Faculty of Information Technology

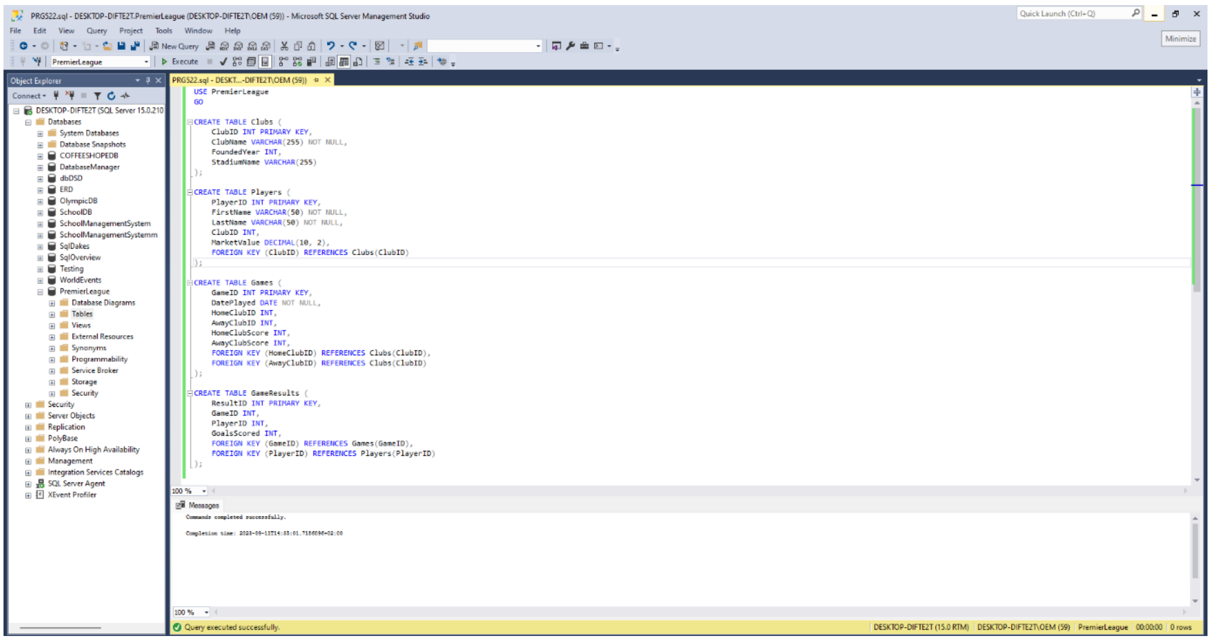
I declare that I am familiar with, and will abide to the Examination rules of CTU [illegible] **Signature**	**SUBJECT NAME: Business Programming** **SUBJECT CODE:** PRG522								
	Formative Assessment 3 **Duration**: 27 Day **Date**: 18-08-2023 **Total Marks**: 70 **Total pages**: -				**Examiner**: Mr Junior Manganyi **Moderator:**				
	Student number								
	2	0	2	3	1	4	0	5	
	Surname: Moshoeshoe				**Initials**: Leduma Tlotliso			/	%

Question 1:

Task 1: Database Design and Normalization

```sql
USE PremierLeague
GO

CREATE TABLE Clubs (
    ClubID INT PRIMARY KEY,
    ClubName VARCHAR(255) NOT NULL,
    FoundedYear INT,
    StadiumName VARCHAR(255)
);

CREATE TABLE Players (
    PlayerID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    ClubID INT,
    MarketValue DECIMAL(10, 2),
    FOREIGN KEY (ClubID) REFERENCES Clubs(ClubID)
);

CREATE TABLE Games (
    GameID INT PRIMARY KEY,
    DatePlayed DATE NOT NULL,
    HomeClubID INT,
    AwayClubID INT,
    HomeClubScore INT,
    AwayClubScore INT,
    FOREIGN KEY (HomeClubID) REFERENCES Clubs(ClubID),
    FOREIGN KEY (AwayClubID) REFERENCES Clubs(ClubID)
);

CREATE TABLE GameResults (
    ResultID INT PRIMARY KEY,
    GameID INT,
    PlayerID INT,
    GoalsScored INT,
    FOREIGN KEY (GameID) REFERENCES Games(GameID),
    FOREIGN KEY (PlayerID) REFERENCES Players(PlayerID)
);
```

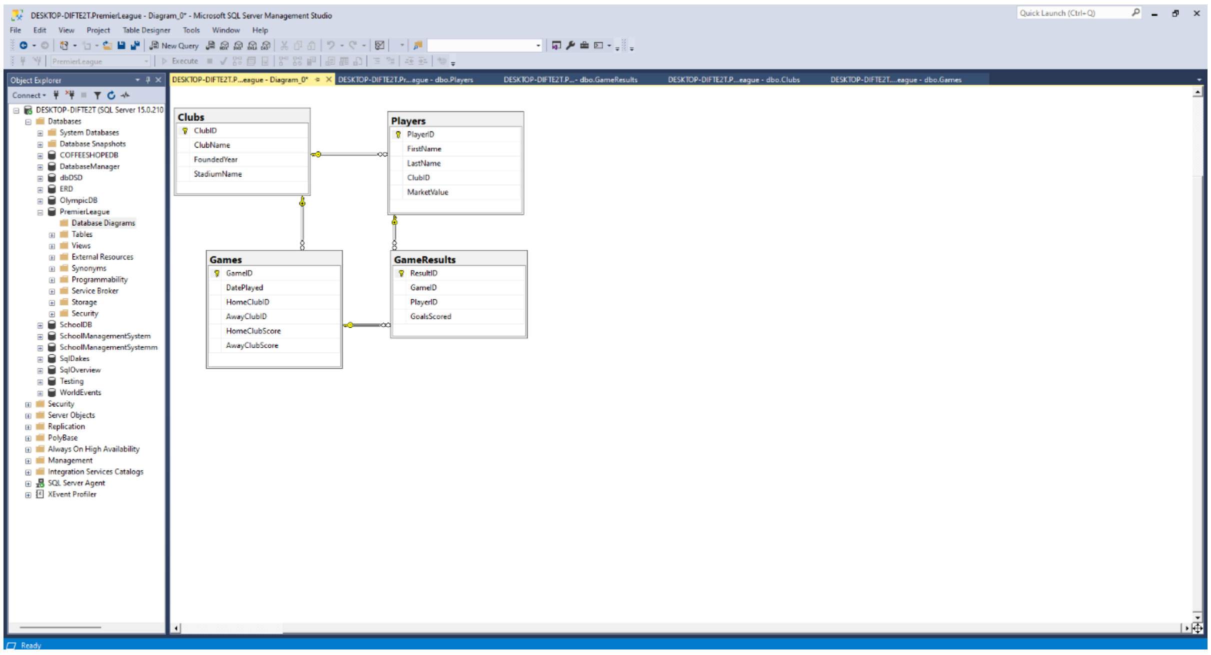
Output:

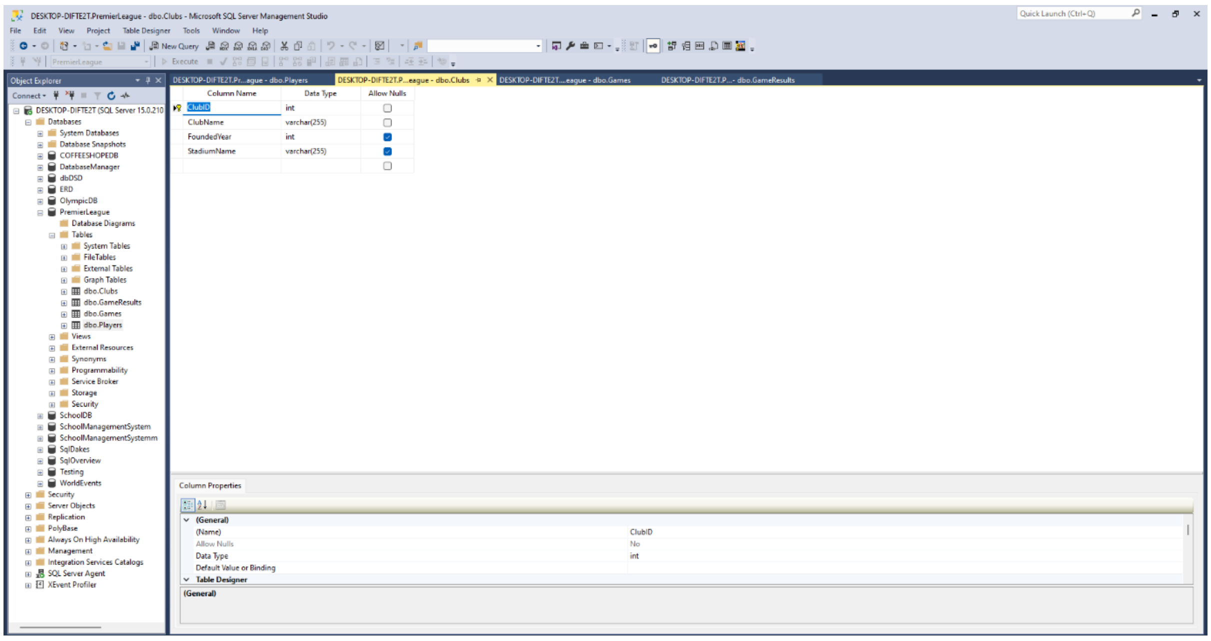


```sql
USE PremierLeague
GO

CREATE TABLE Clubs (
    ClubID INT PRIMARY KEY,
    ClubName VARCHAR(255) NOT NULL,
    FoundedYear INT,
    StadiumName VARCHAR(255)
);

CREATE TABLE Players (
    PlayerID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    ClubID INT,
    MarketValue DECIMAL(10, 2),
    FOREIGN KEY (ClubID) REFERENCES Clubs(ClubID)
);

CREATE TABLE Games (
    GameID INT PRIMARY KEY,
    DatePlayed DATE NOT NULL,
    HomeClubID INT,
    AwayClubID INT,
    HomeClubScore INT,
    AwayClubScore INT,
    FOREIGN KEY (HomeClubID) REFERENCES Clubs(ClubID),
    FOREIGN KEY (AwayClubID) REFERENCES Clubs(ClubID)
);

CREATE TABLE GameResults (
    ResultID INT PRIMARY KEY,
    GameID INT,
    PlayerID INT,
    GoalsScored INT,
    FOREIGN KEY (GameID) REFERENCES Games(GameID),
    FOREIGN KEY (PlayerID) REFERENCES Players(PlayerID)
);
```

```
Commands completed successfully.

Completion time: 2023-09-11T14:55:01.7186096+02:00
```

Query executed successfully.



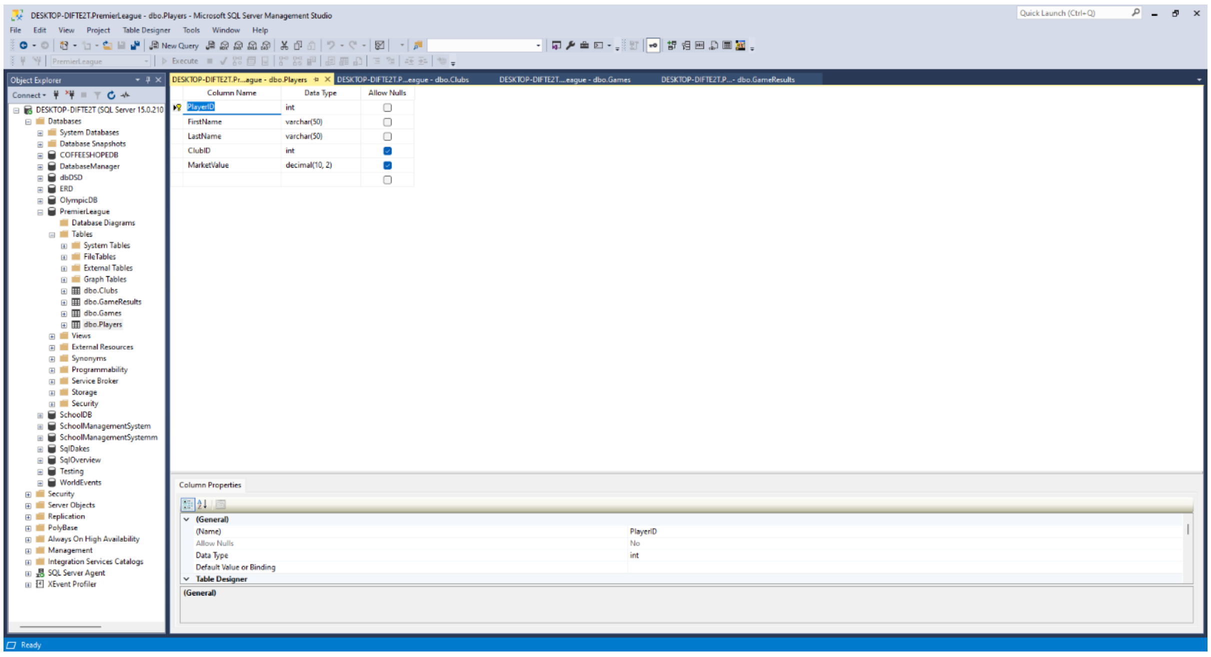

DESKTOP-DIFTE2T.PremierLeague - dbo.Clubs - Microsoft SQL Server Management Studio

	Column Name	Data Type	Allow Nulls
	ClubID	int	☐
	ClubName	varchar(255)	☐
	FoundedYear	int	✓
	StadiumName	varchar(255)	✓
			☐

Column Properties

(General)	
(Name)	ClubID
Allow Nulls	No
Data Type	int
Default Value or Binding	
Table Designer	

(General)

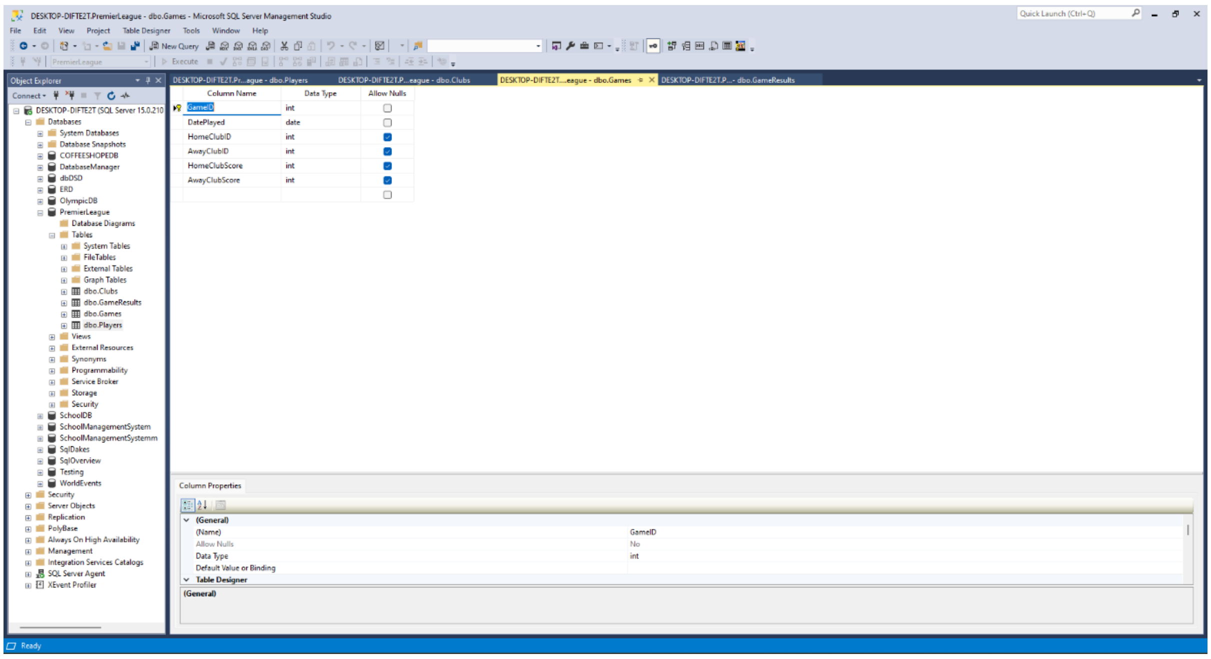

DESKTOP-DIFTE2T.PremierLeague - dbo.Players - Microsoft SQL Server Management Studio

	Column Name	Data Type	Allow Nulls
	PlayerID	int	☐
	FirstName	varchar(50)	☐
	LastName	varchar(50)	☐
	ClubID	int	✓
	MarketValue	decimal(10, 2)	✓
			☐

Column Properties

(General)	
(Name)	PlayerID
Allow Nulls	No
Data Type	int
Default Value or Binding	
Table Designer	

(General)



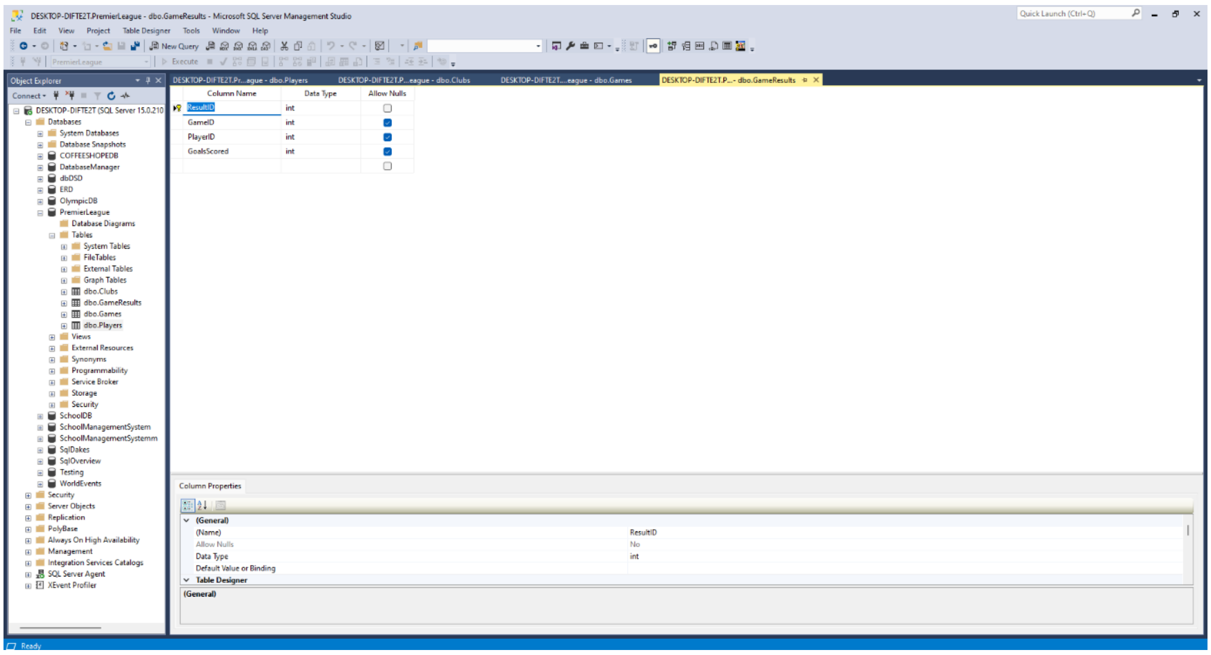
DESKTOP-DIFTE2T.PremierLeague - dbo.Games - Microsoft SQL Server Management Studio

Column Name	Data Type	Allow Nulls
GameID	int	☐
DatePlayed	date	☐
HomeClubID	int	✓
AwayClubID	int	✓
HomeClubScore	int	✓
AwayClubScore	int	✓
		☐

Column Properties

(General)	
(Name)	GameID
Allow Nulls	No
Data Type	int
Default Value or Binding	
Table Designer	

(General)



DESKTOP-DIFTE2T.PremierLeague - dbo.GameResults - Microsoft SQL Server Management Studio

Column Name	Data Type	Allow Nulls
ResultID	int	☐
GameID	int	✓
PlayerID	int	✓
GoalsScored	int	✓
		☐

Column Properties

(General)	
(Name)	ResultID
Allow Nulls	No
Data Type	int
Default Value or Binding	
Table Designer	

(General)

Task 2: Database Creation and Data Population

Clubs_data.sql

```
USE [PremierLeague]
GO

INSERT INTO Clubs (ClubID, ClubName, FoundedYear, StadiumName)
VALUES
    (1, 'Manchester United', 1878, 'Old Trafford'),
    (2, 'Liverpool FC', 1892, 'Anfield'),
    (3, 'Chelsea FC', 1905, 'Stamford Bridge');

GO
```

GamesResults_data.sql

```
USE [PremierLeague]
GO

INSERT INTO GameResults (ResultID, GameID, PlayerID, GoalsScored)
VALUES
    (1, 1, 1, 1),
    (2, 1, 2, 1),
    (3, 2, 3, 1);

GO
```

Games_data.sql

```
USE [PremierLeague]
GO

-- Insert the first row
INSERT INTO [dbo].[Games]
           ([GameID]
           ,[DatePlayed]
           ,[HomeClubID]
           ,[AwayClubID]
           ,[HomeClubScore]
           ,[AwayClubScore])
     VALUES
           (1,
           '2023-09-10',
           1,
           2,
           2,
           1);

-- Insert the second row
INSERT INTO [dbo].[Games]
           ([GameID]
           ,[DatePlayed]
           ,[HomeClubID]
           ,[AwayClubID]
           ,[HomeClubScore]
           ,[AwayClubScore])
     VALUES
```

```
           (2,
           '2023-09-11',
           3,
           4,
           0,
           1);
```

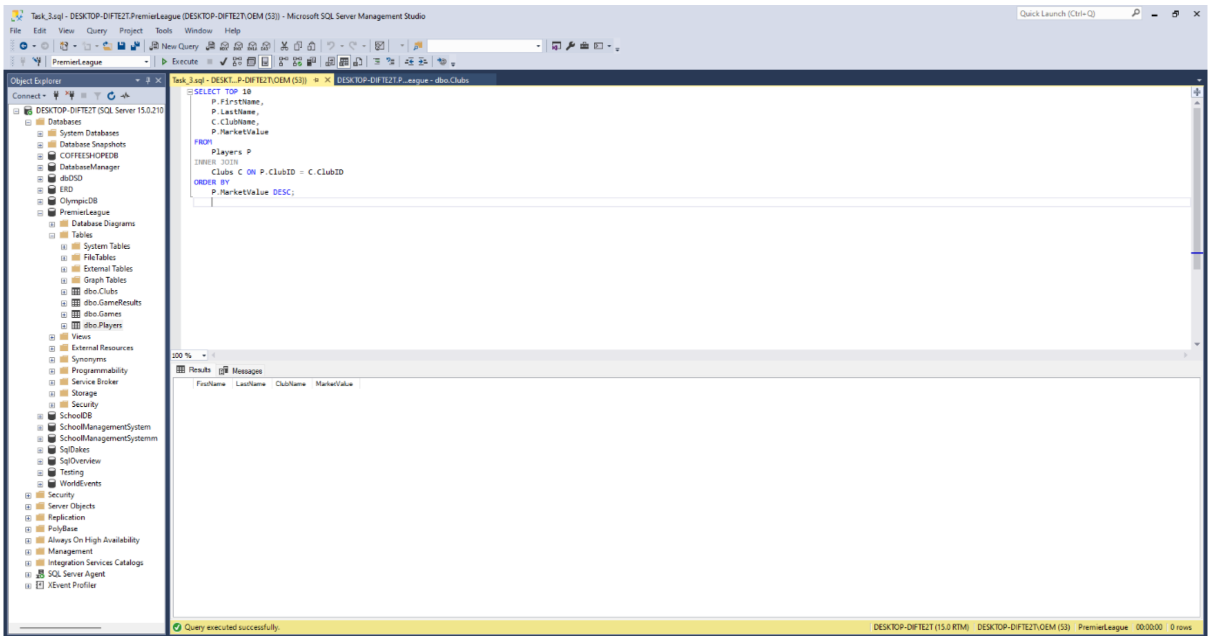
Players_data.sql

```
USE [PremierLeague]
GO

INSERT INTO [dbo].[Players]
           ([PlayerID]
           ,[FirstName]
           ,[LastName]
           ,[ClubID]
           ,[MarketValue])
     VALUES
           (1
           ,'Cristiano'
           ,'Ronaldo'
           ,1
           ,100000000.00),
           (2
           ,'Mohamed'
           ,'Salah'
           ,2
           ,90000000.00),
           (3
           ,'N''Golo'
           ,'Kanté'
           ,3
           ,80000000.00)
GO
```

Task 3: SQL Queries

```sql
SELECT TOP 10
    P.FirstName,
    P.LastName,
    C.ClubName,
    P.MarketValue
FROM
    Players P
INNER JOIN
    Clubs C ON P.ClubID = C.ClubID
ORDER BY
    P.MarketValue DESC;
```

Output:



Task_3.sql - DESKTOP-DIFTE2T.PremierLeague (DESKTOP-DIFTE2T\OEM (53)) - Microsoft SQL Server Management Studio

```sql
SELECT TOP 10
    P.FirstName,
    P.LastName,
    C.ClubName,
    P.MarketValue
FROM
    Players P
INNER JOIN
    Clubs C ON P.ClubID = C.ClubID
ORDER BY
    P.MarketValue DESC;
```

```
   FirstName   LastName   ClubName   MarketValue
```

Query executed successfully.   DESKTOP-DIFTE2T (15.0 RTM) | DESKTOP-DIFTE2T\OEM (53) | PremierLeague | 00:00:00 | 0 rows

Task 4: Database Security and Access Control

i. Use of Database Roles and Privileges to Secure a Database System:

- **Database Roles:** Roles are named groups of related privileges. They simplify the process of granting and managing permissions. Instead of assigning permissions to individual users, you assign them to roles, and then you assign roles to users. Roles can be used to represent different job functions or responsibilities within an organization.

For example, in a football league database:

  - You might create a **football_admin** role for administrators who need to manage player data.
  - You could have a **report_viewer** role for users who need read-only access to game results.

- **Privileges:** Privileges are specific actions that can be performed on database objects, such as SELECT, INSERT, UPDATE, and DELETE. Privileges are assigned to roles or directly to users. They control what actions users or roles can perform on tables, views, and other database objects.

For example, in the context of this database:

  - A user assigned the **football_admin** role might have INSERT and UPDATE privileges on the Players table to manage player information.
  - A user assigned the **report_viewer** role might have SELECT privilege on the GameResults view to view game results..

ii. Available Grant Options and Controlling Access:

SSMS provides various grant options that allow the database administrator to control access effectively. Here's a discussion along with examples:

GRANT SELECT, INSERT, UPDATE, DELETE: These are common privileges used to control access to specific actions on tables.

WITH GRANT OPTION: This option allows a user or role to grant the same privilege to others. It's useful for delegating permission management.

REVOKE: If you need to remove a previously granted privilege, you can use the REVOKE statement.

Grant options and revoking privileges give administrators fine-grained control over who can access and manipulate data in the database.

iii. Role of Views in Controlling Database Access:

Views are a valuable tool for controlling database access and simplifying complex queries:

- **Data Abstraction:** Views can hide sensitive data by exposing only specific columns or rows from underlying tables. For example, you can create a view that exposes player names without revealing their salary information.
- **Simplifying Queries:** Views can encapsulate complex queries, making it easier for users to retrieve data without needing to write complex SQL statements themselves.
- **Access Control:** Views can restrict access to specific data. You can grant users access to views instead of underlying tables, limiting what they can see.

Completed Declaration of Authenticity

I L e d u m a   T l o t l i s o   M o s h o e s h o e hereby

(FULL NAME)

declare that the contents of this assignment P R G 5 2 2 are entirely my work except for the following documents: (List the documents and page numbers of work in this portfolio that were generated in a group)

Activity	Date

Signature: [illegible] Date: 1 2 - 0 9 - 2 0 2 3