INFO – 5707 Data Modeling for Information Professionals (Fall 2024)

Professor - Dr. Lingzi Hong

Project Group – 8

Term Project Final Report

Team Members:

Sri Saaketh Ram Haridasyam	**(11757375)**	**[*Team Coordinator*]**
Nanditha Aitha	**(11759521)**	
Avinash Keerthi	**(11715798)**	
Vinay Daram	**(11641568)**	
Surya Teja Garneni	**(11758506)**	

Revised Objectives

The primary objective of the Pro Kabaddi League Database is to develop a robust, scalable, and comprehensive system that centralizes the management and analysis of all kabaddi-related data. This database aims to streamline league operations, enhance real-time decision-making, and provide actionable insights for all stakeholders, including team managers, coaches, fans, and sponsors. It facilitates efficient tracking and management of player details, team rosters, match scheduling, and venue logistics while ensuring up-to-date information on match outcomes and player performance.

The revised objective emphasizes enhancing fan engagement through personalized interactions based on preferences and behaviors, improving sponsorship management by tracking contracts and financial contributions, and supporting detailed reporting and analytics for strategic planning. By integrating advanced data visualization features, the system enables stakeholders to derive insights such as trend analysis and performance optimization.

Adhering to stringent data integrity, security, and compliance standards, the database ensures the accuracy and reliability of stored information. Designed for seamless integration with external platforms, it provides scalability to accommodate the evolving needs of the league. This comprehensive approach positions the database as a critical tool for enhancing operational efficiency, stakeholder satisfaction, and the long-term success of the Pro Kabaddi League.

Revised Scope of the Project:

The main idea of the Pro Kabaddi League database is to hold, administer, and reveal commercial secrets; it is an organization for the administration of the parties, players, games, venues, fans, sponsors, and seasons. The database of the project counts the following areas:

Team Information Management: Gather and keep the details about each team, for example, team names, period of their establishment, home grounds, and coaching staff.

Player Data and Performance Tracking: Save all the possible pieces of information of each player such as the name, date of birth, position, nationality, height, and weight, also include game statistics such as points scored, and tackles made.

Match Scheduling and Results: Get all the instances of matches of the past. Get the date of the matches, and the stadiums where they were played, as well as the team's positions, points, and final scores. Besides, make a system that tracks the status of the match.

Venue Details and Event Management: From the entire event, get the data about venues, for instance, the names, locations, if they are outdoor venues, if they are also for sports, type of venue used for organizing and managing events.

Fan Engagement and Personalization: The recording of the name, email, and support team of each fan is the only way to personalize the whole engagement through strategies and promotions.

Sponsorship Tracking and Financial Oversight: Manage the entire sponsorship process through maintaining all agreements including sponsor names, sponsorship amounts, and contract durations as well as ensuring that relationships with sponsors are not only effective but also managed properly.

Seasonal Data Archiving and Analysis: The organization can keep the data for the particular season exchanged between two parties, including season year, start and end dates, and total matches, to use historical analysis and performance tracking.

Revised User Requirement:

Comprehensive Data Management: The database should be able to comprehensively store, organize, and manage a multitude of types of information pertaining to teams, players, matches, venues, fans, sponsors, statistics, and other related information in a structured format. To avoid data redundancy and duplication, relationships must be preserved using primary and foreign key constraints for entities to ensure consistency and integrity.

Real-Time Updates: The system should enable real-time updates for match results, player statistics, and ticket sales. This also ensures stakeholders have access to the most up-to-date information for informed decision-making and operational efficiency. Real-time synchronization with other platforms, including mobile applications and websites, is essential for keeping fans updated.

Improved Reporting and Analytics: Users need to be able to create detailed, customized reports regarding player performance, analytics, sponsorship effectiveness, fan engagement, and financial metrics. And advanced analytics tools will deliver insights like seasonal trends, player development patterns, and revenue growth driven by ticket sales and sponsorships.

Enhanced Fan Engagement: The system should personalize interaction with fans using stored preferences, ticket purchase history, and team affiliations. This could be through targeted promotions, reminders for upcoming matches, and engagements based on fan behavior and preference.

Scalable and Secure Architecture: The database should accommodate increasing additions of more teams, additional venues, new seasons, and enhanced fan engagement features without compromising existing data or performance. Sensitive data should be protected by improved security protocols such as encryption, user authentication, and role-based access controls.

Robust Data Backup and Retrieval: Data backup engines should always securely store important information like matches, sponsorship contracts, and ticket sales. The backup data must be encrypted and should be retrievable in case of system failure to ensure minimum data loss and downtime.

Historical Data Management: Long term trend analysis and predictive insights require the archiving of historical records from matches and players which will also help retain sponsorships. The system must also allow retrieval of old data to help aid strategic planning and decision-making.

Improved Search and Filter Features: The system should allow users to easily search for and filter players, games, tickets and sponsorships and retrieve relevant data. You should be able to customize queries for particular scenarios like looking up by date, team (or) financial details.

Compliance and Auditing: The database will be compliant with data protection regulations, ensuring that fan and sponsor data are stored and accessed in accordance with privacy laws. The malicious and unauthorized changes for such sensitive data must be tracked with an audit trail offering enough accountability and backing periodic compliance checks.

Seamless Integration: The system should provide seamless integration with external applications including fan engagement applications, mobile applications, and league websites. Integration provides real-time data sharing, allowing fans to obtain live match information, player statistics, and Ticket availability while supporting league stakeholders with synchronized financial and operational data.

These updated requirements take into consideration a more mature understanding of the capabilities of the database as well as the aims of the project as it transitions from the design into the implementation phase.

Business Rules

1. Each player is assigned to only one team per season.

2. One player must be in one designated team.

3. Each team must have a captain, vice-captain.

4. The Coach should have at least 5 years of experience.

5. To assign a player to a team, the player must go through or pass the minimum requirement of height and weight.

6. One team can have many players.

7. A team can have multiple sponsors.

8. A sponsor can sponsor many teams.

9. A sponsor can sponsor many matches.

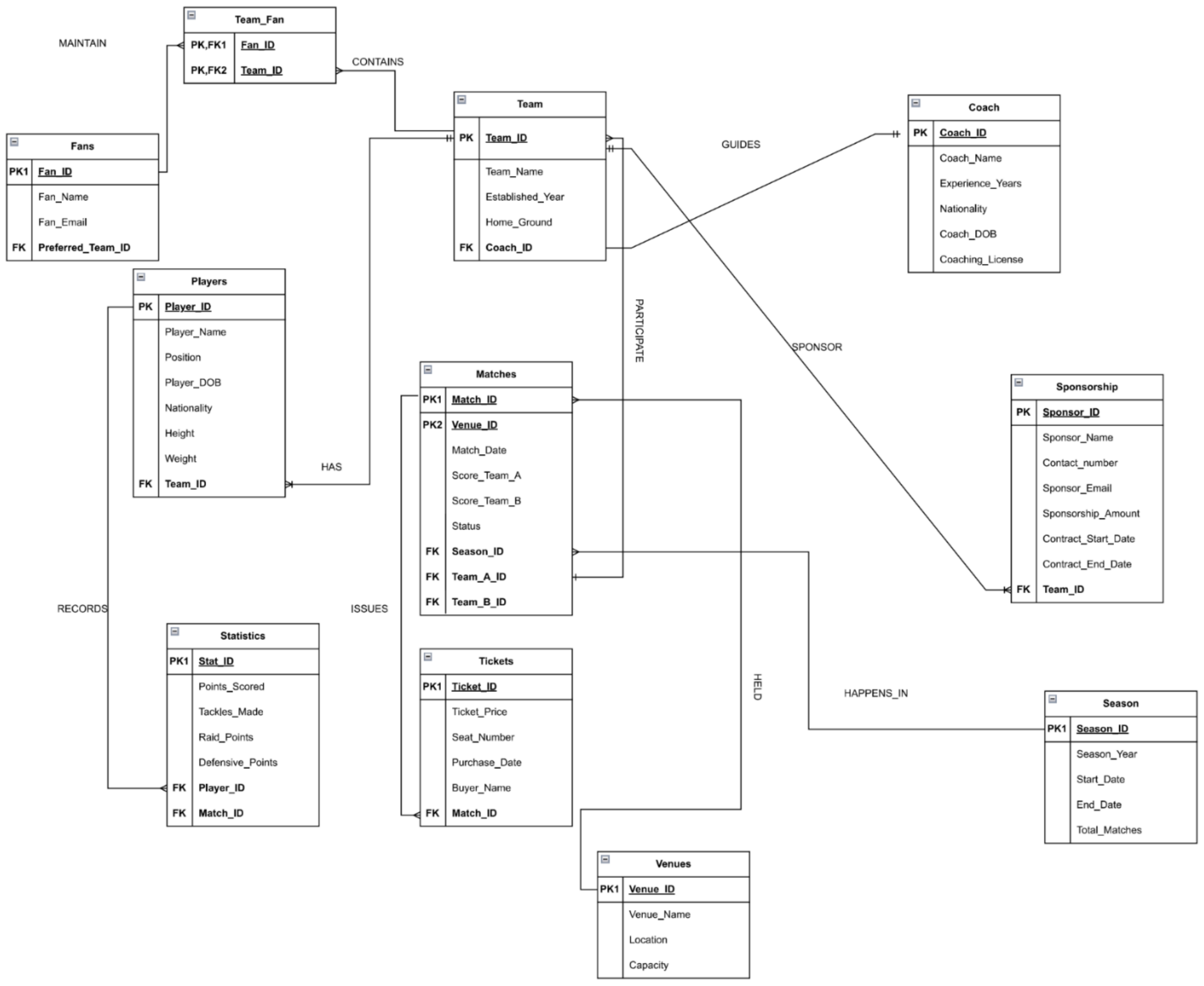
10. A match can have multiple sponsors

11. One match can be held in one venue.

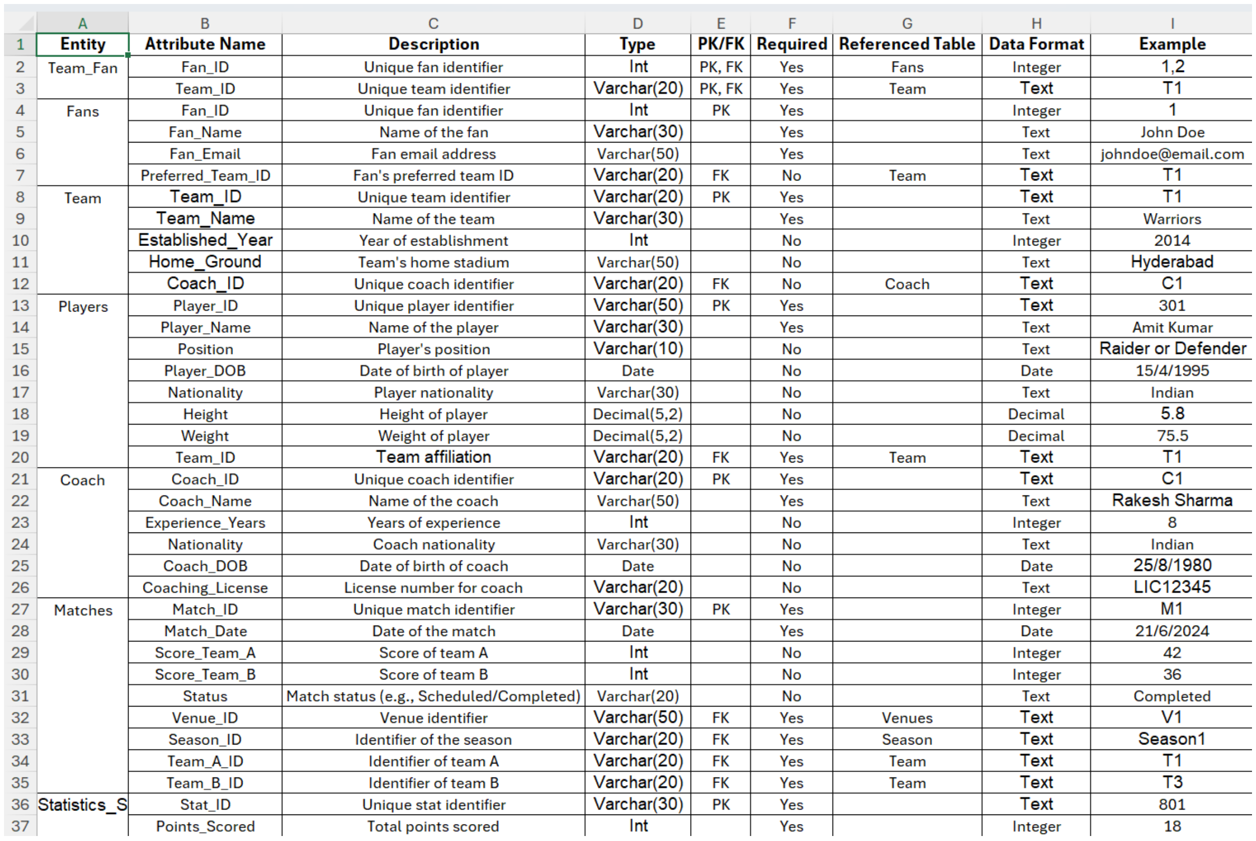
12. One team is allowed to play only once a day.

13. One Team can play many matches over the season league.

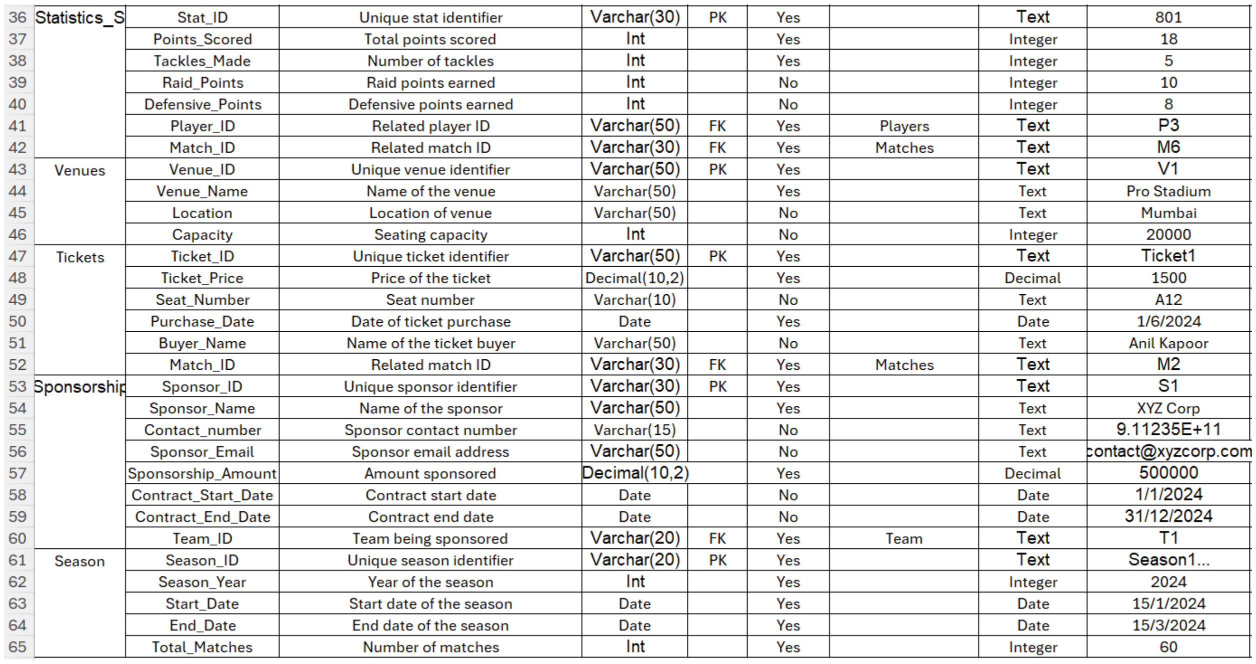
Entity Relationship Diagram



Revised Data Dictionary



	A	B	C	D	E	F	G	H	I
1	Entity	Attribute Name	Description	Type	PK/FK	Required	Referenced Table	Data Format	Example
2	Team_Fan	Fan_ID	Unique fan identifier	Int	PK, FK	Yes	Fans	Integer	1,2
3		Team_ID	Unique team identifier	Varchar(20)	PK, FK	Yes	Team	Text	T1
4	Fans	Fan_ID	Unique fan identifier	Int	PK	Yes		Integer	1
5		Fan_Name	Name of the fan	Varchar(30)		Yes		Text	John Doe
6		Fan_Email	Fan email address	Varchar(50)		Yes		Text	johndoe@email.com
7		Preferred_Team_ID	Fan's preferred team ID	Varchar(20)	FK	No	Team	Text	T1
8	Team	Team_ID	Unique team identifier	Varchar(20)	PK	Yes		Text	T1
9		Team_Name	Name of the team	Varchar(30)		Yes		Text	Warriors
10		Established_Year	Year of establishment	Int		No		Integer	2014
11		Home_Ground	Team's home stadium	Varchar(50)		No		Text	Hyderabad
12		Coach_ID	Unique coach identifier	Varchar(20)	FK	No	Coach	Text	C1
13	Players	Player_ID	Unique player identifier	Varchar(50)	PK	Yes		Text	301
14		Player_Name	Name of the player	Varchar(30)		Yes		Text	Amit Kumar
15		Position	Player's position	Varchar(10)		No		Text	Raider or Defender
16		Player_DOB	Date of birth of player	Date		No		Date	15/4/1995
17		Nationality	Player nationality	Varchar(30)		No		Text	Indian
18		Height	Height of player	Decimal(5,2)		No		Decimal	5.8
19		Weight	Weight of player	Decimal(5,2)		No		Decimal	75.5
20		Team_ID	Team affiliation	Varchar(20)	FK	Yes	Team	Text	T1
21	Coach	Coach_ID	Unique coach identifier	Varchar(20)	PK	Yes		Text	C1
22		Coach_Name	Name of the coach	Varchar(50)		Yes		Text	Rakesh Sharma
23		Experience_Years	Years of experience	Int		No		Integer	8
24		Nationality	Coach nationality	Varchar(30)		No		Text	Indian
25		Coach_DOB	Date of birth of coach	Date		No		Date	25/8/1980
26		Coaching_License	License number for coach	Varchar(20)		No		Text	LIC12345
27	Matches	Match_ID	Unique match identifier	Varchar(30)	PK	Yes		Integer	M1
28		Match_Date	Date of the match	Date		Yes		Date	21/6/2024
29		Score_Team_A	Score of team A	Int		No		Integer	42
30		Score_Team_B	Score of team B	Int		No		Integer	36
31		Status	Match status (e.g., Scheduled/Completed)	Varchar(20)		No		Text	Completed
32		Venue_ID	Venue identifier	Varchar(50)	FK	Yes	Venues	Text	V1
33		Season_ID	Identifier of the season	Varchar(20)	FK	Yes	Season	Text	Season1
34		Team_A_ID	Identifier of team A	Varchar(20)	FK	Yes	Team	Text	T1
35		Team_B_ID	Identifier of team B	Varchar(20)	FK	Yes	Team	Text	T3
36	Statistics_S	Stat_ID	Unique stat identifier	Varchar(30)	PK	Yes		Text	801
37		Points_Scored	Total points scored	Int		Yes		Integer	18



36	Statistics_S	Stat_ID	Unique stat identifier	Varchar(30)	PK	Yes		Text	801
37		Points_Scored	Total points scored	Int		Yes		Integer	18
38		Tackles_Made	Number of tackles	Int		Yes		Integer	5
39		Raid_Points	Raid points earned	Int		No		Integer	10
40		Defensive_Points	Defensive points earned	Int		No		Integer	8
41		Player_ID	Related player ID	Varchar(50)	FK	Yes	Players	Text	P3
42		Match_ID	Related match ID	Varchar(30)	FK	Yes	Matches	Text	M6
43	Venues	Venue_ID	Unique venue identifier	Varchar(50)	PK	Yes		Text	V1
44		Venue_Name	Name of the venue	Varchar(50)		Yes		Text	Pro Stadium
45		Location	Location of venue	Varchar(50)		No		Text	Mumbai
46		Capacity	Seating capacity	Int		No		Integer	20000
47	Tickets	Ticket_ID	Unique ticket identifier	Varchar(50)	PK	Yes		Text	Ticket1
48		Ticket_Price	Price of the ticket	Decimal(10,2)		Yes		Decimal	1500
49		Seat_Number	Seat number	Varchar(10)		No		Text	A12
50		Purchase_Date	Date of ticket purchase	Date		Yes		Date	1/6/2024
51		Buyer_Name	Name of the ticket buyer	Varchar(50)		No		Text	Anil Kapoor
52		Match_ID	Related match ID	Varchar(30)	FK	Yes	Matches	Text	M2
53	Sponsorship	Sponsor_ID	Unique sponsor identifier	Varchar(30)	PK	Yes		Text	S1
54		Sponsor_Name	Name of the sponsor	Varchar(50)		Yes		Text	XYZ Corp
55		Contact_number	Sponsor contact number	Varchar(15)		No		Text	9.11235E+11
56		Sponsor_Email	Sponsor email address	Varchar(50)		No		Text	ontact@xyzcorp.com
57		Sponsorship_Amount	Amount sponsored	Decimal(10,2)		Yes		Decimal	500000
58		Contract_Start_Date	Contract start date	Date		No		Date	1/1/2024
59		Contract_End_Date	Contract end date	Date		No		Date	31/12/2024
60		Team_ID	Team being sponsored	Varchar(20)	FK	Yes	Team	Text	T1
61	Season	Season_ID	Unique season identifier	Varchar(20)	PK	Yes		Text	Season1...
62		Season_Year	Year of the season	Int		Yes		Integer	2024
63		Start_Date	Start date of the season	Date		Yes		Date	15/1/2024
64		End_Date	End date of the season	Date		Yes		Date	15/3/2024
65		Total_Matches	Number of matches	Int		Yes		Integer	60

Software Environment and Database Overview

The database was built in Microsoft SQL Server with 11 entities such as teams, players, matches and sponsors among others. The relationships were defined utilizing primary and foreign keys in order to maintain data integrity, with realistic example data included for meaningful analysis. It was designed and tested using SQL Server Management Studio (SSMS), with sufficient operations and reporting support. The data from each table is shown in an image format after insertion which is shown in the query statements section below.

Database consists of 11 tables which are as follows:

Team Table - Team Table contains team details such as Team_ID, Team_Name, Established_Year, Home_Ground, Coach_ID

Fan Table: Fan_ID, Fan_Name, Fan_Email, Preferred_Team_ID

Player Table: This table contains the details for each player like Player_ID, Player_Name, Position, Player_DOB, Nationality, Height and Weight and Team_ID.

Matches Table: This will hold all match data such as Match_ID, Match_Date, Score_Team_A, Score_Team_B, Status, Venue_ID, Season_ID, Team_A_ID, Team_B_ID.

Venues table: This table records venue details along with Venue_ID, Venue_Name, Location, and Capacity.

Sponsorship Table: This contains the following attributes: Sponsor_ID, Sponsor_Name, Contact_Number, Sponsor_Email, Sponsorship_Amount, Contract_Start_Date, Contract_End_Date & Team_ID

Season Table: This table contains all the relevant information specific to a season like Season_ID, Season_Year, Start_Date, End_Date, Total_Matches.

Tickets Table: For handling ticket sales, comprising Ticket_ID, Ticket_Price, Seat_Number, Purchase_Date, Buyer_Name, and Match_ID attributes.

Statistics_S: This table holds individual statistics of the players for a match like Stat_ID, Points_Scored, Tackles_Made, Raid_Points, Defensive_Points, Player_ID, Match_ID

Coach table: This table would be used to store details of the coaches such as Coach_ID, Coach_Name, Experience_Years, Nationality, Coach_DOB, Coaching_License.

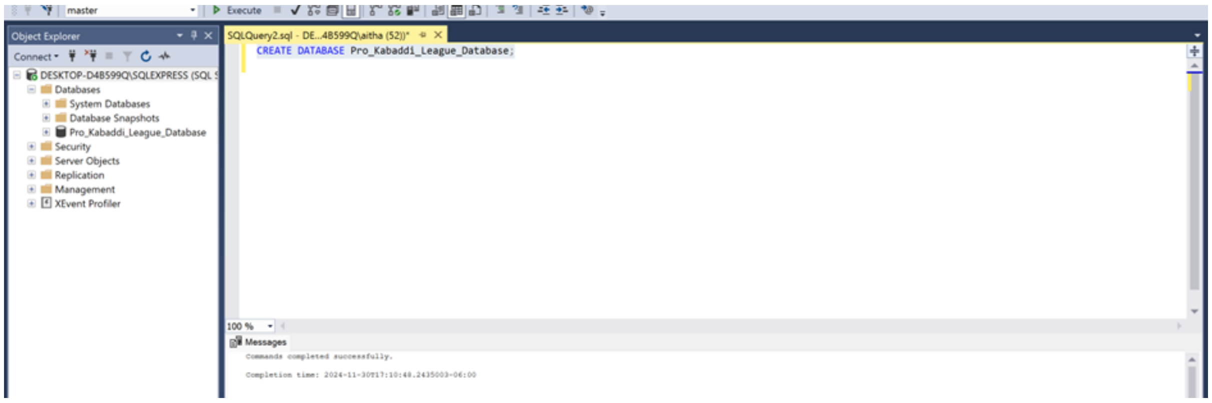
Team_Fan Table: This table connects fans with teams and has attributes Fan_ID and Team_ID.

Query Statements:

Creation of Database:

This query is intended to set up a database for a management system for the Pro Kabaddi League. For managing transactional and analytical data, the configurations provide scalable storage and suitable performance settings.

CREATE DATABASE Pro_Kabaddi_League_Database;



```
CREATE DATABASE Pro_Kabaddi_League_Database;
```

```
Commands completed successfully.

Completion time: 2024-11-30T17:10:48.2435003-06:00
```

Creation of Team Table:

This query aims to provide the TEAM table with preliminary information on Pro Kabaddi League teams. A team is represented by each row, which also contains the corresponding coach's unique identity (Coach_ID), home ground (Home_Ground), year of establishment (Established_Year), name (Team_Name), and unique identifier (Team_ID). In addition to facilitating further database functions like querying team details, connecting with other tables (such coaches or matches), and overseeing league administration, this data is necessary for keeping an organized record of every team in the league.

Query:

USE Pro_Kabaddi_League_Database

CREATE TABLE TEAM(Team_ID Varchar(20) primary key,

Team_Name Varchar(30),

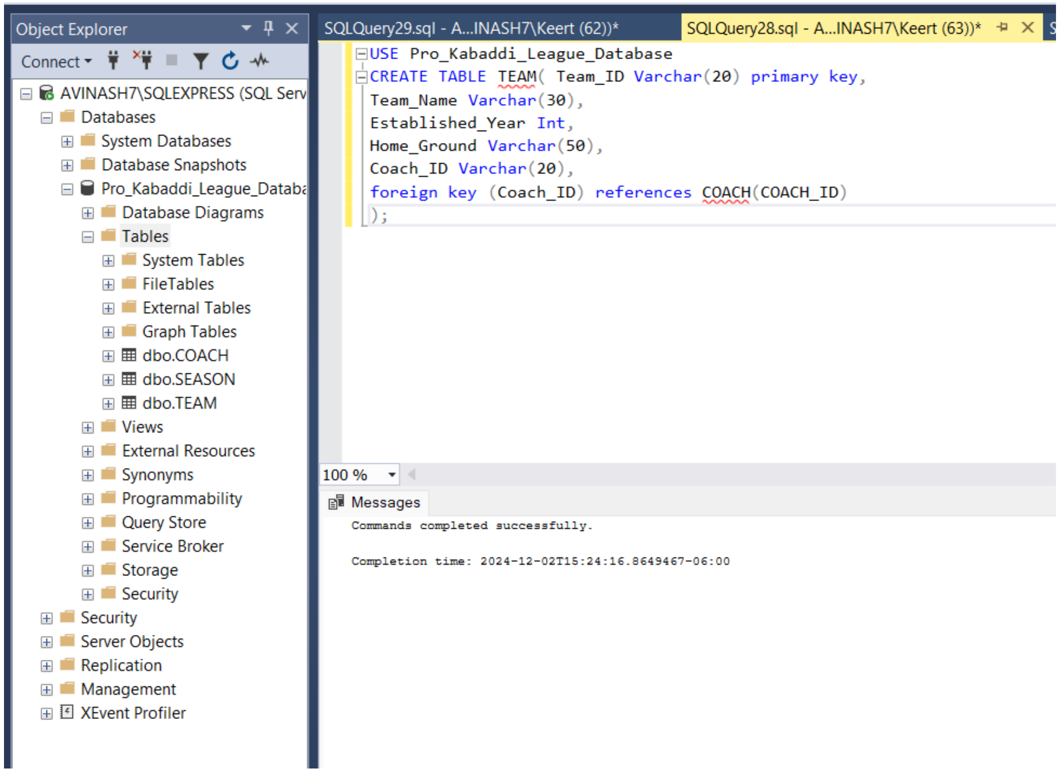
Established_Year Int,

Home_Ground Varchar(50),

Coach_ID Varchar(20),

foreign key (Coach_ID) references COACH(COACH_ID)

);



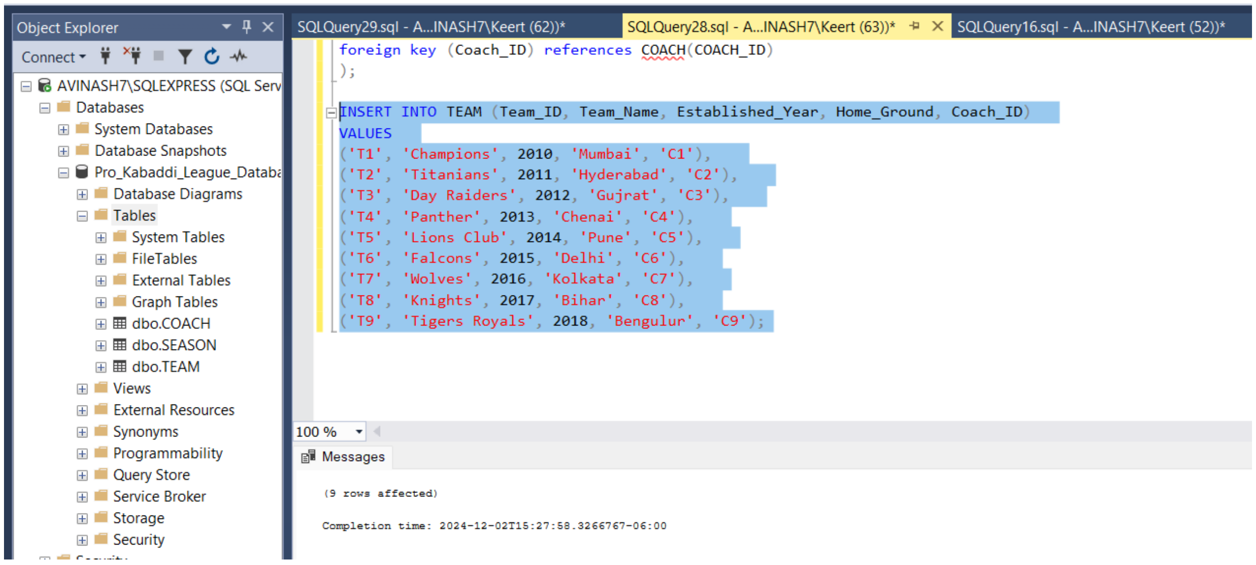
```
USE Pro_Kabaddi_League_Database
CREATE TABLE TEAM( Team_ID Varchar(20) primary key,
Team_Name Varchar(30),
Established_Year Int,
Home_Ground Varchar(50),
Coach_ID Varchar(20),
foreign key (Coach_ID) references COACH(COACH_ID)
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T15:24:16.8649467-06:00
```

Insertion of data into Team Table:

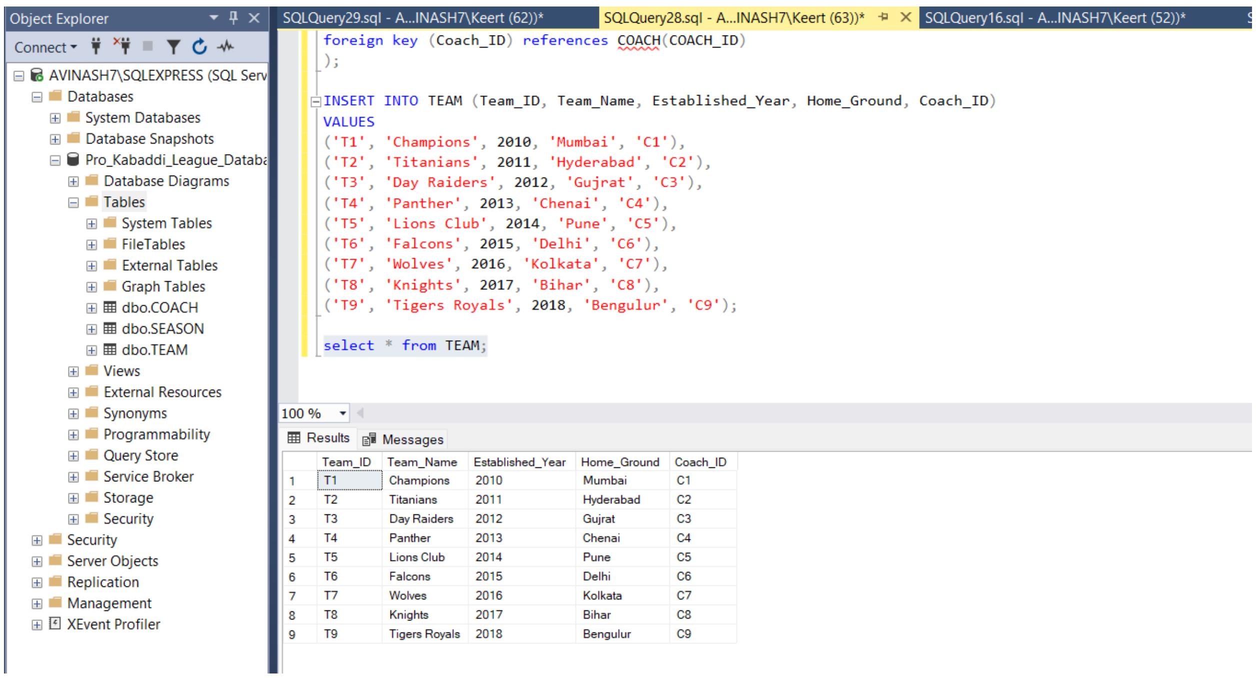
This query's objective is to add entries to the TEAM database, giving it details about the Pro Kabaddi League teams that are playing. A distinct team ID, team name, year of formation, home ground location, and related coach ID are all included in each record. This information forms the basis of the database's management and reference team-related data.

```
INSERT INTO TEAM (Team_ID, Team_Name, Established_Year, Home_Ground, Coach_ID)

VALUES

('T1', 'Champions', 2010, 'Mumbai', 'C1'),

('T2', 'Titanians', 2011, 'Hyderabad', 'C2'),

('T3', 'Day Raiders', 2012, 'Gujrat', 'C3'),

('T4', 'Panther', 2013, 'Chenai', 'C4'),

('T5', 'Lions Club', 2014, 'Pune', 'C5'),

('T6', 'Falcons', 2015, 'Delhi', 'C6'),

('T7', 'Wolves', 2016, 'Kolkata', 'C7'),

('T8', 'Knights', 2017, 'Bihar', 'C8'),

('T9', 'Tigers Royals', 2018, 'Bengulur', 'C9');
```



```
foreign key (Coach_ID) references COACH(COACH_ID)
);

INSERT INTO TEAM (Team_ID, Team_Name, Established_Year, Home_Ground, Coach_ID)
VALUES
('T1', 'Champions', 2010, 'Mumbai', 'C1'),
('T2', 'Titanians', 2011, 'Hyderabad', 'C2'),
('T3', 'Day Raiders', 2012, 'Gujrat', 'C3'),
('T4', 'Panther', 2013, 'Chenai', 'C4'),
('T5', 'Lions Club', 2014, 'Pune', 'C5'),
('T6', 'Falcons', 2015, 'Delhi', 'C6'),
('T7', 'Wolves', 2016, 'Kolkata', 'C7'),
('T8', 'Knights', 2017, 'Bihar', 'C8'),
('T9', 'Tigers Royals', 2018, 'Bengulur', 'C9');
```

```
(9 rows affected)

Completion time: 2024-12-02T15:27:58.3266767-06:00
```



```
foreign key (Coach_ID) references COACH(COACH_ID)
);

INSERT INTO TEAM (Team_ID, Team_Name, Established_Year, Home_Ground, Coach_ID)
VALUES
('T1', 'Champions', 2010, 'Mumbai', 'C1'),
('T2', 'Titanians', 2011, 'Hyderabad', 'C2'),
('T3', 'Day Raiders', 2012, 'Gujrat', 'C3'),
('T4', 'Panther', 2013, 'Chenai', 'C4'),
('T5', 'Lions Club', 2014, 'Pune', 'C5'),
('T6', 'Falcons', 2015, 'Delhi', 'C6'),
('T7', 'Wolves', 2016, 'Kolkata', 'C7'),
('T8', 'Knights', 2017, 'Bihar', 'C8'),
('T9', 'Tigers Royals', 2018, 'Bengulur', 'C9');

select * from TEAM;
```

	Team_ID	Team_Name	Established_Year	Home_Ground	Coach_ID
1	T1	Champions	2010	Mumbai	C1
2	T2	Titanians	2011	Hyderabad	C2
3	T3	Day Raiders	2012	Gujrat	C3
4	T4	Panther	2013	Chenai	C4
5	T5	Lions Club	2014	Pune	C5
6	T6	Falcons	2015	Delhi	C6
7	T7	Wolves	2016	Kolkata	C7
8	T8	Knights	2017	Bihar	C8
9	T9	Tigers Royals	2018	Bengulur	C9

Creation of table Fans:

This query's goal is to create a table in the Pro_Kabaddi_League_Database called FANS. This table is intended to hold data regarding Pro Kabaddi League supporters, such as:

Fan_ID: A primary key that is specific to each fan.

Fan_Name: The fan's name.

Fan_Email: The fan's email address.

Preferred_Team_ID: The fan's favorite team's ID.

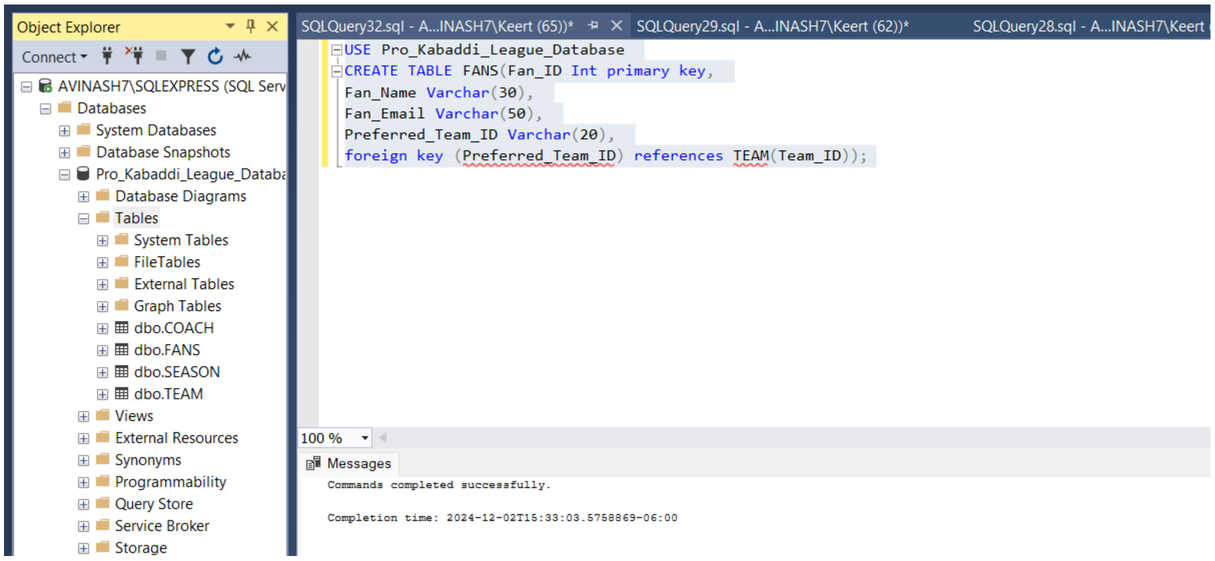
Using a foreign key constraint, the Preferred_Team_ID column creates a connection to the TEAM database. Each desired team ID must match an existing Team_ID in the TEAM database in order to maintain referential integrity.

Analytics, fan interaction, and marketing tactics may all benefit from the recording of fan data and team preferences made possible by this table.

Query:

```
USE Pro_Kabaddi_League_Database

CREATE TABLE FANS(Fan_ID Int primary key,

Fan_Name Varchar(30),

Fan_Email Varchar(50),

Preferred_Team_ID Varchar(20),
```

foreign key (Preferred_Team_ID) references TEAM(Team_ID));



```
USE Pro_Kabaddi_League_Database
CREATE TABLE FANS(Fan_ID Int primary key,
Fan_Name Varchar(30),
Fan_Email Varchar(50),
Preferred_Team_ID Varchar(20),
foreign key (Preferred_Team_ID) references TEAM(Team_ID));
```

```
Commands completed successfully.

Completion time: 2024-12-02T15:33:03.5758869-06:00
```

Insertion data into Fan Table:

This query's goal is to add several records to the FANS database, giving it information about specific Pro Kabaddi League supporters. Every record contains:

Fan_ID: A special number assigned to every fan.

Fan_Name: The fan's name.

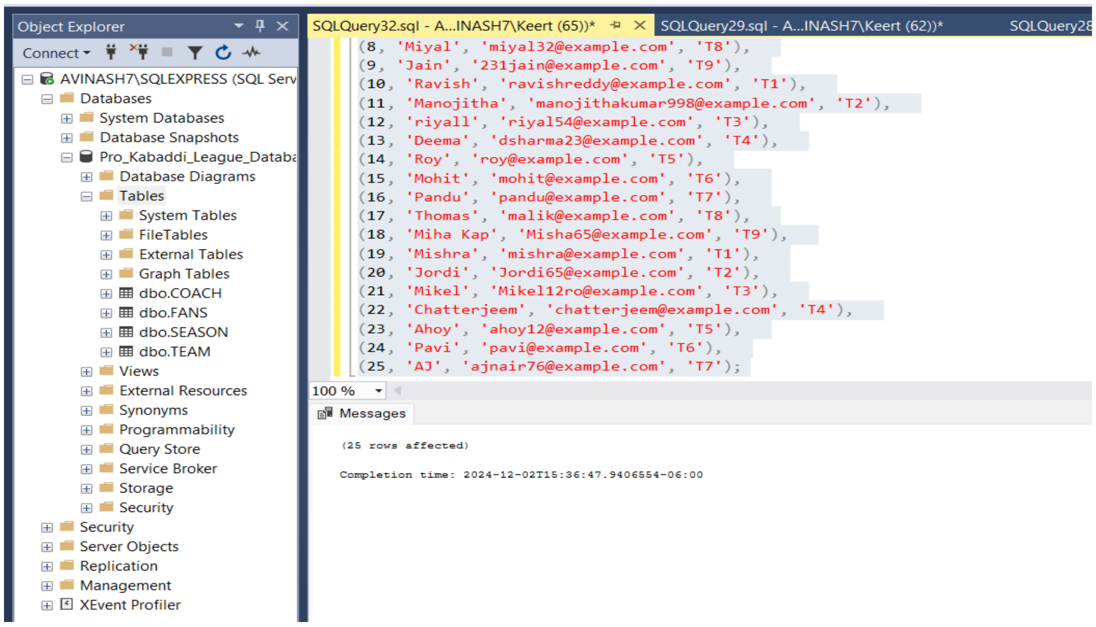
Fan_Email: The fan's email address.

Preferred_Team_ID: The ID that connects the fan's favorite team to the TEAM database.

Relationships between fans and their preferred teams are established by this question, which is essential for interacting with fans, assessing fan preferences, and improving league promotion tactics. Additionally, the data may be utilized to enhance the overall fan experience and run targeted marketing initiatives.

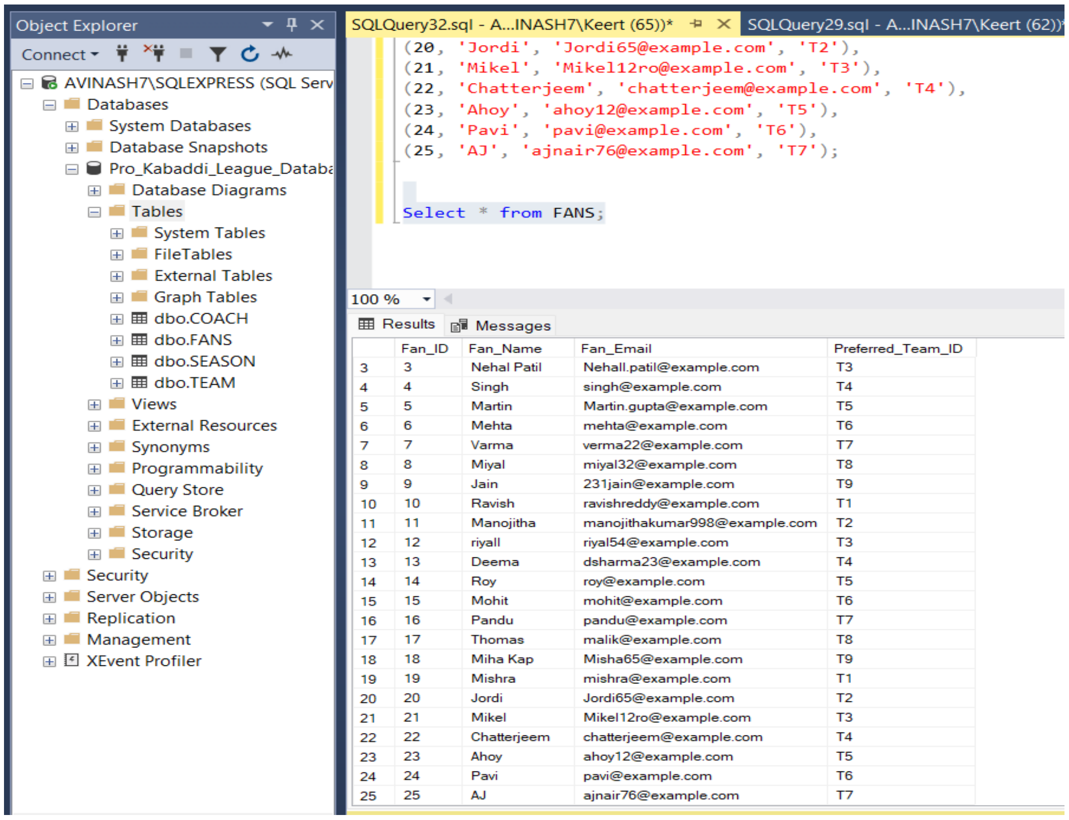
Query:

```
INSERT INTO FANS (Fan_ID, Fan_Name, Fan_Email, Preferred_Team_ID)
 VALUES
 (1, 'Avinash Raj', 'avinash12.ma@example.com', 'T1'),
 (2, 'Naitha Kumari', 'Naitha.kumari@example.com', 'T2'),
 (3, 'Nehal Patil', 'Nehall.patil@example.com', 'T3'),
 (4, 'Singh', 'singh@example.com', 'T4'),
 (5, 'Martin', 'Martin.gupta@example.com', 'T5'),
 (6, 'Mehta', 'mehta@example.com', 'T6'),
 (7, 'Varma', 'verma22@example.com', 'T7'),
 (8, 'Miyal', 'miyal32@example.com', 'T8'),
 (9, 'Jain', '231jain@example.com', 'T9'),
 (10, 'Ravish', 'ravishreddy@example.com', 'T1'),
 (11, 'Manojitha', 'manojithakumar998@example.com', 'T2'),
 (12, 'riyall', 'riyal54@example.com', 'T3'),
 (13, 'Deema', 'dsharma23@example.com', 'T4'),
 (14, 'Roy', 'roy@example.com', 'T5'),

 (15, 'Mohit', 'mohit@example.com', 'T6'),

 (16, 'Pandu', 'pandu@example.com', 'T7'),

 (17, 'Thomas', 'malik@example.com', 'T8'),

 (18, 'Miha Kap', 'Misha65@example.com', 'T9'),

 (19, 'Mishra', 'mishra@example.com', 'T1'),

 (20, 'Jordi', 'Jordi65@example.com', 'T2'),

 (21, 'Mikel', 'Mikel12ro@example.com', 'T3'),

 (22, 'Chatterjeem', 'chatterjeem@example.com', 'T4'),

 (23, 'Ahoy', 'ahoy12@example.com', 'T5'),

 (24, 'Pavi', 'pavi@example.com', 'T6'),

 (25, 'AJ', 'ajnair76@example.com', 'T7');
```



```sql
(8, 'Miyal', 'miyal32@example.com', 'T8'),
(9, 'Jain', '231jain@example.com', 'T9'),
(10, 'Ravish', 'ravishreddy@example.com', 'T1'),
(11, 'Manojitha', 'manojithakumar998@example.com', 'T2'),
(12, 'riyall', 'riyal54@example.com', 'T3'),
(13, 'Deema', 'dsharma23@example.com', 'T4'),
(14, 'Roy', 'roy@example.com', 'T5'),
(15, 'Mohit', 'mohit@example.com', 'T6'),
(16, 'Pandu', 'pandu@example.com', 'T7'),
(17, 'Thomas', 'malik@example.com', 'T8'),
(18, 'Miha Kap', 'Misha65@example.com', 'T9'),
(19, 'Mishra', 'mishra@example.com', 'T1'),
(20, 'Jordi', 'Jordi65@example.com', 'T2'),
(21, 'Mikel', 'Mikel12ro@example.com', 'T3'),
(22, 'Chatterjeem', 'chatterjeem@example.com', 'T4'),
(23, 'Ahoy', 'ahoy12@example.com', 'T5'),
(24, 'Pavi', 'pavi@example.com', 'T6'),
(25, 'AJ', 'ajnair76@example.com', 'T7');
```

```
(25 rows affected)

Completion time: 2024-12-02T15:36:47.9406554-06:00
```



```sql
(20, 'Jordi', 'Jordi65@example.com', 'T2'),
(21, 'Mikel', 'Mikel12ro@example.com', 'T3'),
(22, 'Chatterjeem', 'chatterjeem@example.com', 'T4'),
(23, 'Ahoy', 'ahoy12@example.com', 'T5'),
(24, 'Pavi', 'pavi@example.com', 'T6'),
(25, 'AJ', 'ajnair76@example.com', 'T7');


Select * from FANS;
```

	Fan_ID	Fan_Name	Fan_Email	Preferred_Team_ID
3	3	Nehal Patil	Nehall.patil@example.com	T3
4	4	Singh	singh@example.com	T4
5	5	Martin	Martin.gupta@example.com	T5
6	6	Mehta	mehta@example.com	T6
7	7	Varma	verma22@example.com	T7
8	8	Miyal	miyal32@example.com	T8
9	9	Jain	231jain@example.com	T9
10	10	Ravish	ravishreddy@example.com	T1
11	11	Manojitha	manojithakumar998@example.com	T2
12	12	riyall	riyal54@example.com	T3
13	13	Deema	dsharma23@example.com	T4
14	14	Roy	roy@example.com	T5
15	15	Mohit	mohit@example.com	T6
16	16	Pandu	pandu@example.com	T7
17	17	Thomas	malik@example.com	T8
18	18	Miha Kap	Misha65@example.com	T9
19	19	Mishra	mishra@example.com	T1
20	20	Jordi	Jordi65@example.com	T2
21	21	Mikel	Mikel12ro@example.com	T3
22	22	Chatterjeem	chatterjeem@example.com	T4
23	23	Ahoy	ahoy12@example.com	T5
24	24	Pavi	pavi@example.com	T6
25	25	AJ	ajnair76@example.com	T7

Creation of Team Fan table:

This query's goal is to build a Team_Fan junction table in the Pro_Kabaddi_League_Database. The FANS and TEAM tables have a many-to-many connection thanks to this table. The table is set up as follows

Fan_ID: Defines a distinct fan and uses a foreign key to link to the Fan_ID column in the FANS database.

Team_ID: Denotes a distinct team and is connected to the TEAM table's Team_ID column by a foreign key.

Composite Primary Key: Every fan-team relationship is distinct thanks to the combination of Fan_ID and Team_ID.

Use Case

This table is useful in scenarios where:

- A fan can support multiple teams.
- A team can have multiple fans.

The league can better understand fan participation and preferences thanks to the Team_Fan database, which makes it easier to query and analyze fan-team relationships.

Query:

USE Pro_Kabaddi_League_Database

create table Team_Fan(Fan_ID INT,

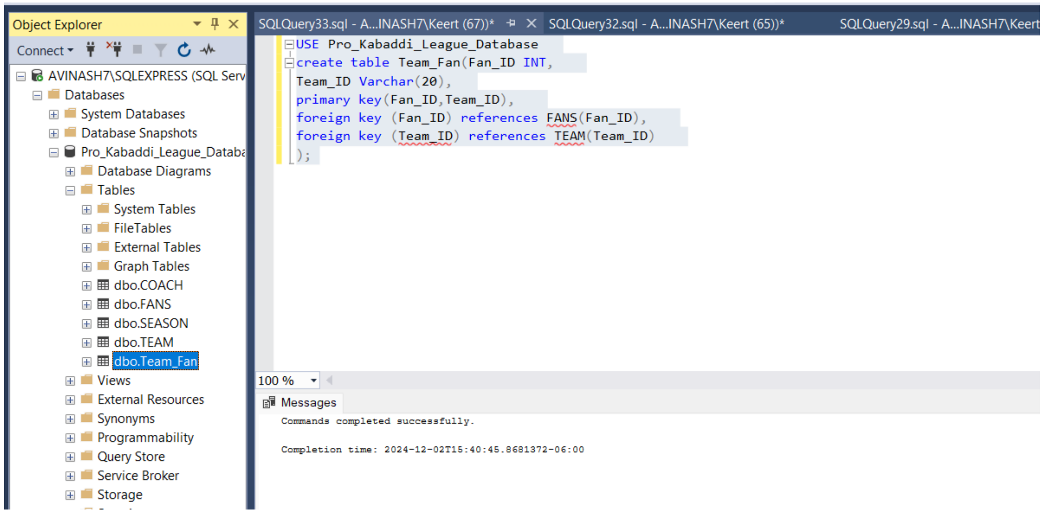
Team_ID Varchar(20),

primary key(Fan_ID,Team_ID),

foreign key (Fan_ID) references FANS(Fan_ID),

foreign key (Team_ID) references TEAM(Team_ID)

);



```
USE Pro_Kabaddi_League_Database
create table Team_Fan(Fan_ID INT,
Team_ID Varchar(20),
primary key(Fan_ID,Team_ID),
foreign key (Fan_ID) references FANS(Fan_ID),
foreign key (Team_ID) references TEAM(Team_ID)
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T15:40:45.8681372-06:00
```

Insertion:

This query aims to create connections between fans and their favorite teams by adding entries to the Team_Fan database. Every record shows a connection between:

- **Fan_ID**: A fan's special identification number.
- **Team_ID**: The fan's special identification number for the team they support.

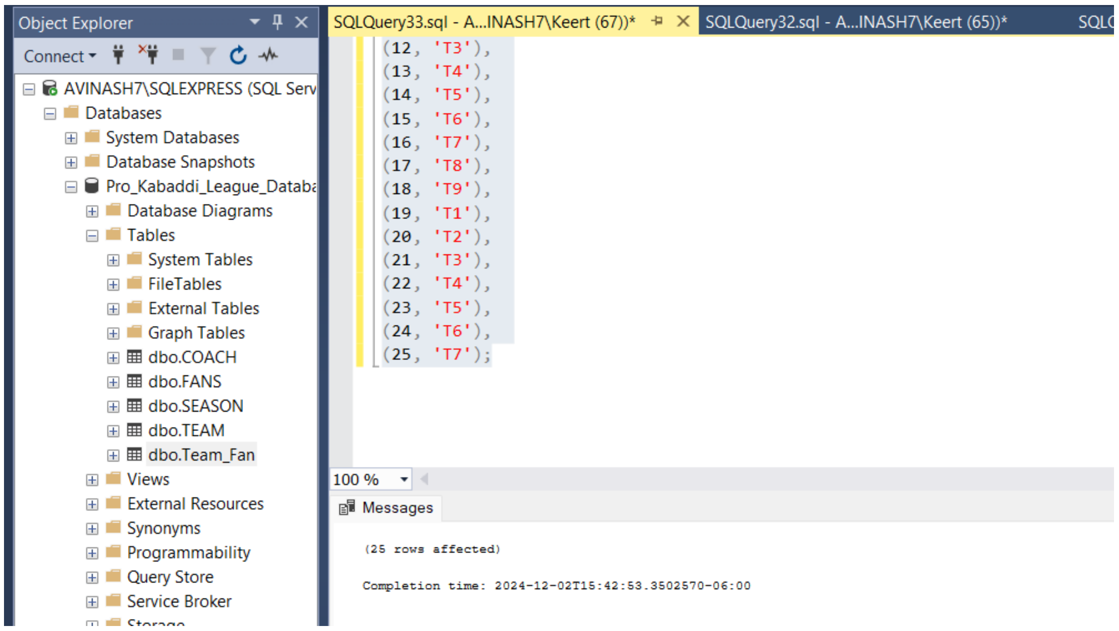
Use Case

This information is essential for comprehending and evaluating the many-to-many connections that exist between teams and their supporters. It permits:

- keeping track of which supporters back certain teams.
- determining which teams receive the greatest support.
- supporting tactics for fan-team contact, include tailored engagement and focused advertising. strategies, such as personalized engagement and targeted campaigns.

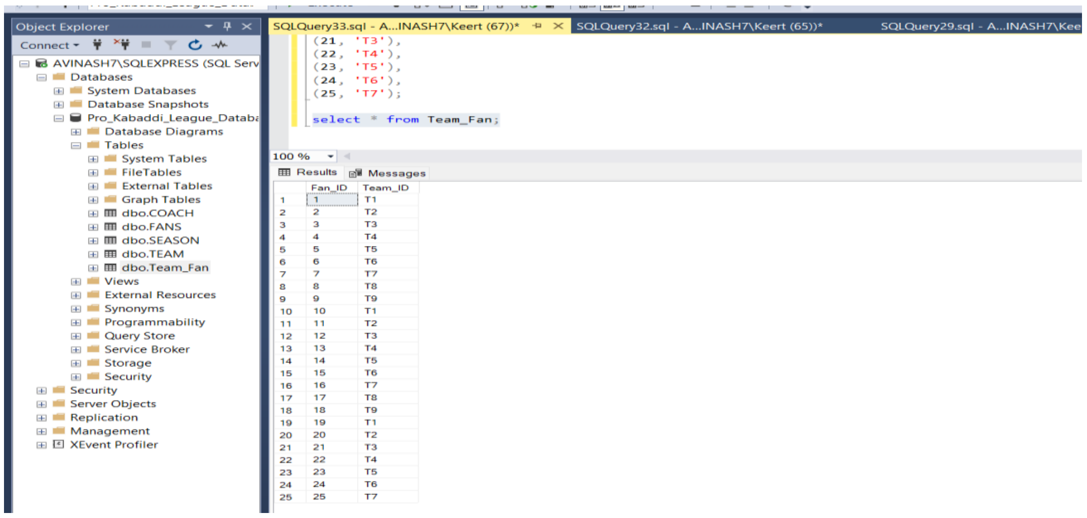
A baseline dataset for additional research on fan-team dynamics in the Pro Kabaddi League is provided by the query.

```
INSERT INTO Team_Fan (Fan_ID, Team_ID)

VALUES

(1, 'T1'),

(2, 'T2'),

(3, 'T3'),

(4, 'T4'),

(5, 'T5'),

(6, 'T6'),

(7, 'T7'),

(8, 'T8'),
(9, 'T9'),

(10, 'T1'),

(11, 'T2'),

(12, 'T3'),

(13, 'T4'),

(14, 'T5'),

(15, 'T6'),

(16, 'T7'),

(17, 'T8'),

(18, 'T9'),

(19, 'T1'),

(20, 'T2'),

(21, 'T3'),

(22, 'T4'),

(23, 'T5'),

(24, 'T6'),

(25, 'T7');
```



```
(12, 'T3'),
(13, 'T4'),
(14, 'T5'),
(15, 'T6'),
(16, 'T7'),
(17, 'T8'),
(18, 'T9'),
(19, 'T1'),
(20, 'T2'),
(21, 'T3'),
(22, 'T4'),
(23, 'T5'),
(24, 'T6'),
(25, 'T7');
```

```
(25 rows affected)

Completion time: 2024-12-02T15:42:53.3502570-06:00
```



```
(21, 'T3'),
(22, 'T4'),
(23, 'T5'),
(24, 'T6'),
(25, 'T7');

select * from Team_Fan;
```

	Fan_ID	Team_ID
1	1	T1
2	2	T2
3	3	T3
4	4	T4
5	5	T5
6	6	T6
7	7	T7
8	8	T8
9	9	T9
10	10	T1
11	11	T2
12	12	T3
13	13	T4
14	14	T5
15	15	T6
16	16	T7
17	17	T8
18	18	T9
19	19	T1
20	20	T2
21	21	T3
22	22	T4
23	23	T5
24	24	T6
25	25	T7

Table creation of Players:

This information is essential for comprehending and evaluating the many-to-many connections that exist between teams and their supporters. It permits:

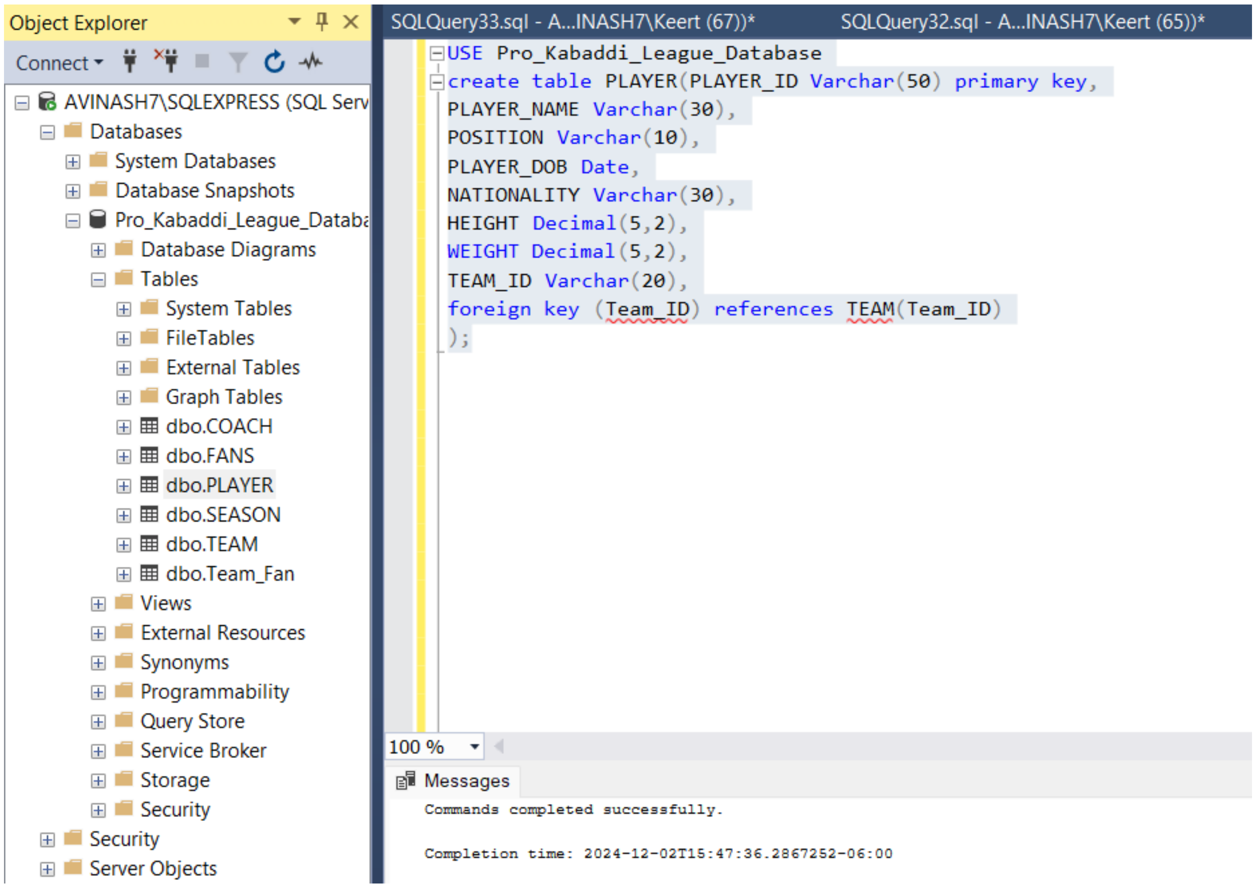
- **PLAYER_ID**: Unique identifier (Primary Key).
- **PLAYER_NAME, POSITION, PLAYER_DOB, NATIONALITY, HEIGHT, WEIGHT**: Player attributes.
- **TEAM_ID**: Foreign key linking players to their respective teams in the TEAM table.

This table creates linkages between team members and arranges player data.

Query:

```
USE Pro_Kabaddi_League_Database

create table PLAYER(PLAYER_ID Varchar(50) primary key,

PLAYER_NAME Varchar(30),

POSITION Varchar(10),

PLAYER_DOB Date,

NATIONALITY Varchar(30),

HEIGHT Decimal(5,2),

WEIGHT Decimal(5,2),

TEAM_ID Varchar(20),

foreign key (Team_ID) references TEAM(Team_ID)
```

);



```sql
USE Pro_Kabaddi_League_Database
create table PLAYER(PLAYER_ID Varchar(50) primary key,
PLAYER_NAME Varchar(30),
POSITION Varchar(10),
PLAYER_DOB Date,
NATIONALITY Varchar(30),
HEIGHT Decimal(5,2),
WEIGHT Decimal(5,2),
TEAM_ID Varchar(20),
foreign key (Team_ID) references TEAM(Team_ID)
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T15:47:36.2867252-06:00
```

Insertion:

This query adds comprehensive information about kabaddi players to the PLAYER database. Every record contains:

- **Player Details:** PLAYER_ID, PLAYER_NAME, POSITION, PLAYER_DOB, NATIONALITY, HEIGHT, and WEIGHT.
- **Team Association**: Players and their respective teams are connected via a foreign key connection created by the TEAM_ID.

Use Case

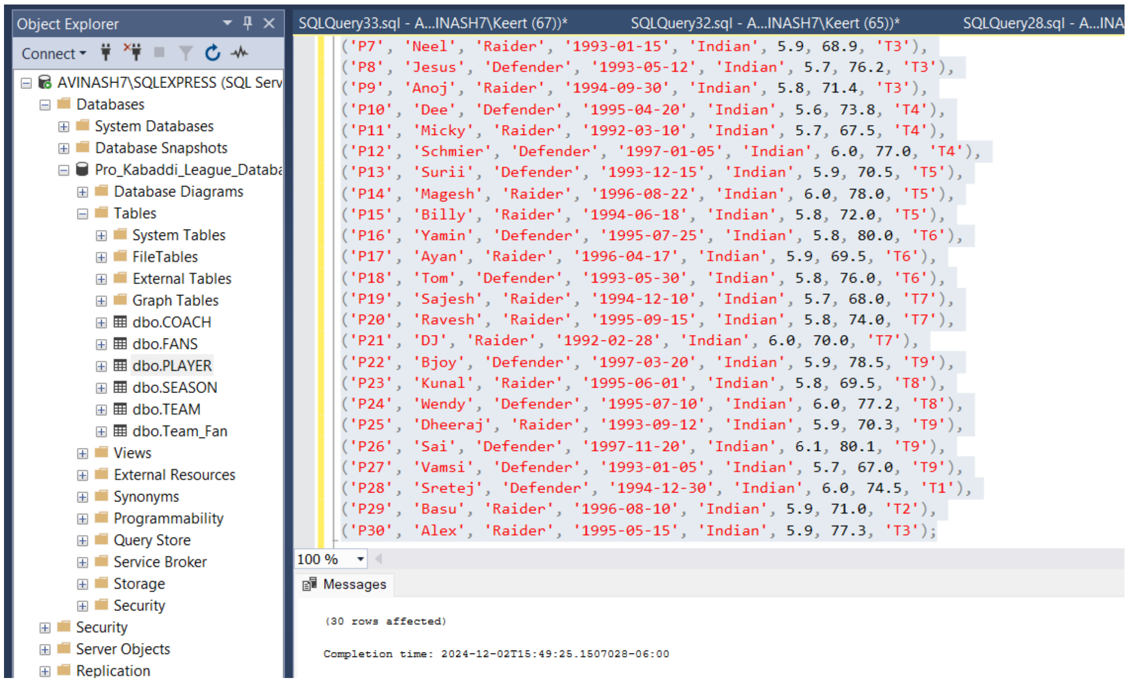
The data supports:

- administration of a team's player roster.
- statistical evaluation based on player characteristics, such as position, weight, and height.
- enabling individual comparisons and team performance monitoring.

```
INSERT INTO PLAYER (PLAYER_ID, PLAYER_NAME, POSITION, PLAYER_DOB,
NATIONALITY, HEIGHT, WEIGHT, TEAM_ID)
VALUES
('P1', 'Sanu', 'Raider', '1993-03-15', 'Indian', 5.8, 70.5, 'T1'),
('P2', 'Sam', 'Defender', '1994-09-20', 'Indian', 6.3, 65.2, 'T2'),
('P3', 'Thomas', 'Raider', '1996-02-18', 'Indian', 6.0, 72.3, 'T1'),
('P4', 'Rakul', 'Defender', '1992-10-12', 'Indian', 5.8, 78.1, 'T2'),
('P5', 'VD', 'Raider', '1997-11-05', 'Indian', 5.7, 70.0, 'T2'),
('P6', 'Ramy', 'Raider', '1995-06-25', 'Indian', 6.1, 80.5, 'T2'),
('P7', 'Neel', 'Raider', '1993-01-15', 'Indian', 5.9, 68.9, 'T3'),
('P8', 'Jesus', 'Defender', '1993-05-12', 'Indian', 5.7, 76.2, 'T3'),
('P9', 'Anoj', 'Raider', '1994-09-30', 'Indian', 5.8, 71.4, 'T3'),
('P10', 'Dee', 'Defender', '1995-04-20', 'Indian', 5.6, 73.8, 'T4'),
('P11', 'Micky', 'Raider', '1992-03-10', 'Indian', 5.7, 67.5, 'T4'),

('P12', 'Schmier', 'Defender', '1997-01-05', 'Indian', 6.0, 77.0, 'T4'),

('P13', 'Surii', 'Defender', '1993-12-15', 'Indian', 5.9, 70.5, 'T5'),

('P14', 'Magesh', 'Raider', '1996-08-22', 'Indian', 6.0, 78.0, 'T5'),

('P15', 'Billy', 'Raider', '1994-06-18', 'Indian', 5.8, 72.0, 'T5'),

('P16', 'Yamin', 'Defender', '1995-07-25', 'Indian', 5.8, 80.0, 'T6'),

('P17', 'Ayan', 'Raider', '1996-04-17', 'Indian', 5.9, 69.5, 'T6'),

('P18', 'Tom', 'Defender', '1993-05-30', 'Indian', 5.8, 76.0, 'T6'),

('P19', 'Sajesh', 'Raider', '1994-12-10', 'Indian', 5.7, 68.0, 'T7'),

('P20', 'Ravesh', 'Raider', '1995-09-15', 'Indian', 5.8, 74.0, 'T7'),

('P21', 'DJ', 'Raider', '1992-02-28', 'Indian', 6.0, 70.0, 'T7'),

('P22', 'Bjoy', 'Defender', '1997-03-20', 'Indian', 5.9, 78.5, 'T9'),

('P23', 'Kunal', 'Raider', '1995-06-01', 'Indian', 5.8, 69.5, 'T8'),

('P24', 'Wendy', 'Defender', '1995-07-10', 'Indian', 6.0, 77.2, 'T8'),

('P25', 'Dheeraj', 'Raider', '1993-09-12', 'Indian', 5.9, 70.3, 'T9'),

('P26', 'Sai', 'Defender', '1997-11-20', 'Indian', 6.1, 80.1, 'T9'),

('P27', 'Vamsi', 'Defender', '1993-01-05', 'Indian', 5.7, 67.0, 'T9'),

('P28', 'Sretej', 'Defender', '1994-12-30', 'Indian', 6.0, 74.5, 'T1'),
```

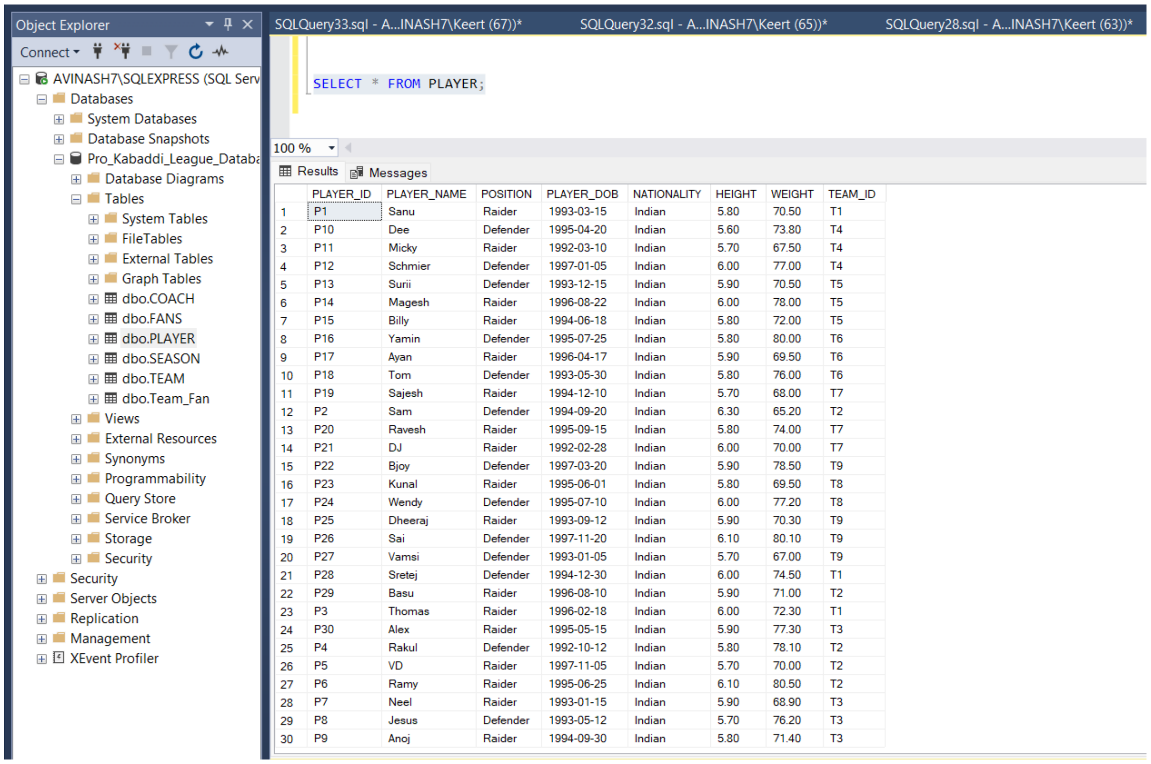
('P29', 'Basu', 'Raider', '1996-08-10', 'Indian', 5.9, 71.0, 'T2'),

('P30', 'Alex', 'Raider', '1995-05-15', 'Indian', 5.9, 77.3, 'T3');



```
('P7', 'Neel', 'Raider', '1993-01-15', 'Indian', 5.9, 68.9, 'T3'),
('P8', 'Jesus', 'Defender', '1993-05-12', 'Indian', 5.7, 76.2, 'T3'),
('P9', 'Anoj', 'Raider', '1994-09-30', 'Indian', 5.8, 71.4, 'T3'),
('P10', 'Dee', 'Defender', '1995-04-20', 'Indian', 5.6, 73.8, 'T4'),
('P11', 'Micky', 'Raider', '1992-03-10', 'Indian', 5.7, 67.5, 'T4'),
('P12', 'Schmier', 'Defender', '1997-01-05', 'Indian', 6.0, 77.0, 'T4'),
('P13', 'Surii', 'Defender', '1993-12-15', 'Indian', 5.9, 70.5, 'T5'),
('P14', 'Magesh', 'Raider', '1996-08-22', 'Indian', 6.0, 78.0, 'T5'),
('P15', 'Billy', 'Raider', '1994-06-18', 'Indian', 5.8, 72.0, 'T5'),
('P16', 'Yamin', 'Defender', '1995-07-25', 'Indian', 5.8, 80.0, 'T6'),
('P17', 'Ayan', 'Raider', '1996-04-17', 'Indian', 5.9, 69.5, 'T6'),
('P18', 'Tom', 'Defender', '1993-05-30', 'Indian', 5.8, 76.0, 'T6'),
('P19', 'Sajesh', 'Raider', '1994-12-10', 'Indian', 5.7, 68.0, 'T7'),
('P20', 'Ravesh', 'Raider', '1995-09-15', 'Indian', 5.8, 74.0, 'T7'),
('P21', 'DJ', 'Raider', '1992-02-28', 'Indian', 6.0, 70.0, 'T7'),
('P22', 'Bjoy', 'Defender', '1997-03-20', 'Indian', 5.9, 78.5, 'T9'),
('P23', 'Kunal', 'Raider', '1995-06-01', 'Indian', 5.8, 69.5, 'T8'),
('P24', 'Wendy', 'Defender', '1995-07-10', 'Indian', 6.0, 77.2, 'T8'),
('P25', 'Dheeraj', 'Raider', '1993-09-12', 'Indian', 5.9, 70.3, 'T9'),
('P26', 'Sai', 'Defender', '1997-11-20', 'Indian', 6.1, 80.1, 'T9'),
('P27', 'Vamsi', 'Defender', '1993-01-05', 'Indian', 5.7, 67.0, 'T9'),
('P28', 'Sretej', 'Defender', '1994-12-30', 'Indian', 6.0, 74.5, 'T1'),
('P29', 'Basu', 'Raider', '1996-08-10', 'Indian', 5.9, 71.0, 'T2'),
('P30', 'Alex', 'Raider', '1995-05-15', 'Indian', 5.9, 77.3, 'T3');
```

```
(30 rows affected)

Completion time: 2024-12-02T15:49:25.1507028-06:00
```



```sql
SELECT * FROM PLAYER;
```

	PLAYER_ID	PLAYER_NAME	POSITION	PLAYER_DOB	NATIONALITY	HEIGHT	WEIGHT	TEAM_ID
1	P1	Sanu	Raider	1993-03-15	Indian	5.80	70.50	T1
2	P10	Dee	Defender	1995-04-20	Indian	5.60	73.80	T4
3	P11	Micky	Raider	1992-03-10	Indian	5.70	67.50	T4
4	P12	Schmier	Defender	1997-01-05	Indian	6.00	77.00	T4
5	P13	Surii	Defender	1993-12-15	Indian	5.90	70.50	T5
6	P14	Magesh	Raider	1996-08-22	Indian	6.00	78.00	T5
7	P15	Billy	Raider	1994-06-18	Indian	5.80	72.00	T5
8	P16	Yamin	Defender	1995-07-25	Indian	5.80	80.00	T6
9	P17	Ayan	Raider	1996-04-17	Indian	5.90	69.50	T6
10	P18	Tom	Defender	1993-05-30	Indian	5.80	76.00	T6
11	P19	Sajesh	Raider	1994-12-10	Indian	5.70	68.00	T7
12	P2	Sam	Defender	1994-09-20	Indian	6.30	65.20	T2
13	P20	Ravesh	Raider	1995-09-15	Indian	5.80	74.00	T7
14	P21	DJ	Raider	1992-02-28	Indian	6.00	70.00	T7
15	P22	Bjoy	Defender	1997-03-20	Indian	5.90	78.50	T9
16	P23	Kunal	Raider	1995-06-01	Indian	5.80	69.50	T8
17	P24	Wendy	Defender	1995-07-10	Indian	6.00	77.20	T8
18	P25	Dheeraj	Raider	1993-09-12	Indian	5.90	70.30	T9
19	P26	Sai	Defender	1997-11-20	Indian	6.10	80.10	T9
20	P27	Vamsi	Defender	1993-01-05	Indian	5.70	67.00	T9
21	P28	Sretej	Defender	1994-12-30	Indian	6.00	74.50	T1
22	P29	Basu	Raider	1996-08-10	Indian	5.90	71.00	T2
23	P3	Thomas	Raider	1996-02-18	Indian	6.00	72.30	T1
24	P30	Alex	Raider	1995-05-15	Indian	5.90	77.30	T3
25	P4	Rakul	Defender	1992-10-12	Indian	5.80	78.10	T2
26	P5	VD	Raider	1997-11-05	Indian	5.70	70.00	T2
27	P6	Ramy	Raider	1995-06-25	Indian	6.10	80.50	T2
28	P7	Neel	Raider	1993-01-15	Indian	5.90	68.90	T3
29	P8	Jesus	Defender	1993-05-12	Indian	5.70	76.20	T3
30	P9	Anoj	Raider	1994-09-30	Indian	5.80	71.40	T3

Creation of Sponsorship Table:

In order to store sponsorship data, this query generates the SPONSORSHIP table in the PRO_KABADDI_LEAGUE_DATABASE. Important characteristics include:

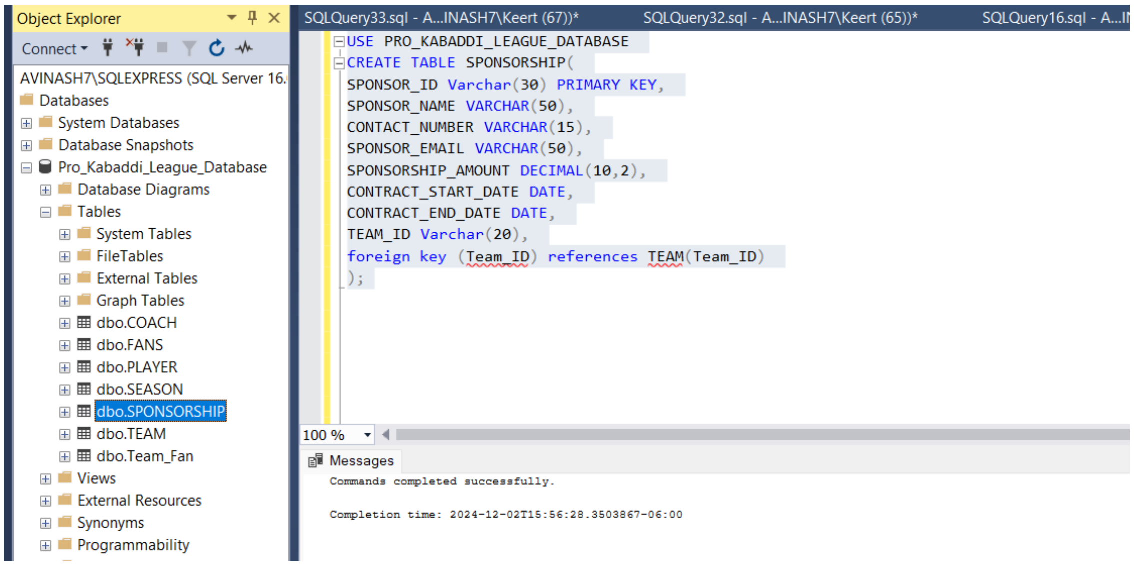
- **Sponsor Details**: SPONSOR_ID (Primary Key), SPONSOR_NAME, CONTACT_NUMBER, SPONSOR_EMAIL.
- **Sponsorship Information**: SPONSORSHIP_AMOUNT, CONTRACT_START_DATE, CONTRACT_END_DATE.
- **Team Association**:Through a foreign key relationship with the TEAM database, TEAM_ID links sponsors with certain teams.

Use Case

The table facilitates:

- Keeping track of sponsorships.
- monitoring grant levels and contract timeframes.
- connecting sponsors with the appropriate teams for analysis and reporting.

```
USE PRO_KABADDI_LEAGUE_DATABASE

CREATE TABLE SPONSORSHIP(

SPONSOR_ID Varchar(50) PRIMARY KEY,

SPONSOR_NAME VARCHAR(50),

CONTACT_NUMBER VARCHAR(15),

SPONSOR_EMAIL VARCHAR(50),

SPONSORSHIP_AMOUNT DECIMAL(10,2),

CONTRACT_START_DATE DATE,

CONTRACT_END_DATE DATE,

TEAM_ID Varchar(50),

foreign key (Team_ID) references TEAM(Team_ID)

);
```



```
USE PRO_KABADDI_LEAGUE_DATABASE
CREATE TABLE SPONSORSHIP(
SPONSOR_ID Varchar(30) PRIMARY KEY,
SPONSOR_NAME VARCHAR(50),
CONTACT_NUMBER VARCHAR(15),
SPONSOR_EMAIL VARCHAR(50),
SPONSORSHIP_AMOUNT DECIMAL(10,2),
CONTRACT_START_DATE DATE,
CONTRACT_END_DATE DATE,
TEAM_ID Varchar(20),
foreign key (Team_ID) references TEAM(Team_ID)
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T15:56:28.3503867-06:00
```

Insertion:

The supplied SQL query links sponsors to their corresponding Pro Kabaddi League teams by inserting several entries into the SPONSORSHIP database. The following are the main fields being filled in:

SPONSOR_ID: Unique identifier for the sponsor.

SPONSOR_NAME: Name of the sponsoring company.

CONTACT_NUMBER: Sponsor's contact phone number.

SPONSOR_EMAIL: Sponsor's email address.

SPONSORSHIP_AMOUNT: The amount of money sponsored for the team.

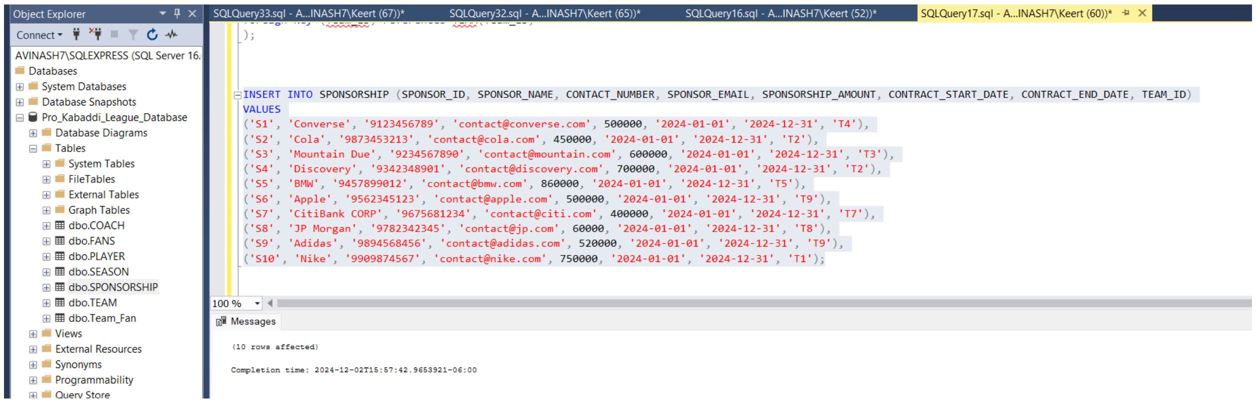
CONTRACT_START_DATE: The date the sponsorship agreement begins.

CONTRACT_END_DATE: The date the sponsorship agreement ends.

TEAM_ID: Refers to the associated team in the league.

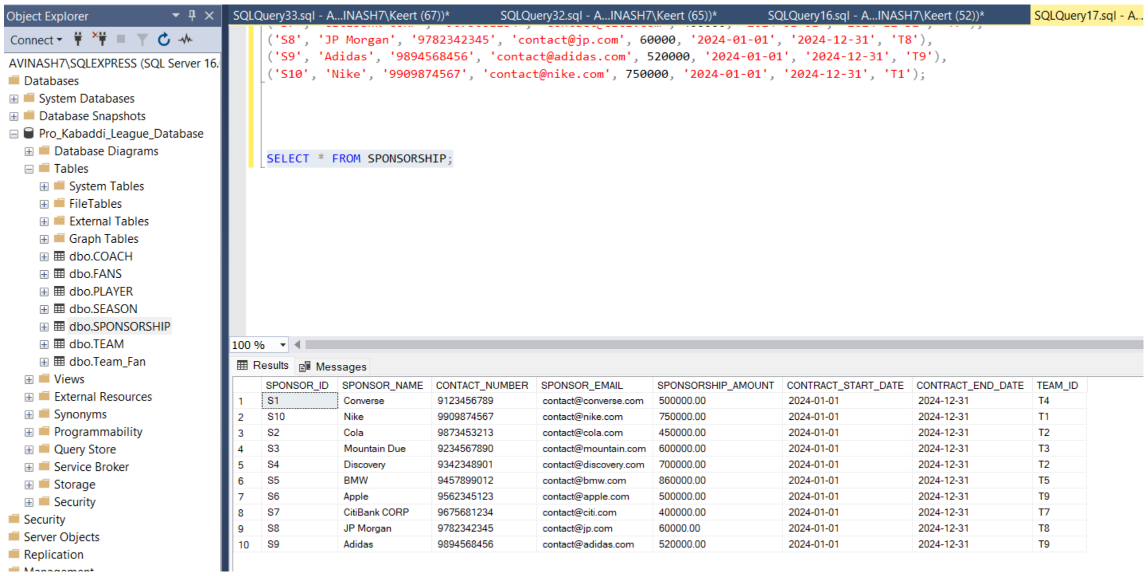
The financial and contractual ties between teams and sponsors may be monitored with the use of this data.

```
INSERT INTO SPONSORSHIP (SPONSOR_ID, SPONSOR_NAME, CONTACT_NUMBER,
SPONSOR_EMAIL, SPONSORSHIP_AMOUNT, CONTRACT_START_DATE,
CONTRACT_END_DATE, TEAM_ID)
VALUES
('S1', 'Converse', '9123456789', 'contact@converse.com', 500000, '2024-01-01', '2024-12-31',
'T4'),
('S2', 'Cola', '9873453213', 'contact@cola.com', 450000, '2024-01-01', '2024-12-31', 'T2'),
('S3', 'Mountain Due', '9234567890', 'contact@mountain.com', 600000, '2024-01-01', '2024-12-
31', 'T3'),
('S4', 'Discovery', '9342348901', 'contact@discovery.com', 700000, '2024-01-01', '2024-12-31',
'T2'),
('S5', 'BMW', '9457899012', 'contact@bmw.com', 860000, '2024-01-01', '2024-12-31', 'T5'),
('S6', 'Apple', '9562345123', 'contact@apple.com', 500000, '2024-01-01', '2024-12-31', 'T9'),
('S7', 'CitiBank CORP', '9675681234', 'contact@citi.com', 400000, '2024-01-01', '2024-12-31',
'T7'),
('S8', 'JP Morgan', '9782342345', 'contact@jp.com', 60000, '2024-01-01', '2024-12-31', 'T8'),
('S9', 'Adidas', '9894568456', 'contact@adidas.com', 520000, '2024-01-01', '2024-12-31', 'T9'),
('S10', 'Nike', '9909874567', 'contact@nike.com', 750000, '2024-01-01', '2024-12-31', 'T1');
```



```sql
);


INSERT INTO SPONSORSHIP (SPONSOR_ID, SPONSOR_NAME, CONTACT_NUMBER, SPONSOR_EMAIL, SPONSORSHIP_AMOUNT, CONTRACT_START_DATE, CONTRACT_END_DATE, TEAM_ID)
VALUES
('S1', 'Converse', '9123456789', 'contact@converse.com', 500000, '2024-01-01', '2024-12-31', 'T4'),
('S2', 'Cola', '9873453213', 'contact@cola.com', 450000, '2024-01-01', '2024-12-31', 'T2'),
('S3', 'Mountain Due', '9234567890', 'contact@mountain.com', 600000, '2024-01-01', '2024-12-31', 'T3'),
('S4', 'Discovery', '9342348901', 'contact@discovery.com', 700000, '2024-01-01', '2024-12-31', 'T2'),
('S5', 'BMW', '9457899012', 'contact@bmw.com', 860000, '2024-01-01', '2024-12-31', 'T5'),
('S6', 'Apple', '9562345123', 'contact@apple.com', 500000, '2024-01-01', '2024-12-31', 'T9'),
('S7', 'CitiBank CORP', '9675681234', 'contact@citi.com', 400000, '2024-01-01', '2024-12-31', 'T7'),
('S8', 'JP Morgan', '9782342345', 'contact@jp.com', 60000, '2024-01-01', '2024-12-31', 'T8'),
('S9', 'Adidas', '9894568456', 'contact@adidas.com', 520000, '2024-01-01', '2024-12-31', 'T9'),
('S10', 'Nike', '9909874567', 'contact@nike.com', 750000, '2024-01-01', '2024-12-31', 'T1');
```

```
(10 rows affected)

Completion time: 2024-12-02T15:57:42.9653921-06:00
```



```sql
('S8', 'JP Morgan', '9782342345', 'contact@jp.com', 60000, '2024-01-01', '2024-12-31', 'T8'),
('S9', 'Adidas', '9894568456', 'contact@adidas.com', 520000, '2024-01-01', '2024-12-31', 'T9'),
('S10', 'Nike', '9909874567', 'contact@nike.com', 750000, '2024-01-01', '2024-12-31', 'T1');



SELECT * FROM SPONSORSHIP;
```

	SPONSOR_ID	SPONSOR_NAME	CONTACT_NUMBER	SPONSOR_EMAIL	SPONSORSHIP_AMOUNT	CONTRACT_START_DATE	CONTRACT_END_DATE	TEAM_ID
1	S1	Converse	9123456789	contact@converse.com	500000.00	2024-01-01	2024-12-31	T4
2	S10	Nike	9909874567	contact@nike.com	750000.00	2024-01-01	2024-12-31	T1
3	S2	Cola	9873453213	contact@cola.com	450000.00	2024-01-01	2024-12-31	T2
4	S3	Mountain Due	9234567890	contact@mountain.com	600000.00	2024-01-01	2024-12-31	T3
5	S4	Discovery	9342348901	contact@discovery.com	700000.00	2024-01-01	2024-12-31	T2
6	S5	BMW	9457899012	contact@bmw.com	860000.00	2024-01-01	2024-12-31	T5
7	S6	Apple	9562345123	contact@apple.com	500000.00	2024-01-01	2024-12-31	T9
8	S7	CitiBank CORP	9675681234	contact@citi.com	400000.00	2024-01-01	2024-12-31	T7
9	S8	JP Morgan	9782342345	contact@jp.com	60000.00	2024-01-01	2024-12-31	T8
10	S9	Adidas	9894568456	contact@adidas.com	520000.00	2024-01-01	2024-12-31	T9

Creation of Venue Entity table:

This table is essential for keeping track of the locations, names, capacities, and unique IDs of the venues used in Pro Kabaddi League matches. The information kept in this table will be useful for managing and referencing locations while planning games and events.

USE PRO_KABADDI_LEAGUE_DATABASE

CREATE TABLE Venues(

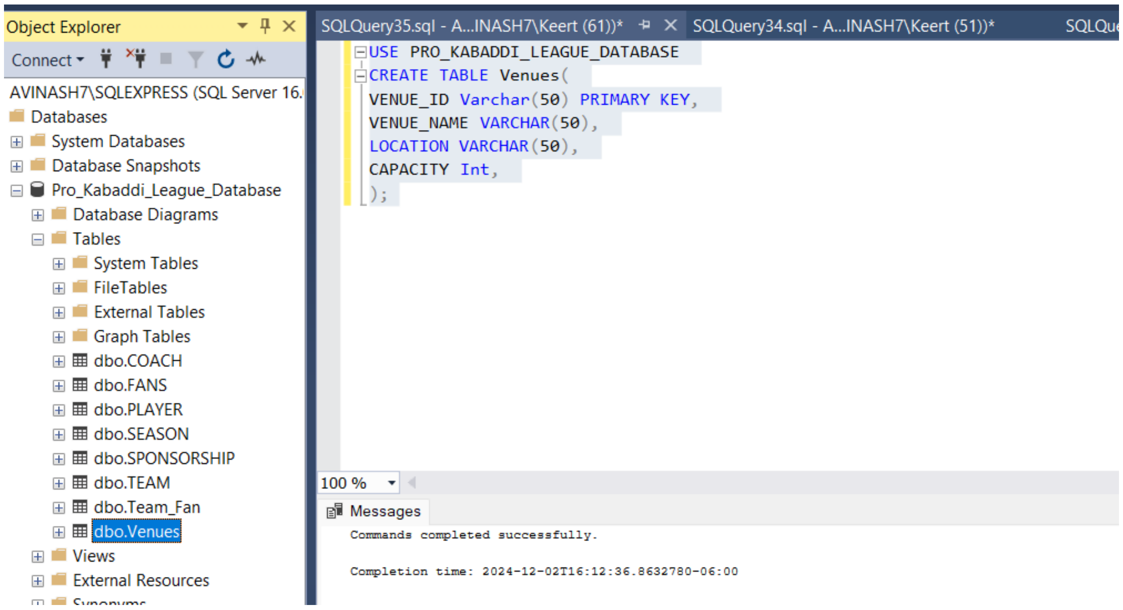
VENUE_ID Varchar(50) PRIMARY KEY,

VENUE_NAME VARCHAR(50),

LOCATION VARCHAR(50),

CAPACITY INT,

);



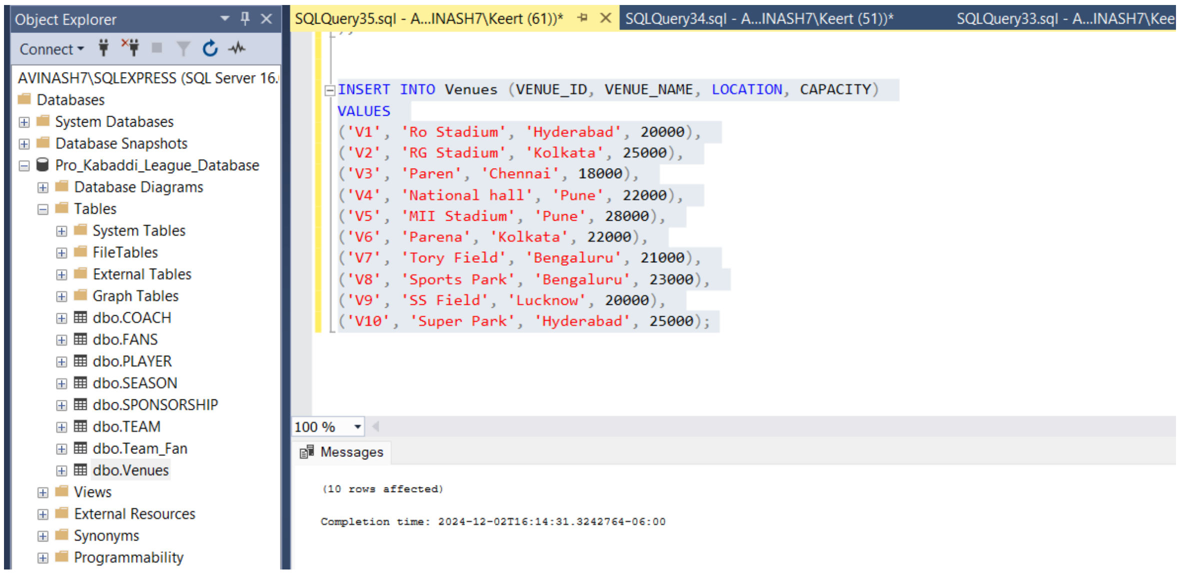
```
USE PRO_KABADDI_LEAGUE_DATABASE
CREATE TABLE Venues(
VENUE_ID Varchar(50) PRIMARY KEY,
VENUE_NAME VARCHAR(50),
LOCATION VARCHAR(50),
CAPACITY Int,
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T16:12:36.8632780-06:00
```

Insertion:

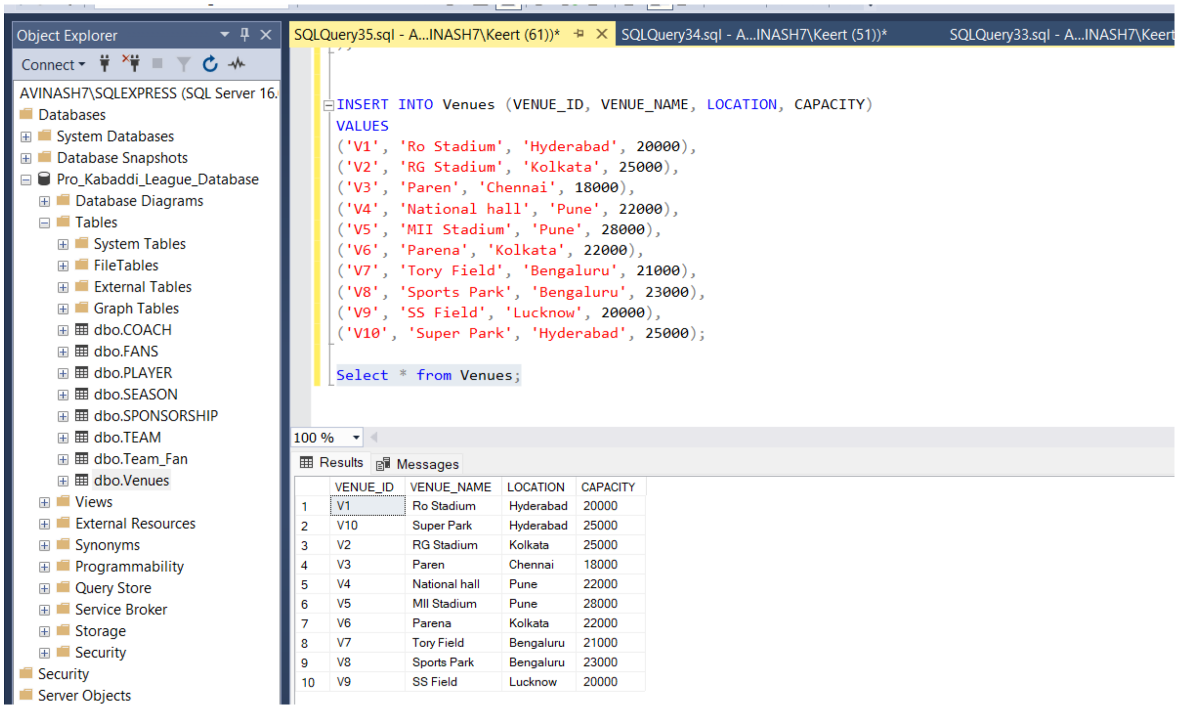
This query adds information about several venues to the Venues database. Every venue has a distinct VENUE_ID that contains details about its name, address, and capacity. This information may be used in the future for scheduling, assigning match locations, and managing leagues.

```sql
INSERT INTO Venues (VENUE_ID, VENUE_NAME, LOCATION, CAPACITY)

VALUES

('V1', 'Ro Stadium', 'Hyderabad', 20000),

('V2', 'RG Stadium', 'Kolkata', 25000),

('V3', 'Paren', 'Chennai', 18000),

('V4', 'National hall', 'Pune', 22000),

('V5', 'MII Stadium', 'Pune', 28000),

('V6', 'Parena', 'Kolkata', 22000),

('V7', 'Tory Field', 'Bengaluru', 21000),

('V8', 'Sports Park', 'Bengaluru', 23000),

('V9', 'SS Field', 'Lucknow', 20000),

('V10', 'Super Park', 'Hyderabad', 25000);
```



```
INSERT INTO Venues (VENUE_ID, VENUE_NAME, LOCATION, CAPACITY)
VALUES
('V1', 'Ro Stadium', 'Hyderabad', 20000),
('V2', 'RG Stadium', 'Kolkata', 25000),
('V3', 'Paren', 'Chennai', 18000),
('V4', 'National hall', 'Pune', 22000),
('V5', 'MII Stadium', 'Pune', 28000),
('V6', 'Parena', 'Kolkata', 22000),
('V7', 'Tory Field', 'Bengaluru', 21000),
('V8', 'Sports Park', 'Bengaluru', 23000),
('V9', 'SS Field', 'Lucknow', 20000),
('V10', 'Super Park', 'Hyderabad', 25000);
```

```
(10 rows affected)

Completion time: 2024-12-02T16:14:31.3242764-06:00
```



```
INSERT INTO Venues (VENUE_ID, VENUE_NAME, LOCATION, CAPACITY)
VALUES
('V1', 'Ro Stadium', 'Hyderabad', 20000),
('V2', 'RG Stadium', 'Kolkata', 25000),
('V3', 'Paren', 'Chennai', 18000),
('V4', 'National hall', 'Pune', 22000),
('V5', 'MII Stadium', 'Pune', 28000),
('V6', 'Parena', 'Kolkata', 22000),
('V7', 'Tory Field', 'Bengaluru', 21000),
('V8', 'Sports Park', 'Bengaluru', 23000),
('V9', 'SS Field', 'Lucknow', 20000),
('V10', 'Super Park', 'Hyderabad', 25000);

Select * from Venues;
```

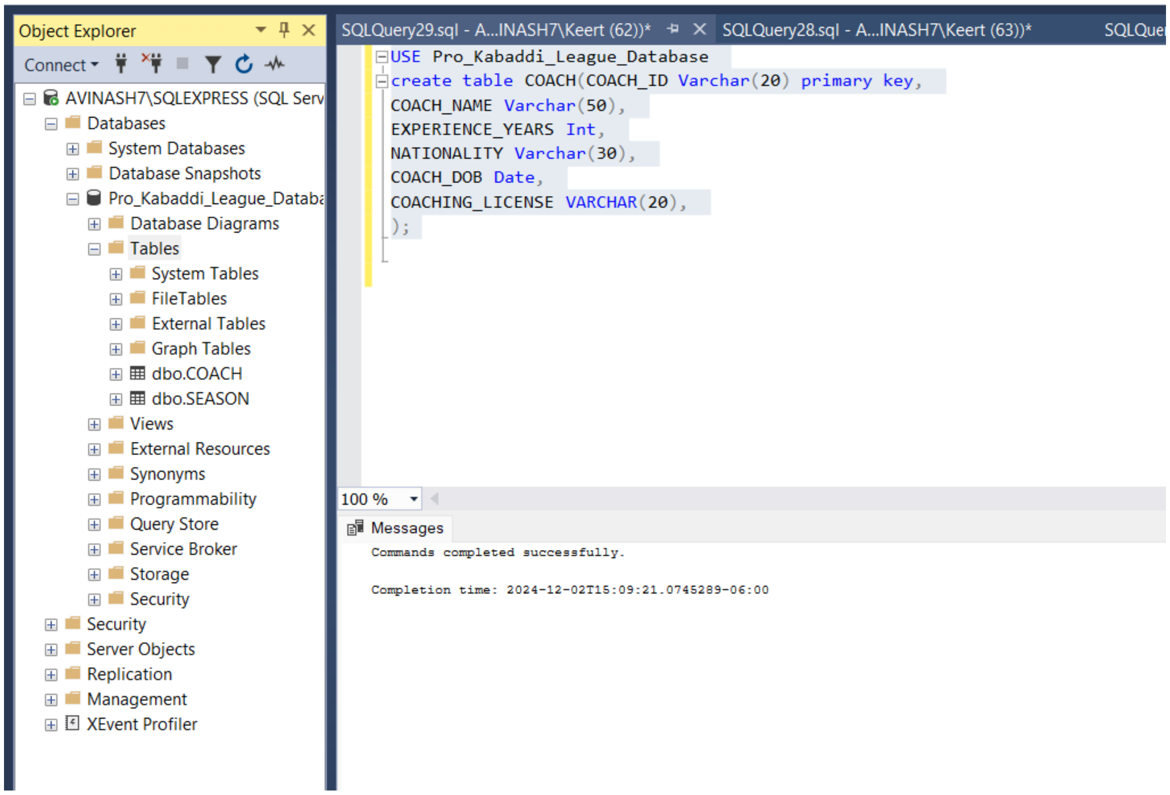
	VENUE_ID	VENUE_NAME	LOCATION	CAPACITY
1	V1	Ro Stadium	Hyderabad	20000
2	V10	Super Park	Hyderabad	25000
3	V2	RG Stadium	Kolkata	25000
4	V3	Paren	Chennai	18000
5	V4	National hall	Pune	22000
6	V5	MII Stadium	Pune	28000
7	V6	Parena	Kolkata	22000
8	V7	Tory Field	Bengaluru	21000
9	V8	Sports Park	Bengaluru	23000
10	V9	SS Field	Lucknow	20000

Creation of Coach Table:

Important information regarding the Pro Kabaddi League coaches is kept in the COACH table.

Every entry in this table includes details on the coach's background, identity, and team. This table makes it easier to manage coaches and their teams by assigning each coach to a team using the Team_ID.

```
USE Pro_Kabaddi_League_Database

create table COACH(COACH_ID Varchar(20) primary key,

COACH_NAME Varchar(50),

EXPERIENCE_YEARS Int,

NATIONALITY Varchar(30),

COACH_DOB Date,

COACHING_LICENSE VARCHAR(20),

);
```



```
USE Pro_Kabaddi_League_Database
create table COACH(COACH_ID Varchar(20) primary key,
COACH_NAME Varchar(50),
EXPERIENCE_YEARS Int,
NATIONALITY Varchar(30),
COACH_DOB Date,
COACHING_LICENSE VARCHAR(20),
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T15:09:21.0745289-06:00
```

Insertion:

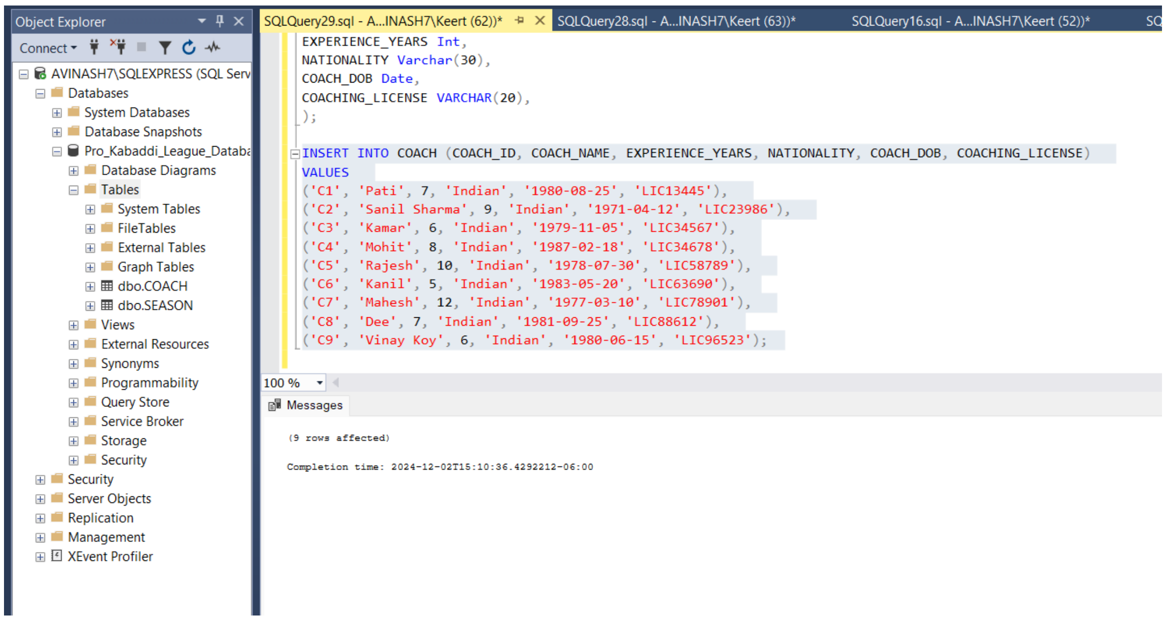
9 entries, each representing a different coach, are inserted into the COACH database by this query. The values line up with the COACH table's columns:

- **COACH_ID**: Unique identifier for each coach.
- **COACH_NAME**: Name of the coach.
- **EXPERIENCE_YEARS**: Number of years of coaching experience.
- **NATIONALITY**: Nationality of the coach.
- **COACH_DOB**: Date of birth of the coach.
- **COACHING_LICENSE**: The coaching license number.
- **TEAM_ID**: The ID of the team associated with the coach.

Every entry adds a coach with the necessary characteristics and uses the TEAM_ID to link them to a team. Connecting coaches with the clubs they oversee in the Pro Kabaddi League database will be helpful.

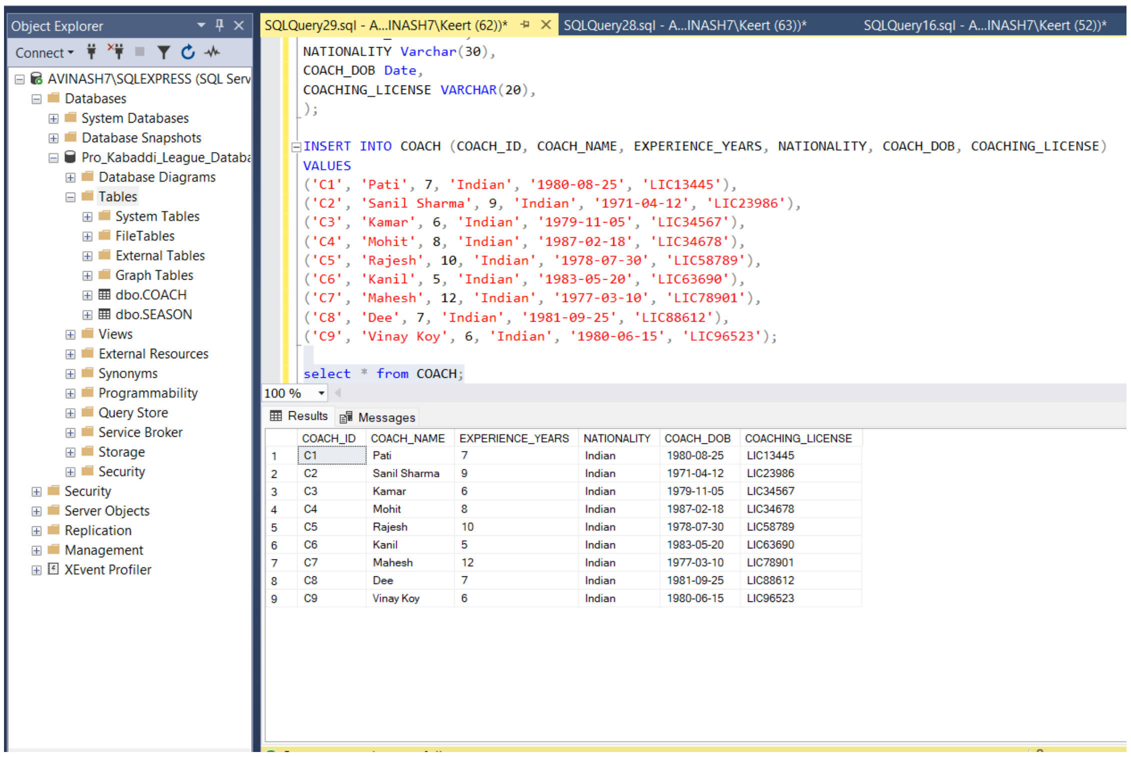
You may use this explanation to explain how data is inserted into the COACH table in your final document.

```
INSERT INTO COACH (COACH_ID, COACH_NAME, EXPERIENCE_YEARS,
NATIONALITY, COACH_DOB, COACHING_LICENSE)
VALUES
('C1', 'Pati', 7, 'Indian', '1980-08-25', 'LIC13445'),
('C2', 'Sanil Sharma', 9, 'Indian', '1971-04-12', 'LIC23986'),
('C3', 'Kamar', 6, 'Indian', '1979-11-05', 'LIC34567'),
('C4', 'Mohit', 8, 'Indian', '1987-02-18', 'LIC34678'),
('C5', 'Rajesh', 10, 'Indian', '1978-07-30', 'LIC58789'),
('C6', 'Kanil', 5, 'Indian', '1983-05-20', 'LIC63690'),
('C7', 'Mahesh', 12, 'Indian', '1977-03-10', 'LIC78901'),
('C8', 'Dee', 7, 'Indian', '1981-09-25', 'LIC88612'),
('C9', 'Vinay Koy', 6, 'Indian', '1980-06-15', 'LIC96523');
```



```
EXPERIENCE_YEARS Int,
NATIONALITY Varchar(30),
COACH_DOB Date,
COACHING_LICENSE VARCHAR(20),
);

INSERT INTO COACH (COACH_ID, COACH_NAME, EXPERIENCE_YEARS, NATIONALITY, COACH_DOB, COACHING_LICENSE)
VALUES
('C1', 'Pati', 7, 'Indian', '1980-08-25', 'LIC13445'),
('C2', 'Sanil Sharma', 9, 'Indian', '1971-04-12', 'LIC23986'),
('C3', 'Kamar', 6, 'Indian', '1979-11-05', 'LIC34567'),
('C4', 'Mohit', 8, 'Indian', '1987-02-18', 'LIC34678'),
('C5', 'Rajesh', 10, 'Indian', '1978-07-30', 'LIC58789'),
('C6', 'Kanil', 5, 'Indian', '1983-05-20', 'LIC63690'),
('C7', 'Mahesh', 12, 'Indian', '1977-03-10', 'LIC78901'),
('C8', 'Dee', 7, 'Indian', '1981-09-25', 'LIC88612'),
('C9', 'Vinay Koy', 6, 'Indian', '1980-06-15', 'LIC96523');
```

```
(9 rows affected)

Completion time: 2024-12-02T15:10:36.4292212-06:00
```



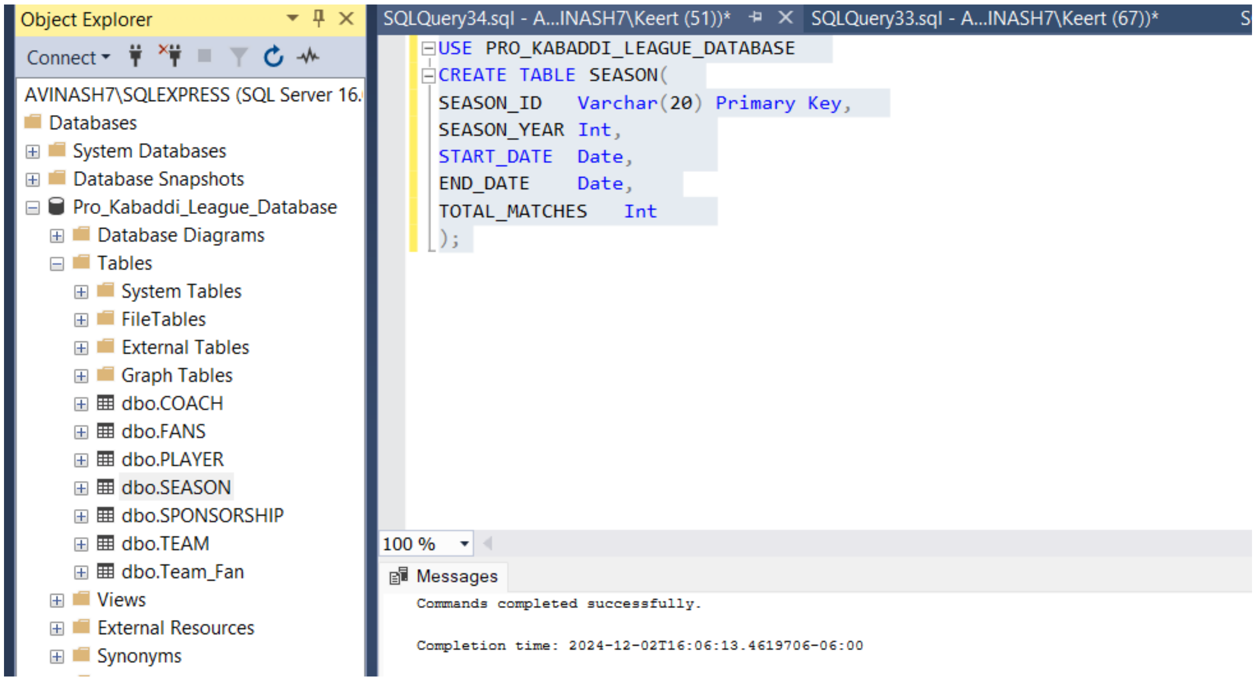
```
NATIONALITY Varchar(30),
COACH_DOB Date,
COACHING_LICENSE VARCHAR(20),
);

INSERT INTO COACH (COACH_ID, COACH_NAME, EXPERIENCE_YEARS, NATIONALITY, COACH_DOB, COACHING_LICENSE)
VALUES
('C1', 'Pati', 7, 'Indian', '1980-08-25', 'LIC13445'),
('C2', 'Sanil Sharma', 9, 'Indian', '1971-04-12', 'LIC23986'),
('C3', 'Kamar', 6, 'Indian', '1979-11-05', 'LIC34567'),
('C4', 'Mohit', 8, 'Indian', '1987-02-18', 'LIC34678'),
('C5', 'Rajesh', 10, 'Indian', '1978-07-30', 'LIC58789'),
('C6', 'Kanil', 5, 'Indian', '1983-05-20', 'LIC63690'),
('C7', 'Mahesh', 12, 'Indian', '1977-03-10', 'LIC78901'),
('C8', 'Dee', 7, 'Indian', '1981-09-25', 'LIC88612'),
('C9', 'Vinay Koy', 6, 'Indian', '1980-06-15', 'LIC96523');

select * from COACH;
```

	COACH_ID	COACH_NAME	EXPERIENCE_YEARS	NATIONALITY	COACH_DOB	COACHING_LICENSE
1	C1	Pati	7	Indian	1980-08-25	LIC13445
2	C2	Sanil Sharma	9	Indian	1971-04-12	LIC23986
3	C3	Kamar	6	Indian	1979-11-05	LIC34567
4	C4	Mohit	8	Indian	1987-02-18	LIC34678
5	C5	Rajesh	10	Indian	1978-07-30	LIC58789
6	C6	Kanil	5	Indian	1983-05-20	LIC63690
7	C7	Mahesh	12	Indian	1977-03-10	LIC78901
8	C8	Dee	7	Indian	1981-09-25	LIC88612
9	C9	Vinay Koy	6	Indian	1980-06-15	LIC96523

Creation of Season table:

The purpose of the SEASON database is to effectively organize and query data pertaining to every Pro Kabaddi season. It will be crucial in offering thorough insights for any study, report creation, or any inquiries concerning the league's past statistics. This table constitutes a fundamental part of the database because of its well-defined structure, which covers the duration of each season as well as the quantity of games played.

```
USE PRO_KABADDI_LEAGUE_DATABASE

CREATE TABLE SEASON(

SEASON_ID  Varchar(20)     Primary Key,

SEASON_YEAR       Int,

START_DATE        Date,

END_DATE   Date,

TOTAL_MATCHES  Int

);
```



```
USE PRO_KABADDI_LEAGUE_DATABASE
CREATE TABLE SEASON(
SEASON_ID    Varchar(20) Primary Key,
SEASON_YEAR  Int,
START_DATE   Date,
END_DATE     Date,
TOTAL_MATCHES    Int
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T16:06:13.4619706-06:00
```

Insertion of data in season table:

In order to monitor the specifics of a certain Pro Kabaddi League season, this query is necessary in order to add a new record to the SEASON database. By including this data, we give the fundamental knowledge required to link league statistics to the appropriate season, which is essential for any reporting or analysis. In this instance, the query marks the 2024 season with distinct identities, date ranges, and the number of matches.

INSERT INTO SEASON

VALUES ('SEASON1', 2020, '2020-01-01', '2020-12-31', 120);

INSERT INTO SEASON

VALUES ('SEASON2', 2021, '2021-01-01', '2021-12-31', 120);

INSERT INTO SEASON

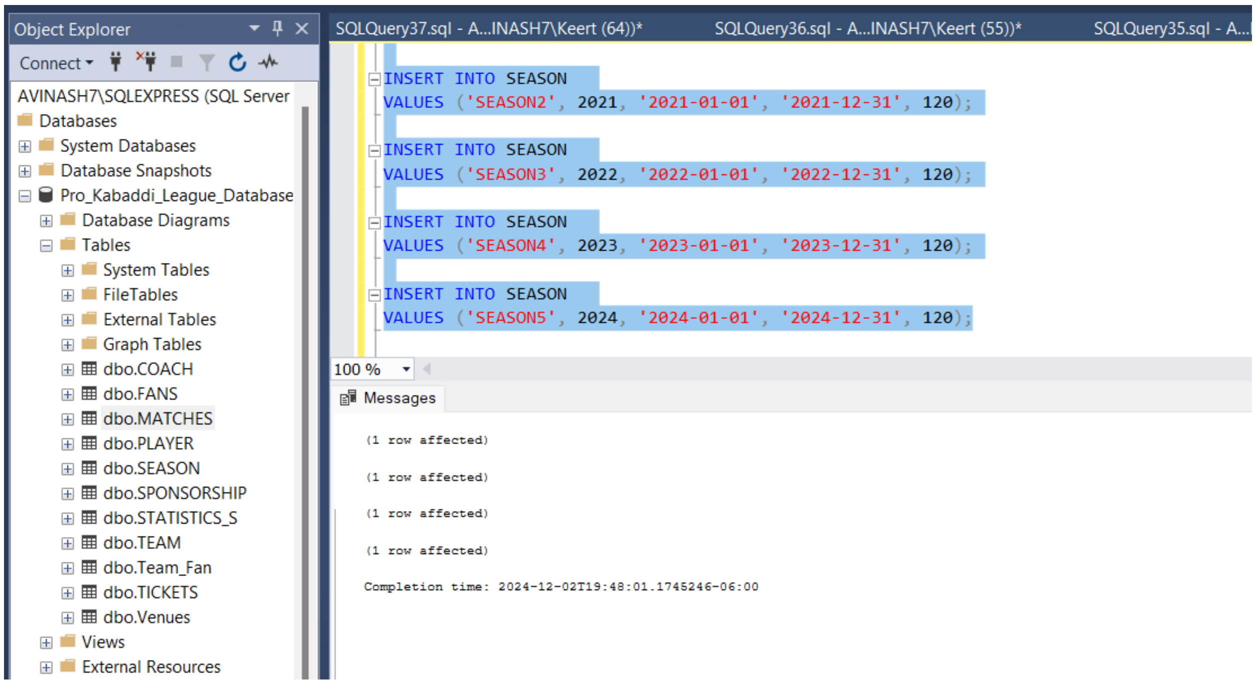
VALUES ('SEASON3', 2022, '2022-01-01', '2022-12-31', 120);

INSERT INTO SEASON

VALUES ('SEASON4', 2023, '2023-01-01', '2023-12-31', 120);

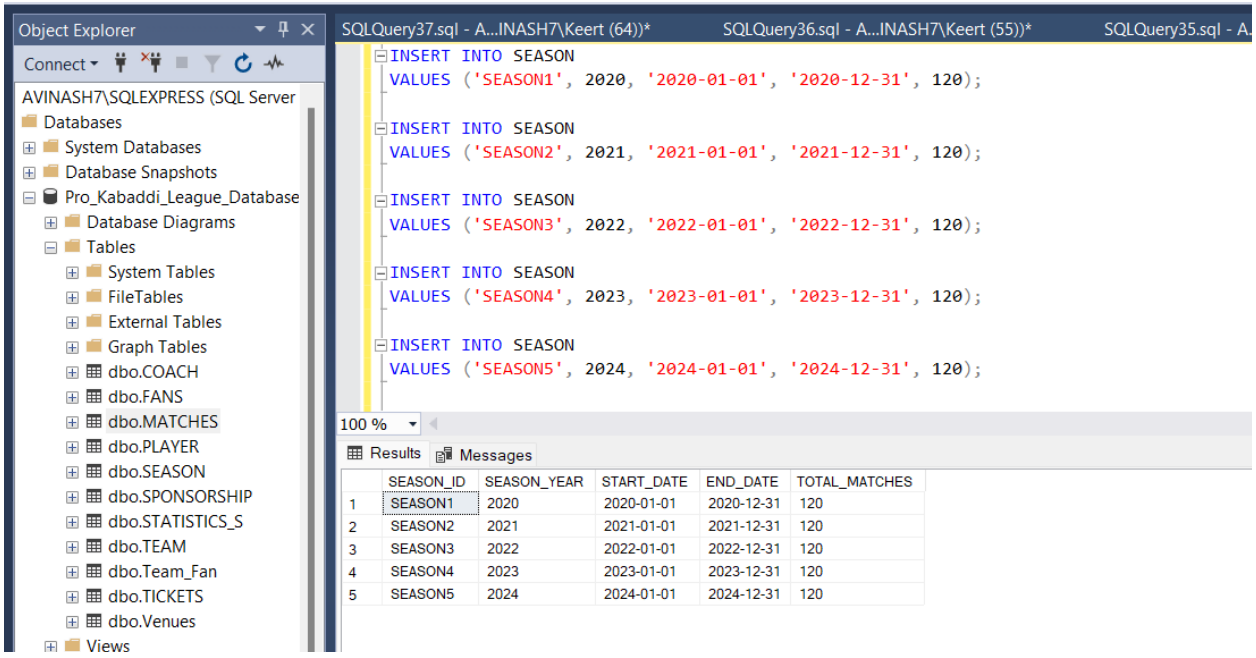
INSERT INTO SEASON

VALUES ('SEASON5', 2024, '2024-01-01', '2024-12-31', 120);



```
INSERT INTO SEASON
VALUES ('SEASON2', 2021, '2021-01-01', '2021-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON3', 2022, '2022-01-01', '2022-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON4', 2023, '2023-01-01', '2023-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON5', 2024, '2024-01-01', '2024-12-31', 120);
```

```
(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

Completion time: 2024-12-02T19:48:01.1745246-06:00
```



```
INSERT INTO SEASON
VALUES ('SEASON1', 2020, '2020-01-01', '2020-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON2', 2021, '2021-01-01', '2021-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON3', 2022, '2022-01-01', '2022-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON4', 2023, '2023-01-01', '2023-12-31', 120);

INSERT INTO SEASON
VALUES ('SEASON5', 2024, '2024-01-01', '2024-12-31', 120);
```

	SEASON_ID	SEASON_YEAR	START_DATE	END_DATE	TOTAL_MATCHES
1	SEASON1	2020	2020-01-01	2020-12-31	120
2	SEASON2	2021	2021-01-01	2021-12-31	120
3	SEASON3	2022	2022-01-01	2022-12-31	120
4	SEASON4	2023	2023-01-01	2023-12-31	120
5	SEASON5	2024	2024-01-01	2024-12-31	120

Creation of Match table:

To save comprehensive information about every Pro Kabaddi League encounter, the MATCHES table is necessary. It consists of:

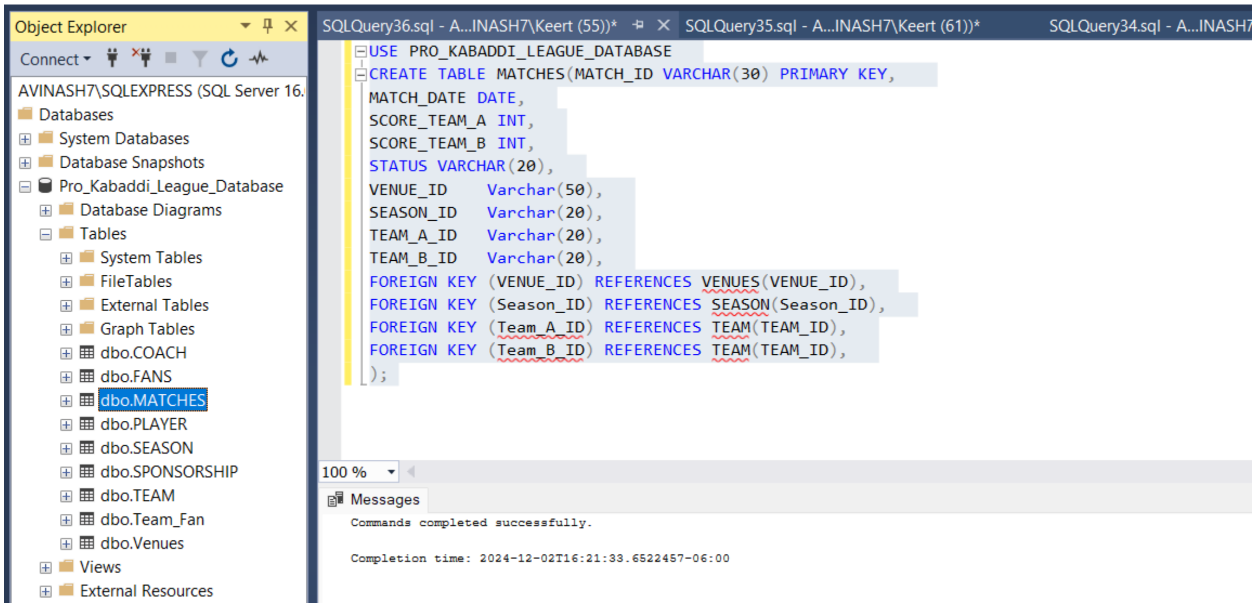
- **Match Details**: Such includes the match's current status, the date, and each team's scores.
- **Relationships**: The table links to the VENUES, SEASON, and TEAM tables by foreign keys, making it easier to determine the location of the match, the season it happened in, and the participating teams.

USE PRO_KABADDI_LEAGUE_DATABASE

CREATE TABLE MATCHES(MATCH_ID VARCHAR(30) PRIMARY KEY,

MATCH_DATE DATE,

```
SCORE_TEAM_A INT,

SCORE_TEAM_B INT,

STATUS VARCHAR(20),

VENUE_ID    Varchar(50),

SEASON_ID  Varchar(20),

TEAM_A_ID  Varchar(20),

TEAM_B_ID  Varchar(20),

FOREIGN KEY (VENUE_ID) REFERENCES VENUES(VENUE_ID),

FOREIGN KEY (Season_ID) REFERENCES SEASON(Season_ID),

FOREIGN KEY (Team_A_ID) REFERENCES TEAM(TEAM_ID),

FOREIGN KEY (Team_B_ID) REFERENCES TEAM(TEAM_ID),

);
```



```
USE PRO_KABADDI_LEAGUE_DATABASE
CREATE TABLE MATCHES(MATCH_ID VARCHAR(30) PRIMARY KEY,
MATCH_DATE DATE,
SCORE_TEAM_A INT,
SCORE_TEAM_B INT,
STATUS VARCHAR(20),
VENUE_ID    Varchar(50),
SEASON_ID   Varchar(20),
TEAM_A_ID   Varchar(20),
TEAM_B_ID   Varchar(20),
FOREIGN KEY (VENUE_ID) REFERENCES VENUES(VENUE_ID),
FOREIGN KEY (Season_ID) REFERENCES SEASON(Season_ID),
FOREIGN KEY (Team_A_ID) REFERENCES TEAM(TEAM_ID),
FOREIGN KEY (Team_B_ID) REFERENCES TEAM(TEAM_ID),
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T16:21:33.6522457-06:00
```

Insertion of data in matches table:

The MATCHES database is filled with real match data from these queries, which is essential for monitoring and evaluating Pro Kabaddi League match outcomes. Data entry into the table allows us to create reports and carry out a number of studies, including:

- **Match Results**: Finding out which team won or lost by accessing the scores of finished matches.
- **Venue and Season Information**: Detailed information on the location, time, and season of each match.
- **Team Performance**: Examining various teams' scores to gauge how well they've performed throughout time.

A snapshot of the games played throughout the inaugural season, including the results of different games between different clubs, is provided by the added data.

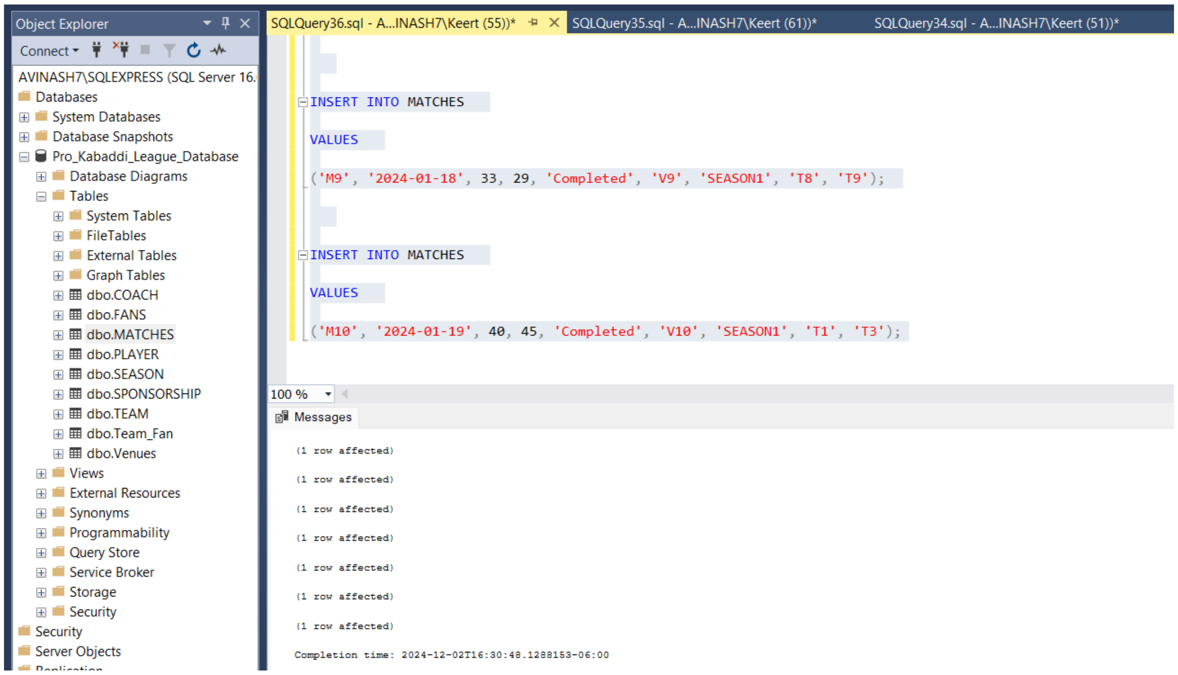
```
INSERT INTO MATCHES

VALUES

('M1', '2024-01-10', 35, 28, 'Completed', 'V1', 'SEASON1', 'T1', 'T2');



INSERT INTO MATCHES

VALUES

('M2', '2024-01-11', 40, 36, 'Completed', 'V2', 'SEASON1', 'T3', 'T4');



INSERT INTO MATCHES

VALUES

('M3', '2024-01-12', 22, 25, 'Completed', 'V3', 'SEASON1', 'T5', 'T6');



INSERT INTO MATCHES

VALUES

('M4', '2024-01-13', 50, 48, 'Completed', 'V4', 'SEASON1', 'T7', 'T8');



INSERT INTO MATCHES
```

```
VALUES

('M5', '2024-01-14', 30, 30, 'Draw', 'V5', 'SEASON1', 'T9', 'T1');


INSERT INTO MATCHES

VALUES

('M6', '2024-01-15', 42, 38, 'Completed', 'V6', 'SEASON1', 'T2', 'T3');


INSERT INTO MATCHES

VALUES

('M7', '2024-01-16', 27, 32, 'Completed', 'V7', 'SEASON1', 'T4', 'T5');


INSERT INTO MATCHES

VALUES

('M8', '2024-01-17', 29, 20, 'Completed', 'V8', 'SEASON1', 'T6', 'T7');


INSERT INTO MATCHES

VALUES

('M9', '2024-01-18', 33, 29, 'Completed', 'V9', 'SEASON1', 'T8', 'T9');
```

INSERT INTO MATCHES

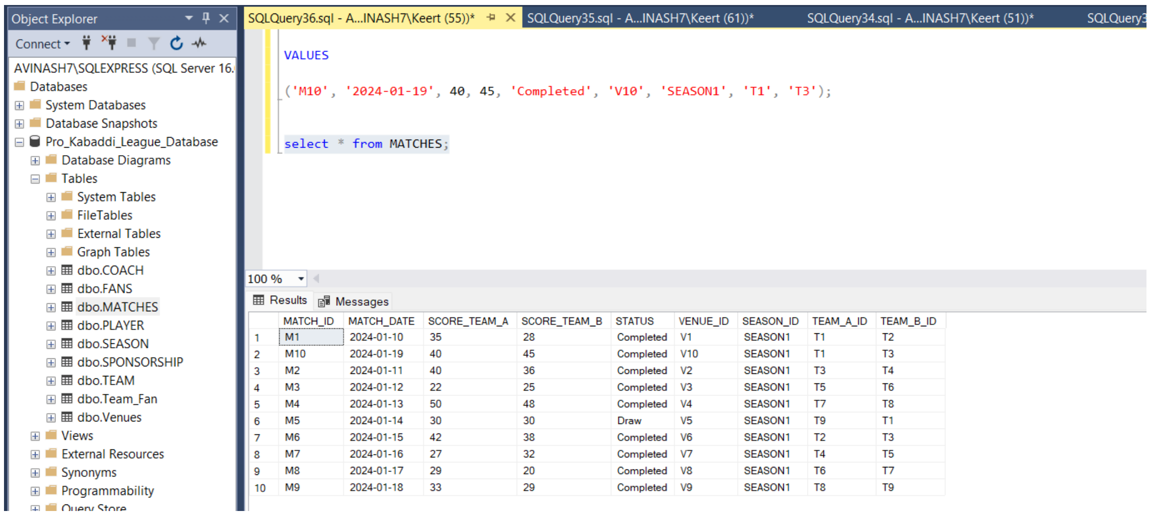
VALUES

('M10', '2024-01-19', 40, 45, 'Completed', 'V10', 'SEASON1', 'T1', 'T3');



```
INSERT INTO MATCHES

VALUES

('M9', '2024-01-18', 33, 29, 'Completed', 'V9', 'SEASON1', 'T8', 'T9');


INSERT INTO MATCHES

VALUES

('M10', '2024-01-19', 40, 45, 'Completed', 'V10', 'SEASON1', 'T1', 'T3');
```

```
(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

Completion time: 2024-12-02T16:30:48.1288153-06:00
```



```
VALUES

('M10', '2024-01-19', 40, 45, 'Completed', 'V10', 'SEASON1', 'T1', 'T3');

select * from MATCHES;
```

	MATCH_ID	MATCH_DATE	SCORE_TEAM_A	SCORE_TEAM_B	STATUS	VENUE_ID	SEASON_ID	TEAM_A_ID	TEAM_B_ID
1	M1	2024-01-10	35	28	Completed	V1	SEASON1	T1	T2
2	M10	2024-01-19	40	45	Completed	V10	SEASON1	T1	T3
3	M2	2024-01-11	40	36	Completed	V2	SEASON1	T3	T4
4	M3	2024-01-12	22	25	Completed	V3	SEASON1	T5	T6
5	M4	2024-01-13	50	48	Completed	V4	SEASON1	T7	T8
6	M5	2024-01-14	30	30	Draw	V5	SEASON1	T9	T1
7	M6	2024-01-15	42	38	Completed	V6	SEASON1	T2	T3
8	M7	2024-01-16	27	32	Completed	V7	SEASON1	T4	T5
9	M8	2024-01-17	29	20	Completed	V8	SEASON1	T6	T7
10	M9	2024-01-18	33	29	Completed	V9	SEASON1	T8	T9

Creation of Tickets table:

In-depth data on the tickets sold for every Pro Kabaddi League game is kept in the TICKETS table. There are several uses for this data:

- **Tracking Ticket Sales**: Keeping tabs on how many tickets are sold for each game, as well as their cost.
- **Match Attendance**: Finding out who bought tickets and how many were sold for a specific match.
- **Financial Analysis**: Figuring out how much money is made from ticket sales for every game or season.
- **Seat Allocation**: Linking particular seats to ticket purchases in order to guarantee that every seat is appropriately allotted for the game.

You may effectively track purchasers, manage ticket sales, and combine ticket data with other match-related database data by establishing this table.

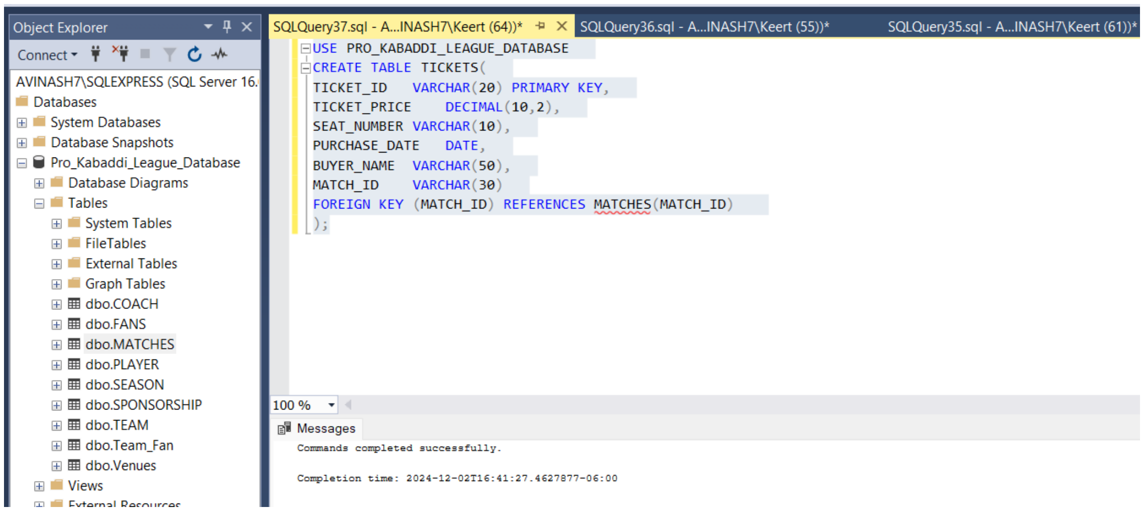
```
USE PRO_KABADDI_LEAGUE_DATABASE

CREATE TABLE TICKETS(

TICKET_ID    VARCHAR(20) PRIMARY KEY,

TICKET_PRICE        DECIMAL(10,2),

SEAT_NUMBER        VARCHAR(10),

PURCHASE_DATE   DATE,
```

BUYER_NAME   VARCHAR(50),

MATCH_ID   VARCHAR(30)

FOREIGN KEY (MATCH_ID) REFERENCES MATCHES(MATCH_ID)

);



```
USE PRO_KABADDI_LEAGUE_DATABASE
CREATE TABLE TICKETS(
TICKET_ID    VARCHAR(20) PRIMARY KEY,
TICKET_PRICE     DECIMAL(10,2),
SEAT_NUMBER VARCHAR(10),
PURCHASE_DATE    DATE,
BUYER_NAME   VARCHAR(50),
MATCH_ID     VARCHAR(30)
FOREIGN KEY (MATCH_ID) REFERENCES MATCHES(MATCH_ID)
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T16:41:27.4627877-06:00
```

Insertion of data in tickets table:

The TICKETS database now has many rows of ticket sale data thanks to this query. This information is being added in order to fill the table with example records that reflect ticket purchases for different Pro Kabaddi League games.

Each record contains:

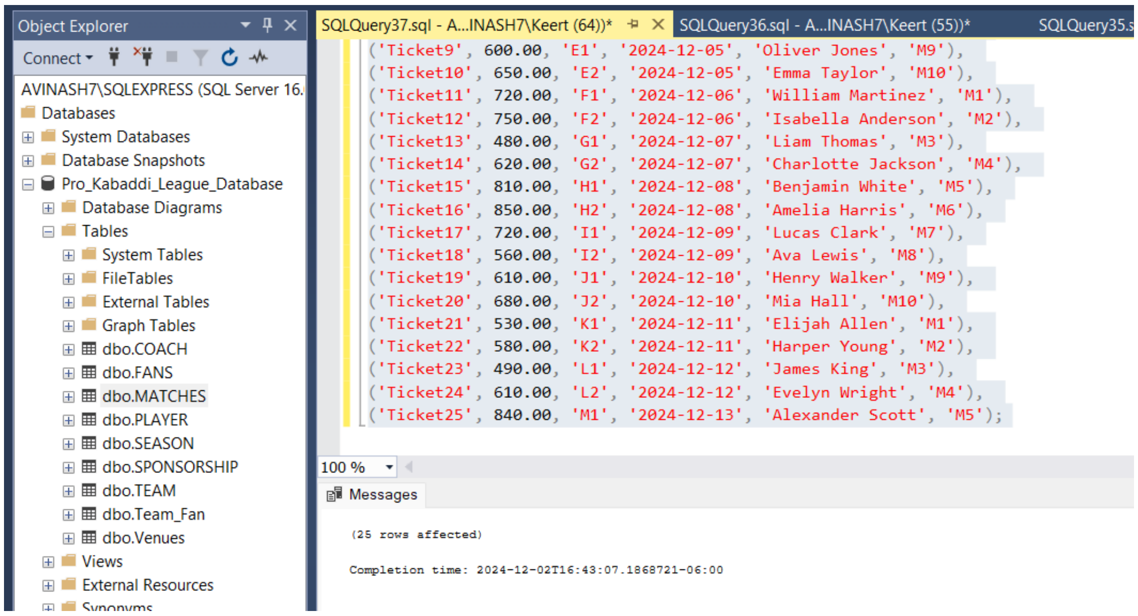
- Details about the buyer's information, seating configurations, and ticket costs.
- To make sure the tickets are connected to the right game, the MATCH_ID field associates each ticket with a particular match.

The inserted data can then be used for:

- examining each match's ticket sales.
- keeping track of each match's ticket sales earnings.
- creating reports on match attendance and seat occupancy.

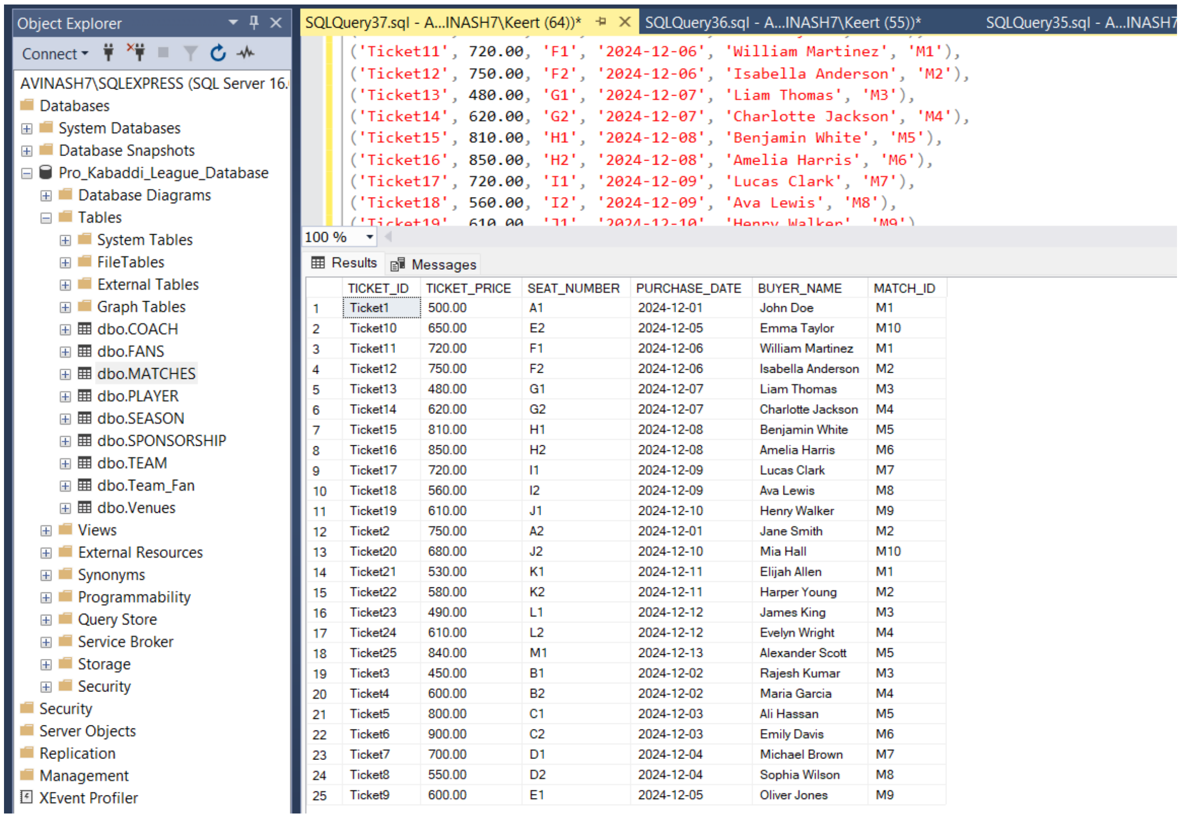
By adding this information, the database is enhanced with crucial details for the Pro Kabaddi League's effective ticket administration and sales analysis.

```
INSERT INTO TICKETS

VALUES

('Ticket1', 500.00, 'A1', '2024-12-01', 'John Doe', 'M1'),

('Ticket2', 750.00, 'A2', '2024-12-01', 'Jane Smith', 'M2'),

('Ticket3', 450.00, 'B1', '2024-12-02', 'Rajesh Kumar', 'M3'),

('Ticket4', 600.00, 'B2', '2024-12-02', 'Maria Garcia', 'M4'),

('Ticket5', 800.00, 'C1', '2024-12-03', 'Ali Hassan', 'M5'),

('Ticket6', 900.00, 'C2', '2024-12-03', 'Emily Davis', 'M6'),

('Ticket7', 700.00, 'D1', '2024-12-04', 'Michael Brown', 'M7'),

('Ticket8', 550.00, 'D2', '2024-12-04', 'Sophia Wilson', 'M8'),

('Ticket9', 600.00, 'E1', '2024-12-05', 'Oliver Jones', 'M9'),
('Ticket10', 650.00, 'E2', '2024-12-05', 'Emma Taylor', 'M10'),

('Ticket11', 720.00, 'F1', '2024-12-06', 'William Martinez', 'M1'),

('Ticket12', 750.00, 'F2', '2024-12-06', 'Isabella Anderson', 'M2'),

('Ticket13', 480.00, 'G1', '2024-12-07', 'Liam Thomas', 'M3'),

('Ticket14', 620.00, 'G2', '2024-12-07', 'Charlotte Jackson', 'M4'),

('Ticket15', 810.00, 'H1', '2024-12-08', 'Benjamin White', 'M5'),

('Ticket16', 850.00, 'H2', '2024-12-08', 'Amelia Harris', 'M6'),

('Ticket17', 720.00, 'I1', '2024-12-09', 'Lucas Clark', 'M7'),

('Ticket18', 560.00, 'I2', '2024-12-09', 'Ava Lewis', 'M8'),

('Ticket19', 610.00, 'J1', '2024-12-10', 'Henry Walker', 'M9'),

('Ticket20', 680.00, 'J2', '2024-12-10', 'Mia Hall', 'M10'),

('Ticket21', 530.00, 'K1', '2024-12-11', 'Elijah Allen', 'M1'),

('Ticket22', 580.00, 'K2', '2024-12-11', 'Harper Young', 'M2'),

('Ticket23', 490.00, 'L1', '2024-12-12', 'James King', 'M3'),

('Ticket24', 610.00, 'L2', '2024-12-12', 'Evelyn Wright', 'M4'),

('Ticket25', 840.00, 'M1', '2024-12-13', 'Alexander Scott', 'M5');
```



```
('Ticket9', 600.00, 'E1', '2024-12-05', 'Oliver Jones', 'M9'),
('Ticket10', 650.00, 'E2', '2024-12-05', 'Emma Taylor', 'M10'),
('Ticket11', 720.00, 'F1', '2024-12-06', 'William Martinez', 'M1'),
('Ticket12', 750.00, 'F2', '2024-12-06', 'Isabella Anderson', 'M2'),
('Ticket13', 480.00, 'G1', '2024-12-07', 'Liam Thomas', 'M3'),
('Ticket14', 620.00, 'G2', '2024-12-07', 'Charlotte Jackson', 'M4'),
('Ticket15', 810.00, 'H1', '2024-12-08', 'Benjamin White', 'M5'),
('Ticket16', 850.00, 'H2', '2024-12-08', 'Amelia Harris', 'M6'),
('Ticket17', 720.00, 'I1', '2024-12-09', 'Lucas Clark', 'M7'),
('Ticket18', 560.00, 'I2', '2024-12-09', 'Ava Lewis', 'M8'),
('Ticket19', 610.00, 'J1', '2024-12-10', 'Henry Walker', 'M9'),
('Ticket20', 680.00, 'J2', '2024-12-10', 'Mia Hall', 'M10'),
('Ticket21', 530.00, 'K1', '2024-12-11', 'Elijah Allen', 'M1'),
('Ticket22', 580.00, 'K2', '2024-12-11', 'Harper Young', 'M2'),
('Ticket23', 490.00, 'L1', '2024-12-12', 'James King', 'M3'),
('Ticket24', 610.00, 'L2', '2024-12-12', 'Evelyn Wright', 'M4'),
('Ticket25', 840.00, 'M1', '2024-12-13', 'Alexander Scott', 'M5');
```

```
(25 rows affected)

Completion time: 2024-12-02T16:43:07.1868721-06:00
```



```
('Ticket11', 720.00, 'F1', '2024-12-06', 'William Martinez', 'M1'),
('Ticket12', 750.00, 'F2', '2024-12-06', 'Isabella Anderson', 'M2'),
('Ticket13', 480.00, 'G1', '2024-12-07', 'Liam Thomas', 'M3'),
('Ticket14', 620.00, 'G2', '2024-12-07', 'Charlotte Jackson', 'M4'),
('Ticket15', 810.00, 'H1', '2024-12-08', 'Benjamin White', 'M5'),
('Ticket16', 850.00, 'H2', '2024-12-08', 'Amelia Harris', 'M6'),
('Ticket17', 720.00, 'I1', '2024-12-09', 'Lucas Clark', 'M7'),
('Ticket18', 560.00, 'I2', '2024-12-09', 'Ava Lewis', 'M8'),
('Ticket19', 610.00, 'J1', '2024-12-10', 'Henry Walker', 'M9'),
```

	TICKET_ID	TICKET_PRICE	SEAT_NUMBER	PURCHASE_DATE	BUYER_NAME	MATCH_ID
1	Ticket1	500.00	A1	2024-12-01	John Doe	M1
2	Ticket10	650.00	E2	2024-12-05	Emma Taylor	M10
3	Ticket11	720.00	F1	2024-12-06	William Martinez	M1
4	Ticket12	750.00	F2	2024-12-06	Isabella Anderson	M2
5	Ticket13	480.00	G1	2024-12-07	Liam Thomas	M3
6	Ticket14	620.00	G2	2024-12-07	Charlotte Jackson	M4
7	Ticket15	810.00	H1	2024-12-08	Benjamin White	M5
8	Ticket16	850.00	H2	2024-12-08	Amelia Harris	M6
9	Ticket17	720.00	I1	2024-12-09	Lucas Clark	M7
10	Ticket18	560.00	I2	2024-12-09	Ava Lewis	M8
11	Ticket19	610.00	J1	2024-12-10	Henry Walker	M9
12	Ticket2	750.00	A2	2024-12-01	Jane Smith	M2
13	Ticket20	680.00	J2	2024-12-10	Mia Hall	M10
14	Ticket21	530.00	K1	2024-12-11	Elijah Allen	M1
15	Ticket22	580.00	K2	2024-12-11	Harper Young	M2
16	Ticket23	490.00	L1	2024-12-12	James King	M3
17	Ticket24	610.00	L2	2024-12-12	Evelyn Wright	M4
18	Ticket25	840.00	M1	2024-12-13	Alexander Scott	M5
19	Ticket3	450.00	B1	2024-12-02	Rajesh Kumar	M3
20	Ticket4	600.00	B2	2024-12-02	Maria Garcia	M4
21	Ticket5	800.00	C1	2024-12-03	Ali Hassan	M5
22	Ticket6	900.00	C2	2024-12-03	Emily Davis	M6
23	Ticket7	700.00	D1	2024-12-04	Michael Brown	M7
24	Ticket8	550.00	D2	2024-12-04	Sophia Wilson	M8
25	Ticket9	600.00	E1	2024-12-05	Oliver Jones	M9

Creation of STATISTICS_S table:

Players' complete performance data from each Pro Kabaddi League match is supposed to be included in the STATISTICS_S table. This table, which offers details on individual contributions, may be used by teams, league organizers, and analysts to track player performance over the course of several games.

The stored data can be used for:

- **Player Performance Analysis**: Coaches and analysts can assess a player's strengths and weaknesses by looking at metrics like points scored, tackles, and defensive points.
- **Match Reporting**: The table makes it possible to provide thorough reports on each match's player performance.
- **Statistical Summaries**: Overall performance patterns, such as average points scored or tackles made throughout a season, can be displayed by aggregating the data.

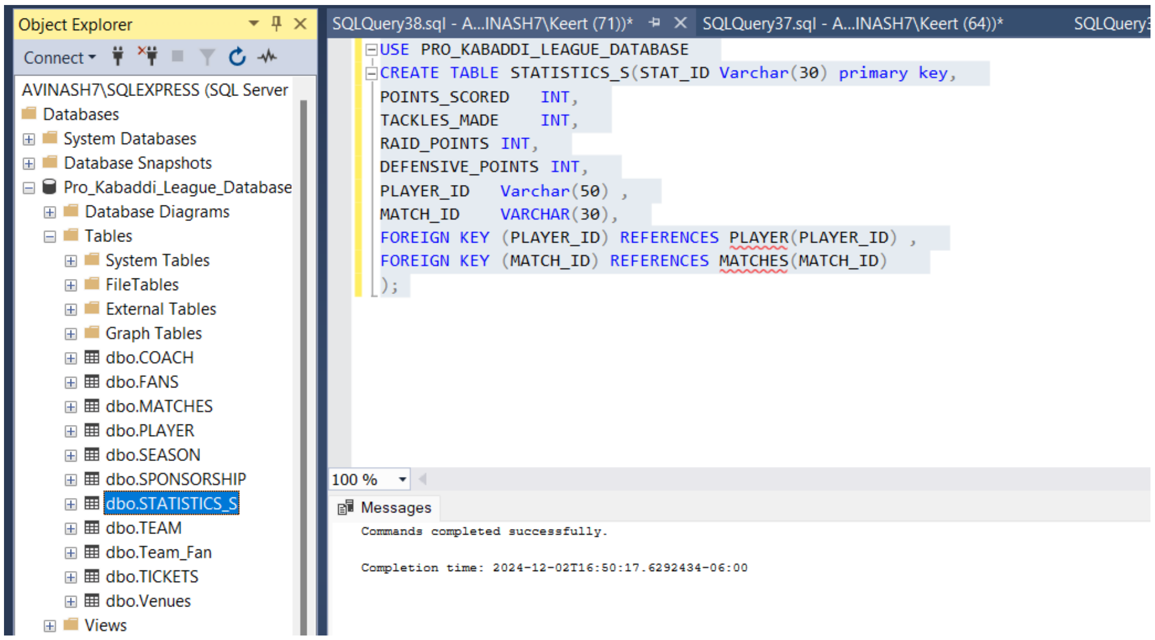
```
USE PRO_KABADDI_LEAGUE_DATABASE

CREATE TABLE STATISTICS_S(STAT_ID Varchar(30) primary key,

POINTS_SCORED    INT,

TACKLES_MADE     INT,

RAID_POINTS      INT,

DEFENSIVE_POINTS INT,

PLAYER_ID   Varchar(50) ,
```

MATCH_ID   VARCHAR(30),

FOREIGN KEY (PLAYER_ID) REFERENCES PLAYER(PLAYER_ID) ,

FOREIGN KEY (MATCH_ID) REFERENCES MATCHES(MATCH_ID)

);



```
USE PRO_KABADDI_LEAGUE_DATABASE
CREATE TABLE STATISTICS_S(STAT_ID Varchar(30) primary key,
POINTS_SCORED    INT,
TACKLES_MADE     INT,
RAID_POINTS INT,
DEFENSIVE_POINTS INT,
PLAYER_ID     Varchar(50) ,
MATCH_ID      VARCHAR(30),
FOREIGN KEY (PLAYER_ID) REFERENCES PLAYER(PLAYER_ID) ,
FOREIGN KEY (MATCH_ID) REFERENCES MATCHES(MATCH_ID)
);
```

```
Commands completed successfully.

Completion time: 2024-12-02T16:50:17.6292434-06:00
```

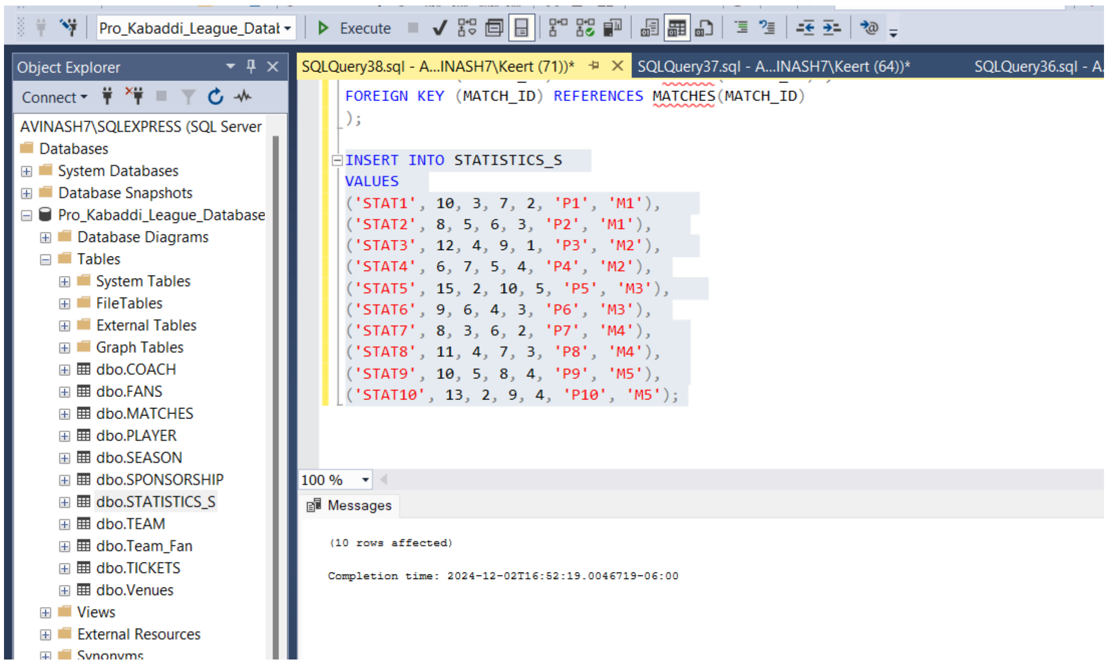
Insertion Data into STATISTICS_S table :

Recording Player Performance: TThis query records each match's points scored, tackles made, raid points, and defensive points by inserting player performance data into the STATISTICS_S table.

Linking Data to Matches and Players: To guarantee data consistency across database tables, the PLAYER_ID and MATCH_ID entries associate these statistics with particular players and matches.

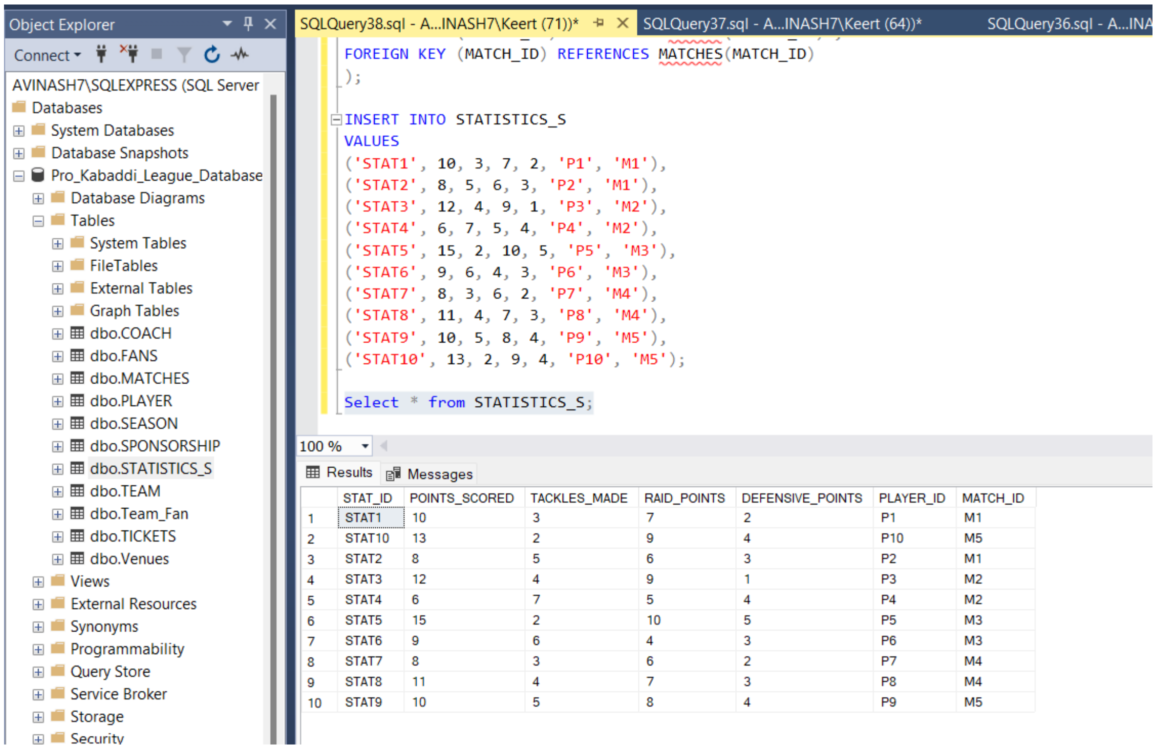
Facilitating Match Analysis: This query adds to the overall study of each player's performance by saving player statistics, which may be used to evaluate each player's unique contributions to the game and season.

```
INSERT INTO STATISTICS_S

VALUES

('STAT1', 10, 3, 7, 2, 'P1', 'M1'),

('STAT2', 8, 5, 6, 3, 'P2', 'M1'),

('STAT3', 12, 4, 9, 1, 'P3', 'M2'),

('STAT4', 6, 7, 5, 4, 'P4', 'M2'),

('STAT5', 15, 2, 10, 5, 'P5', 'M3'),

('STAT6', 9, 6, 4, 3, 'P6', 'M3'),

('STAT7', 8, 3, 6, 2, 'P7', 'M4'),

('STAT8', 11, 4, 7, 3, 'P8', 'M4'),

('STAT9', 10, 5, 8, 4, 'P9', 'M5'),

('STAT10', 13, 2, 9, 4, 'P10', 'M5');
```



```
FOREIGN KEY (MATCH_ID) REFERENCES MATCHES(MATCH_ID)
);

INSERT INTO STATISTICS_S
VALUES
('STAT1', 10, 3, 7, 2, 'P1', 'M1'),
('STAT2', 8, 5, 6, 3, 'P2', 'M1'),
('STAT3', 12, 4, 9, 1, 'P3', 'M2'),
('STAT4', 6, 7, 5, 4, 'P4', 'M2'),
('STAT5', 15, 2, 10, 5, 'P5', 'M3'),
('STAT6', 9, 6, 4, 3, 'P6', 'M3'),
('STAT7', 8, 3, 6, 2, 'P7', 'M4'),
('STAT8', 11, 4, 7, 3, 'P8', 'M4'),
('STAT9', 10, 5, 8, 4, 'P9', 'M5'),
('STAT10', 13, 2, 9, 4, 'P10', 'M5');
```

```
(10 rows affected)

Completion time: 2024-12-02T16:52:19.0046719-06:00
```



```
FOREIGN KEY (MATCH_ID) REFERENCES MATCHES(MATCH_ID)
);

INSERT INTO STATISTICS_S
VALUES
('STAT1', 10, 3, 7, 2, 'P1', 'M1'),
('STAT2', 8, 5, 6, 3, 'P2', 'M1'),
('STAT3', 12, 4, 9, 1, 'P3', 'M2'),
('STAT4', 6, 7, 5, 4, 'P4', 'M2'),
('STAT5', 15, 2, 10, 5, 'P5', 'M3'),
('STAT6', 9, 6, 4, 3, 'P6', 'M3'),
('STAT7', 8, 3, 6, 2, 'P7', 'M4'),
('STAT8', 11, 4, 7, 3, 'P8', 'M4'),
('STAT9', 10, 5, 8, 4, 'P9', 'M5'),
('STAT10', 13, 2, 9, 4, 'P10', 'M5');

Select * from STATISTICS_S;
```

	STAT_ID	POINTS_SCORED	TACKLES_MADE	RAID_POINTS	DEFENSIVE_POINTS	PLAYER_ID	MATCH_ID
1	STAT1	10	3	7	2	P1	M1
2	STAT10	13	2	9	4	P10	M5
3	STAT2	8	5	6	3	P2	M1
4	STAT3	12	4	9	1	P3	M2
5	STAT4	6	7	5	4	P4	M2
6	STAT5	15	2	10	5	P5	M3
7	STAT6	9	6	4	3	P6	M3
8	STAT7	8	3	6	2	P7	M4
9	STAT8	11	4	7	3	P8	M4
10	STAT9	10	5	8	4	P9	M5

SQL Select statements

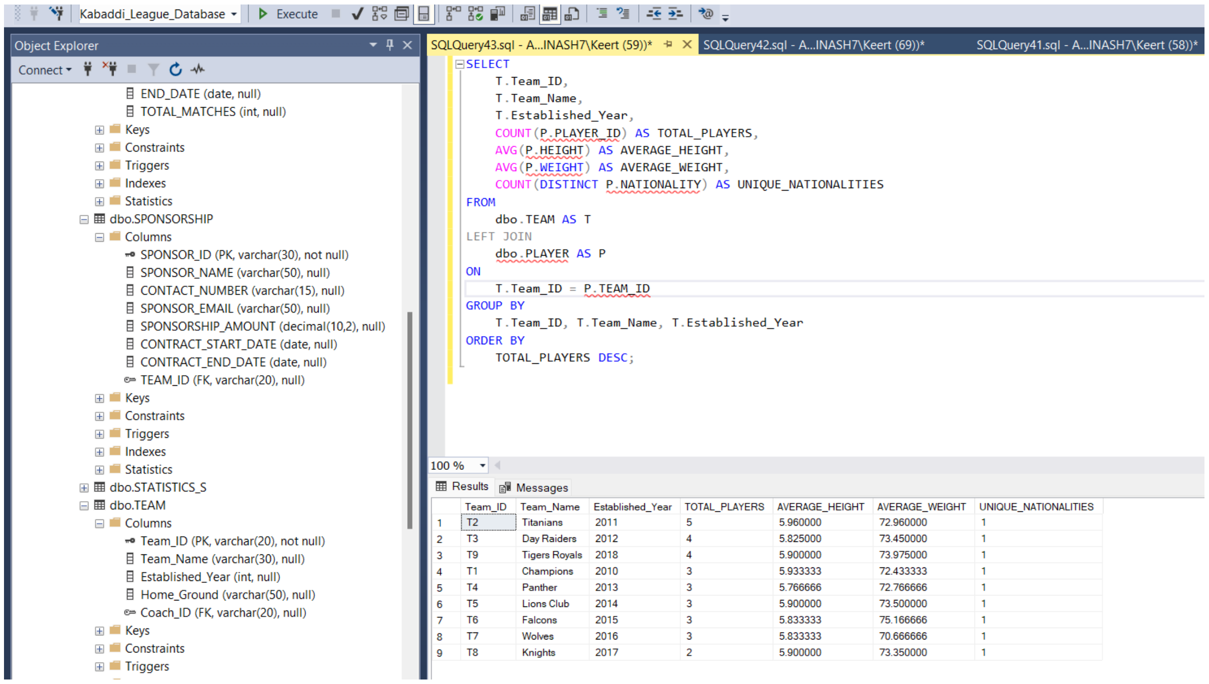
1. **Team Overview with Player Statistics:**

This query also provides a summary of each team with its ID, name, and year of formation in addition to the aggregated player data. It sets the average height and weight of each squad, how many players overall, how many different nationalities. The results — which consist of team diversity and lineup composition data — are listed in descending order by total participants.

```
SELECT
    T.Team_ID,
    T.Team_Name,
    T.Established_Year,
    COUNT(P.PLAYER_ID) AS TOTAL_PLAYERS,
    AVG(P.HEIGHT) AS AVERAGE_HEIGHT,
    AVG(P.WEIGHT) AS AVERAGE_WEIGHT,
    COUNT(DISTINCT P.NATIONALITY) AS UNIQUE_NATIONALITIES
 FROM
    dbo.TEAM AS T
 LEFT JOIN
    dbo.PLAYER AS P
 ON
    T.Team_ID = P.TEAM_ID
 GROUP BY
    T.Team_ID, T.Team_Name, T.Established_Year
```

ORDER BY

TOTAL_PLAYERS DESC;


Kabaddi_League_Database

Object Explorer

```
        END_DATE (date, null)
        TOTAL_MATCHES (int, null)
      Keys
      Constraints
      Triggers
      Indexes
      Statistics
    dbo.SPONSORSHIP
      Columns
        SPONSOR_ID (PK, varchar(30), not null)
        SPONSOR_NAME (varchar(50), null)
        CONTACT_NUMBER (varchar(15), null)
        SPONSOR_EMAIL (varchar(50), null)
        SPONSORSHIP_AMOUNT (decimal(10,2), null)
        CONTRACT_START_DATE (date, null)
        CONTRACT_END_DATE (date, null)
        TEAM_ID (FK, varchar(20), null)
      Keys
      Constraints
      Triggers
      Indexes
      Statistics
    dbo.STATISTICS_S
    dbo.TEAM
      Columns
        Team_ID (PK, varchar(20), not null)
        Team_Name (varchar(30), null)
        Established_Year (int, null)
        Home_Ground (varchar(50), null)
        Coach_ID (FK, varchar(20), null)
      Keys
      Constraints
      Triggers
```

SQLQuery43.sql - A...INASH7\Keert (59))*

```sql
SELECT
    T.Team_ID,
    T.Team_Name,
    T.Established_Year,
    COUNT(P.PLAYER_ID) AS TOTAL_PLAYERS,
    AVG(P.HEIGHT) AS AVERAGE_HEIGHT,
    AVG(P.WEIGHT) AS AVERAGE_WEIGHT,
    COUNT(DISTINCT P.NATIONALITY) AS UNIQUE_NATIONALITIES
FROM
    dbo.TEAM AS T
LEFT JOIN
    dbo.PLAYER AS P
ON
    T.Team_ID = P.TEAM_ID
GROUP BY
    T.Team_ID, T.Team_Name, T.Established_Year
ORDER BY
    TOTAL_PLAYERS DESC;
```

	Team_ID	Team_Name	Established_Year	TOTAL_PLAYERS	AVERAGE_HEIGHT	AVERAGE_WEIGHT	UNIQUE_NATIONALITIES
1	T2	Titanians	2011	5	5.960000	72.960000	1
2	T3	Day Raiders	2012	4	5.825000	73.450000	1
3	T9	Tigers Royals	2018	4	5.900000	73.975000	1
4	T1	Champions	2010	3	5.933333	72.433333	1
5	T4	Panther	2013	3	5.766666	72.766666	1
6	T5	Lions Club	2014	3	5.900000	73.500000	1
7	T6	Falcons	2015	3	5.833333	75.166666	1
8	T7	Wolves	2016	3	5.833333	70.666666	1
9	T8	Knights	2017	2	5.900000	73.350000	1

2. **Top 5 Players Based on Total Points Scored:**

This query retrieves comprehensive performance data for the top five players, in descending order of the total points accrued. It includes each player's name, unique identification number, playing position, country of origin, number of games played, average points, total points, number of tackles, raid points, and defensive points. Provides details about each player's unique impact on the game categorized by player information.

SELECT TOP 5

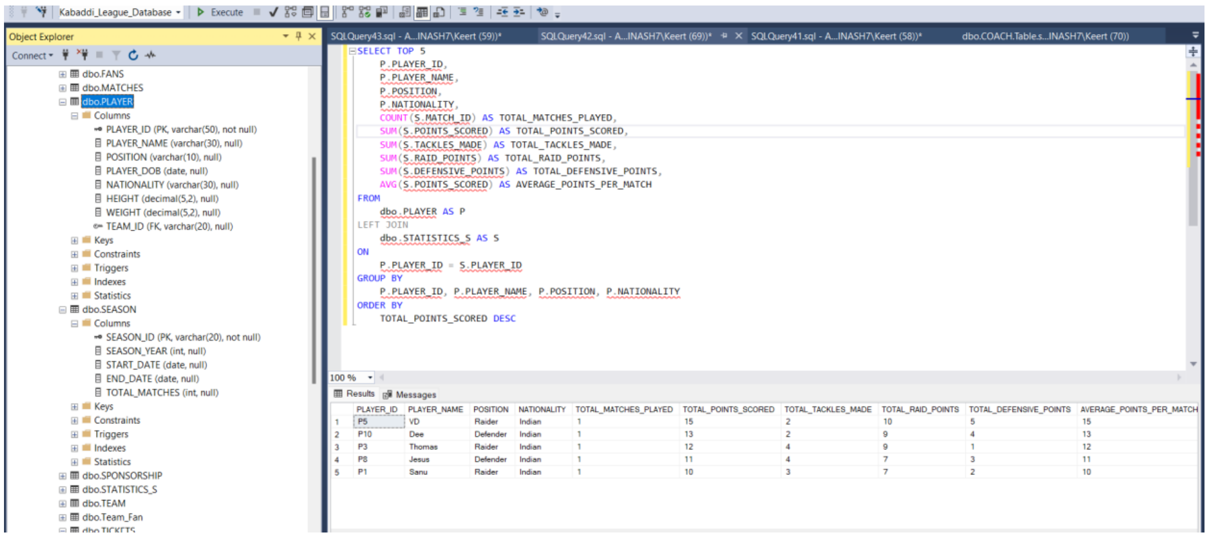
P.PLAYER_ID,

P.PLAYER_NAME,

P.POSITION,

P.NATIONALITY,

```
    COUNT(S.MATCH_ID) AS TOTAL_MATCHES_PLAYED,
    SUM(S.POINTS_SCORED) AS TOTAL_POINTS_SCORED,
    SUM(S.TACKLES_MADE) AS TOTAL_TACKLES_MADE,
    SUM(S.RAID_POINTS) AS TOTAL_RAID_POINTS,
    SUM(S.DEFENSIVE_POINTS) AS TOTAL_DEFENSIVE_POINTS,
    AVG(S.POINTS_SCORED) AS AVERAGE_POINTS_PER_MATCH
FROM
    dbo.PLAYER AS P
LEFT JOIN
    dbo.STATISTICS_S AS S
ON
    P.PLAYER_ID = S.PLAYER_ID
GROUP BY
    P.PLAYER_ID, P.PLAYER_NAME, P.POSITION, P.NATIONALITY
ORDER BY
    TOTAL_POINTS_SCORED DESC
```



```
Object Explorer
  dbo.FANS
  dbo.MATCHES
  dbo.PLAYER
    Columns
      PLAYER_ID (PK, varchar(50), not null)
      PLAYER_NAME (varchar(30), null)
      POSITION (varchar(10), null)
      PLAYER_DOB (date, null)
      NATIONALITY (varchar(30), null)
      HEIGHT (decimal(5,2), null)
      WEIGHT (decimal(5,2), null)
      TEAM_ID (FK, varchar(20), null)
    Keys
    Constraints
    Triggers
    Indexes
    Statistics
  dbo.SEASON
    Columns
      SEASON_ID (PK, varchar(20), not null)
      SEASON_YEAR (int, null)
      START_DATE (date, null)
      END_DATE (date, null)
      TOTAL_MATCHES (int, null)
    Keys
    Constraints
    Triggers
    Indexes
    Statistics
  dbo.SPONSORSHIP
  dbo.STATISTICS_S
  dbo.TEAM
  dbo.Team_Fan
```

```
SELECT TOP 5
    P.PLAYER_ID,
    P.PLAYER_NAME,
    P.POSITION,
    P.NATIONALITY,
    COUNT(S.MATCH_ID) AS TOTAL_MATCHES_PLAYED,
    SUM(S.POINTS_SCORED) AS TOTAL_POINTS_SCORED,
    SUM(S.TACKLES_MADE) AS TOTAL_TACKLES_MADE,
    SUM(S.RAID_POINTS) AS TOTAL_RAID_POINTS,
    SUM(S.DEFENSIVE_POINTS) AS TOTAL_DEFENSIVE_POINTS,
    AVG(S.POINTS_SCORED) AS AVERAGE_POINTS_PER_MATCH
FROM
    dbo.PLAYER AS P
LEFT JOIN
    dbo.STATISTICS_S AS S
ON
    P.PLAYER_ID = S.PLAYER_ID
GROUP BY
    P.PLAYER_ID, P.PLAYER_NAME, P.POSITION, P.NATIONALITY
ORDER BY
    TOTAL_POINTS_SCORED DESC
```

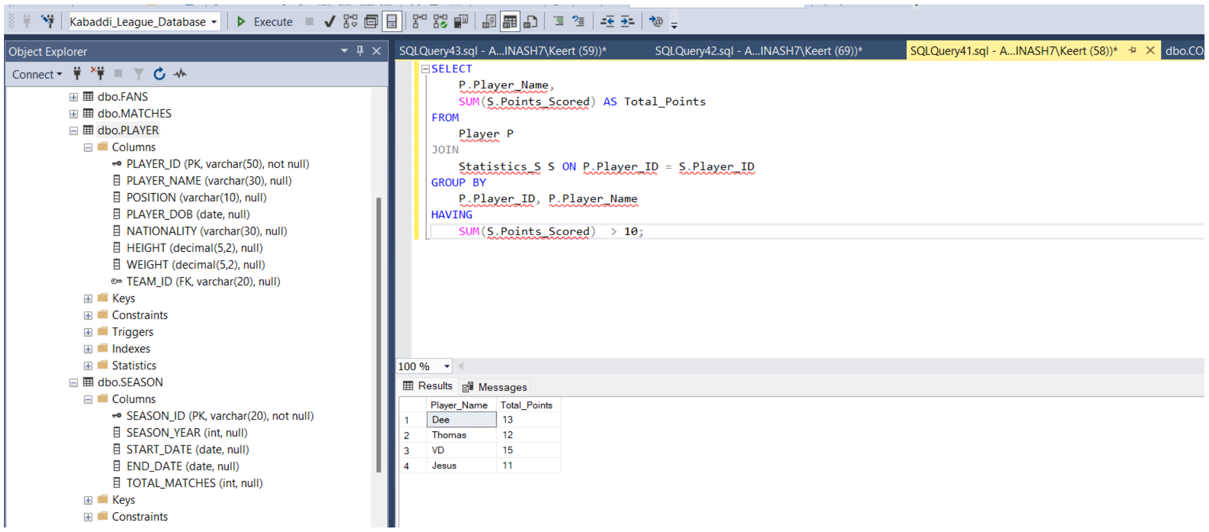
	PLAYER_ID	PLAYER_NAME	POSITION	NATIONALITY	TOTAL_MATCHES_PLAYED	TOTAL_POINTS_SCORED	TOTAL_TACKLES_MADE	TOTAL_RAID_POINTS	TOTAL_DEFENSIVE_POINTS	AVERAGE_POINTS_PER_MATCH
1	P5	VD	Raider	Indian	1	15	2	10	5	15
2	P10	Dee	Defender	Indian	1	13	2	9	4	13
3	P3	Thomas	Raider	Indian	1	12	4	9	1	12
4	P8	Jesus	Defender	Indian	1	11	4	7	3	11
5	P1	Sanu	Raider	Indian	1	10	3	7	2	10

3. **Players Scoring More Than 10 Points:**

This query finds players that have scored more than 10 points overall throughout all games. It includes their names and total points earned, sorted based on player data. Between overall scoring contributions, this gives you an idea of repeatable players.

```
SELECT
    P.Player_Name,
    SUM(S.Points_Scored) AS Total_Points
 FROM
    Player P
 JOIN
    Statistics_S S ON P.Player_ID = S.Player_ID
 GROUP BY
    P.Player_ID, P.Player_Name
 HAVING
```

SUM(S.Points_Scored) > 10;


```
dbo.FANS
dbo.MATCHES
dbo.PLAYER
  Columns
    PLAYER_ID (PK, varchar(50), not null)
    PLAYER_NAME (varchar(30), null)
    POSITION (varchar(10), null)
    PLAYER_DOB (date, null)
    NATIONALITY (varchar(30), null)
    HEIGHT (decimal(5,2), null)
    WEIGHT (decimal(5,2), null)
    TEAM_ID (FK, varchar(20), null)
  Keys
  Constraints
  Triggers
  Indexes
  Statistics
dbo.SEASON
  Columns
    SEASON_ID (PK, varchar(20), not null)
    SEASON_YEAR (int, null)
    START_DATE (date, null)
    END_DATE (date, null)
    TOTAL_MATCHES (int, null)
  Keys
  Constraints
```

```sql
SELECT
    P.Player_Name,
    SUM(S.Points_Scored) AS Total_Points
FROM
    Player P
JOIN
    Statistics_S S ON P.Player_ID = S.Player_ID
GROUP BY
    P.Player_ID, P.Player_Name
HAVING
    SUM(S.Points_Scored)  > 10;
```

	Player_Name	Total_Points
1	Dee	13
2	Thomas	12
3	VD	15
4	Jesus	11

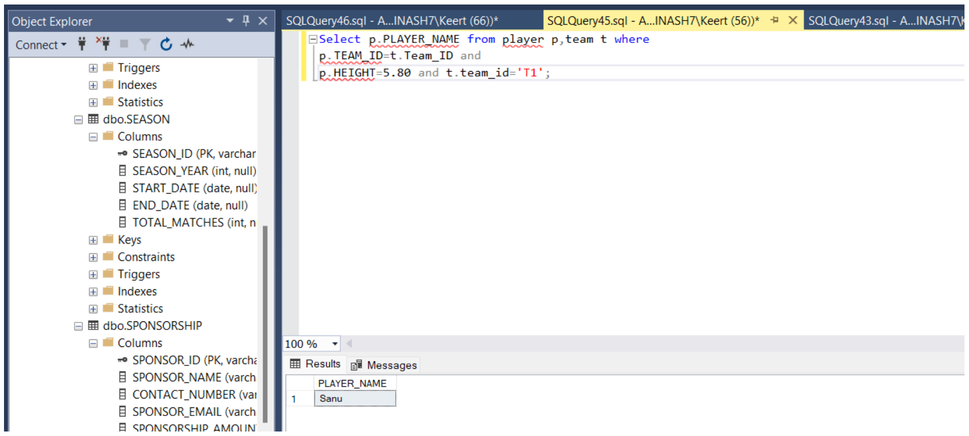
4. **Players with Specific Height in Team T1:**

This is a query that retrieves the players' names on team T1 with a height of 5.80. It uses a join on TEAM_ID to filter players by height and determine the teams they belong to based on this particular data set.

Select p.PLAYER_NAME from player p,team t where

p.TEAM_ID=t.Team_ID and

p.HEIGHT=5.80 and t.team_id='T1';


```
  Triggers
  Indexes
  Statistics
dbo.SEASON
  Columns
    SEASON_ID (PK, varchar
    SEASON_YEAR (int, null)
    START_DATE (date, null)
    END_DATE (date, null)
    TOTAL_MATCHES (int, n
  Keys
  Constraints
  Triggers
  Indexes
  Statistics
dbo.SPONSORSHIP
  Columns
    SPONSOR_ID (PK, varcha
    SPONSOR_NAME (varch
    CONTACT_NUMBER (var
    SPONSOR_EMAIL (varch
    SPONSORSHIP_AMOUN
```

```sql
Select p.PLAYER_NAME from player p,team t where
p.TEAM_ID=t.Team_ID and
p.HEIGHT=5.80 and t.team_id='T1';
```

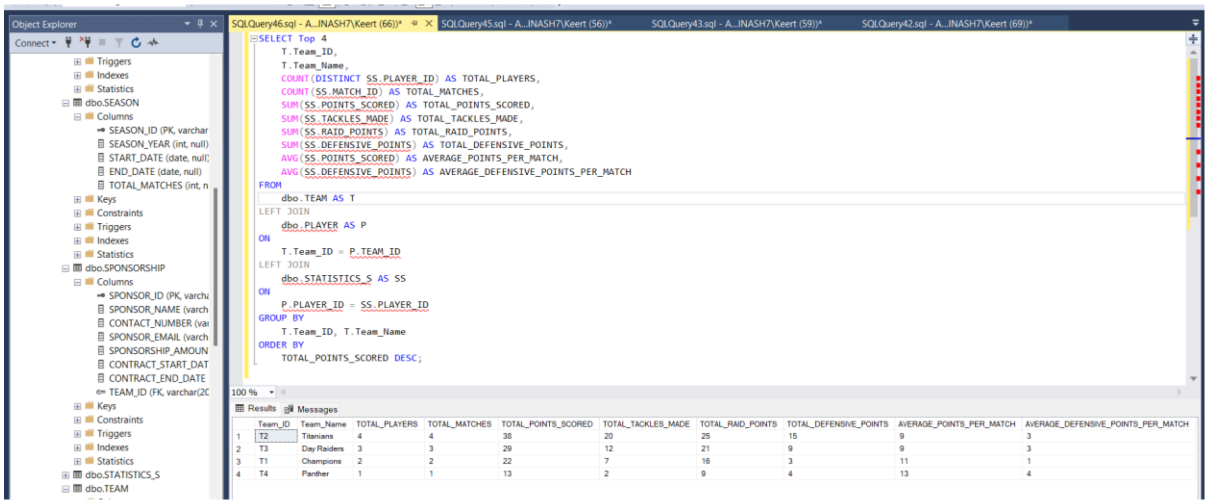
	PLAYER_NAME
1	Sanu

5. **Top 4 Teams Based on Total Points Scored:**

Here is one way you could retrieve performance stats for the top 4 teams in terms of total points scored, in order. It has team ID, team name, total players, total matches, total points, tackles, points scored in raids, points scored in defence and average points and average points in defense per match These results provide a comparative view of how teams fared each game — both in terms of scoring and defense.

```
SELECT Top 4
    T.Team_ID,
    T.Team_Name,
    COUNT(DISTINCT SS.PLAYER_ID) AS TOTAL_PLAYERS,
    COUNT(SS.MATCH_ID) AS TOTAL_MATCHES,
    SUM(SS.POINTS_SCORED) AS TOTAL_POINTS_SCORED,
    SUM(SS.TACKLES_MADE) AS TOTAL_TACKLES_MADE,
    SUM(SS.RAID_POINTS) AS TOTAL_RAID_POINTS,
    SUM(SS.DEFENSIVE_POINTS) AS TOTAL_DEFENSIVE_POINTS,
    AVG(SS.POINTS_SCORED) AS AVERAGE_POINTS_PER_MATCH,
    AVG(SS.DEFENSIVE_POINTS) AS
AVERAGE_DEFENSIVE_POINTS_PER_MATCH
 FROM
    dbo.TEAM AS T
 LEFT JOIN
    dbo.PLAYER AS P
 ON
```

```
    T.Team_ID = P.TEAM_ID
LEFT JOIN
    dbo.STATISTICS_S AS SS
ON
    P.PLAYER_ID = SS.PLAYER_ID
GROUP BY
    T.Team_ID, T.Team_Name
ORDER BY
    TOTAL_POINTS_SCORED DESC;
```



```
SELECT Top 4
    T.Team_ID,
    T.Team_Name,
    COUNT(DISTINCT SS.PLAYER_ID) AS TOTAL_PLAYERS,
    COUNT(SS.MATCH_ID) AS TOTAL_MATCHES,
    SUM(SS.POINTS_SCORED) AS TOTAL_POINTS_SCORED,
    SUM(SS.TACKLES_MADE) AS TOTAL_TACKLES_MADE,
    SUM(SS.RAID_POINTS) AS TOTAL_RAID_POINTS,
    SUM(SS.DEFENSIVE_POINTS) AS TOTAL_DEFENSIVE_POINTS,
    AVG(SS.POINTS_SCORED) AS AVERAGE_POINTS_PER_MATCH,
    AVG(SS.DEFENSIVE_POINTS) AS AVERAGE_DEFENSIVE_POINTS_PER_MATCH
FROM
    dbo.TEAM AS T
LEFT JOIN
    dbo.PLAYER AS P
ON
    T.Team_ID = P.TEAM_ID
LEFT JOIN
    dbo.STATISTICS_S AS SS
ON
    P.PLAYER_ID = SS.PLAYER_ID
GROUP BY
    T.Team_ID, T.Team_Name
ORDER BY
    TOTAL_POINTS_SCORED DESC;
```

	Team_ID	Team_Name	TOTAL_PLAYERS	TOTAL_MATCHES	TOTAL_POINTS_SCORED	TOTAL_TACKLES_MADE	TOTAL_RAID_POINTS	TOTAL_DEFENSIVE_POINTS	AVERAGE_POINTS_PER_MATCH	AVERAGE_DEFENSIVE_POINTS_PER_MATCH
1	T2	Titanians	4	4	38	20	25	15	9	3
2	T3	Day Raiders	3	3	29	12	21	9	9	3
3	T1	Champions	2	2	22	7	16	3	11	1
4	T4	Panther	1	1	13	2	9	4	13	4