining** (Pw_ID, **Player_ID**, Player_Winnig)

20. **Player_Team** (Pt_ID, **Player_ID**, **Team_ID**)

21. **Record**(Record_ID, Record_Date, Record_Price_Pool, **Tournament_ID**)

22. **Tournament**(Tournament_ID, Tournament_Name, Tournament_StartingDate,Tournament_EndingDate, Tournament_Location, Tournament_Prize_Pool)

23. **Game**(Game_ID, Game_Name, Game_Icon, Game_ReleaseDate, Game_Platform, Game_PricePool, Game_Genre, Game_Publisher)

24. **Tournament_Game**(Tournament_ID, Game_ID)

25. **Organization_Tournament**(Organization_ID, Tournament_ID)

26. **Team_Game**(Team_ID, Game_ID)

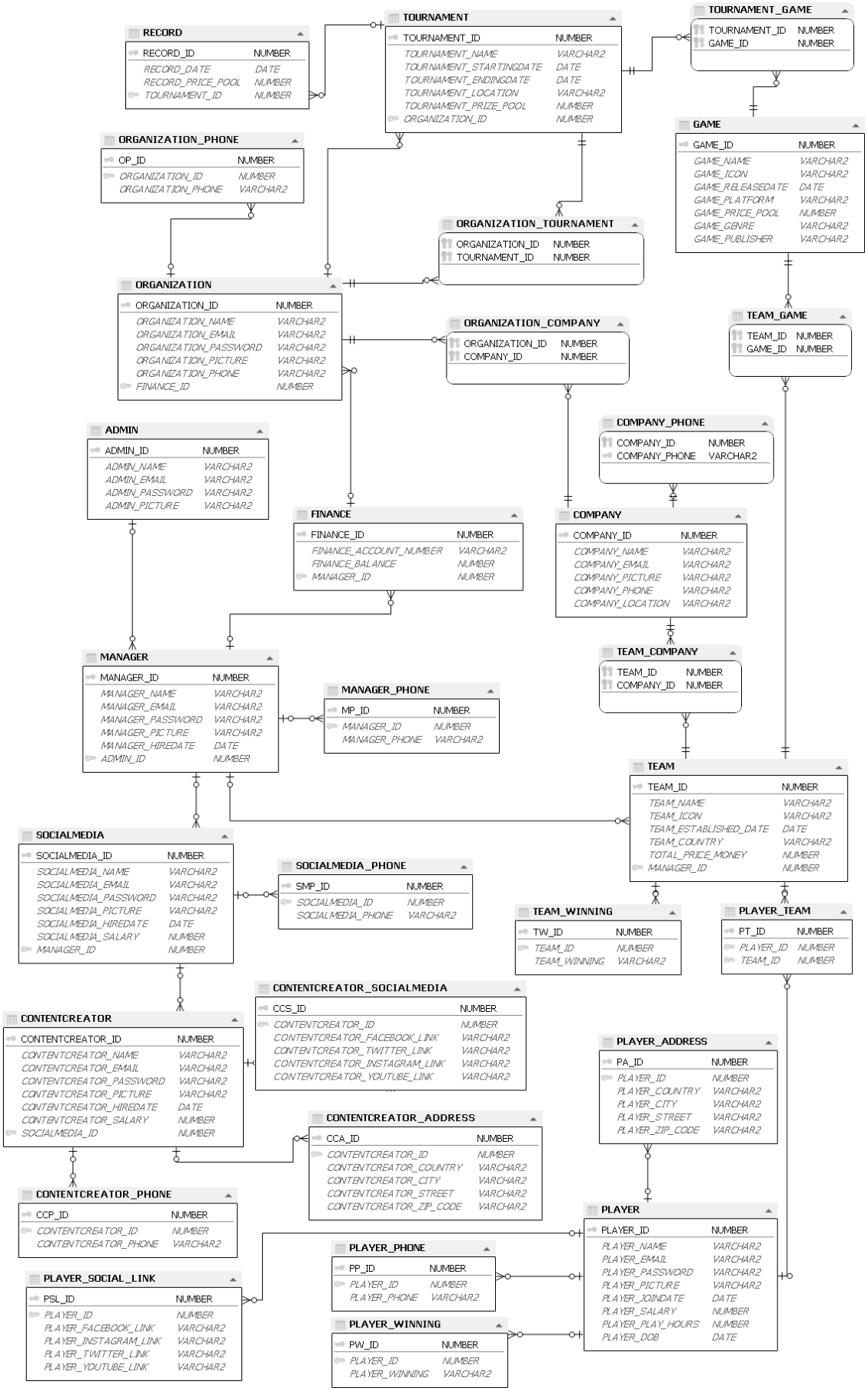
27. **Company**(Company_ID, Company_Name, Company_Email, Company_Picture, Company_Phone, location)

28. **Company_Phone**(Company_ID, Company_Phone)

29. **Organization_Company**(Organization_ID, Company_ID)

30. **Team_Company**(Team_ID, Company_ID)

4.10 Schema Diagram



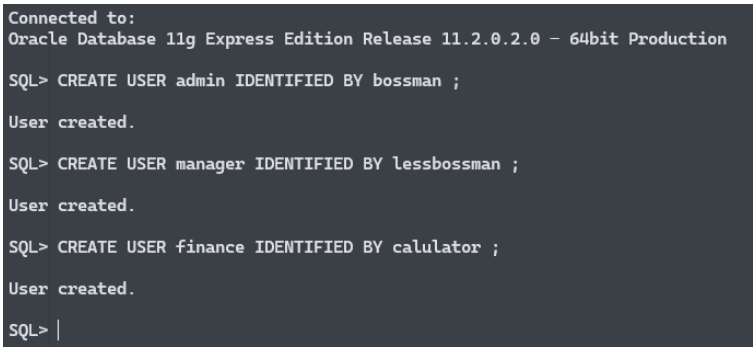
Schema Diagram

SQL Queries

5.1 User Creation

```
1 -- Create the users
2 CREATE USER admin IDENTIFIED BY bossman ;
3 CREATE USER manager IDENTIFIED BY lessbossman ;
4 CREATE USER finance IDENTIFIED BY calulator ;
```

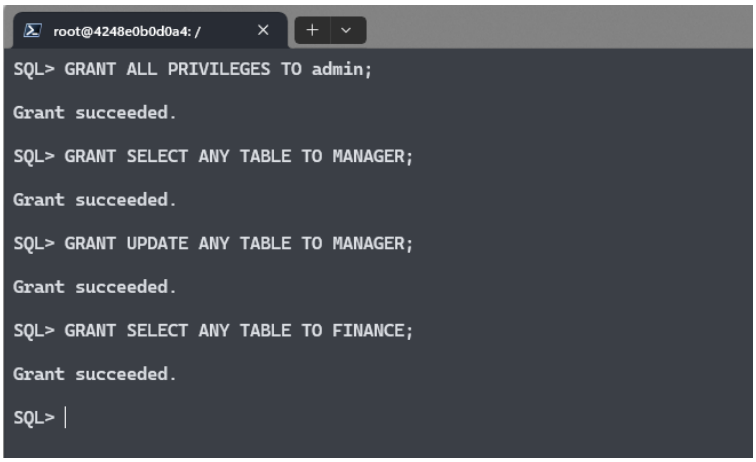
User Creation



Screenshot of User Creation from SQL Plus

```
1 -- Grant All privileges to admin user
2 GRANT ALL PRIVILEGES TO admin;
3
4 -- Grant DML privileges to manager user
5 GRANT SELECT ANY TABLE TO MANAGER;
6 GRANT UPDATE ANY TABLE TO MANAGER;
7
8 -- Grant Read only privileges to finance user
9 GRANT SELECT ANY TABLE TO FINANCE;
```

User Creation



Screenshot of User Privileges from SQL Plus

5.2 Table Creation

```
1   CREATE TABLE Admin (
2       Admin_ID INT PRIMARY KEY,
3       Admin_Name VARCHAR(100),
4       Admin_Email VARCHAR(100),
5       Admin_Password VARCHAR(100),
6       Admin_Picture VARCHAR(100)
7   );
```

Create Admin table

Results | Explain | Describe | Saved SQL | History

Object Type TABLE Object ADMIN

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ADMIN	ADMIN_ID	NUMBER	22	-	0	1	-	-	-
	ADMIN_NAME	VARCHAR2	100	-	-	-	✓	-	-
	ADMIN_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	ADMIN_PASSWORD	VARCHAR2	100	-	-	-	✓	-	-
	ADMIN_PICTURE	VARCHAR2	100	-	-	-	✓	-	-

Admin table description

```
1   CREATE TABLE Manager (
2       Manager_ID INT PRIMARY KEY,
3       Manager_Name VARCHAR(100),
4       Manager_Email VARCHAR(100),
5       Manager_Password VARCHAR(100),
6       Manager_Picture VARCHAR(100),
7       Manager_Hiredate DATE,
8       Admin_ID INT,
9       FOREIGN KEY (Admin_id) REFERENCES Manager (Admin_ID)
10  );
```

Create Manager table

Object Type TABLE Object MANAGER

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MANAGER	MANAGER_ID	NUMBER	22	-	0	1	-	-	-
	MANAGER_NAME	VARCHAR2	100	-	-	-	✓	-	-
	MANAGER_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	MANAGER_PASSWORD	VARCHAR2	100	-	-	-	✓	-	-
	MANAGER_PICTURE	VARCHAR2	100	-	-	-	✓	-	-
	MANAGER_HIREDATE	DATE	7	-	-	-	✓	-	-
	ADMIN_ID	NUMBER	22	-	0	-	✓	-	-

Manager table description

```
1   CREATE TABLE Manager_Phone (
2       Mp_ID INT PRIMARY KEY,
3       Manager_ID INT,
4       Manager_Phone VARCHAR(20),
5       FOREIGN KEY (Manager_ID) REFERENCES Manager (Manager_ID)
6   );
```

Create Manager Phone table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **MANAGER_PHONE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MANAGER_PHONE	MP_ID	NUMBER	22	-	0	1	-	-	-
	MANAGER_ID	NUMBER	22	-	0	-	✓	-	-
	MANAGER_PHONE	VARCHAR2	20	-	-	-	✓	-	-

Manager Phone table description

```sql
CREATE TABLE Finance (
    Finance_ID INT PRIMARY KEY,
    Finance_Account_Number VARCHAR(100),
    Finance_Balance DECIMAL(10, 2),
    Manager_ID INT,
    FOREIGN KEY (Manager_ID) REFERENCES Manager (Manager_ID)
);
```

Create Finance table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **FINANCE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FINANCE	FINANCE_ID	NUMBER	22	-	0	1	-	-	-
	FINANCE_ACCOUNT_NUMBER	VARCHAR2	100	-	-	-	✓	-	-
	FINANCE_BALANCE	NUMBER	-	10	2	-	✓	-	-
	MANAGER_ID	NUMBER	22	-	0	-	✓	-	-

Finance table description

```sql
CREATE TABLE Team (
    Team_ID INT PRIMARY KEY,
    Team_Name VARCHAR(100),
    Team_Icon VARCHAR(100),
    Team_Established_Date DATE,
    Team_Country VARCHAR(100),
    Total_Price_Money DECIMAL(10, 2),
    Manager_ID INT,
    FOREIGN KEY (Manager_ID) REFERENCES Manager (Manager_ID)
);
```

Create Team table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **TEAM**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TEAM	TEAM_ID	NUMBER	22	-	0	1	-	-	-
	TEAM_NAME	VARCHAR2	100	-	-	-	✓	-	-
	TEAM_ICON	VARCHAR2	100	-	-	-	✓	-	-
	TEAM_ESTABLISHED_DATE	DATE	7	-	-	-	✓	-	-
	TEAM_COUNTRY	VARCHAR2	100	-	-	-	✓	-	-
	TOTAL_PRICE_MONEY	NUMBER	-	10	2	-	✓	-	-
	MANAGER_ID	NUMBER	22	-	0	-	✓	-	-

Team table description

```sql
CREATE TABLE Team_Winning (
    Tw_ID INT PRIMARY KEY,
    Team_ID INT,
    Team_Winning VARCHAR(100),
    FOREIGN KEY (Team_ID) REFERENCES Team (Team_ID)
);
```

Create Team Winning table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **TEAM_WINNING**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TEAM_WINNING	TW_ID	NUMBER	22	-	0	1	-	-	-
	TEAM_ID	NUMBER	22	-	0	-	✓	-	-
	TEAM_WINNING	VARCHAR2	100	-	-	-	✓	-	-

Team Winning table description

```sql
CREATE TABLE SocialMedia (
    SocialMedia_ID INT PRIMARY KEY,
    SocialMedia_Name VARCHAR(100),
    SocialMedia_Email VARCHAR(100),
    SocialMedia_Password VARCHAR(100),
    SocialMedia_Picture VARCHAR(100),
    SocialMedia_Hiredate DATE,
    SocialMedia_Salary DECIMAL(10, 2),
    Manager_ID INT,
    FOREIGN KEY (Manager_ID) REFERENCES Manager (Manager_ID)
);
```

Create SocialMedia table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **SOCIALMEDIA**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SOCIALMEDIA	SOCIALMEDIA_ID	NUMBER	22	-	0	1	-	-	-
	SOCIALMEDIA_NAME	VARCHAR2	100	-	-	-	✓	-	-
	SOCIALMEDIA_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	SOCIALMEDIA_PASSWORD	VARCHAR2	100	-	-	-	✓	-	-
	SOCIALMEDIA_PICTURE	VARCHAR2	100	-	-	-	✓	-	-
	SOCIALMEDIA_HIREDATE	DATE	7	-	-	-	✓	-	-
	SOCIALMEDIA_SALARY	NUMBER	-	10	2	-	✓	-	-
	MANAGER_ID	NUMBER	22	-	0	-	✓	-	-

SocialMedia table description

```sql
CREATE TABLE ContentCreator (
    ContentCreator_ID INT PRIMARY KEY,
    ContentCreator_Name VARCHAR(100),
    ContentCreator_Email VARCHAR(100),
    ContentCreator_Password VARCHAR(100),
    ContentCreator_Picture VARCHAR(100),
    ContentCreator_Hiredate DATE,
    ContentCreator_Salary DECIMAL(10, 2),
    SocialMedia_ID INT,
    FOREIGN KEY (SocialMedia_ID) REFERENCES SocialMedia (SocialMedia_ID)
);
```

Create ContentCreator table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **CONTENTCREATOR**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CONTENTCREATOR	CONTENTCREATOR_ID	NUMBER	22	-	0	1	-	-	-
	CONTENTCREATOR_NAME	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_PASSWORD	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_PICTURE	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_HIREDATE	DATE	7	-	-	-	✓	-	-
	CONTENTCREATOR_SALARY	NUMBER	-	10	2	-	✓	-	-
	SOCIALMEDIA_ID	NUMBER	22	-	0	-	✓	-	-

ContentCreator table description

```sql
CREATE TABLE ContentCreator_SocialMedia (
    Ccs_ID INT PRIMARY KEY,
    ContentCreator_ID INT,
    ContentCreator_Facebook_Link VARCHAR(100),
    ContentCreator_Twitter_Link VARCHAR(100),
    ContentCreator_Instagram_Link VARCHAR(100),
    ContentCreator_Youtube_Link VARCHAR(100),
    FOREIGN KEY (ContentCreator_ID) REFERENCES ContentCreator (ContentCreator_ID)
);
```

Create ContentCreator SocialMedia table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **CONTENTCREATOR_SOCIALMEDIA**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CONTENTCREATOR_SOCIALMEDIA	CCS_ID	NUMBER	22	-	0	1	-	-	-
	CONTENTCREATOR_ID	NUMBER	22	-	0	-	✓	-	-
	CONTENTCREATOR_FACEBOOK_LINK	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_TWITTER_LINK	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_INSTAGRAM_LINK	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_YOUTUBE_LINK	VARCHAR2	100	-	-	-	✓	-	-

ContentCreator SocialMedia table description

```sql
CREATE TABLE ContentCreator_Address (
    Cca_ID INT PRIMARY KEY,
    ContentCreator_ID INT,
    ContentCreator_Country VARCHAR(100),
    ContentCreator_City VARCHAR(100),
    ContentCreator_Street VARCHAR(100),
    ContentCreator_Zip_Code VARCHAR(20),
    FOREIGN KEY (ContentCreator_ID) REFERENCES ContentCreator (ContentCreator_ID)
);
```

Create ContentCreator Address table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **CONTENTCREATOR_ADDRESS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CONTENTCREATOR_ADDRESS	CCA_ID	NUMBER	22	-	0	1	-	-	-
	CONTENTCREATOR_ID	NUMBER	22	-	0	-	✓	-	-
	CONTENTCREATOR_COUNTRY	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_CITY	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_STREET	VARCHAR2	100	-	-	-	✓	-	-
	CONTENTCREATOR_ZIP_CODE	VARCHAR2	20	-	-	-	✓	-	-

ContentCreator Address table description

```
1    CREATE TABLE ContentCreator_Phone (
2        Ccp_ID INT PRIMARY KEY,
3        ContentCreator_ID INT,
4        ContentCreator_Phone VARCHAR(20),
5        FOREIGN KEY (ContentCreator_ID) REFERENCES ContentCreator (ContentCreator_ID)
6    );
```

Create ContentCreator Phone table

```sql
CREATE TABLE SocialMedia_Phone (
    Smp_ID INT PRIMARY KEY,
    SocialMedia_ID INT,
    SocialMedia_Phone VARCHAR(20),
    FOREIGN KEY (SocialMedia_ID) REFERENCES SocialMedia (SocialMedia_ID)
);
```

Create SocialMedia Phone table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **SOCIALMEDIA_PHONE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SOCIALMEDIA_PHONE	SMP_ID	NUMBER	22	-	0	1	-	-	-
	SOCIALMEDIA_ID	NUMBER	22	-	0	-	✓	-	-
	SOCIALMEDIA_PHONE	VARCHAR2	20	-	-	-	✓	-	-

SocialMedia Phone table description

```sql
CREATE TABLE Organization (
    Organization_ID INT PRIMARY KEY,
    Organization_Name VARCHAR(100),
    Organization_Email VARCHAR(100),
    Organization_Password VARCHAR(100),
    Organization_Picture VARCHAR(100),
    Organization_Phone VARCHAR(20),
    Finance_ID INT,
    FOREIGN KEY (Finance_ID) REFERENCES Finance (Finance_ID)
);
```

Create Organization table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **ORGANIZATION**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ORGANIZATION	ORGANIZATION_ID	NUMBER	22	-	0	1	-	-	-
	ORGANIZATION_NAME	VARCHAR2	100	-	-	-	✓	-	-
	ORGANIZATION_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	ORGANIZATION_PASSWORD	VARCHAR2	100	-	-	-	✓	-	-
	ORGANIZATION_PICTURE	VARCHAR2	100	-	-	-	✓	-	-
	ORGANIZATION_PHONE	VARCHAR2	20	-	-	-	✓	-	-
	FINANCE_ID	NUMBER	22	-	0	-	✓	-	-

Organization table description

```sql
CREATE TABLE Organization_Phone (
    Op_ID INT PRIMARY KEY,
    Organization_ID INT,
    Organization_Phone VARCHAR(20),
    FOREIGN KEY (Organization_ID) REFERENCES Organization (Organization_ID)
);
```

Create Organization Phone table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **ORGANIZATION_PHONE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ORGANIZATION_PHONE	OP_ID	NUMBER	22	-	0	1	-	-	-
	ORGANIZATION_ID	NUMBER	22	-	0	-	✓	-	-
	ORGANIZATION_PHONE	VARCHAR2	20	-	-	-	✓	-	-

Organization Phone table description

```sql
CREATE TABLE Player (
    Player_ID INT PRIMARY KEY,
    Player_Name VARCHAR(100),
    Player_Email VARCHAR(100),
    Player_Password VARCHAR(100),
    Player_Picture VARCHAR(100),
    Player_JoinDate DATE,
    Player_Salary DECIMAL(10, 2),
    Player_Play_Hours INT,
    Player_DOB DATE
);
```

Create Player table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **PLAYER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER	PLAYER_ID	NUMBER	22	-	0	1	-	-	-
	PLAYER_NAME	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_PASSWORD	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_PICTURE	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_JOINDATE	DATE	7	-	-	-	✓	-	-
	PLAYER_SALARY	NUMBER	-	10	2	-	✓	-	-
	PLAYER_PLAY_HOURS	NUMBER	22	-	0	-	✓	-	-
	PLAYER_DOB	DATE	7	-	-	-	✓	-	-

Player table description

```sql
CREATE TABLE Player_Address (
    Pa_ID INT PRIMARY KEY,
    Player_ID INT,
    Player_Country VARCHAR(100),
    Player_City VARCHAR(100),
    Player_Street VARCHAR(100),
    Player_Zip_Code VARCHAR(20),
    FOREIGN KEY (Player_ID) REFERENCES Player (Player_ID)
);
```

Create Player Address table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **PLAYER_ADDRESS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER_ADDRESS	PA_ID	NUMBER	22	-	0	1	-	-	-
	PLAYER_ID	NUMBER	22	-	0	-	✓	-	-
	PLAYER_COUNTRY	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_CITY	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_STREET	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_ZIP_CODE	VARCHAR2	20	-	-	-	✓	-	-

Player Address table description

```sql
CREATE TABLE Player_Social_Link (
    Psl_ID INT PRIMARY KEY,
    Player_ID INT,
    Player_Facebook_Link VARCHAR(100),
    Player_Instagram_Link VARCHAR(100),
    Player_Twitter_Link VARCHAR(100),
    Player_Youtube_Link VARCHAR(100),
    FOREIGN KEY (Player_ID) REFERENCES Player (Player_ID)
);
```

Create Player Social Link table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **PLAYER_SOCIAL_LINK**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER_SOCIAL_LINK	PSL_ID	NUMBER	22	-	0	1	-	-	-
	PLAYER_ID	NUMBER	22	-	0	-	✓	-	-
	PLAYER_FACEBOOK_LINK	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_INSTAGRAM_LINK	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_TWITTER_LINK	VARCHAR2	100	-	-	-	✓	-	-
	PLAYER_YOUTUBE_LINK	VARCHAR2	100	-	-	-	✓	-	-

Player Social Link table description

```sql
CREATE TABLE Player_Phone (
    Pp_ID INT PRIMARY KEY,
    Player_ID INT,
    Player_Phone VARCHAR(20),
    FOREIGN KEY (Player_ID) REFERENCES Player (Player_ID)
);
```

Create Player Phone table

Results Explain Describe Saved SQL History

Object Type TABLE Object PLAYER_PHONE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER_PHONE	PP_ID	NUMBER	22	-	0	1	-	-	-
	PLAYER_ID	NUMBER	22	-	0	-	✓	-	-
	PLAYER_PHONE	VARCHAR2	20	-	-	-	✓	-	-

Player Phone table description

```sql
CREATE TABLE Player_Winning (
    Pw_ID INT PRIMARY KEY,
    Player_ID INT,
    Player_Winning VARCHAR(100),
    FOREIGN KEY (Player_ID) REFERENCES Player (Player_ID)
);
```

Create Player Winning table

Results Explain Describe Saved SQL History

Object Type TABLE Object PLAYER_WINNING

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER_WINNING	PW_ID	NUMBER	22	-	0	1	-	-	-
	PLAYER_ID	NUMBER	22	-	0	-	✓	-	-
	PLAYER_WINNING	VARCHAR2	100	-	-	-	✓	-	-

Player Winning table description

```sql
CREATE TABLE Player_Team (
    Pt_ID INT PRIMARY KEY,
    Player_ID INT,
    Team_ID INT,
    FOREIGN KEY (Player_ID) REFERENCES Player (Player_ID),
    FOREIGN KEY (Team_ID) REFERENCES Team (Team_ID)
);
```

Create Player Team table

Results Explain Describe Saved SQL History

Object Type TABLE Object PLAYER_TEAM

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER_TEAM	PT_ID	NUMBER	22	-	0	1	-	-	-
	PLAYER_ID	NUMBER	22	-	0	-	✓	-	-
	TEAM_ID	NUMBER	22	-	0	-	✓	-	-

Player Team table description

```sql
CREATE TABLE Tournament (
    Tournament_ID INT PRIMARY KEY,
    Tournament_Name VARCHAR(100),
    Tournament_StartingDate DATE,
    Tournament_EndingDate DATE,
    Tournament_Location VARCHAR(100),
    Tournament_Prize_Pool DECIMAL(10, 2),
    Organization_ID INT,
    FOREIGN KEY (Organization_ID) REFERENCES Organization (Organization_ID)
);
```

Create Tournament table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **TOURNAMENT**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TOURNAMENT	TOURNAMENT_ID	NUMBER	22	-	0	1	-	-	-
	TOURNAMENT_NAME	VARCHAR2	100	-	-	-	✓	-	-
	TOURNAMENT_STARTINGDATE	DATE	7	-	-	-	✓	-	-
	TOURNAMENT_ENDINGDATE	DATE	7	-	-	-	✓	-	-
	TOURNAMENT_LOCATION	VARCHAR2	100	-	-	-	✓	-	-
	TOURNAMENT_PRIZE_POOL	NUMBER	-	10	2	-	✓	-	-
	ORGANIZATION_ID	NUMBER	22	-	0	-	✓	-	-

Tournament table description

```sql
CREATE TABLE Record (
    Record_ID INT PRIMARY KEY,
    Record_Date DATE,
    Record_Price_Pool DECIMAL(10, 2),
    Tournament_ID INT,
    FOREIGN KEY (Tournament_ID) REFERENCES Tournament (Tournament_ID)
);
```

Create Record table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **RECORD**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
RECORD	RECORD_ID	NUMBER	22	-	0	1	-	-	-
	RECORD_DATE	DATE	7	-	-	-	✓	-	-
	RECORD_PRICE_POOL	NUMBER	-	10	2	-	✓	-	-
	TOURNAMENT_ID	NUMBER	22	-	0	-	✓	-	-

Record table description

```sql
CREATE TABLE Game (
    Game_ID INT PRIMARY KEY,
    Game_Name VARCHAR(100),
    Game_Icon VARCHAR(100),
    Game_ReleaseDate DATE,
    Game_Platform VARCHAR(100),
    Game_Price_Pool DECIMAL(10, 2),
    Game_Genre VARCHAR(100),
    Game_Publisher VARCHAR(100)
);
```

Create Game table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **GAME**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GAME	GAME_ID	NUMBER	22	-	0	1	-	-	-
	GAME_NAME	VARCHAR2	100	-	-	-	✓	-	-
	GAME_ICON	VARCHAR2	100	-	-	-	✓	-	-
	GAME_RELEASEDATE	DATE	7	-	-	-	✓	-	-
	GAME_PLATFORM	VARCHAR2	100	-	-	-	✓	-	-
	GAME_PRICE_POOL	NUMBER	-	10	2	-	✓	-	-
	GAME_GENRE	VARCHAR2	100	-	-	-	✓	-	-
	GAME_PUBLISHER	VARCHAR2	100	-	-	-	✓	-	-

Game table description

```
1  CREATE TABLE Tournament_Game (
2      Tournament_ID INT,
3      Game_ID INT,
4      PRIMARY KEY (Tournament_ID, Game_ID),
5      FOREIGN KEY (Tournament_ID) REFERENCES Tournament (Tournament_ID),
6      FOREIGN KEY (Game_ID) REFERENCES Game (Game_ID)
7  );
```

Create Tournament Game table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **TOURNAMENT_GAME**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TOURNAMENT_GAME	TOURNAMENT_ID	NUMBER	22	-	0	1	-	-	-
	GAME_ID	NUMBER	22	-	0	2	-	-	-

Tournament Game table description

```
1  CREATE TABLE Organization_Tournament (
2      Organization_ID INT,
3      Tournament_ID INT,
4      PRIMARY KEY (Organization_ID, Tournament_ID),
5      FOREIGN KEY (Organization_ID) REFERENCES Organization (Organization_ID),
6      FOREIGN KEY (Tournament_ID) REFERENCES Tournament (Tournament_ID)
7  );
```

Create Organization Tournament table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **ORGANIZATION_TOURNAMENT**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ORGANIZATION_TOURNAMENT	ORGANIZATION_ID	NUMBER	22	-	0	1	-	-	-
	TOURNAMENT_ID	NUMBER	22	-	0	2	-	-	-

Organization Tournament table description

```sql
CREATE TABLE Team_Game (
    Team_ID INT,
    Game_ID INT,
    PRIMARY KEY (Team_ID, Game_ID),
    FOREIGN KEY (Team_ID) REFERENCES Team (Team_ID),
    FOREIGN KEY (Game_ID) REFERENCES Game (Game_ID)
);
```

Create Team Game table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **TEAM_GAME**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TEAM_GAME	TEAM_ID	NUMBER	22	-	0	1	-	-	-
	GAME_ID	NUMBER	22	-	0	2	-	-	-

Team Game table description

```sql
CREATE TABLE Company (
    Company_ID INT PRIMARY KEY,
    Company_Name VARCHAR(100),
    Company_Email VARCHAR(100),
    Company_Picture VARCHAR(100),
    Company_Phone VARCHAR(20),
    Company_Location VARCHAR(100)
);
```

Create Company table

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **COMPANY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
COMPANY	COMPANY_ID	NUMBER	22	-	0	1	-	-	-
	COMPANY_NAME	VARCHAR2	100	-	-	-	✓	-	-
	COMPANY_EMAIL	VARCHAR2	100	-	-	-	✓	-	-
	COMPANY_PICTURE	VARCHAR2	100	-	-	-	✓	-	-
	COMPANY_PHONE	VARCHAR2	20	-	-	-	✓	-	-
	COMPANY_LOCATION	VARCHAR2	100	-	-	-	✓	-	-

Company table description

```sql
CREATE TABLE Company_Phone (
    Company_ID INT,
    Company_Phone VARCHAR(20),
    PRIMARY KEY (Company_ID, Company_Phone),
    FOREIGN KEY (Company_ID) REFERENCES Company (Company_ID)
);
```

Create Company Phone table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **COMPANY_PHONE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
COMPANY_PHONE	COMPANY_ID	NUMBER	22	-	0	1	-	-	-
	COMPANY_PHONE	VARCHAR2	20	-	-	2	-	-	-

Company Phone table description

```
1  CREATE TABLE Organization_Company (
2      Organization_ID INT,
3      Company_ID INT,
4      PRIMARY KEY (Organization_ID, Company_ID),
5      FOREIGN KEY (Organization_ID) REFERENCES Organization (Organization_ID),
6      FOREIGN KEY (Company_ID) REFERENCES Company (Company_ID)
7  );
```

Create Organization Company table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **ORGANIZATION_COMPANY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ORGANIZATION_COMPANY	ORGANIZATION_ID	NUMBER	22	-	0	1	-	-	-
	COMPANY_ID	NUMBER	22	-	0	2	-	-	-

Organization Company table description

```
1  CREATE TABLE Team_Company (
2      Team_ID INT,
3      Company_ID INT,
4      PRIMARY KEY (Team_ID, Company_ID),
5      FOREIGN KEY (Team_ID) REFERENCES Team (Team_ID),
6      FOREIGN KEY (Company_ID) REFERENCES Company (Company_ID)
7  );
```

Create Team Company table

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **TEAM_COMPANY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TEAM_COMPANY	TEAM_ID	NUMBER	22	-	0	1	-	-	-
	COMPANY_ID	NUMBER	22	-	0	2	-	-	-

Team Company table description

5.3 Sequence Creation

```
-- Create sequence for Manager table
CREATE SEQUENCE seq_manager_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Admin table
CREATE SEQUENCE seq_admin_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Manager_Phone table
CREATE SEQUENCE seq_mp_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Finance table
CREATE SEQUENCE seq_finance_id START WITH 1 INCREMENT BY 1;

-- Create sequence for Team table
CREATE SEQUENCE seq_team_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Team_Winning table
CREATE SEQUENCE seq_tw_id START WITH 1 INCREMENT BY 1;
-- Create sequence for SocialMedia table
CREATE SEQUENCE seq_socialmedia_id START WITH 1 INCREMENT BY 1;
-- Create sequence for ContentCreator table
CREATE SEQUENCE seq_contentcreator_id START WITH 1 INCREMENT BY 1;
-- Create sequence for ContentCreator_SocialMedia table
CREATE SEQUENCE seq_ccs_id START WITH 1 INCREMENT BY 1;
-- Create sequence for ContentCreator_Address table
CREATE SEQUENCE seq_cca_id START WITH 1 INCREMENT BY 1;
-- Create sequence for ContentCreator_Phone table
CREATE SEQUENCE seq_ccp_id START WITH 1 INCREMENT BY 1;

-- Create sequence for SocialMedia_Phone table
CREATE SEQUENCE seq_smp_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Organization table
CREATE SEQUENCE seq_organization_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Organization_Phone table
CREATE SEQUENCE seq_op_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Player table
CREATE SEQUENCE seq_player_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Player_Address table
CREATE SEQUENCE seq_pa_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Player_Social_Link table
CREATE SEQUENCE seq_psl_id START WITH 1 INCREMENT BY 1;

-- Create sequence for Player_Phone table
CREATE SEQUENCE seq_pp_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Player_Winning table
CREATE SEQUENCE seq_pw_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Record table
CREATE SEQUENCE seq_record_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Tournament table
CREATE SEQUENCE seq_tournament_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Game table
CREATE SEQUENCE seq_game_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Company table
CREATE SEQUENCE seq_company_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Company_Phone table
CREATE SEQUENCE seq_cp_id START WITH 1 INCREMENT BY 1;
-- Create sequence for Organization_Company table
CREATE SEQUENCE seq_oc_id START WITH 1 INCREMENT BY 1;
--
```

Sequence Creation

5.4 Index for Table

```
    -- Create index for Manager table
    CREATE INDEX idx_manager_email ON Manager (Manager_Email);
    -- Create index for Admin table
    CREATE INDEX idx_admin_email ON Admin (Admin_Email);
    -- Create index for Manager_Phone table
    CREATE INDEX idx_manager_phone_manager_id ON Manager_Phone (Manager_ID);
    -- Create index for Finance table
    CREATE INDEX idx_finance_manager_id ON Finance (Manager_ID);
    -- Create index for Team table
    CREATE INDEX idx_team_manager_id ON Team (Manager_ID);

    -- Create index for Team_Winning table
    CREATE INDEX idx_team_winning_team_id ON Team_Winning (Team_ID);
    -- Create index for SocialMedia table
    CREATE INDEX idx_socialmedia_name ON SocialMedia (SocialMedia_Name);
    -- Create index for ContentCreator table
    CREATE INDEX idx_contentcreator_name ON ContentCreator (ContentCreator_Name);
    -- Create index for ContentCreator_SocialMedia table
    CREATE INDEX idx_ccs_contentcreator_id ON ContentCreator_SocialMedia (ContentCreator_ID)
        ;
    -- Create index for ContentCreator_Address table
    CREATE INDEX idx_cca_contentcreator_id ON ContentCreator_Address (ContentCreator_ID);
    -- Create index for ContentCreator_Phone table
    CREATE INDEX idx_ccp_contentcreator_id ON ContentCreator_Phone (ContentCreator_ID);
    -- Create index for SocialMedia_Phone table
    CREATE INDEX idx_smp_socialmedia_id ON SocialMedia_Phone (SocialMedia_ID);
    -- Create index for Organization table
    CREATE INDEX idx_organization_name ON Organization (Organization_Name);
    -- Create index for Organization_Phone table
    CREATE INDEX idx_op_organization_id ON Organization_Phone (Organization_ID);
    -- Create index for Player table
    CREATE INDEX idx_player_name ON Player (Player_Name);

    -- Create index for Player_Address table
    CREATE INDEX idx_pa_player_id ON Player_Address (Player_ID);
    -- Create index for Player_Social_Link table
    CREATE INDEX idx_psl_player_id ON Player_Social_Link (Player_ID);
    -- Create index for Player_Phone table
    CREATE INDEX idx_pp_player_id ON Player_Phone (Player_ID);
    -- Create index for Player_Winning table
    CREATE INDEX idx_pw_player_id ON Player_Winning (Player_ID);
    -- Create index for Record table
    CREATE INDEX idx_record_date ON Record (Record_Date);

    -- Create index for Tournament table
    CREATE INDEX idx_tournament_name ON Tournament (Tournament_Name);
    -- Create index for Game table
    CREATE INDEX idx_game_name ON Game (Game_Name);
    -- Create index for Company table
    CREATE INDEX idx_company_name ON Company (Company_Name);
    -- Create index for Company_Phone table
    CREATE INDEX idx_cp_company_id ON Company_Phone (Company_ID);
    -- Create index for Organization_Company table
    CREATE INDEX idx_oc_organization_id ON Organization_Company (Organization_ID);
    --
```

Index for Table

5.5 Alter Table for effective indexing

```
-- Alter Manager table to add index
ALTER TABLE Manager ADD CONSTRAINT idx_manager_email UNIQUE (Manager_Email);
-- Alter Admin table to add index
ALTER TABLE Admin ADD CONSTRAINT idx_admin_email UNIQUE (Admin_Email);
-- Alter Manager_Phone table to add index
ALTER TABLE Manager_Phone ADD CONSTRAINT idx_manager_phone_manager_id UNIQUE (Manager_ID);
-- Alter Finance table to add index
ALTER TABLE Finance ADD CONSTRAINT idx_finance_manager_id UNIQUE (Manager_ID);
-- Alter Team table to add index
ALTER TABLE Team ADD CONSTRAINT idx_team_manager_id UNIQUE (Manager_ID);
-- Alter Team_Winning table to add index
ALTER TABLE Team_Winning ADD CONSTRAINT idx_team_winning_team_id UNIQUE (Team_ID);

-- Alter SocialMedia table to add index
ALTER TABLE SocialMedia ADD CONSTRAINT idx_socialmedia_name UNIQUE (SocialMedia_Name);
-- Alter ContentCreator table to add index
ALTER TABLE ContentCreator ADD CONSTRAINT idx_contentcreator_name UNIQUE (ContentCreator_Name);
-- Alter ContentCreator_SocialMedia table to add index
ALTER TABLE ContentCreator_SocialMedia ADD CONSTRAINT idx_ccs_contentcreator_id UNIQUE (ContentCreator_ID);
-- Alter ContentCreator_Address table to add index
ALTER TABLE ContentCreator_Address ADD CONSTRAINT idx_cca_contentcreator_id UNIQUE (ContentCreator_ID);
-- Alter ContentCreator_Phone table to add index
ALTER TABLE ContentCreator_Phone ADD CONSTRAINT idx_ccp_contentcreator_id UNIQUE (ContentCreator_ID);
-- Alter SocialMedia_Phone table to add index
ALTER TABLE SocialMedia_Phone ADD CONSTRAINT idx_smp_socialmedia_id UNIQUE (SocialMedia_ID);
-- Alter Organization table to add index
ALTER TABLE Organization ADD CONSTRAINT idx_organization_name UNIQUE (Organization_Name);
-- Alter Organization_Phone table to add index
ALTER TABLE Organization_Phone ADD CONSTRAINT idx_op_organization_id UNIQUE (Organization_ID);

-- Alter Player table to add index
ALTER TABLE Player ADD CONSTRAINT idx_player_name UNIQUE (Player_Name);
-- Alter Player_Address table to add index
ALTER TABLE Player_Address ADD CONSTRAINT idx_pa_player_id UNIQUE (Player_ID);
-- Alter Player_Social_Link table to add index
ALTER TABLE Player_Social_Link ADD CONSTRAINT idx_psl_player_id UNIQUE (Player_ID);
-- Alter Player_Phone table to add index
ALTER TABLE Player_Phone ADD CONSTRAINT idx_pp_player_id UNIQUE (Player_ID);
-- Alter Player_Winning table to add index
ALTER TABLE Player_Winning ADD CONSTRAINT idx_pw_player_id UNIQUE (Player_ID);
-- Alter Record table to add index
ALTER TABLE Record ADD CONSTRAINT idx_record_date UNIQUE (Record_Date);

-- Alter Tournament table to add index
ALTER TABLE Tournament ADD CONSTRAINT idx_tournament_name UNIQUE (Tournament_Name);
-- Alter Game table to add index
ALTER TABLE Game ADD CONSTRAINT idx_game_name UNIQUE (Game_Name);
-- Alter Company table to add index
ALTER TABLE Company ADD CONSTRAINT idx_company_name UNIQUE (Company_Name);
-- Alter Company_Phone table to add index
ALTER TABLE Company_Phone ADD CONSTRAINT idx_cp_company_id UNIQUE (Company_ID);
-- Alter Organization_Company table to add index
ALTER TABLE Organization_Company ADD CONSTRAINT idx_oc_organization_id UNIQUE (Organization_ID);
--
```

Alter Table

5.6 Data Insertion

```sql
INSERT INTO Admin (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
VALUES (seq_admin_id.NEXTVAL, 'Admin 1', 'admin1@example.com', 'adminpass1', 'admin1.jpg');

INSERT INTO Admin (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
VALUES (seq_admin_id.NEXTVAL, 'Admin 2', 'admin2@example.com', 'adminpass2', 'admin2.jpg');

INSERT INTO Admin (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
VALUES (seq_admin_id.NEXTVAL, 'Admin 3', 'admin3@example.com', 'adminpass3', 'admin3.jpg');

INSERT INTO Admin (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
VALUES (seq_admin_id.NEXTVAL, 'Admin 4', 'admin4@example.com', 'adminpass4', 'admin4.jpg');

INSERT INTO Admin (Admin_ID, Admin_Name, Admin_Email, Admin_Password, Admin_Picture)
VALUES (seq_admin_id.NEXTVAL, 'Admin 5', 'admin5@example.com', 'adminpass5', 'admin5.jpg');
```

Inserting data into Admin table

Results | Explain | Describe | Saved SQL | History

ADMIN_ID	ADMIN_NAME	ADMIN_EMAIL	ADMIN_PASSWORD	ADMIN_PICTURE
1	Admin 1	admin1@example.com	adminpass1	admin1.jpg
2	Admin 2	admin2@example.com	adminpass2	admin2.jpg
3	Admin 3	admin3@example.com	adminpass3	admin3.jpg
4	Admin 4	admin4@example.com	adminpass4	admin4.jpg
5	Admin 5	admin5@example.com	adminpass5	admin5.jpg

5 rows returned in 0.00 seconds

Inserted data of Admin table

```sql
INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Password,
    Manager_Picture, Manager_Hiredate, Admin_ID)
VALUES (seq_manager_id.NEXTVAL, 'John Doe', 'john.doe@example.com', 'password123', 'profile.
    jpg', TO_DATE('2022-01-01', 'YYYY-MM-DD'), 1);

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Password,
    Manager_Picture, Manager_Hiredate, Admin_ID)
VALUES (seq_manager_id.NEXTVAL, 'Jane Smith', 'jane.smith@example.com', 'password456', '
    profile2.jpg', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 2);

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Password,
    Manager_Picture, Manager_Hiredate, Admin_ID)
VALUES (seq_manager_id.NEXTVAL, 'Mike Johnson', 'mike.johnson@example.com', 'password789', '
    profile3.jpg', TO_DATE('2022-03-01', 'YYYY-MM-DD'), 1);

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Password,
    Manager_Picture, Manager_Hiredate, Admin_ID)
VALUES (seq_manager_id.NEXTVAL, 'Sarah Williams', 'sarah.williams@example.com', 'password123
    ', 'profile4.jpg', TO_DATE('2022-04-01', 'YYYY-MM-DD'), 2);

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Password,
    Manager_Picture, Manager_Hiredate, Admin_ID)
VALUES (seq_manager_id.NEXTVAL, 'Robert Davis', 'robert.davis@example.com', 'password456', '
    profile5.jpg', TO_DATE('2022-05-01', 'YYYY-MM-DD'), 1);
```

Inserting data into manager tables

Results | Explain | Describe | Saved SQL | History

MANAGER_ID	MANAGER_NAME	MANAGER_EMAIL	MANAGER_PASSWORD	MANAGER_PICTURE	MANAGER_HIREDATE	ADMIN_ID
1	John Doe	john.doe@example.com	password123	profile.jpg	01-JAN-22	1
2	Jane Smith	jane.smith@example.com	password456	profile2.jpg	01-FEB-22	2
3	Mike Johnson	mike.johnson@example.com	password789	profile3.jpg	01-MAR-22	1
4	Sarah Williams	sarah.williams@example.com	password123	profile4.jpg	01-APR-22	2
5	Robert Davis	robert.davis@example.com	password456	profile5.jpg	01-MAY-22	1

5 rows returned in 0.00 seconds

Inserted data of manager table

```
INSERT INTO Manager_Phone (Mp_ID, Manager_ID, Manager_Phone)
VALUES (seq_mp_id.NEXTVAL, 1, '1234567890');

INSERT INTO Manager_Phone (Mp_ID, Manager_ID, Manager_Phone)
VALUES (seq_mp_id.NEXTVAL, 2, '0987654321');

INSERT INTO Manager_Phone (Mp_ID, Manager_ID, Manager_Phone)
VALUES (seq_mp_id.NEXTVAL, 3, '1112223333');

INSERT INTO Manager_Phone (Mp_ID, Manager_ID, Manager_Phone)
VALUES (seq_mp_id.NEXTVAL, 4, '4445556666');

INSERT INTO Manager_Phone (Mp_ID, Manager_ID, Manager_Phone)
VALUES (seq_mp_id.NEXTVAL, 5, '7778889999');
```

Inserting data into manager phone tables

Results | Explain | Describe | Saved SQL | History

MP_ID	MANAGER_ID	MANAGER_PHONE
1	1	1234567890
2	2	0987654321
3	3	1112223333
4	4	4445556666
5	5	7778889999

5 rows returned in 0.00 seconds

Inserted data of manager phone table

```
INSERT INTO Finance (Finance_ID, Finance_Account_Number, Finance_Balance, Manager_ID)
VALUES (seq_finance_id.NEXTVAL, 'ABC123456', 10000, 1);

INSERT INTO Finance (Finance_ID, Finance_Account_Number, Finance_Balance, Manager_ID)
VALUES (seq_finance_id.NEXTVAL, 'DEF789012', 20000, 2);

INSERT INTO Finance (Finance_ID, Finance_Account_Number, Finance_Balance, Manager_ID)
VALUES (seq_finance_id.NEXTVAL, 'GHI345678', 15000, 3);

INSERT INTO Finance (Finance_ID, Finance_Account_Number, Finance_Balance, Manager_ID)
VALUES (seq_finance_id.NEXTVAL, 'JKL901234', 18000, 4);

INSERT INTO Finance (Finance_ID, Finance_Account_Number, Finance_Balance, Manager_ID)
VALUES (seq_finance_id.NEXTVAL, 'MNO567890', 22000, 5);
```

Inserting data into Finance tables

Results | Explain | Describe | Saved SQL | History

FINANCE_ID	FINANCE_ACCOUNT_NUMBER	FINANCE_BALANCE	MANAGER_ID
1	ABC123456	10000	1
2	DEF789012	20000	2
3	GHI345678	15000	3
4	JKL901234	18000	4
5	MNO567890	22000	5

5 rows returned in 0.00 seconds

Inserted data of Finance table

```
1
2  INSERT INTO Team (Team_ID, Team_Name, Team_Icon, Team_Established_Date, Team_Country,
       Total_Price_Money, Manager_ID)
3  VALUES (seq_team_id.NEXTVAL, 'Team A', 'teamA.png', TO_DATE('2022-01-01', 'YYYY-MM-DD'), '
       USA', 50000, 1);
4
5  INSERT INTO Team (Team_ID, Team_Name, Team_Icon, Team_Established_Date, Team_Country,
       Total_Price_Money, Manager_ID)
6  VALUES (seq_team_id.NEXTVAL, 'Team B', 'teamB.png', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 'UK
       ', 60000, 2);
7
8  INSERT INTO Team (Team_ID, Team_Name, Team_Icon, Team_Established_Date, Team_Country,
       Total_Price_Money, Manager_ID)
9  VALUES (seq_team_id.NEXTVAL, 'Team C', 'teamC.png', TO_DATE('2022-03-01', 'YYYY-MM-DD'), '
       Australia', 45000, 3);
10
11 INSERT INTO Team (Team_ID, Team_Name, Team_Icon, Team_Established_Date, Team_Country,
       Total_Price_Money, Manager_ID)
12 VALUES (seq_team_id.NEXTVAL, 'Team D', 'teamD.png', TO_DATE('2022-04-01', 'YYYY-MM-DD'), '
       Canada', 55000, 4);
13
14 INSERT INTO Team (Team_ID, Team_Name, Team_Icon, Team_Established_Date, Team_Country,
       Total_Price_Money, Manager_ID)
15 VALUES (seq_team_id.NEXTVAL, 'Team E', 'teamE.png', TO_DATE('2022-05-01', 'YYYY-MM-DD'), '
       Germany', 70000, 5);
```

Inserting data into Team tables

Results | Explain | Describe | Saved SQL | History

TEAM_ID	TEAM_NAME	TEAM_ICON	TEAM_ESTABLISHED_DATE	TEAM_COUNTRY	TOTAL_PRICE_MONEY	MANAGER_ID
1	Team A	teamA.png	01-JAN-22	USA	50000	1
2	Team B	teamB.png	01-FEB-22	UK	60000	2
3	Team C	teamC.png	01-MAR-22	Australia	45000	3
4	Team D	teamD.png	01-APR-22	Canada	55000	4
5	Team E	teamE.png	01-MAY-22	Germany	70000	5

5 rows returned in 0.00 seconds

Inserted data of Team table

```
1
2  INSERT INTO Team_Winning (Tw_ID, Team_ID, Team_Winning)
3  VALUES (seq_tw_id.NEXTVAL, 1, 'Championship 2022');
4
5  INSERT INTO Team_Winning (Tw_ID, Team_ID, Team_Winning)
6  VALUES (seq_tw_id.NEXTVAL, 2, 'Tournament 2023');
7
8  INSERT INTO Team_Winning (Tw_ID, Team_ID, Team_Winning)
9  VALUES (seq_tw_id.NEXTVAL, 3, 'Cup 2022');
10
11 INSERT INTO Team_Winning (Tw_ID, Team_ID, Team_Winning)
12 VALUES (seq_tw_id.NEXTVAL, 4, 'League 2022');
13
14 INSERT INTO Team_Winning (Tw_ID, Team_ID, Team_Winning)
15 VALUES (seq_tw_id.NEXTVAL, 5, 'Championship 2023');
```

Inserting data into Team_Winning tables

Results | Explain | Describe | Saved SQL | History

TW_ID	TEAM_ID	TEAM_WINNING
1	1	Championship 2022
2	2	Tournament 2023
3	3	Cup 2022
4	4	League 2022
5	5	Championship 2023

5 rows returned in 0.00 seconds

Inserted data of Team Winning table

```
1
2  INSERT INTO SocialMedia (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email,
       SocialMedia_Password, SocialMedia_Picture, SocialMedia_Hiredate, SocialMedia_Salary,
       MANAGER_ID)
3  VALUES (seq_socialmedia_id.NEXTVAL, 'SocialMediaUser221', 'social.user1@example.com', '
       socialpass1', 'social1.jpg', TO_DATE('2022-01-01', 'YYYY-MM-DD'), 5000, 1);
4
5  INSERT INTO SocialMedia (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email,
       SocialMedia_Password, SocialMedia_Picture, SocialMedia_Hiredate, SocialMedia_Salary,
       MANAGER_ID)
6  VALUES (seq_socialmedia_id.NEXTVAL, 'SocialMediaUser442', 'social.user2@example.com', '
       socialpass2', 'social2.jpg', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 6000, 2);
7
8  INSERT INTO SocialMedia (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email,
       SocialMedia_Password, SocialMedia_Picture, SocialMedia_Hiredate, SocialMedia_Salary,
       MANAGER_ID)
9  VALUES (seq_socialmedia_id.NEXTVAL, 'SocialMediaUser423', 'social.user3@example.com', '
       socialpass3', 'social3.jpg', TO_DATE('2022-03-01', 'YYYY-MM-DD'), 7000, 3);
10
11 INSERT INTO SocialMedia (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email,
       SocialMedia_Password, SocialMedia_Picture, SocialMedia_Hiredate, SocialMedia_Salary,
       MANAGER_ID)
12 VALUES (seq_socialmedia_id.NEXTVAL, 'SocialMediaUser4234', 'social.user4@example.com', '
       socialpass4', 'social4.jpg', TO_DATE('2022-04-01', 'YYYY-MM-DD'), 8000, 4);
13
14 INSERT INTO SocialMedia (SocialMedia_ID, SocialMedia_Name, SocialMedia_Email,
       SocialMedia_Password, SocialMedia_Picture, SocialMedia_Hiredate, SocialMedia_Salary,
       MANAGER_ID)
15 VALUES (seq_socialmedia_id.NEXTVAL, 'SocialMediaUser5523', 'social.user5@example.com', '
       socialpass5', 'social5.jpg', TO_DATE('2022-05-01', 'YYYY-MM-DD'), 9000, 5);
```

Inserting data into SocialMedia tables

Results | Explain | Describe | Saved SQL | History

SOCIALMEDIA_ID	SOCIALMEDIA_NAME	SOCIALMEDIA_EMAIL	SOCIALMEDIA_PASSWORD	SOCIALMEDIA_PICTURE	SOCIALMEDIA_HIREDATE	SOCIALMEDIA_SALARY	MANAGER_ID
1	SocialMediaUser221	social.user1@example.com	socialpass1	social1.jpg	01-JAN-22	5000	1
2	SocialMediaUser442	social.user2@example.com	socialpass2	social2.jpg	01-FEB-22	6000	2
3	SocialMediaUser423	social.user3@example.com	socialpass3	social3.jpg	01-MAR-22	7000	3
4	SocialMediaUser4234	social.user4@example.com	socialpass4	social4.jpg	01-APR-22	8000	4
5	SocialMediaUser5523	social.user5@example.com	socialpass5	social5.jpg	01-MAY-22	9000	5

5 rows returned in 0.00 seconds

Inserted data of Social Media table

```sql
INSERT INTO ContentCreator (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email,
    ContentCreator_Password, ContentCreator_Picture, ContentCreator_Hiredate,
    ContentCreator_Salary, SOCIALMEDIA_ID)
VALUES (seq_contentcreator_id.NEXTVAL, 'ContentCreator 1', 'cc1@example.com', 'ccpass1', '
    cc1.jpg', TO_DATE('2022-01-01', 'YYYY-MM-DD'), 3000, 1)

INSERT INTO ContentCreator (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email,
    ContentCreator_Password, ContentCreator_Picture, ContentCreator_Hiredate,
    ContentCreator_Salary, SOCIALMEDIA_ID)
VALUES (seq_contentcreator_id.NEXTVAL, 'ContentCreator 2', 'cc2@example.com', 'ccpass2', '
    cc2.jpg', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 3500, 2)

INSERT INTO ContentCreator (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email,
    ContentCreator_Password, ContentCreator_Picture, ContentCreator_Hiredate,
    ContentCreator_Salary, SOCIALMEDIA_ID)
VALUES (seq_contentcreator_id.NEXTVAL, 'ContentCreator 3', 'cc3@example.com', 'ccpass3', '
    cc3.jpg', TO_DATE('2022-03-01', 'YYYY-MM-DD'), 4000, 3)

INSERT INTO ContentCreator (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email,
    ContentCreator_Password, ContentCreator_Picture, ContentCreator_Hiredate,
    ContentCreator_Salary, SOCIALMEDIA_ID)
VALUES (seq_contentcreator_id.NEXTVAL, 'ContentCreator 4', 'cc4@example.com', 'ccpass4', '
    cc4.jpg', TO_DATE('2022-04-01', 'YYYY-MM-DD'), 4500, 4)

INSERT INTO ContentCreator (ContentCreator_ID, ContentCreator_Name, ContentCreator_Email,
    ContentCreator_Password, ContentCreator_Picture, ContentCreator_Hiredate,
    ContentCreator_Salary, SOCIALMEDIA_ID)
VALUES (seq_contentcreator_id.NEXTVAL, 'ContentCreator 5', 'cc5@example.com', 'ccpass5', '
    cc5.jpg', TO_DATE('2022-05-01', 'YYYY-MM-DD'), 5000, 5)
```

Inserting data into ContentCreator tables

Results Explain Describe Saved SQL History

CONTENTCREATOR_ID	CONTENTCREATOR_NAME	CONTENTCREATOR_EMAIL	CONTENTCREATOR_PASSWORD	CONTENTCREATOR_PICTURE	CONTENTCREATOR_HIREDATE	CONTENTCREATOR_SALARY	SOCIALMEDIA_ID
1	ContentCreator 1	cc1@example.com	ccpass1	cc1.jpg	01-JAN-22	3000	1
2	ContentCreator 2	cc2@example.com	ccpass2	cc2.jpg	01-FEB-22	3500	2
3	ContentCreator 3	cc3@example.com	ccpass3	cc3.jpg	01-MAR-22	4000	3
4	ContentCreator 4	cc4@example.com	ccpass4	cc4.jpg	01-APR-22	4500	4
5	ContentCreator 5	cc5@example.com	ccpass5	cc5.jpg	01-MAY-22	5000	5

5 rows returned in 0.00 seconds

Inserted data of Content Creator table

```sql
INSERT INTO ContentCreator_SocialMedia (Ccs_ID, ContentCreator_ID,
    ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_Instagram_Link
    , ContentCreator_Youtube_Link)
VALUES (seq_ccs_id.NEXTVAL, 1, 'https://facebook.com/cc1', 'https://twitter.com/cc1', 'https
    ://instagram.com/cc1', 'https://youtube.com/cc1');

INSERT INTO ContentCreator_SocialMedia (Ccs_ID, ContentCreator_ID,
    ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_Instagram_Link
    , ContentCreator_Youtube_Link)
VALUES (seq_ccs_id.NEXTVAL, 2, 'https://facebook.com/cc2', 'https://twitter.com/cc2', 'https
    ://instagram.com/cc2', 'https://youtube.com/cc2');

INSERT INTO ContentCreator_SocialMedia (Ccs_ID, ContentCreator_ID,
    ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_Instagram_Link
    , ContentCreator_Youtube_Link)
VALUES (seq_ccs_id.NEXTVAL, 3, 'https://facebook.com/cc3', 'https://twitter.com/cc3', 'https
    ://instagram.com/cc3', 'https://youtube.com/cc3');

INSERT INTO ContentCreator_SocialMedia (Ccs_ID, ContentCreator_ID,
    ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_Instagram_Link
    , ContentCreator_Youtube_Link)
VALUES (seq_ccs_id.NEXTVAL, 4, 'https://facebook.com/cc4', 'https://twitter.com/cc4', 'https
    ://instagram.com/cc4', 'https://youtube.com/cc4');

INSERT INTO ContentCreator_SocialMedia (Ccs_ID, ContentCreator_ID,
    ContentCreator_Facebook_Link, ContentCreator_Twitter_Link, ContentCreator_Instagram_Link
    , ContentCreator_Youtube_Link)
VALUES (seq_ccs_id.NEXTVAL, 5, 'https://facebook.com/cc5', 'https://twitter.com/cc5', 'https
    ://instagram.com/cc5', 'https://youtube.com/cc5');
```

Inserting data into ContentCreator_SocialMedia tables

Results | Explain | Describe | Saved SQL | History

CCS_ID	CONTENTCREATOR_ID	CONTENTCREATOR_FACEBOOK_LINK	CONTENTCREATOR_TWITTER_LINK	CONTENTCREATOR_INSTAGRAM_LINK	CONTENTCREATOR_YOUTUBE_LINK
1	1	https://facebook.com/cc1	https://twitter.com/cc1	https://instagram.com/cc1	https://youtube.com/cc1
2	2	https://facebook.com/cc2	https://twitter.com/cc2	https://instagram.com/cc2	https://youtube.com/cc2
3	3	https://facebook.com/cc3	https://twitter.com/cc3	https://instagram.com/cc3	https://youtube.com/cc3
4	4	https://facebook.com/cc4	https://twitter.com/cc4	https://instagram.com/cc4	https://youtube.com/cc4
5	5	https://facebook.com/cc5	https://twitter.com/cc5	https://instagram.com/cc5	https://youtube.com/cc5

5 rows returned in 0.00 seconds

Inserted data of Content Creator Social Media table

```
1  INSERT INTO ContentCreator_Address (Cca_ID, ContentCreator_ID, ContentCreator_Country,
       ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
2  VALUES (seq_cca_id.NEXTVAL, 1, 'USA', 'New York', '123 Street', '10001');
3
4  INSERT INTO ContentCreator_Address (Cca_ID, ContentCreator_ID, ContentCreator_Country,
       ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
5  VALUES (seq_cca_id.NEXTVAL, 2, 'USA', 'Los Angeles', '456 Avenue', '90001');
6
7  INSERT INTO ContentCreator_Address (Cca_ID, ContentCreator_ID, ContentCreator_Country,
       ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
8  VALUES (seq_cca_id.NEXTVAL, 3, 'UK', 'London', '789 Road', 'SW1A 1AA');
9
10 INSERT INTO ContentCreator_Address (Cca_ID, ContentCreator_ID, ContentCreator_Country,
       ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
11 VALUES (seq_cca_id.NEXTVAL, 4, 'Canada', 'Toronto', '321 Boulevard', 'M5V 2T3');
12
13 INSERT INTO ContentCreator_Address (Cca_ID, ContentCreator_ID, ContentCreator_Country,
       ContentCreator_City, ContentCreator_Street, ContentCreator_Zip_Code)
14 VALUES (seq_cca_id.NEXTVAL, 5, 'Germany', 'Berlin', '987 Strasse', '12345');
```

Inserting data into ContentCreator_Address tables

Results | Explain | Describe | Saved SQL | History

CCA_ID	CONTENTCREATOR_ID	CONTENTCREATOR_COUNTRY	CONTENTCREATOR_CITY	CONTENTCREATOR_STREET	CONTENTCREATOR_ZIP_CODE
1	1	USA	New York	123 Street	10001
2	2	USA	Los Angeles	456 Avenue	90001
3	3	UK	London	789 Road	SW1A 1AA
4	4	Canada	Toronto	321 Boulevard	M5V 2T3
5	5	Germany	Berlin	987 Strasse	12345

5 rows returned in 0.00 seconds

Inserted data of Content Creator Address table

```
1  INSERT INTO ContentCreator_Phone (Ccp_ID, ContentCreator_ID, ContentCreator_Phone)
2  VALUES (seq_ccp_id.NEXTVAL, 1, '9876543210');
3
4  INSERT INTO ContentCreator_Phone (Ccp_ID, ContentCreator_ID, ContentCreator_Phone)
5  VALUES (seq_ccp_id.NEXTVAL, 2, '1234567890');
6
7  INSERT INTO ContentCreator_Phone (Ccp_ID, ContentCreator_ID, ContentCreator_Phone)
8  VALUES (seq_ccp_id.NEXTVAL, 3, '5551234567');
9
10 INSERT INTO ContentCreator_Phone (Ccp_ID, ContentCreator_ID, ContentCreator_Phone)
11 VALUES (seq_ccp_id.NEXTVAL, 4, '7775558888');
12
13 INSERT INTO ContentCreator_Phone (Ccp_ID, ContentCreator_ID, ContentCreator_Phone)
14 VALUES (seq_ccp_id.NEXTVAL, 5, '9990001111');
```

Inserting data into ContentCreator_Phone tables

Results | Explain | Describe | Saved SQL | History

CCP_ID	CONTENTCREATOR_ID	CONTENTCREATOR_PHONE
1	1	9876543210
2	2	1234567890
3	3	5551234567
4	4	7775558888
5	5	9990001111

5 rows returned in 0.00 seconds

Inserted data of Content Creator Phone table

```
INSERT INTO SocialMedia_Phone (Smp_ID, SocialMedia_ID, SocialMedia_Phone)
VALUES (seq_smp_id.NEXTVAL, 1, '5551234567');

INSERT INTO SocialMedia_Phone (Smp_ID, SocialMedia_ID, SocialMedia_Phone)
VALUES (seq_smp_id.NEXTVAL, 2, '6669876543');

INSERT INTO SocialMedia_Phone (Smp_ID, SocialMedia_ID, SocialMedia_Phone)
VALUES (seq_smp_id.NEXTVAL, 3, '7774561230');

INSERT INTO SocialMedia_Phone (Smp_ID, SocialMedia_ID, SocialMedia_Phone)
VALUES (seq_smp_id.NEXTVAL, 4, '8887890123');

INSERT INTO SocialMedia_Phone (Smp_ID, SocialMedia_ID, SocialMedia_Phone)
VALUES (seq_smp_id.NEXTVAL, 5, '9996547890');
```

Inserting data into SocialMedia_Phone tables

Results | Explain | Describe | Saved SQL | History

SMP_ID	SOCIALMEDIA_ID	SOCIALMEDIA_PHONE
1	1	5551234567
2	2	6669876543
3	3	7774561230
4	4	8887890123
5	5	9996547890

5 rows returned in 0.00 seconds

Inserted data of Social Media Phone table

```
INSERT INTO Organization (Organization_ID, Organization_Name, Organization_Email,
    Organization_Password, Organization_Picture, Organization_Phone, Finance_ID)
VALUES (seq_organization_id.NEXTVAL, 'Organization XYZ', 'info@xyz.com', 'orgpass', 'org.jpg
    ', '1234567890', 1);

INSERT INTO Organization (Organization_ID, Organization_Name,Organization_Email,
    Organization_Password, Organization_Picture, Organization_Phone, Finance_ID)
VALUES (seq_organization_id.NEXTVAL, 'Organization ABC', 'info@abc.com', 'orgpass', 'org2.
    jpg', '0987654321', 2);

INSERT INTO Organization (Organization_ID, Organization_Name, Organization_Email,
    Organization_Password, Organization_Picture, Organization_Phone, Finance_ID)
VALUES (seq_organization_id.NEXTVAL, 'Organization DEF', 'info@def.com', 'orgpass', 'org3.
    jpg', '1112223333', 3);

INSERT INTO Organization (Organization_ID, Organization_Name, Organization_Email,
    Organization_Password, Organization_Picture, Organization_Phone, Finance_ID)
VALUES (seq_organization_id.NEXTVAL, 'Organization GHI', 'info@ghi.com', 'orgpass', 'org4.
    jpg', '4445556666', 4);

INSERT INTO Organization (Organization_ID, Organization_Name, Organization_Email,
    Organization_Password, Organization_Picture, Organization_Phone, Finance_ID)
VALUES (seq_organization_id.NEXTVAL, 'Organization JKL', 'info@jkl.com', 'orgpass', 'org5.
    jpg', '7778889999', 5);
```

Inserting data into Organization tables

Results | Explain | Describe | Saved SQL | History

ORGANIZATION_ID	ORGANIZATION_NAME	ORGANIZATION_EMAIL	ORGANIZATION_PASSWORD	ORGANIZATION_PICTURE	ORGANIZATION_PHONE	FINANCE_ID
1	Organization XYZ	info@xyz.com	orgpass	org.jpg	1234567890	1
2	Organization ABC	info@abc.com	orgpass	org2.jpg	0987654321	2
3	Organization DEF	info@def.com	orgpass	org3.jpg	1112223333	3
4	Organization GHI	info@ghi.com	orgpass	org4.jpg	4445556666	4
5	Organization JKL	info@jkl.com	orgpass	org5.jpg	7778889999	5

5 rows returned in 0.00 seconds

Inserted data of Organization table

```
1  INSERT INTO Organization_Phone (Op_ID, Organization_ID, Organization_Phone)
2  VALUES (seq_op_id.NEXTVAL, 1, '9998887777');
3
4  INSERT INTO Organization_Phone (Op_ID, Organization_ID, Organization_Phone)
5  VALUES (seq_op_id.NEXTVAL, 2, '8887776666');
6
7  INSERT INTO Organization_Phone (Op_ID, Organization_ID, Organization_Phone)
8  VALUES (seq_op_id.NEXTVAL, 3, '7776665555');
9
10 INSERT INTO Organization_Phone (Op_ID, Organization_ID, Organization_Phone)
11 VALUES (seq_op_id.NEXTVAL, 4, '6665554444');
12
13 INSERT INTO Organization_Phone (Op_ID, Organization_ID, Organization_Phone)
14 VALUES (seq_op_id.NEXTVAL, 5, '5554443333');
```

Inserting data into Organization_Phone tables

Results | Explain | Describe | Saved SQL | History

OP_ID	ORGANIZATION_ID	ORGANIZATION_PHONE
1	1	9998887777
2	2	8887776666
3	3	7776665555
4	4	6665554444
5	5	5554443333

5 rows returned in 0.00 seconds

Inserted data of Organization Phone table

```
1  INSERT INTO Player (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture,
       Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
2  VALUES (seq_player_id.NEXTVAL, 'Player 1', 'player1@example.com', 'playerpass', 'player1.jpg
       ', TO_DATE('2022-01-01', 'YYYY-MM-DD'), 5000, 100, TO_DATE('1990-01-01', 'YYYY-MM-DD'));
3
4  INSERT INTO Player (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture,
       Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
5  VALUES (seq_player_id.NEXTVAL, 'Player 2', 'player2@example.com', 'playerpass', 'player2.jpg
       ', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 6000, 200, TO_DATE('1992-05-10', 'YYYY-MM-DD'));
6
7  INSERT INTO Player (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture,
       Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
8  VALUES (seq_player_id.NEXTVAL, 'Player 3', 'player3@example.com', 'playerpass', 'player3.jpg
       ', TO_DATE('2022-03-01', 'YYYY-MM-DD'), 7000, 150, TO_DATE('1994-09-20', 'YYYY-MM-DD'));
9
10 INSERT INTO Player (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture,
       Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
11 VALUES (seq_player_id.NEXTVAL, 'Player 4', 'player4@example.com', 'playerpass', 'player4.jpg
       ', TO_DATE('2022-04-01', 'YYYY-MM-DD'), 8000, 120, TO_DATE('1996-12-05', 'YYYY-MM-DD'));
12
13 INSERT INTO Player (Player_ID, Player_Name, Player_Email, Player_Password, Player_Picture,
       Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
14 VALUES (seq_player_id.NEXTVAL, 'Player 5', 'player5@example.com', 'playerpass', 'player5.jpg
       ', TO_DATE('2022-05-01', 'YYYY-MM-DD'), 9000, 180, TO_DATE('1998-03-15', 'YYYY-MM-DD'));
```

Inserting data into Player tables

Results | Explain | Describe | Saved SQL | History

PLAYER_ID	PLAYER_NAME	PLAYER_EMAIL	PLAYER_PASSWORD	PLAYER_PICTURE	PLAYER_JOINDATE	PLAYER_SALARY	PLAYER_PLAY_HOURS	PLAYER_DOB
1	Player 1	player1@example.com	playerpass	player1.jpg	01-JAN-22	5000	100	01-JAN-90
2	Player 2	player2@example.com	playerpass	player2.jpg	01-FEB-22	6000	200	10-MAY-92
3	Player 3	player3@example.com	playerpass	player3.jpg	01-MAR-22	7000	150	20-SEP-94
4	Player 4	player4@example.com	playerpass	player4.jpg	01-APR-22	8000	120	05-DEC-96
5	Player 5	player5@example.com	playerpass	player5.jpg	01-MAY-22	9000	180	15-MAR-98

5 rows returned in 0.00 seconds

Inserted data of Player table

```sql
INSERT INTO Player_Address (Pa_ID, Player_ID, Player_Country, Player_City, Player_Street,
    Player_Zip_Code)
VALUES (seq_pa_id.NEXTVAL, 1, 'USA', 'New York', '123 Street', '10001');
INSERT INTO Player_Address (Pa_ID, Player_ID, Player_Country, Player_City, Player_Street,
    Player_Zip_Code)
VALUES (seq_pa_id.NEXTVAL, 2, 'USA', 'Los Angeles', '456 Avenue', '90001');
INSERT INTO Player_Address (Pa_ID, Player_ID, Player_Country, Player_City, Player_Street,
    Player_Zip_Code)
VALUES (seq_pa_id.NEXTVAL, 3, 'UK', 'London', '789 Road', 'SW1A 1AA');
INSERT INTO Player_Address (Pa_ID, Player_ID, Player_Country, Player_City, Player_Street,
    Player_Zip_Code)
VALUES (seq_pa_id.NEXTVAL, 4, 'Canada', 'Toronto', '321 Boulevard', 'M5V 2T3');
INSERT INTO Player_Address (Pa_ID, Player_ID, Player_Country, Player_City, Player_Street,
    Player_Zip_Code)
VALUES (seq_pa_id.NEXTVAL, 5, 'Germany', 'Berlin', '987 Strasse', '12345');
```

Inserting data into Player_Address tables

Results | Explain | Describe | Saved SQL | History

PA_ID	PLAYER_ID	PLAYER_COUNTRY	PLAYER_CITY	PLAYER_STREET	PLAYER_ZIP_CODE
1	1	USA	New York	123 Street	10001
2	2	USA	Los Angeles	456 Avenue	90001
3	3	UK	London	789 Road	SW1A 1AA
4	4	Canada	Toronto	321 Boulevard	M5V 2T3
5	5	Germany	Berlin	987 Strasse	12345

5 rows returned in 0.00 seconds

Inserted data of Player Address table

```sql
INSERT INTO PLAYER_SOCIAL_LINK (Psl_ID, Player_ID, Player_Facebook_Link,
    Player_Instagram_Link, Player_Twitter_Link, Player_Youtube_Link)
VALUES (seq_psl_id.NEXTVAL, 1, 'https://facebook.com/player1', 'https://instagram.com/
    player1', 'https://twitter.com/player1', 'https://youtube.com/player1');

INSERT INTO PLAYER_SOCIAL_LINK (Psl_ID, Player_ID, Player_Facebook_Link,
    Player_Instagram_Link, Player_Twitter_Link, Player_Youtube_Link)
VALUES (seq_psl_id.NEXTVAL, 2, 'https://facebook.com/player2', 'https://instagram.com/
    player2', 'https://twitter.com/player2', 'https://youtube.com/player2');

INSERT INTO PLAYER_SOCIAL_LINK (Psl_ID, Player_ID, Player_Facebook_Link,
    Player_Instagram_Link, Player_Twitter_Link, Player_Youtube_Link)
VALUES (seq_psl_id.NEXTVAL, 3, 'https://facebook.com/player3', 'https://instagram.com/
    player3', 'https://twitter.com/player3', 'https://youtube.com/player3');

INSERT INTO PLAYER_SOCIAL_LINK (Psl_ID, Player_ID, Player_Facebook_Link,
    Player_Instagram_Link, Player_Twitter_Link, Player_Youtube_Link)
VALUES (seq_psl_id.NEXTVAL, 4, 'https://facebook.com/player4', 'https://instagram.com/
    player4', 'https://twitter.com/player4', 'https://youtube.com/player4');

INSERT INTO PLAYER_SOCIAL_LINK (Psl_ID, Player_ID, Player_Facebook_Link,
    Player_Instagram_Link, Player_Twitter_Link, Player_Youtube_Link)
VALUES (seq_psl_id.NEXTVAL, 5, 'https://facebook.com/player5', 'https://instagram.com/
    player5', 'https://twitter.com/player5', 'https://youtube.com/player5');
```

Inserting data into Player_SocialLink tables

Results | Explain | Describe | Saved SQL | History

PSL_ID	PLAYER_ID	PLAYER_FACEBOOK_LINK	PLAYER_INSTAGRAM_LINK	PLAYER_TWITTER_LINK	PLAYER_YOUTUBE_LINK
1	1	https://facebook.com/player1	https://instagram.com/player1	https://twitter.com/player1	https://youtube.com/player1
2	2	https://facebook.com/player2	https://instagram.com/player2	https://twitter.com/player2	https://youtube.com/player2
3	3	https://facebook.com/player3	https://instagram.com/player3	https://twitter.com/player3	https://youtube.com/player3
4	4	https://facebook.com/player4	https://instagram.com/player4	https://twitter.com/player4	https://youtube.com/player4
5	5	https://facebook.com/player5	https://instagram.com/player5	https://twitter.com/player5	https://youtube.com/player5

5 rows returned in 0.00 seconds

Inserted data of Player Social Link table

```sql
INSERT INTO Player_Phone (Pp_ID, Player_ID, Player_Phone)
VALUES (seq_pp_id.NEXTVAL, 1, '1112223333');

INSERT INTO Player_Phone (Pp_ID, Player_ID, Player_Phone)
VALUES (seq_pp_id.NEXTVAL, 2, '2223334444');

INSERT INTO Player_Phone (Pp_ID, Player_ID, Player_Phone)
VALUES (seq_pp_id.NEXTVAL, 3, '3334445555');

INSERT INTO Player_Phone (Pp_ID, Player_ID, Player_Phone)
VALUES (seq_pp_id.NEXTVAL, 4, '4445556666');

INSERT INTO Player_Phone (Pp_ID, Player_ID, Player_Phone)
VALUES (seq_pp_id.NEXTVAL, 5, '5556667777');
```

Inserting data into Player_Phone tables

Results | Explain | Describe | Saved SQL | History

PP_ID	PLAYER_ID	PLAYER_PHONE
1	1	1112223333
2	2	2223334444
3	3	3334445555
4	4	4445556666
5	5	5556667777

5 rows returned in 0.00 seconds

Inserted data of Player Phone table

```sql
INSERT INTO Player_Winning (Pw_ID, Player_ID, Player_Winning)
VALUES (seq_pw_id.NEXTVAL, 1, 'Tournament 2022');

INSERT INTO Player_Winning (Pw_ID, Player_ID, Player_Winning)
VALUES (seq_pw_id.NEXTVAL, 2, 'Championship 2023');

INSERT INTO Player_Winning (Pw_ID, Player_ID, Player_Winning)
VALUES (seq_pw_id.NEXTVAL, 3, 'Cup 2022');

INSERT INTO Player_Winning (Pw_ID, Player_ID, Player_Winning)
VALUES (seq_pw_id.NEXTVAL, 4, 'Tournament 2023');

INSERT INTO Player_Winning (Pw_ID, Player_ID, Player_Winning)
VALUES (seq_pw_id.NEXTVAL, 5, 'League 2022');
```

Inserting data into Player_Winning tables

Results | Explain | Describe | Saved SQL | History

PW_ID	PLAYER_ID	PLAYER_WINNING
1	1	Tournament 2022
2	2	Championship 2023
3	3	Cup 2022
4	4	Tournament 2023
5	5	League 2022

5 rows returned in 0.00 seconds

Inserted data of Player Winning table

```
INSERT INTO Player_Team (Pt_ID, Player_ID, Team_ID)
VALUES (seq_pt_id.NEXTVAL, 1, 1);

INSERT INTO Player_Team (Pt_ID, Player_ID, Team_ID)
VALUES (seq_pt_id.NEXTVAL, 2, 2);

INSERT INTO Player_Team (Pt_ID, Player_ID, Team_ID)
VALUES (seq_pt_id.NEXTVAL, 3, 3);

INSERT INTO Player_Team (Pt_ID, Player_ID, Team_ID)
VALUES (seq_pt_id.NEXTVAL, 4, 4);

INSERT INTO Player_Team (Pt_ID, Player_ID, Team_ID)
VALUES (seq_pt_id.NEXTVAL, 5, 5);
```

Inserting data into Player_Team tables

Results | Explain | Describe | Saved SQL | History

PT_ID	PLAYER_ID	TEAM_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5

5 rows returned in 0.00 seconds

Inserted data of Player Team table

5.7 Single Row Functions

1. Retrieve the email domain for the admin with Admin_ID = 1

```
1  SELECT Admin_Email, SUBSTR(Admin_Email, INSTR(Admin_Email, '@') + 1) AS Email_Domain
2  FROM Admin
3  WHERE Admin_ID = 1;
```

Query 1

Result of Query 1

2. Get the hire date of the manager named 'John Doe' formatted in a specific way

```
1  SELECT Manager_Name, TO_CHAR(Manager_Hiredate, 'DD-Mon-YYYY') AS Hire_Date
2  FROM Manager
3  WHERE Manager_Name = 'John Doe';
```

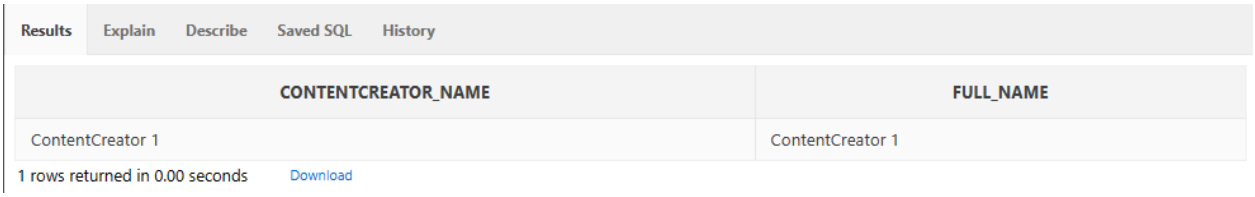
Query 2

Result of Query 2

3. Concatenate first and last name of the content creator with ContentCreator_ID = 1

```
1  SELECT ContentCreator_Name, CONCAT(CONCAT(SUBSTR(ContentCreator_Name, 1, INSTR(
       ContentCreator_Name, ' ') - 1), ' '), SUBSTR(ContentCreator_Name, INSTR(
       ContentCreator_Name, ' ') + 1)) AS Full_Name
2  FROM ContentCreator
3  WHERE ContentCreator_ID = 1;
```

Query 3



Result of Query 3

5.8 Group Functions

1. Calculate the average balance for all finance records

```
SELECT AVG(Finance_Balance) AS Average_Balance
FROM Finance;
```

Query 1

Results | Explain | Describe | Saved SQL | History

AVERAGE_BALANCE
17000

1 rows returned in 0.00 seconds Download

Result of Query 1

2. Count the number of teams established in each country

```
SELECT Team_Country, COUNT(*) AS Team_Count
FROM Team
GROUP BY Team_Country;
```

Query 2

Results | Explain | Describe | Saved SQL | History

TEAM_COUNTRY	TEAM_COUNT
USA	1
Germany	1
Australia	1
Canada	1
UK	1

5 rows returned in 0.10 seconds Download

Result of Query 2

3. Calculate the total salary expense for content creators

```
SELECT SUM(ContentCreator_Salary) AS Total_Salary_Expense
FROM ContentCreator;
```

Query 3

Results | Explain | Describe | Saved SQL | History

TOTAL_SALARY_EXPENSE
20000

1 rows returned in 0.00 seconds Download

Result of Query 3

5.9 SubQuery

1. Retrieve the managers associated with teams established before a specific date

```
    SELECT Manager_Name
    FROM Manager
    WHERE Manager_ID IN (
        SELECT Manager_ID
        FROM Team
        WHERE Team_Established_Date < TO_DATE('2022-03-01', 'YYYY-MM-DD')
);
```

Query 1

Results | Explain | Describe | Saved SQL | History

MANAGER_NAME
John Doe
Jane Smith

2 rows returned in 0.00 seconds Download

Result of Query 1

2. Get the content creators who have a higher salary than the average salary of all content creators

```
    SELECT ContentCreator_Name
    FROM ContentCreator
    WHERE ContentCreator_Salary > (
        SELECT AVG(ContentCreator_Salary)
        FROM ContentCreator
    );
```

Query 2

Results | Explain | Describe | Saved SQL | History

CONTENTCREATOR_NAME
ContentCreator 4
ContentCreator 5

2 rows returned in 0.00 seconds Download

Result of Query 2

3. Retrieve the teams managed by managers who have won a championship

```
    SELECT Team_Name
    FROM Team
    WHERE Manager_ID IN (
    SELECT Manager_ID
    FROM Manager
    WHERE Manager_ID IN (
        SELECT DISTINCT Manager_ID
        FROM Team_Winning
        WHERE Team_Winning LIKE '%Championship%'
    )
);
```

Query 3

Results | Explain | Describe | Saved SQL | History

TEAM_NAME
Team A
Team B
Team D
Team E
Team C

5 rows returned in 0.15 seconds Download

Result of Query 3

5.10 Join Queries

1. Get the team name and manager name for each team

```
SELECT t.Team_Name, m.Manager_Name
FROM Team t
JOIN Manager m ON t.Manager_ID = m.Manager_ID;
```

Query 1

Results | Explain | Describe | Saved SQL | History

TEAM_NAME	MANAGER_NAME
Team A	John Doe
Team B	Jane Smith
Team C	Mike Johnson
Team D	Sarah Williams
Team E	Robert Davis

5 rows returned in 0.00 seconds Download

Result of Query 1

2. Retrieve the player name, team name, and country for each player

```
SELECT p.Player_Name, t.Team_Name, t.Team_Country
FROM Player p
JOIN Player_Team pt ON p.Player_ID = pt.Player_ID
JOIN Team t ON pt.Team_ID = t.Team_ID;
```

Query 2

Results | Explain | Describe | Saved SQL | History

PLAYER_NAME	TEAM_NAME	TEAM_COUNTRY
Player 1	Team A	USA
Player 2	Team B	UK
Player 3	Team C	Australia
Player 4	Team D	Canada
Player 5	Team E	Germany

5 rows returned in 0.00 seconds Download

Result of Query 2

3. Get the content creator name, social media name, and email for each content creator

```
1    SELECT cc.ContentCreator_Name, sm.SocialMedia_Name, sm.SocialMedia_Email
2    FROM ContentCreator cc
3    JOIN SocialMedia sm ON cc.SOCIALMEDIA_ID = sm.SocialMedia_ID;
```

Query 3

Results | Explain | Describe | Saved SQL | History

CONTENTCREATOR_NAME	SOCIALMEDIA_NAME	SOCIALMEDIA_EMAIL
ContentCreator 1	SocialMediaUser221	social.user1@example.com
ContentCreator 2	SocialMediaUser442	social.user2@example.com
ContentCreator 3	SocialMediaUser423	social.user3@example.com
ContentCreator 4	SocialMediaUser4234	social.user4@example.com
ContentCreator 5	SocialMediaUser5523	social.user5@example.com

5 rows returned in 0.01 seconds Download

Result of Query 3

5.11 Creating View

1. Create a view to display the details of managers and their associated teams

```
CREATE VIEW ManagerTeamView AS
SELECT m.Manager_Name, t.Team_Name, t.Team_Country
FROM Manager m
JOIN Team t ON m.Manager_ID = t.Manager_ID;
```

ManagerTeamView

Results | Explain | Describe | Saved SQL | History

MANAGER_NAME	TEAM_NAME	TEAM_COUNTRY
John Doe	Team A	USA
Jane Smith	Team B	UK
Mike Johnson	Team C	Australia
Sarah Williams	Team D	Canada
Robert Davis	Team E	Germany

5 rows returned in 0.00 seconds Download

Result of ManagerTeamView

2. Create a view to show the average salary of content creators

```
CREATE VIEW AverageSalaryView AS
SELECT AVG(ContentCreator_Salary) AS Average_Salary
FROM ContentCreator;
```

AvgSalaryView

Results | Explain | Describe | Saved SQL | History

AVERAGE_SALARY
4000

1 rows returned in 0.00 seconds Download

Result of AverageSalaryView

3. Create a view to list the players and their corresponding teams

```
CREATE VIEW PlayerTeamView AS
JOIN Team t ON pt.Team_ID = t.Team_ID;
SELECT p.Player_Name, t.Team_Name
FROM Player p
JOIN Player_Team pt ON p.Player_ID = pt.Player_ID
```

PlayerTeamView

Results | Explain | Describe | Saved SQL | History

PLAYER_NAME	TEAM_NAME
Player 1	Team A
Player 2	Team B
Player 3	Team C
Player 4	Team D
Player 5	Team E

5 rows returned in 0.00 seconds Download

Result of PlayerTeamView

5.12 Synonyms

```
1 -- Create synonym for the ORGANIZATION_TOURNAMENT table
2 CREATE SYNONYM org_tour FOR Organization_Tournament;
3
4 -- Create synonym for the CONTENTCREATOR_SOCIALMEDIA table
5 CREATE SYNONYM cc_sm FOR ContentCreator_SocialMedia;
6
7 -- Create synonym for the CONTENTCREATOR_PHONE table
8 CREATE SYNONYM cc_ph FOR ContentCreator_Phone;
```

Synonyms

Results | Explain | Describe | Saved SQL | History

OWNER	SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
ESPORTFTW	CC_PH	ESPORTFTW	CONTENTCREATOR_PHONE	-
ESPORTFTW	CC_SM	ESPORTFTW	CONTENTCREATOR_SOCIALMEDIA	-
ESPORTFTW	ORG_TOUR	ESPORTFTW	ORGANIZATION_TOURNAMENT	-

3 rows returned in 1.38 seconds Download

List of Synonyms

5.13 PL/SQL

5.13.1 Functions

---

1. Calculate Age using a Function

```
    CREATE OR REPLACE FUNCTION calculate_age(birth_date DATE) RETURN NUMBER IS
    age NUMBER;
BEGIN
    age := TRUNC(MONTHS_BETWEEN(SYSDATE, birth_date) / 12);
    RETURN age;
END;
/
```

Query 1

2. Get Team Member Count using a Function

```
    CREATE OR REPLACE FUNCTION get_team_member_count(team_id INT) RETURN INT IS
    member_count INT;
BEGIN
    SELECT COUNT(*) INTO member_count
    FROM Player_Team
    WHERE Team_ID = team_id;
    RETURN member_count;
END;
/
```

Query 2

3. Get Total Earnings using a Function

```
    CREATE OR REPLACE FUNCTION get_total_earnings(player_id INT) RETURN DECIMAL IS
    total_earnings DECIMAL(10, 2);
BEGIN
    SELECT SUM(Player_Winning) INTO total_earnings
    FROM Player_Winning
    WHERE Player_ID = player_id;
    RETURN total_earnings;
END;
/
```

Query 3

5.13.2 Procedure

1. Create a procedure to insert data into the Player table with address details

```
CREATE OR REPLACE PROCEDURE Insert_Player_Data (
    p_Ign VARCHAR2,
    p_Name VARCHAR2,
    p_Email VARCHAR2,
    p_Password VARCHAR2,
    p_Picture VARCHAR2,
    p_JoinDate DATE,
    p_Salary DECIMAL,
    p_Play_Hours INT,
    p_DOB DATE,
    p_Country VARCHAR2,
    p_City VARCHAR2,
    p_Street VARCHAR2,
    p_Zip_Code VARCHAR2
)
IS
    v_Player_ID INT;
BEGIN
    -- Insert data into Player table
    INSERT INTO Player (Player_ID, Player_Ign, Player_Name, Player_Email, Player_Password,
        Player_Picture, Player_JoinDate, Player_Salary, Player_Play_Hours, Player_DOB)
    VALUES (SEQ_PLAYER_ID.nextval, p_Ign, p_Name, p_Email, p_Password, p_Picture, p_JoinDate
        , p_Salary, p_Play_Hours, p_DOB);

    -- Get the last inserted Player_ID
    SELECT MAX(Player_ID) INTO v_Player_ID FROM Player;

    -- Insert data into Player_Address table
    INSERT INTO Player_Address (Pa_ID, Player_ID, Player_Country, Player_City, Player_Street
        , Player_Zip_Code)
    VALUES (SEQ_CCA_ID.nextval, v_Player_ID, p_Country, p_City, p_Street, p_Zip_Code);

    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Data inserted successfully.');
EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('Error inserting data: ' || SQLERRM);
END;
/
```

Query 1

2. Create a procedure to update data in the Player and Player_Address tables

```
    -- To update data in Player and Player_Address tables
CREATE OR REPLACE PROCEDURE Update_Player_Data (
    p_Player_ID INT,
    p_Name VARCHAR2,
    p_Play_Hours INT,
    p_DOB DATE,
    p_City VARCHAR2,
    p_Street VARCHAR2,
    p_Zip_Code VARCHAR2
)
IS
BEGIN
    -- Update data in Player table
    UPDATE Player
    SET Player_Name = p_Name,
        Player_Play_Hours = p_Play_Hours,
        Player_DOB = p_DOB
    WHERE Player_ID = p_Player_ID;

    -- Update data in Player_Address table
    UPDATE Player_Address
    SET Player_City = p_City,
        Player_Street = p_Street,
        Player_Zip_Code = p_Zip_Code
    WHERE Player_ID = p_Player_ID;

    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Data updated successfully.');
EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('Error updating data: ' || SQLERRM);
END;
/
```

Query 2

3. Update Team Name using Procedure

```
    CREATE OR REPLACE PROCEDURE update_team_name(
    p_team_id INT,
    p_new_name VARCHAR2
) IS
BEGIN
    UPDATE Team
    SET Team_Name = p_new_name
    WHERE Team_ID = p_team_id;
    COMMIT;
END;
/
```

Query 3

5.13.3 Record

1. Create a Player Record Type and show the details of a player.

```
    DECLARE
    TYPE player_rec IS RECORD (
        player_id INT,
        player_name VARCHAR2(100),
        player_email VARCHAR2(100)
    );
    p_info player_rec;
BEGIN
    SELECT Player_ID, Player_Name, Player_Email
    INTO p_info
    FROM Player
    WHERE Player_ID = 1;
    DBMS_OUTPUT.PUT_LINE('Player ID: ' || p_info.player_id);
    DBMS_OUTPUT.PUT_LINE('Player Name: ' || p_info.player_name);
    DBMS_OUTPUT.PUT_LINE('Player Email: ' || p_info.player_email);
END;
/
```

Query 1

2. Create a Team Record Type and show the details of a team.

```
    DECLARE
    TYPE team_rec IS RECORD (
        team_id INT,
        team_name VARCHAR2(100),
        team_country VARCHAR2(100)
    );
    t_info team_rec;
BEGIN
    SELECT Team_ID, Team_Name, Team_Country
    INTO t_info
    FROM Team
    WHERE Team_ID = 1;
    DBMS_OUTPUT.PUT_LINE('Team ID: ' || t_info.team_id);
    DBMS_OUTPUT.PUT_LINE('Team Name: ' || t_info.team_name);
    DBMS_OUTPUT.PUT_LINE('Team Country: ' || t_info.team_country);
END;
/
```

Query 2

3. Create a Tournament Record Type and show the details of a tournament.

```
    DECLARE
    TYPE tournament_rec IS RECORD (
        tournament_id INT,
        tournament_name VARCHAR2(100),
        prize_pool DECIMAL(10, 2)
    );
    t_info tournament_rec;
BEGIN
    SELECT Tournament_ID, Tournament_Name, Tournament_Prize_Pool
    INTO t_info
    FROM Tournament
    WHERE Tournament_ID = 1;
    DBMS_OUTPUT.PUT_LINE('Tournament ID: ' || t_info.tournament_id);
    DBMS_OUTPUT.PUT_LINE('Tournament Name: ' || t_info.tournament_name);
    DBMS_OUTPUT.PUT_LINE('Prize Pool: ' || t_info.prize_pool);
END;
/
```

Query 3

5.13.4 Cursors

1. Create a Players Cursor and show the details of all the players.

```
    DECLARE
    CURSOR player_cursor IS
    SELECT Player_ID, Player_Name, Player_Email
    FROM Player;
    p_info player_cursor%ROWTYPE;
BEGIN
    OPEN player_cursor;
    LOOP
        FETCH player_cursor INTO p_info;
        EXIT WHEN player_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Player ID: ' || p_info.Player_ID);
        DBMS_OUTPUT.PUT_LINE('Player Name: ' || p_info.Player_Name);
        DBMS_OUTPUT.PUT_LINE('Player Email: ' || p_info.Player_Email);
    END LOOP;
    CLOSE player_cursor;
END;
/
```

Query 1

2. Create a Tournament Teams Cursor and show the details of all the teams in a tournament.

```
    DECLARE
    CURSOR team_cursor (p_tournament_id INT) IS
    SELECT Team_ID, Team_Name, Team_Country
    FROM Team
    WHERE Team_ID IN (SELECT Team_ID FROM Player_Team WHERE Tournament_ID = p_tournament_id)
        ;
    t_info team_cursor%ROWTYPE;
BEGIN
    FOR t_info IN team_cursor(1) -- Provide a tournament ID
    LOOP
        DBMS_OUTPUT.PUT_LINE('Team ID: ' || t_info.Team_ID);
        DBMS_OUTPUT.PUT_LINE('Team Name: ' || t_info.Team_Name);
        DBMS_OUTPUT.PUT_LINE('Team Country: ' || t_info.Team_Country);
    END LOOP;
END;
/
```

Query 2

3. Create an Organization Finance Cursor and show the details of all the organizations and their finances.

```
    DECLARE
    CURSOR finance_cursor IS
    SELECT o.Organization_Name, f.Finance_Balance
    FROM Organization o
    JOIN Finance f ON o.Finance_ID = f.Finance_ID;
    fin_info finance_cursor%ROWTYPE;
BEGIN
    OPEN finance_cursor;
    LOOP
        FETCH finance_cursor INTO fin_info;
        EXIT WHEN finance_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Organization Name: ' || fin_info.Organization_Name);
        DBMS_OUTPUT.PUT_LINE('Finance Balance: ' || fin_info.Finance_Balance);
    END LOOP;
    CLOSE finance_cursor;
END;
/
```

Query 3

5.13.5 Triggers

1. Create a trigger to update the salary of a player when the player's play hours are updated.

```
    CREATE OR REPLACE TRIGGER player_salary_update
    BEFORE UPDATE ON Player
    FOR EACH ROW
    BEGIN
        IF :NEW.Player_Salary < :OLD.Player_Salary THEN
            RAISE_APPLICATION_ERROR(-20001, 'Salary decrease not allowed.');
        END IF;
    END;
    /
```

Query 1

2. Create a trigger to update the prize pool of a tournament when the tournament's start date is updated.

```
    CREATE OR REPLACE TRIGGER tournament_prize_pool_trigger
BEFORE INSERT ON Tournament
FOR EACH ROW
BEGIN
    IF :NEW.Tournament_Prize_Pool <= 0 THEN
        :NEW.Tournament_Prize_Pool := NULL;
    END IF;
END;
/
```

Query 2

3. Create a trigger to update the team member limit of a team when the team's country is updated.

```
    CREATE OR REPLACE TRIGGER team_member_limit_trigger
BEFORE INSERT ON Player_Team
FOR EACH ROW
DECLARE
    team_size INT;
BEGIN
    SELECT COUNT(*) INTO team_size
    FROM Player_Team
    WHERE Team_ID = :NEW.Team_ID;

    IF team_size >= 7 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Team member limit reached.');
    END IF;
END;
/
```

Query 3

5.13.6 Package

1. Create a package to manage players. The package should contain the following procedures:

```
    CREATE OR REPLACE PACKAGE player_management_pkg AS
    PROCEDURE add_new_player(
        p_name VARCHAR2,
        p_email VARCHAR2,
        p_dob DATE,
        p_salary DECIMAL
    );

    PROCEDURE update_player_salary(
        p_player_id INT,
        p_new_salary DECIMAL
    );

    FUNCTION get_player_age(
        p_player_id INT
    ) RETURN INT;

    FUNCTION get_player_total_earnings(
        p_player_id INT
    ) RETURN DECIMAL;

    PROCEDURE remove_player(
        p_player_id INT
    );
END player_management_pkg;
/
```

Query 1

2. Create a package to manage teams. The package should contain the following procedures:

```
    CREATE OR REPLACE PACKAGE team_management_pkg AS
    PROCEDURE update_team_name(
        p_team_id INT,
        p_new_name VARCHAR2
    );

    PROCEDURE add_player_to_team(
        p_player_id INT,
        p_team_id INT
    );

    FUNCTION get_team_member_count(
        p_team_id INT
    ) RETURN INT;

    PROCEDURE remove_player_from_team(
        p_player_id INT,
        p_team_id INT
    );

    FUNCTION get_team_players(
        p_team_id INT
    ) RETURN SYS_REFCURSOR;
END team_management_pkg;
/
```

Query 2

3. Create a package to manage tournaments. The package should contain the following procedures:

```
    CREATE OR REPLACE PACKAGE tournament_management_pkg AS
    PROCEDURE create_new_tournament(
        p_name VARCHAR2,
        p_start_date DATE,
        p_end_date DATE,
        p_location VARCHAR2,
        p_prize_pool DECIMAL
    );

    PROCEDURE delete_tournament(
        p_tournament_id INT
    );

    FUNCTION get_tournament_prize_pool(
        p_tournament_id INT
    ) RETURN DECIMAL;

    FUNCTION get_tournament_winner(
        p_tournament_id INT
    ) RETURN VARCHAR2;

    PROCEDURE update_tournament_location(
        p_tournament_id INT,
        p_new_location VARCHAR2
    );
END tournament_management_pkg;
/
```

Query 3

Relational Algebra

1. Find the name of the manager whose manager id is 5.

$$\Pi_{\text{Manager\_Name}}(\sigma_{\text{Manager\_ID=5(Manager)}})$$

2. Find the Salary of 'Player 1'.

$$\Pi_{\text{Player\_Salary}}(\sigma_{\text{name="Player 1"(Player)}})$$

3. Find Player id whose birthday is on 1992-05-10.

$$\Pi_{\text{Player\_ID}}(\sigma_{\text{Player\_DOB="1992-05-10"(Player)}})$$

4. Find the Country, City, Street, and Zip code where Content Creator ID is equal to 4.

$$\Pi_{\text{ContentCreator\_Country, ContentCreator\_City,ContentCreator\_Street, ContentCreator\_Zip\_Code}}$$

$$(\sigma_{\text{ContentCreator\_ID=4(ContentCreator\_Address)}})$$

5. Find the Team ID that won the Championship 2023.

$$\Pi_{\text{Team\_ID}}(\sigma_{\text{Team\_Winning ="Championship 2023"(Team\_Winning)}})$$

Conclusion

In conclusion, the project for the development and implementation of an Esports Management System has outlined a revolutionary platform that aims to transform the management and organization of esports teams, players, tournaments, and sponsors. The proposed system seeks to improve user experience, encourage community engagement, and streamline operations within the esports industry by leveraging advanced technology and comprehensive functionalities.

For the final term, we plans to enhance the existing project by developing a web application using Dotnet and Sveltekit frameworks. This transition to a web app will provide greater accessibility and flexibility to users, allowing them to access the Esports Management System from any device with an internet connection.