INDIAN PREMIER LEAGUE STATS USING NEO4j

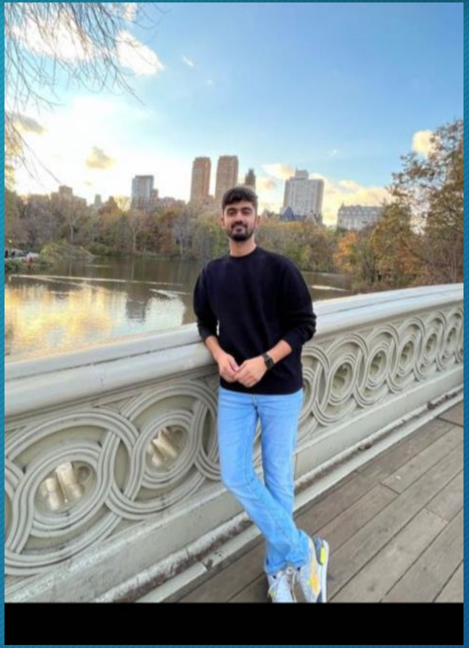
GROUP -1- PLAYERS

Pruthvi Raj Reddy

MS IN DATA SCIENCE

Piyush Gupta

MS IN DATA SCIENCE



Rajeev Anvar Rais

MS IN DATA SCIENCE

CONTENT

INTRODUCTION

- Since its start in 2008, the Indian Premier League (IPL), a professional Twenty20 cricket competition in India, has amassed enormous popularity and a devoted fan base among cricket enthusiasts. The Indian Premier League (IPL) has completely altered the landscape of cricket thanks to its one-of-a-kind combination of sports and entertainment and its extensive pool of outstanding players worldwide.
- Our project explores and analyzes the complex relationships within IPL data, such as player performances, team dynamics, and coaching staff's impact. To achieve this, we utilize a Graph database, an ideal choice for storing and querying corresponding data, providing powerful insights into the intricate interactions within the IPL ecosystem.
- By leveraging the capabilities of a Graph Database, we will investigate various aspects of the IPL, such as top performers, team strategies, coaching influence, and more. This analysis will not only offer a comprehensive understanding of the IPL's inner workings but also uncover intriguing patterns that could shape the future of this prestigious cricket league.

PROBLEM DEFINITION

The Indian Premier League (IPL) is a multifaceted cricket league, with numerous components influencing its dynamics, including player performances, team compositions, and coaching staff strategies. Analyzing the interwoven relationships within the IPL data is essential for understanding the factors contributing to a team's success or failure and identifying potential improvements in technique, selection, and performance.

We aim to examine the IPL data, focusing on the connections between players, teams, and coaches, and uncover insights that can contribute to a deeper understanding of the league's structure and performance trends. We aim to answer questions such as:

- ❑ How do players' performances affect their teams' overall success?
- ❑ How does the coaching staff impact players' performances and team dynamics?
- ❑ How do team compositions and strategies evolve in response to various factors, such as opponents' strengths and weaknesses?
- ❑ Can we identify patterns or trends that could be instrumental in devising effective strategies for future matches?

By addressing these questions, we intend to comprehensively analyze the IPL ecosystem, revealing valuable insights and patterns that can potentially influence the league's future trajectory.

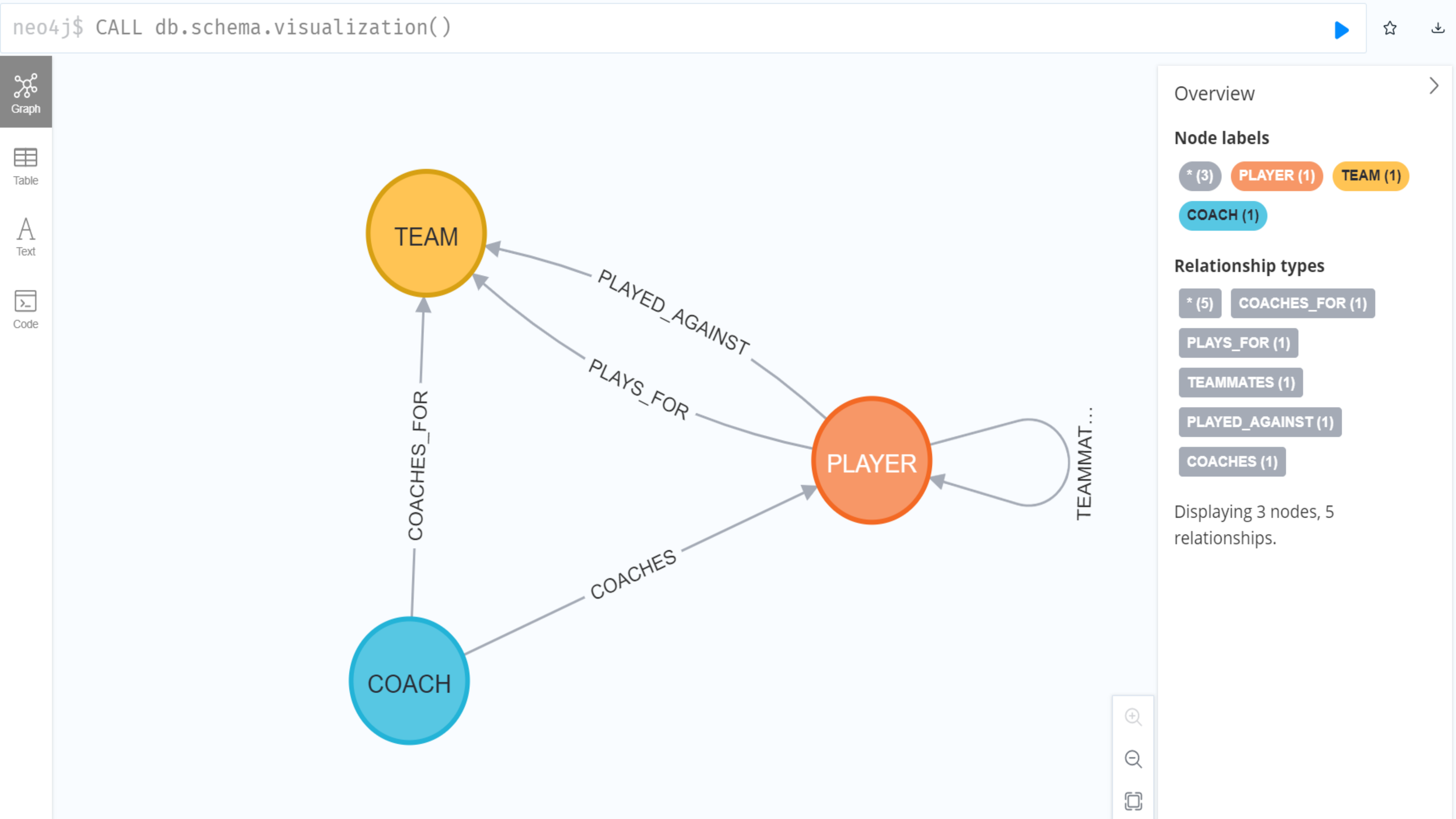
GRAPH DATABASE IS THE BEST FIT

A Graph Database is an exceptional choice for analyzing the Indian Premier League (IPL) data because it can efficiently model, store, and query complex relationships between various entities. It offers several advantages in the context of our problem:

- Intuitive Data Representation: Graph Databases represent data as nodes (entities) and relationships (edges) in a graph structure. This realistic representation aligns perfectly with the IPL's corresponding data, such as players, teams, and coaches, facilitating a more accessible and comprehensive analysis.
- Scalability: The IPL data is vast and ever-growing, with new players, teams, and matches added each season. Graph Databases are designed to handle large datasets and scale efficiently, ensuring the analysis remains efficient and up-to-date.
- Efficient Querying: Graph Databases provide exceptional performance in querying connected data. This capability is crucial for exploring the IPL's intricate relationships and answering complex questions about players' performances, team dynamics, and the coaching staff's impact.
- Flexible Data Model: The IPL's structure and data are subject to change over time, with evolving rules, team compositions, and strategies. Graph Databases offer a flexible data model that can quickly adapt to these changes, ensuring the analysis remains relevant and accurate.

By leveraging the unique strengths of a Graph Database, we can delve deep into the IPL data, uncovering valuable insights and patterns that can contribute to a better understanding of the league's complex interactions and potentially shape its future success.

MODEL OF OUR GRAPH DATABASE


```
neo4j$ CALL db.schema.visualization()
```

Overview

Node labels

* (3) PLAYER (1) TEAM (1) COACH (1)

Relationship types

* (5) COACHES_FOR (1) PLAYS_FOR (1) TEAMMATES (1) PLAYED_AGAINST (1) COACHES (1)

Displaying 3 nodes, 5 relationships.

PERFORMING ALL TASK

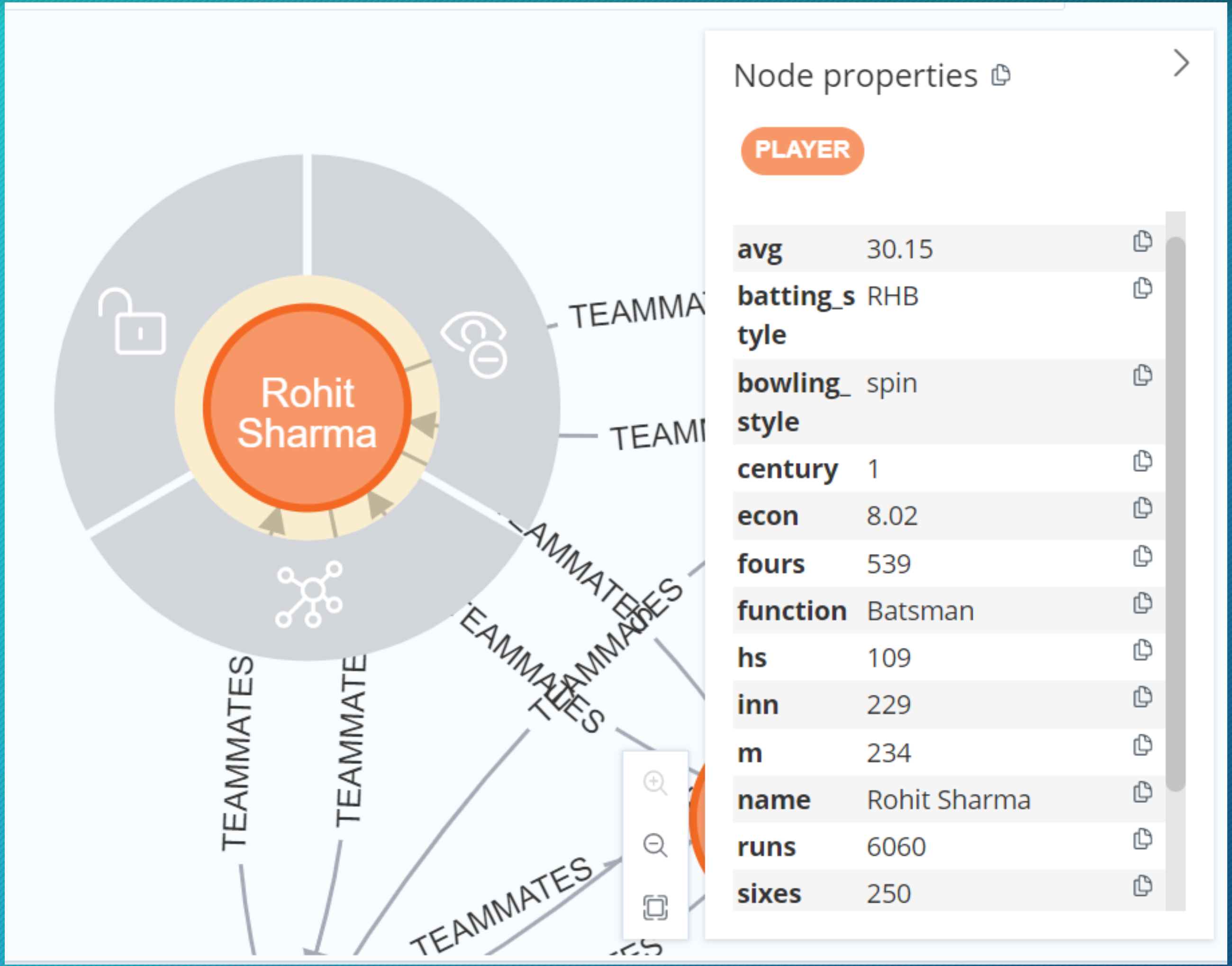
CREATE PLAYER DATABASE

Created a dataset of **16 Indian Premier League players** and added their properties such as name, year of birth, role, batting style, bowling style, matches, innings, runs, highest score, batting average, strike rate, centuries, fours, sixes, wickets, and economy rate to its nodes.

```
CREATE
(Rohit:PLAYER{name:"Rohit Sharma", year: 1987, function: "Batsman", batting_style: "R
HB", bowling_style: "spin",m:234,inn:229,runs:6060,hs:109,avg:30.15,sr:130.04,century:1,fo
urs:539,sixes:250,wkts:15,econ:8.02});
```

For Example:- Rohit Sharma: A right-handed batsman and occasional spin bowler, born in 1987, Rohit has played in 234 matches, scoring 6060 runs with a highest score of 109, and has taken 15 wickets with an economy rate of 8.02.

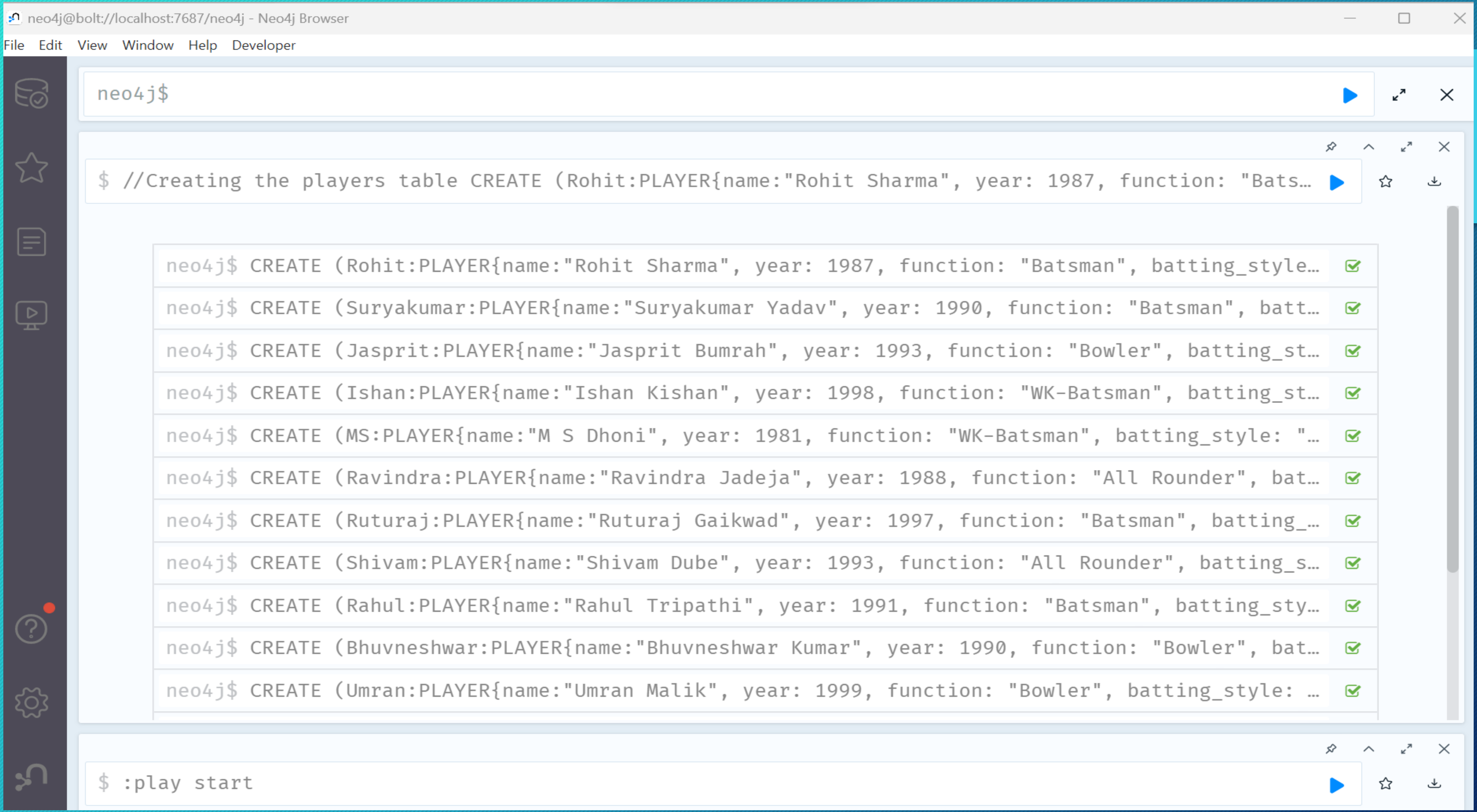
PLAYER NODE PROPERTIES



Node properties

PLAYER

avg	30.15
batting_style	RHB
bowling_style	spin
century	1
econ	8.02
fours	539
function	Batsman
hs	109
inn	229
m	234
name	Rohit Sharma
runs	6060
sixes	250

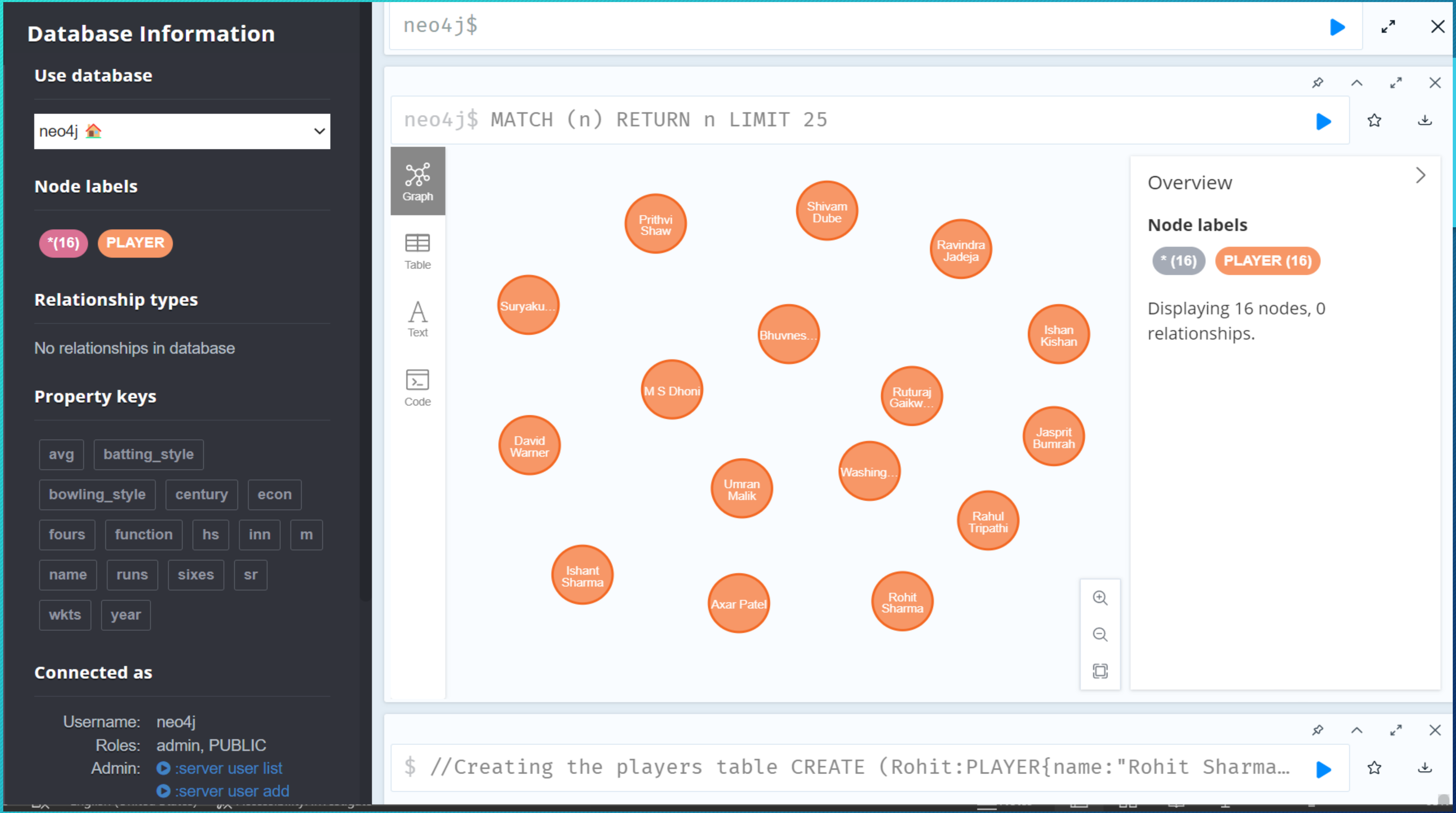

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser
File Edit View Window Help Developer

```
neo4j$
```

```
$ //Creating the players table CREATE (Rohit:PLAYER{name:"Rohit Sharma", year: 1987, function: "Bats…
```

```
neo4j$ CREATE (Rohit:PLAYER{name:"Rohit Sharma", year: 1987, function: "Batsman", batting_style…
neo4j$ CREATE (Suryakumar:PLAYER{name:"Suryakumar Yadav", year: 1990, function: "Batsman", batt…
neo4j$ CREATE (Jasprit:PLAYER{name:"Jasprit Bumrah", year: 1993, function: "Bowler", batting_st…
neo4j$ CREATE (Ishan:PLAYER{name:"Ishan Kishan", year: 1998, function: "WK-Batsman", batting_st…
neo4j$ CREATE (MS:PLAYER{name:"M S Dhoni", year: 1981, function: "WK-Batsman", batting_style: "…
neo4j$ CREATE (Ravindra:PLAYER{name:"Ravindra Jadeja", year: 1988, function: "All Rounder", bat…
neo4j$ CREATE (Ruturaj:PLAYER{name:"Ruturaj Gaikwad", year: 1997, function: "Batsman", batting_…
neo4j$ CREATE (Shivam:PLAYER{name:"Shivam Dube", year: 1993, function: "All Rounder", batting_s…
neo4j$ CREATE (Rahul:PLAYER{name:"Rahul Tripathi", year: 1991, function: "Batsman", batting_sty…
neo4j$ CREATE (Bhuvneshwar:PLAYER{name:"Bhuvneshwar Kumar", year: 1990, function: "Bowler", bat…
neo4j$ CREATE (Umran:PLAYER{name:"Umran Malik", year: 1999, function: "Bowler", batting_style: …
```

```
$ :play start
```



Database Information

Use database

neo4j

Node labels

*(16) PLAYER

Relationship types

No relationships in database

Property keys

avg batting_style bowling_style century econ fours function hs inn m name runs sixes sr wkts year

Connected as

Username: neo4j
Roles: admin, PUBLIC
Admin: :server user list
:server user add

```
neo4j$
```

```
neo4j$ MATCH (n) RETURN n LIMIT 25
```

Overview

Node labels

* (16) PLAYER (16)

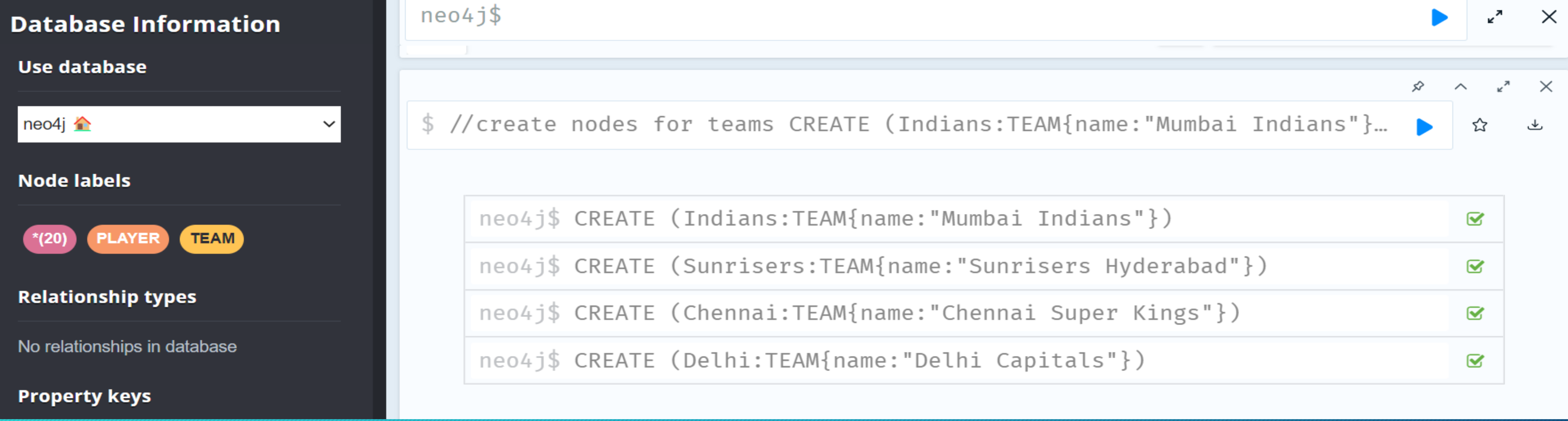
Displaying 16 nodes, 0 relationships.

```
$ //Creating the players table CREATE (Rohit:PLAYER{name:"Rohit Sharma…
```

CREATE TEAM DATABASE

Created nodes for 4 Indian Premier League teams with their respective names as properties:

- **Mumbai Indians**: Representing the city of Mumbai, the Mumbai Indians is one of the most successful teams in the IPL, having won multiple titles since its inception. The team is known for its consistent performance and a strong lineup of players.
- **Sunrisers Hyderabad**: Representing the city of Hyderabad, the Sunrisers Hyderabad has significantly impacted the IPL since its entry. The team has won the title once and is recognized for its strong bowling unit and balanced team composition.
- **Chennai Super Kings**: Representing the city of Chennai, the Chennai Super Kings is another successful IPL team, having won multiple championships. Known for its experienced players and strong leadership, the team enjoys a massive fan following and has consistently been a strong contender in the league.
- **Delhi Capitals**: Representing the city of Delhi, the Delhi Capitals has seen a resurgence in recent years, becoming a strong contender in the IPL. The team is known for its young talent and aggressive playing style, often making it to the playoffs and challenging the top teams in the league.


Database Information

Use database

neo4j

Node labels

*(20) PLAYER TEAM

Relationship types

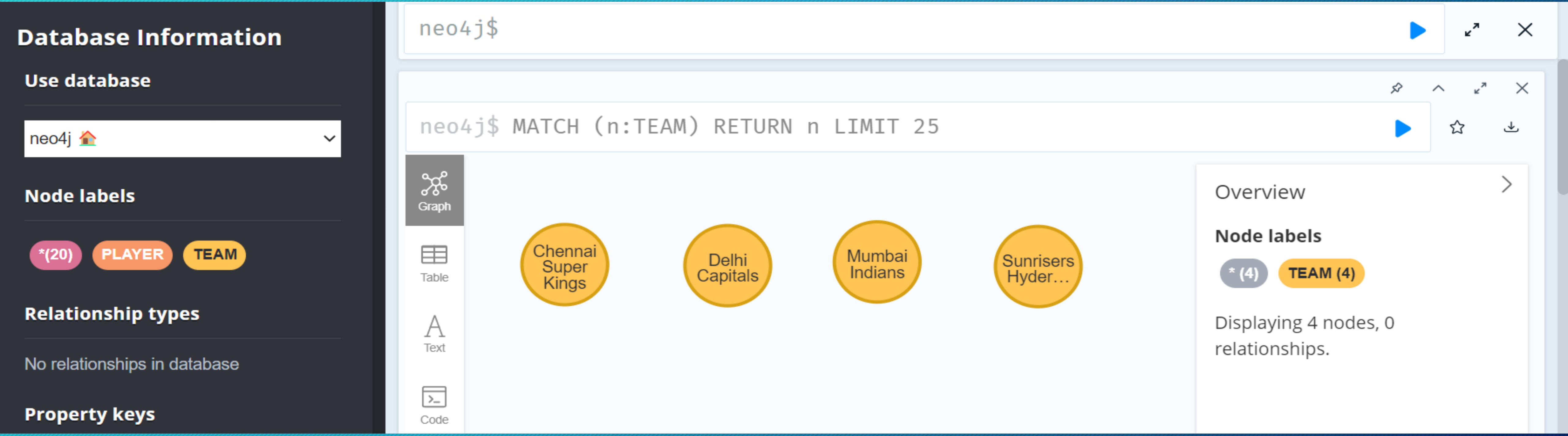
No relationships in database

Property keys

```
neo4j$
```

```
$ //create nodes for teams CREATE (Indians:TEAM{name:"Mumbai Indians"}…
```

```
neo4j$ CREATE (Indians:TEAM{name:"Mumbai Indians"})
neo4j$ CREATE (Sunrisers:TEAM{name:"Sunrisers Hyderabad"})
neo4j$ CREATE (Chennai:TEAM{name:"Chennai Super Kings"})
neo4j$ CREATE (Delhi:TEAM{name:"Delhi Capitals"})
```


Database Information

Use database

neo4j

Node labels

*(20) PLAYER TEAM

Relationship types

No relationships in database

Property keys

```
neo4j$
```

```
neo4j$ MATCH (n:TEAM) RETURN n LIMIT 25
```

Overview

Node labels

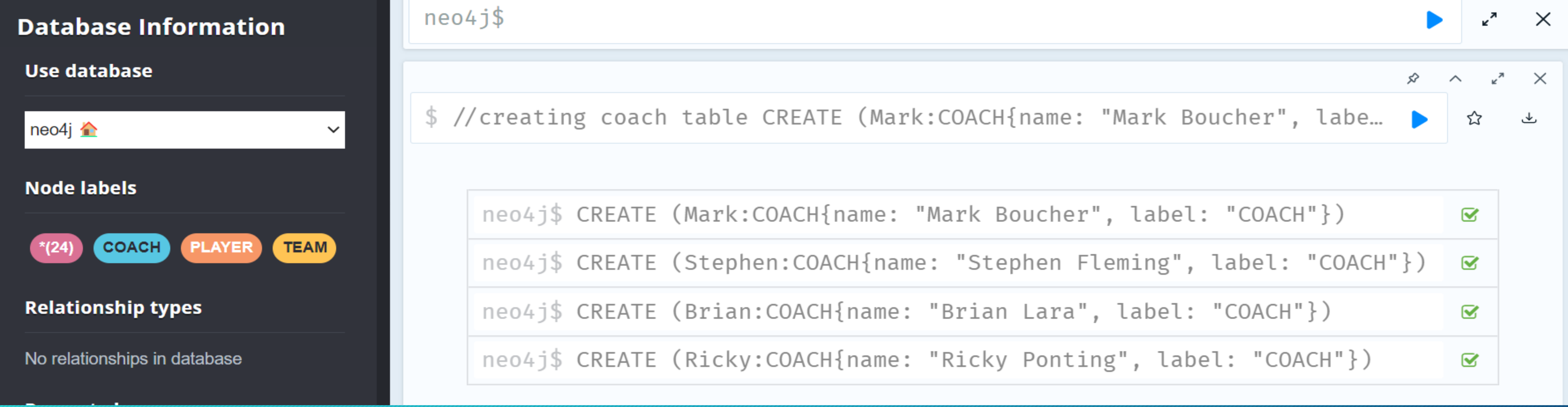
* (4) TEAM (4)

Displaying 4 nodes, 0 relationships.

CREATE COACH DATABASE

Created nodes for 4 renowned cricket coaches with their names as properties:

- **Mark Boucher**: A former South African cricketer and wicketkeeper, Mark Boucher is now an experienced coach who brings his vast knowledge and strategic acumen to the teams he works with.
- **Stephen Fleming**: A former New Zealand cricketer and captain, Stephen Fleming is known for his tactical brilliance and calm demeanor. As a coach, he effectively mentors players and helps teams achieve success.
- **Brian Lara:** A legendary West Indian batsman, Brian Lara holds numerous cricket records. As a coach, his deep understanding of the game and expertise in batting techniques make him an invaluable asset to any team.
- **Ricky Ponting**: A former Australian cricket captain and one of the most successful cricketers in history, Ricky Ponting brings his exceptional leadership skills and winning mindset to coaching, motivating players to perform at their best



Database Information

Use database

neo4j

Node labels

*(24) COACH PLAYER TEAM

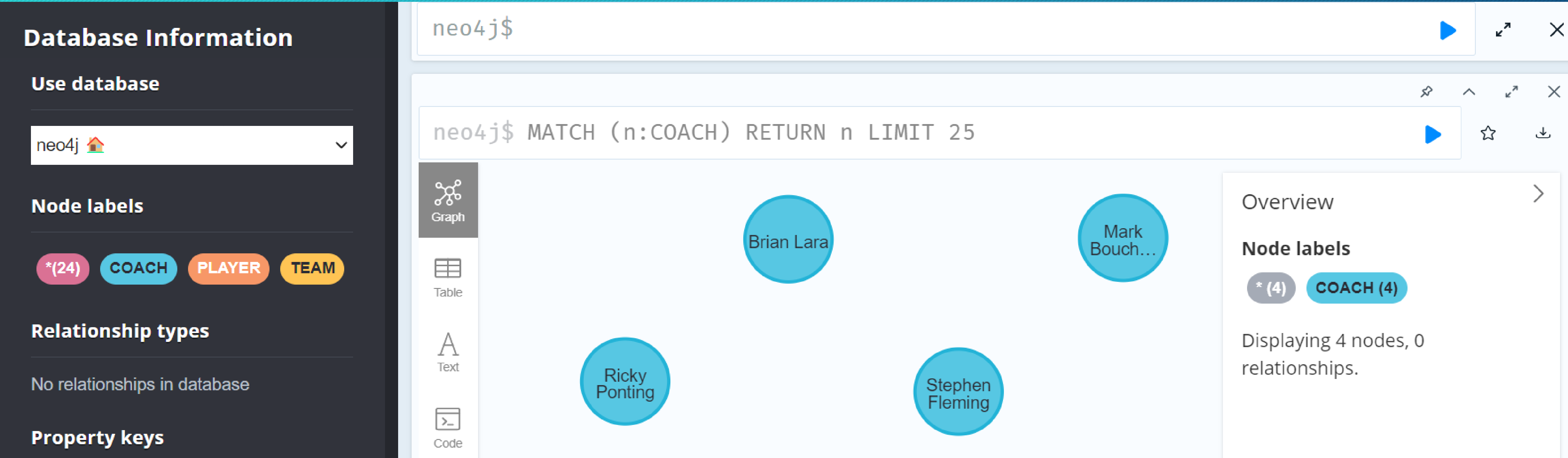
Relationship types

No relationships in database

```
neo4j$
```

```
$ //creating coach table CREATE (Mark:COACH{name: "Mark Boucher", labe…
```

```
neo4j$ CREATE (Mark:COACH{name: "Mark Boucher", label: "COACH"})
neo4j$ CREATE (Stephen:COACH{name: "Stephen Fleming", label: "COACH"})
neo4j$ CREATE (Brian:COACH{name: "Brian Lara", label: "COACH"})
neo4j$ CREATE (Ricky:COACH{name: "Ricky Ponting", label: "COACH"})
```



Database Information

Use database

neo4j

Node labels

*(24) COACH PLAYER TEAM

Relationship types

No relationships in database

Property keys

```
neo4j$
```

```
neo4j$ MATCH (n:COACH) RETURN n LIMIT 25
```

Overview

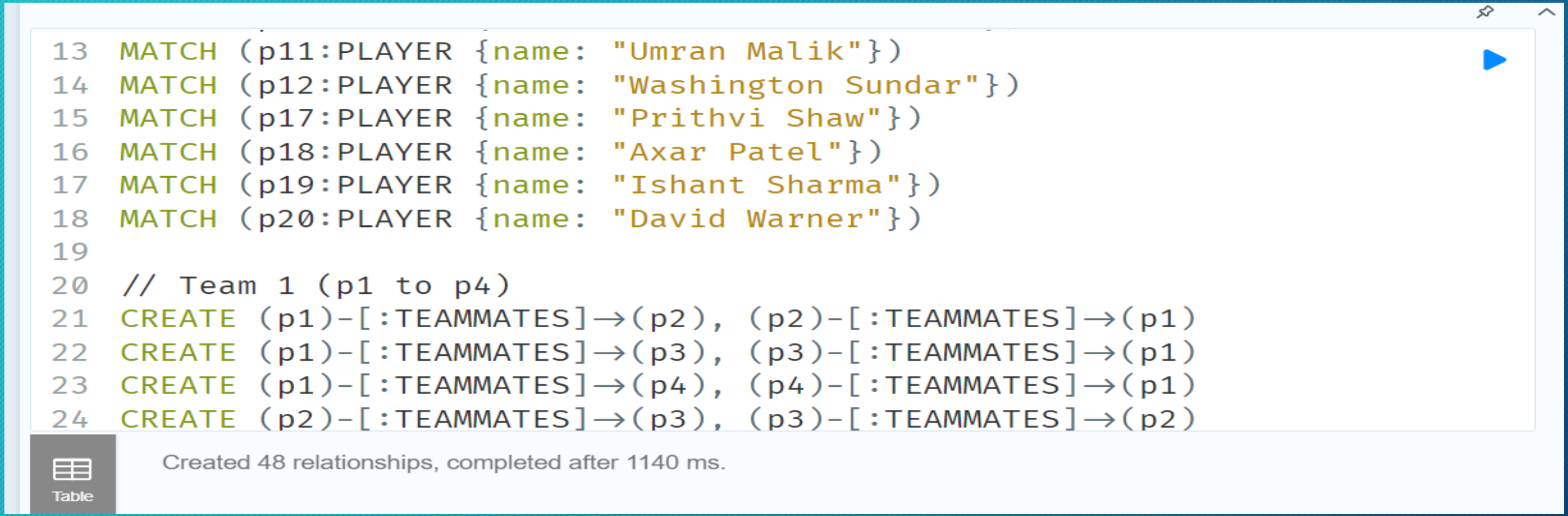
Node labels

* (4) COACH (4)

Displaying 4 nodes, 0 relationships.

CREATE RELATIONSHIPS B/W TEAMMATES

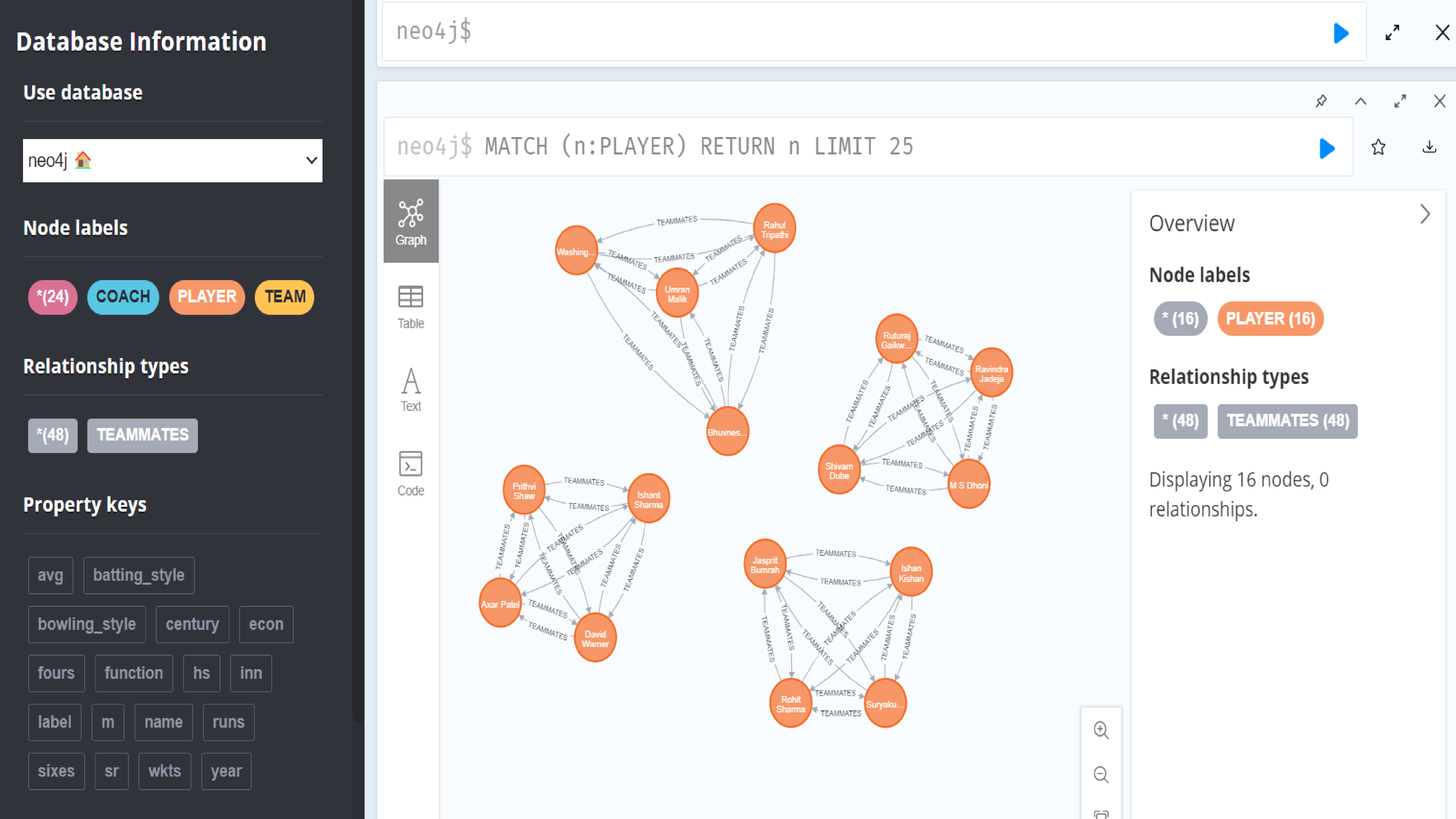
Creating relationships between teammates in the graph database represents the connections among players who play together on the same team. These relationships help to visualize and analyze the interactions between players within a team, such as their performance as a unit, their understanding of each other's playing styles, and their on-field chemistry. By establishing these connections, we can gain insights into team dynamics, strengths, weaknesses, and overall team performance in the Indian Premier League.



```
MATCH (p11:PLAYER {name: "Umran Malik"})
MATCH (p12:PLAYER {name: "Washington Sundar"})
MATCH (p17:PLAYER {name: "Prithvi Shaw"})
MATCH (p18:PLAYER {name: "Axar Patel"})
MATCH (p19:PLAYER {name: "Ishant Sharma"})
MATCH (p20:PLAYER {name: "David Warner"})

// Team 1 (p1 to p4)
CREATE (p1)-[:TEAMMATES]→(p2), (p2)-[:TEAMMATES]→(p1)
CREATE (p1)-[:TEAMMATES]→(p3), (p3)-[:TEAMMATES]→(p1)
CREATE (p1)-[:TEAMMATES]→(p4), (p4)-[:TEAMMATES]→(p1)
CREATE (p2)-[:TEAMMATES]→(p3), (p3)-[:TEAMMATES]→(p2)
```

Created 48 relationships, completed after 1140 ms.


Database Information

Use database

neo4j

Node labels

*(24) COACH PLAYER TEAM

Relationship types

*(48) TEAMMATES

Property keys

avg batting_style bowling_style century econ fours function hs inn label m name runs sixes sr wkts year

```
neo4j$
```

```
neo4j$ MATCH (n:PLAYER) RETURN n LIMIT 25
```

Overview

Node labels

* (16) PLAYER (16)

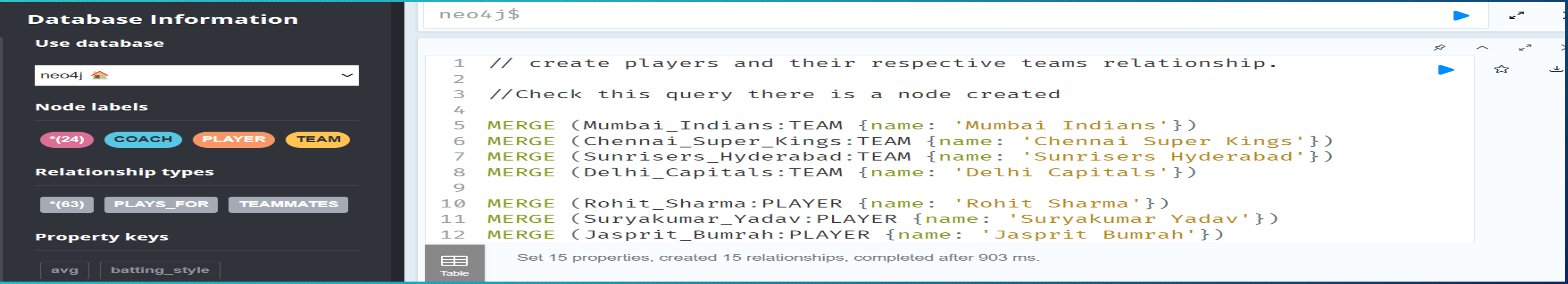
Relationship types

* (48) TEAMMATES (48)

Displaying 16 nodes, 0 relationships.

CREATE RELATIONSHIPS B/W TEAMMATES

Creating nodes for four teams and 16 players, and establishes a relationship between them through the "PLAYS_FOR" relationship type. The "MERGE" keyword is used to ensure that nodes are only created if they do not already exist in the database.Each player node has a "name" property, while each team node has a "name" property that is used to uniquely identify it. The "PLAYS_FOR" relationship has a "role" property that specifies the role of the player in the team.Overall, this query establishes the players' association with their respective teams in the Indian Premier League, allowing us to query and analyze data related to individual players or teams in the league.



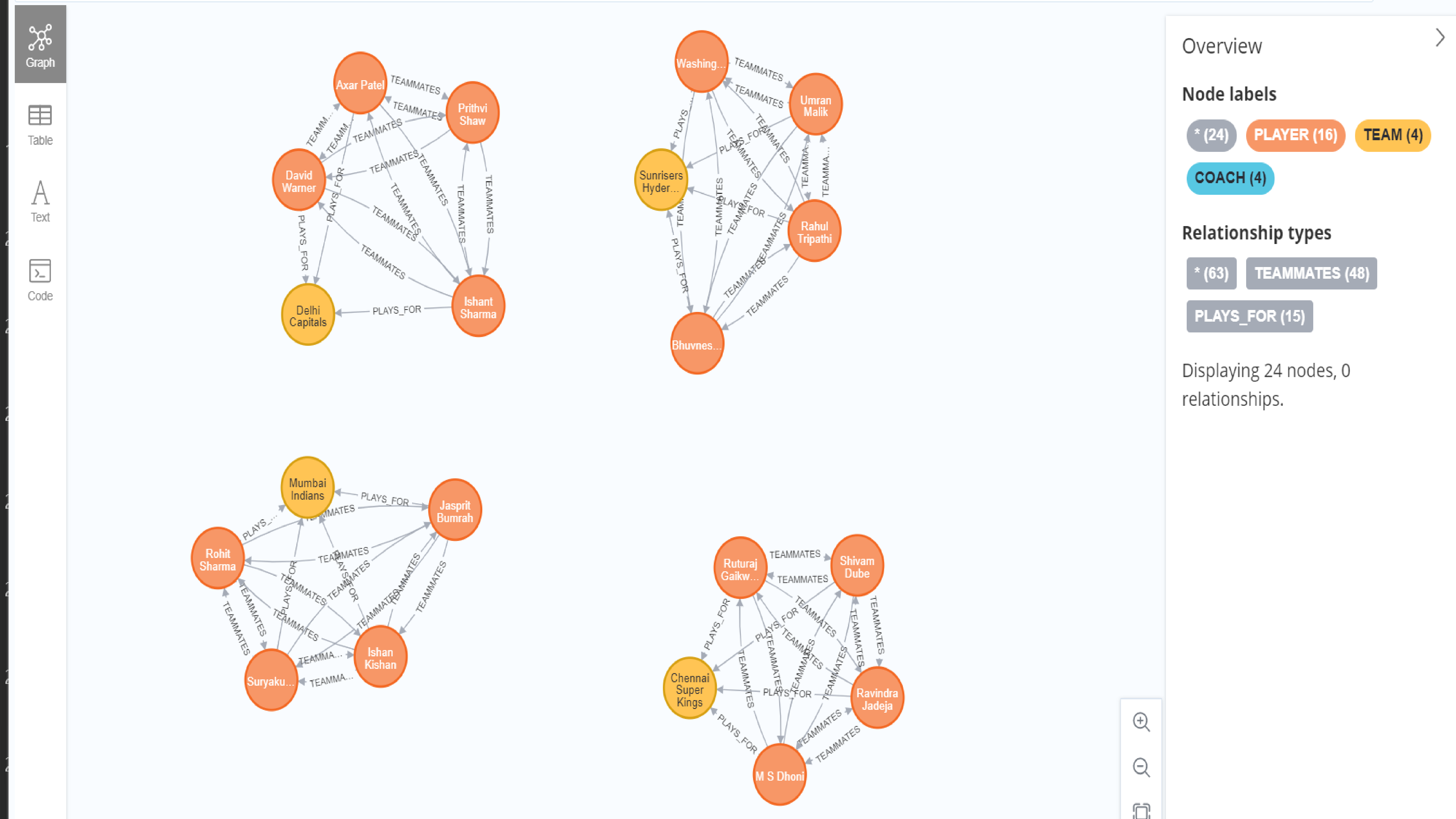
```
// create players and their respective teams relationship.

//Check this query there is a node created

MERGE (Mumbai_Indians:TEAM {name: 'Mumbai Indians'})
MERGE (Chennai_Super_Kings:TEAM {name: 'Chennai Super Kings'})
MERGE (Sunrisers_Hyderabad:TEAM {name: 'Sunrisers Hyderabad'})
MERGE (Delhi_Capitals:TEAM {name: 'Delhi Capitals'})

MERGE (Rohit_Sharma:PLAYER {name: 'Rohit Sharma'})
MERGE (Suryakumar_Yadav:PLAYER {name: 'Suryakumar Yadav'})
MERGE (Jasprit_Bumrah:PLAYER {name: 'Jasprit Bumrah'})
```

```
Set 15 properties, created 15 relationships, completed after 903 ms.
```



Overview

Node labels

* (24) PLAYER (16) TEAM (4) COACH (4)

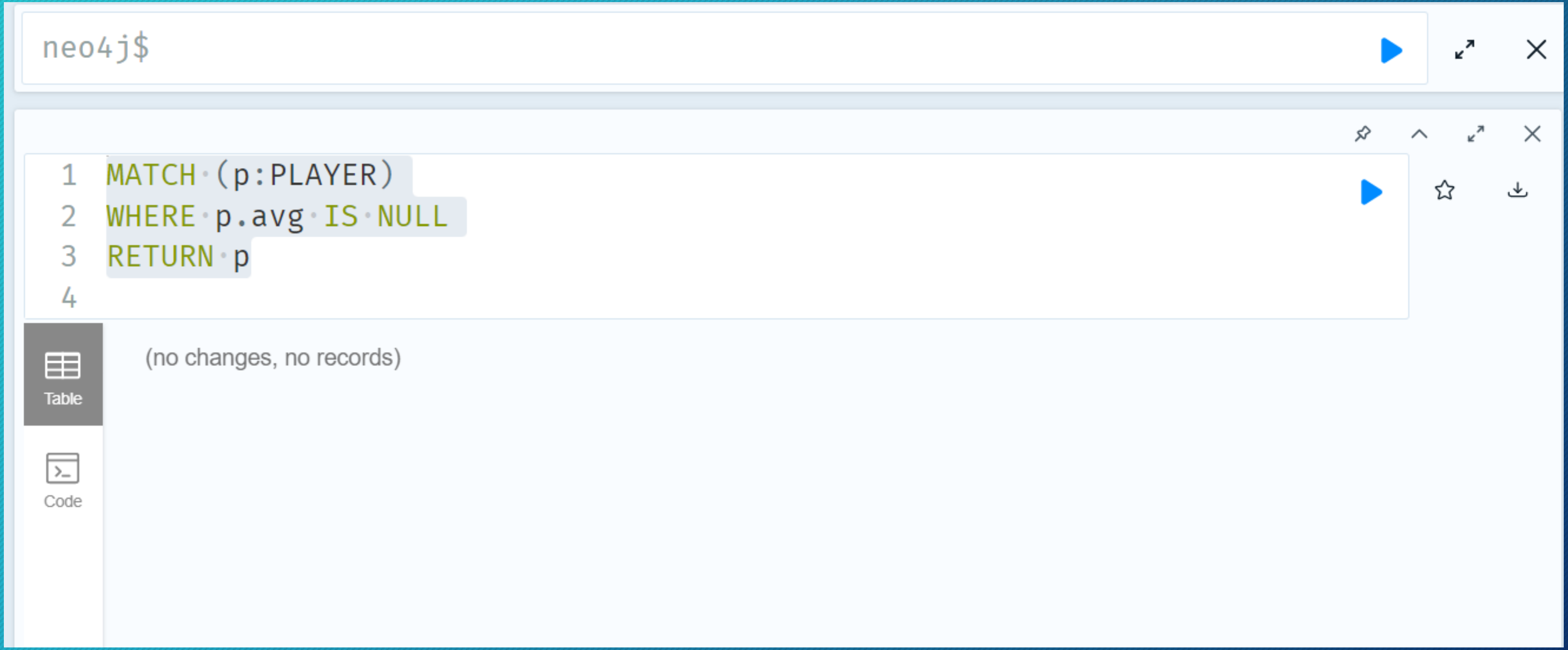
Relationship types

* (63) TEAMMATES (48) PLAYS_FOR (15)

Displaying 24 nodes, 0 relationships.

CHECK FOR ANY NULL VALUES

```
MATCH (p:PLAYER)
WHERE p.avg IS NULL
RETURN p
```



```
neo4j$
```

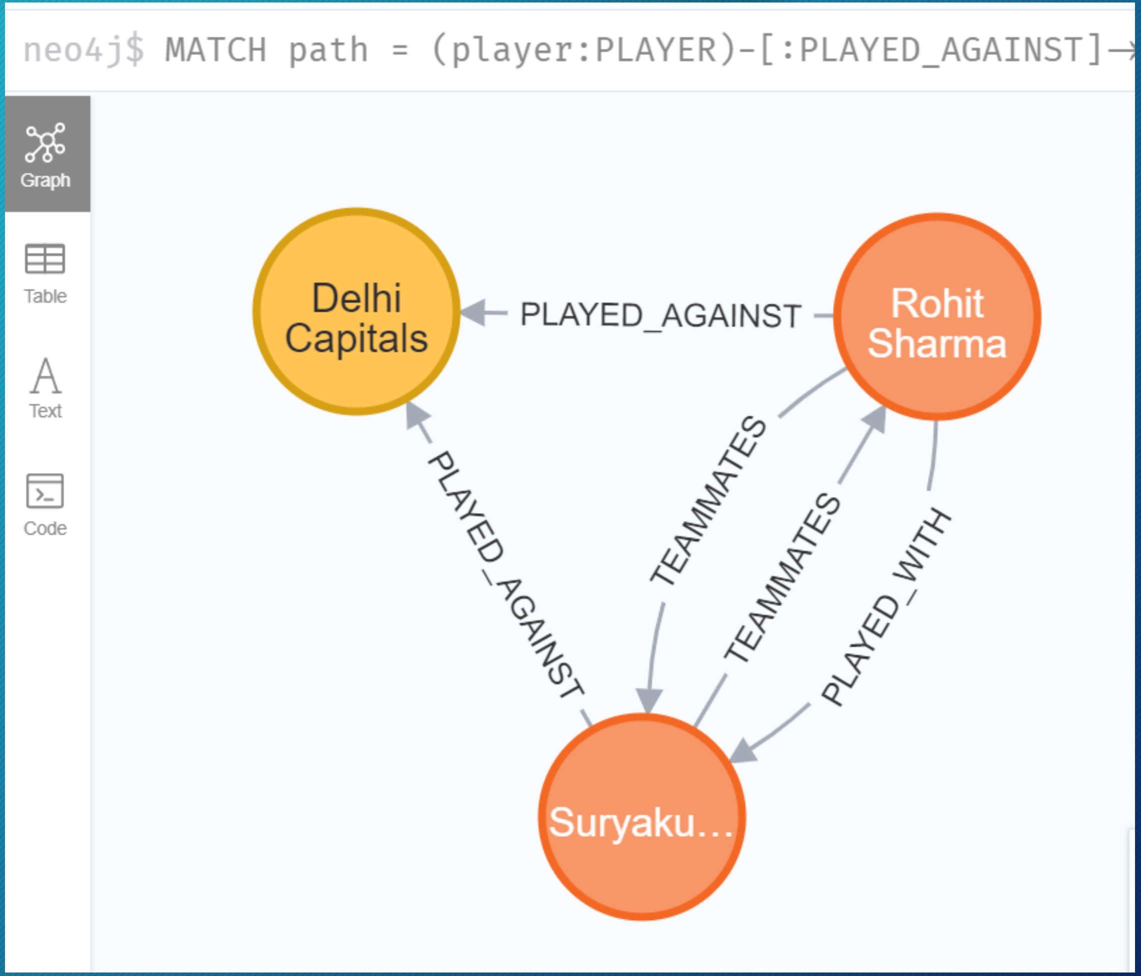
```
MATCH (p:PLAYER)
WHERE p.avg IS NULL
RETURN p
```

```
(no changes, no records)
```

Show that your model has loops

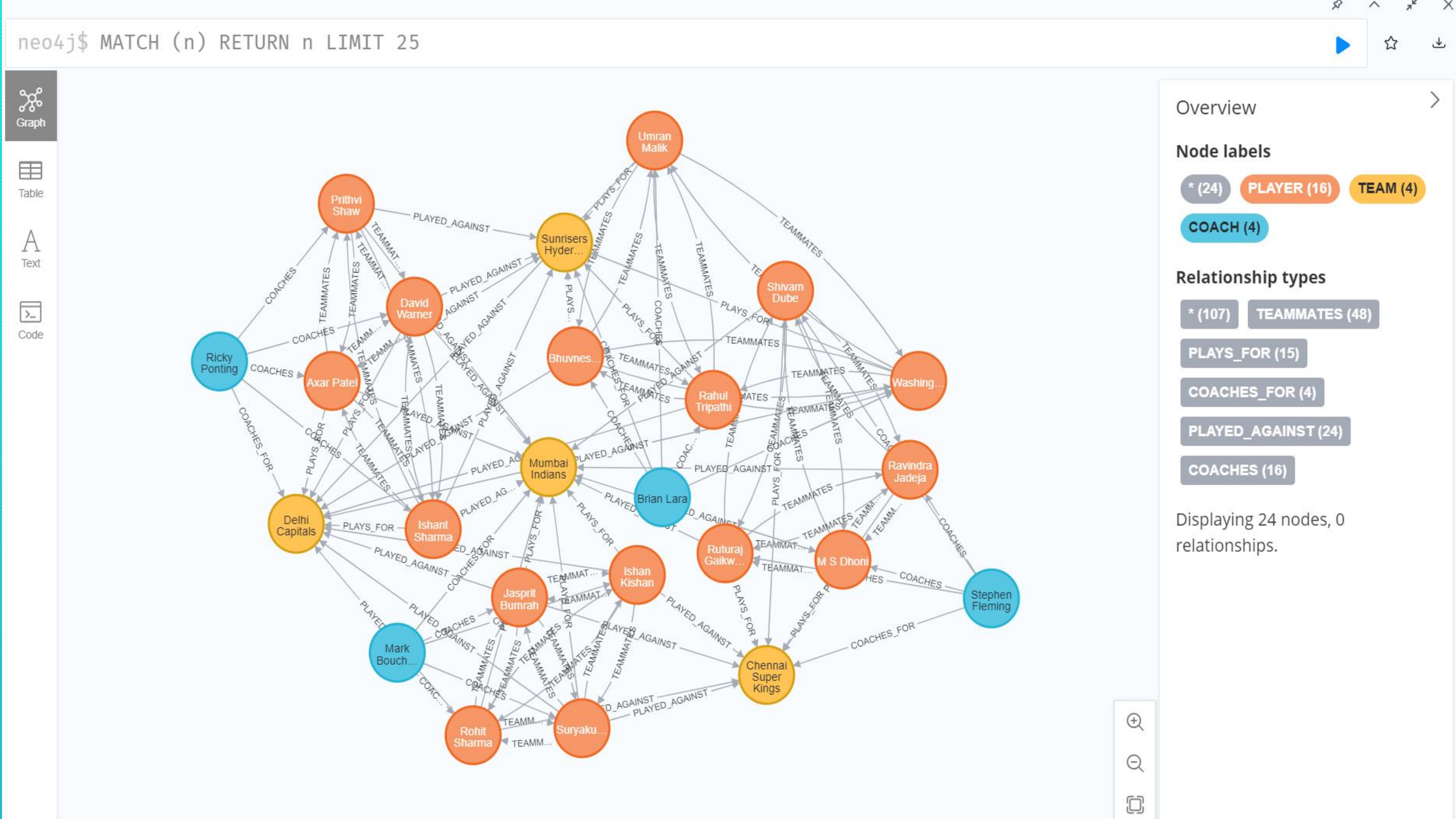
```
MATCH path = (player:PLAYER)-[:PLAYED_AGAINST]->(team:TEAM)<-[:PLAYED_AGAINST]-(player2:PLAYER)-[:PLAYED_WITH]-(player)RETURN path LIMIT 1;
```

The following query will return the path where a player has played against a team and played with another player who also played against the same team and This query should return a single path with a loop that includes the "PLAYED_AGAINST" and "PLAYED_WITH" relationships.



```
neo4j$ MATCH path = (player:PLAYER)-[:PLAYED_AGAINST]→
```

FINAL TRANSFORM GRAPH DATABASE



```
neo4j$ MATCH (n) RETURN n LIMIT 25
```

Overview

Node labels

* (24) PLAYER (16) TEAM (4) COACH (4)

Relationship types

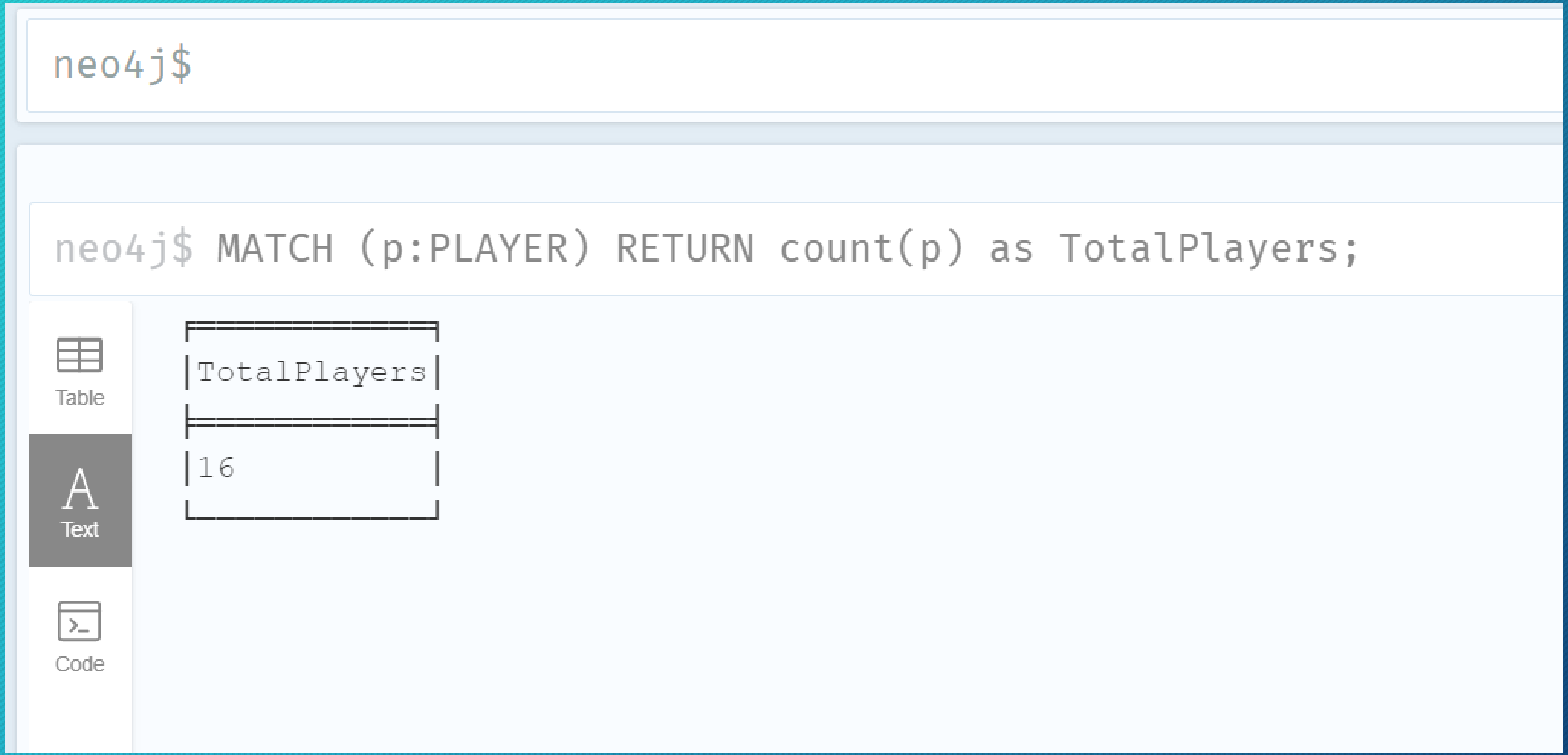
* (107) TEAMMATES (48) PLAYS_FOR (15) COACHES_FOR (4) PLAYED_AGAINST (24) COACHES (16)

Displaying 24 nodes, 0 relationships.

PERFORM AGGREGATION OPERATION

Find the total number of players in the dataset:

MATCH (p:PLAYER)RETURN count(p) as TotalPlayers;

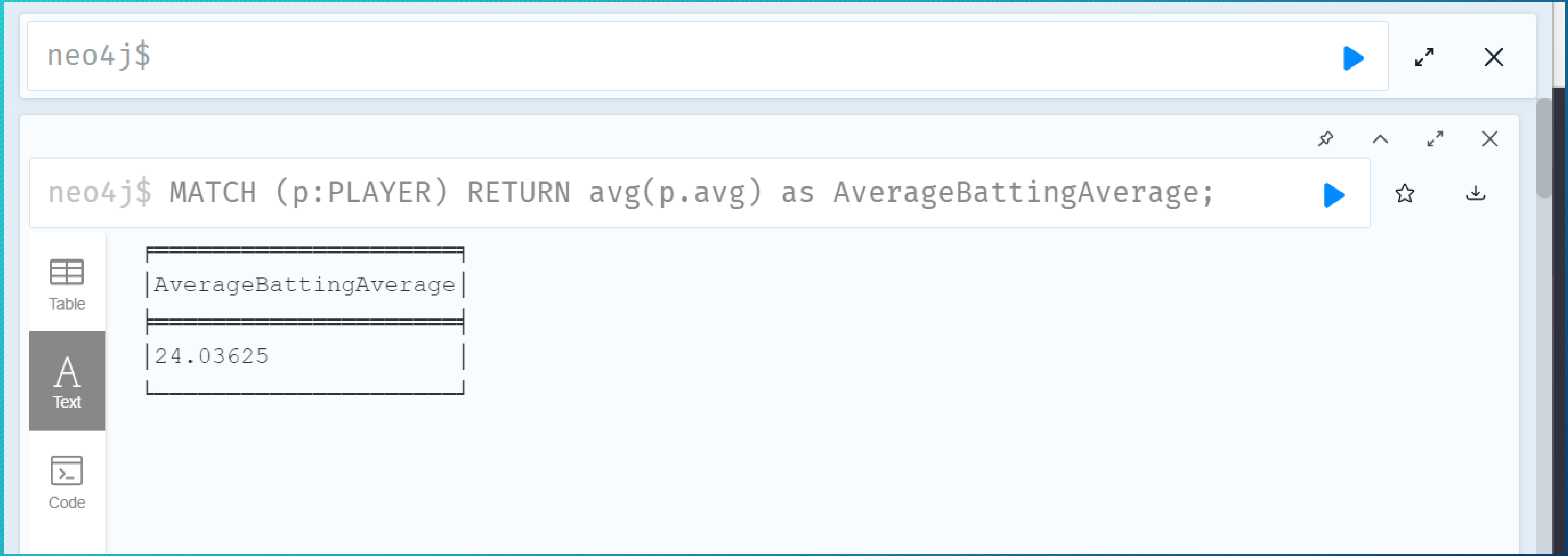


```
neo4j$

neo4j$ MATCH (p:PLAYER) RETURN count(p) as TotalPlayers;
╒════════════╕
│TotalPlayers│
╞════════════╡
│16          │
└────────────┘
```

Calculate the average batting average for all players:

MATCH (p:PLAYER)RETURN avg(p.avg) as AverageBattingAverage;;



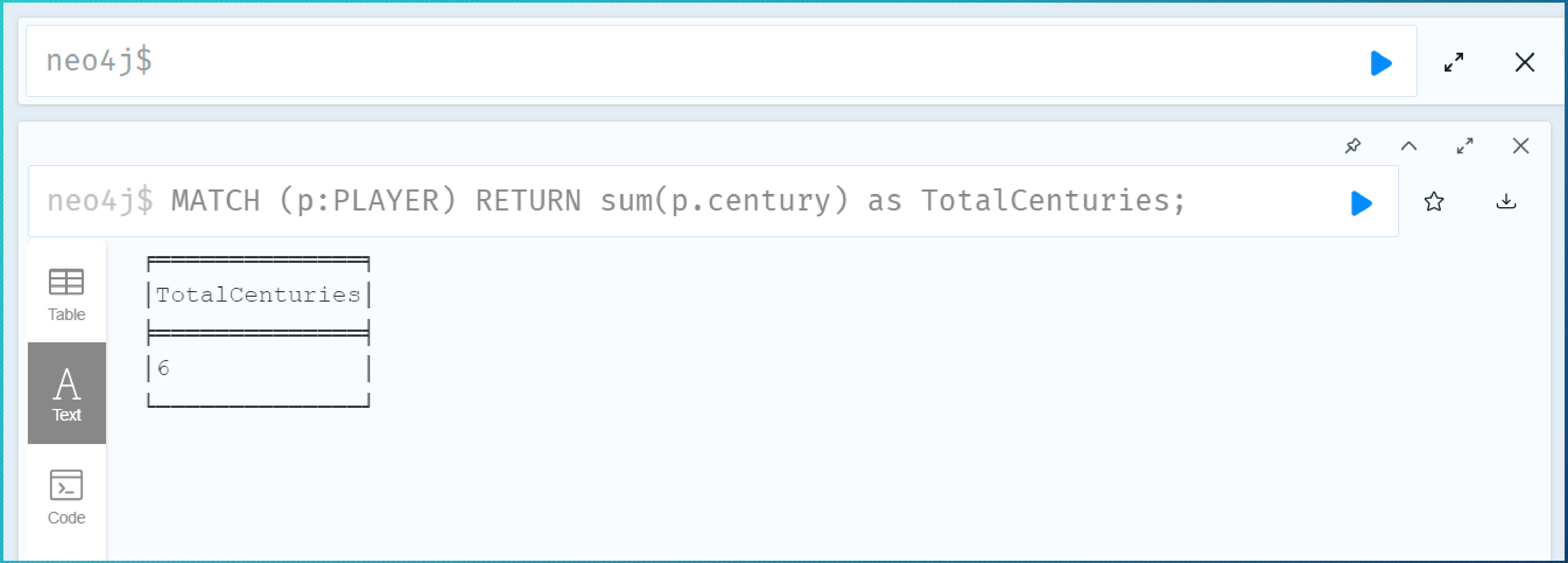
```
neo4j$
```

```
neo4j$ MATCH (p:PLAYER) RETURN avg(p.avg) as AverageBattingAverage;
```

```
╒═════════════════════╕
│AverageBattingAverage│
╞═════════════════════╡
│24.03625             │
└─────────────────────┘
```

Find the total number of centuries scored by all players:

MATCH (p:PLAYER)RETURN sum(p.century) as TotalCenturies;



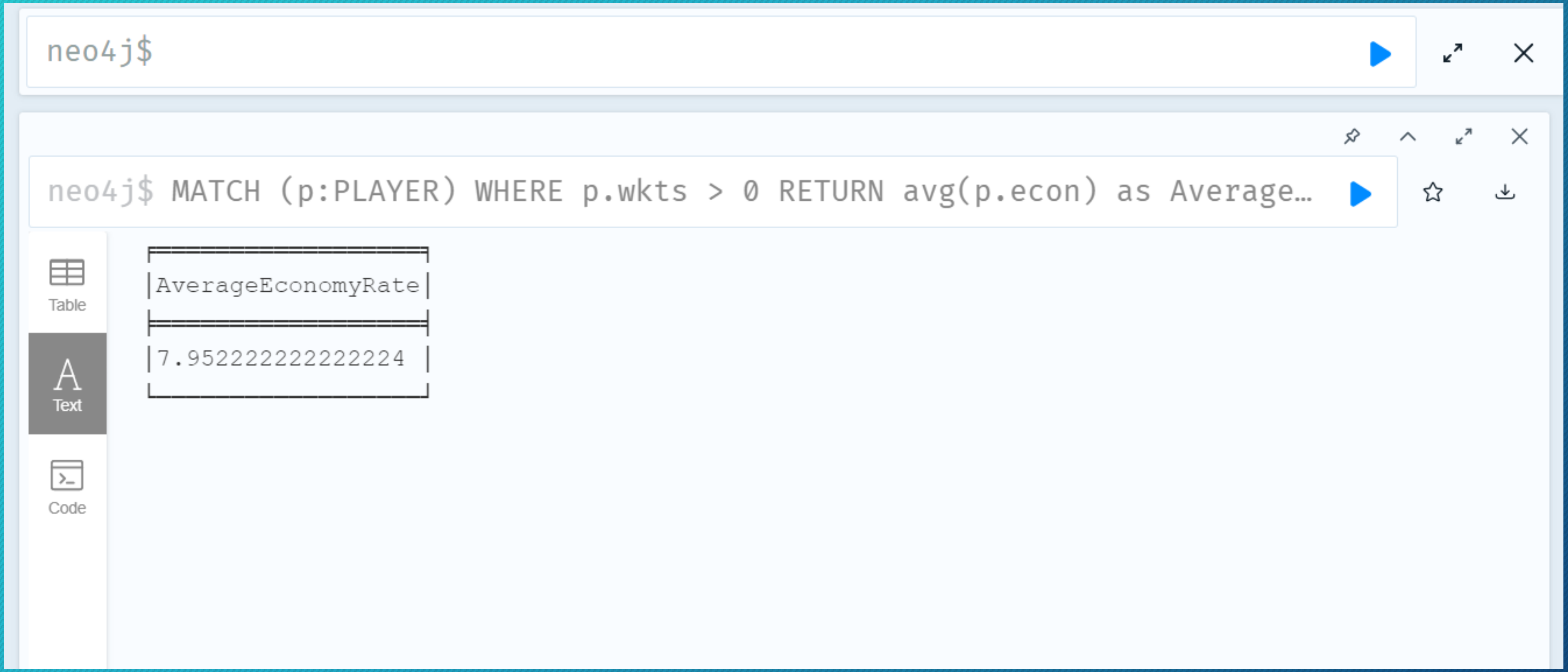
```
neo4j$
```

```
neo4j$ MATCH (p:PLAYER) RETURN sum(p.century) as TotalCenturies;
```

```
╒══════════════╕
│TotalCenturies│
╞══════════════╡
│6             │
└──────────────┘
```

Calculate the average economy rate for all bowlers:

MATCH (p:PLAYER)WHERE p.wkts > 0RETURN avg(p.econ) as AverageEconomyRate;



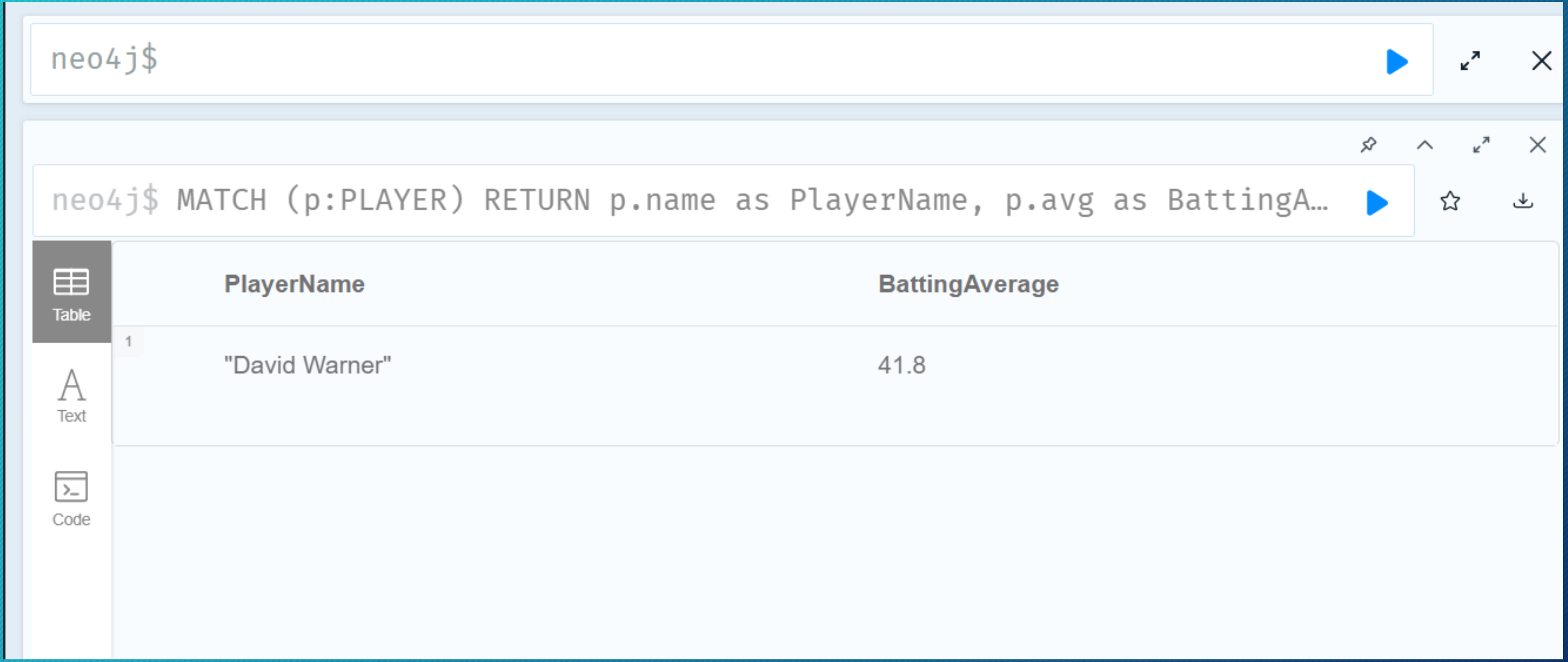
```
neo4j$
```

```
neo4j$ MATCH (p:PLAYER) WHERE p.wkts > 0 RETURN avg(p.econ) as Average…
```

```
╒══════════════════╕
│AverageEconomyRate│
╞══════════════════╡
│7.952222222222224 │
└──────────────────┘
```

Find the player with the highest batting average:

```
MATCH (p:PLAYER)
RETURN p.name as PlayerName, p.avg as BattingAverage
ORDER BY p.avg DESC
LIMIT 1;
```



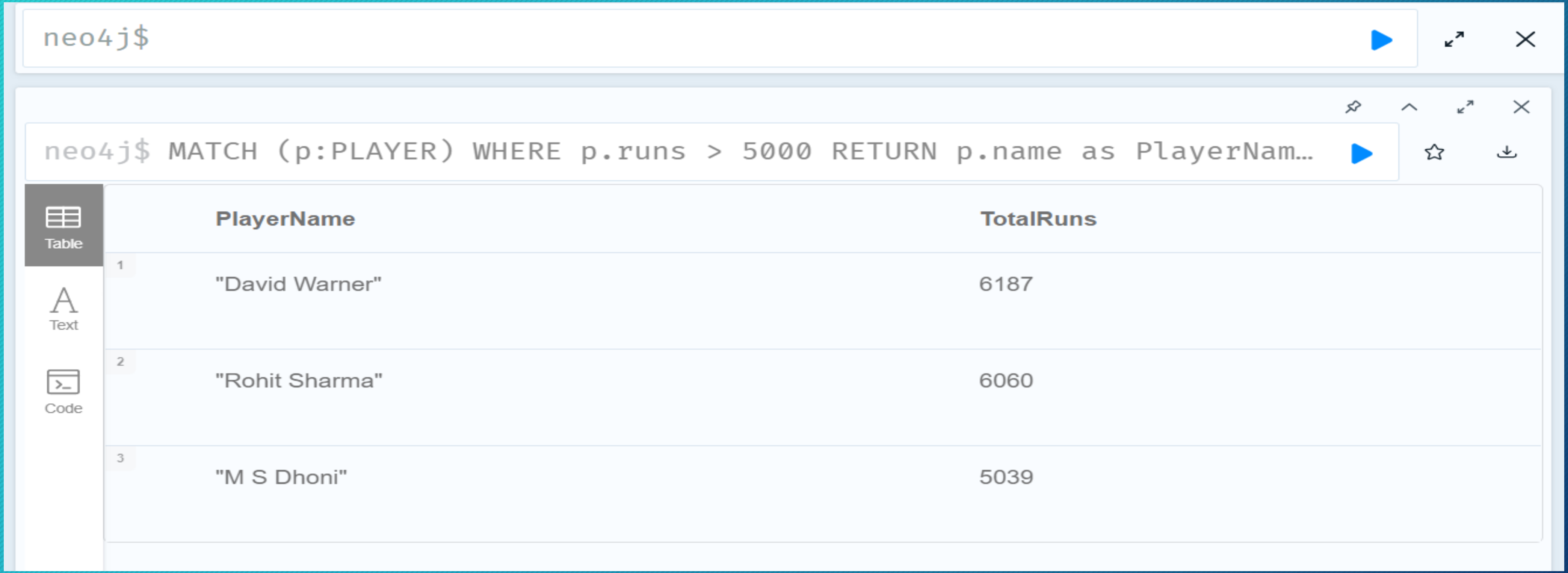
```
neo4j$
neo4j$ MATCH (p:PLAYER) RETURN p.name as PlayerName, p.avg as BattingA…
```

	PlayerName	BattingAverage
1	"David Warner"	41.8

QUERY OF DATABASE

List all the players who have scored more than 5000 runs:

MATCH (p:PLAYER)WHERE p.runs > 5000RETURN p.name as PlayerName, p.runs as TotalRunsORDER BY p.runs DESC;

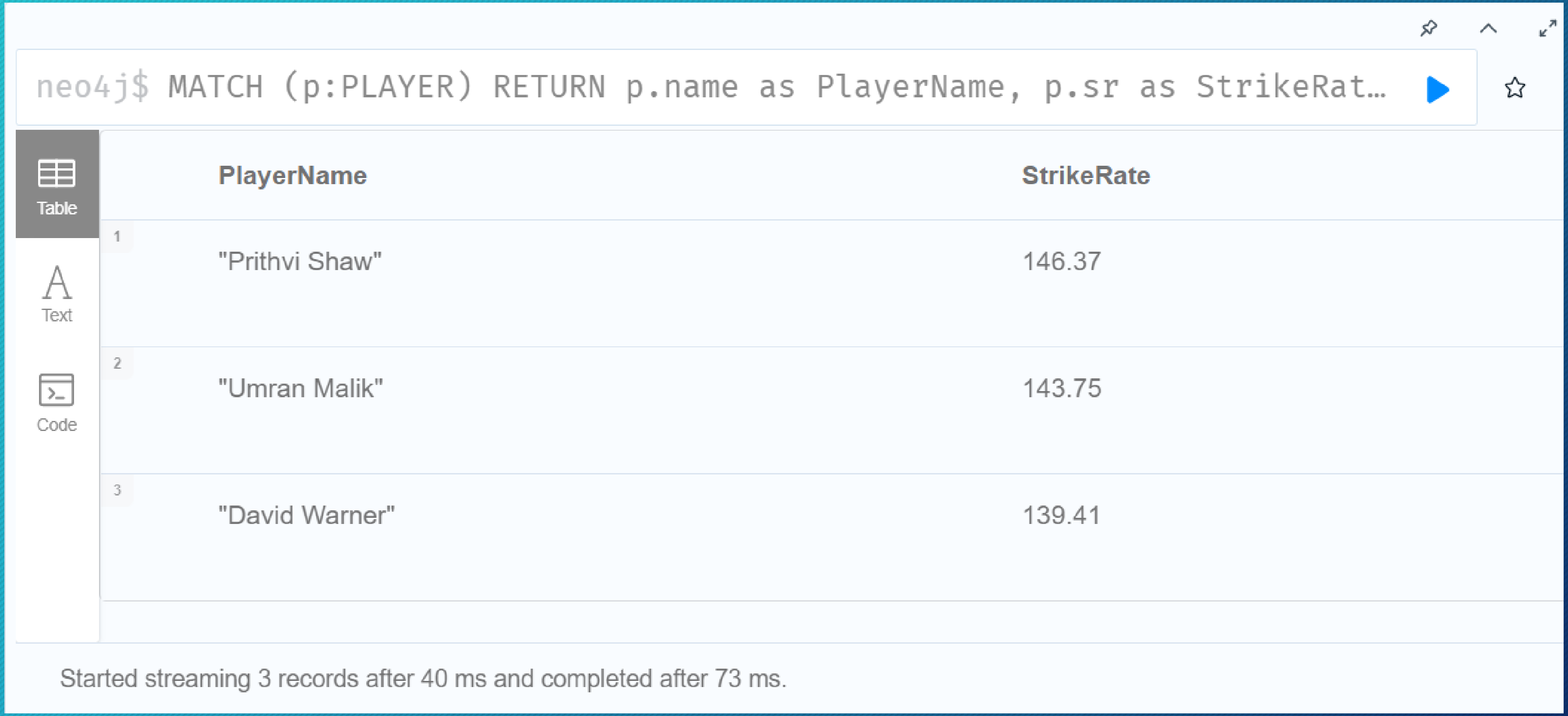


```
neo4j$ MATCH (p:PLAYER) WHERE p.runs > 5000 RETURN p.name as PlayerNam…
```

	PlayerName	TotalRuns
1	"David Warner"	6187
2	"Rohit Sharma"	6060
3	"M S Dhoni"	5039

Find the top 3 players with the highest strike rates:

```
MATCH (p:PLAYER)
RETURN p.name as PlayerName, p.sr as StrikeRate
ORDER BY p.sr DESC
LIMIT 3;
```



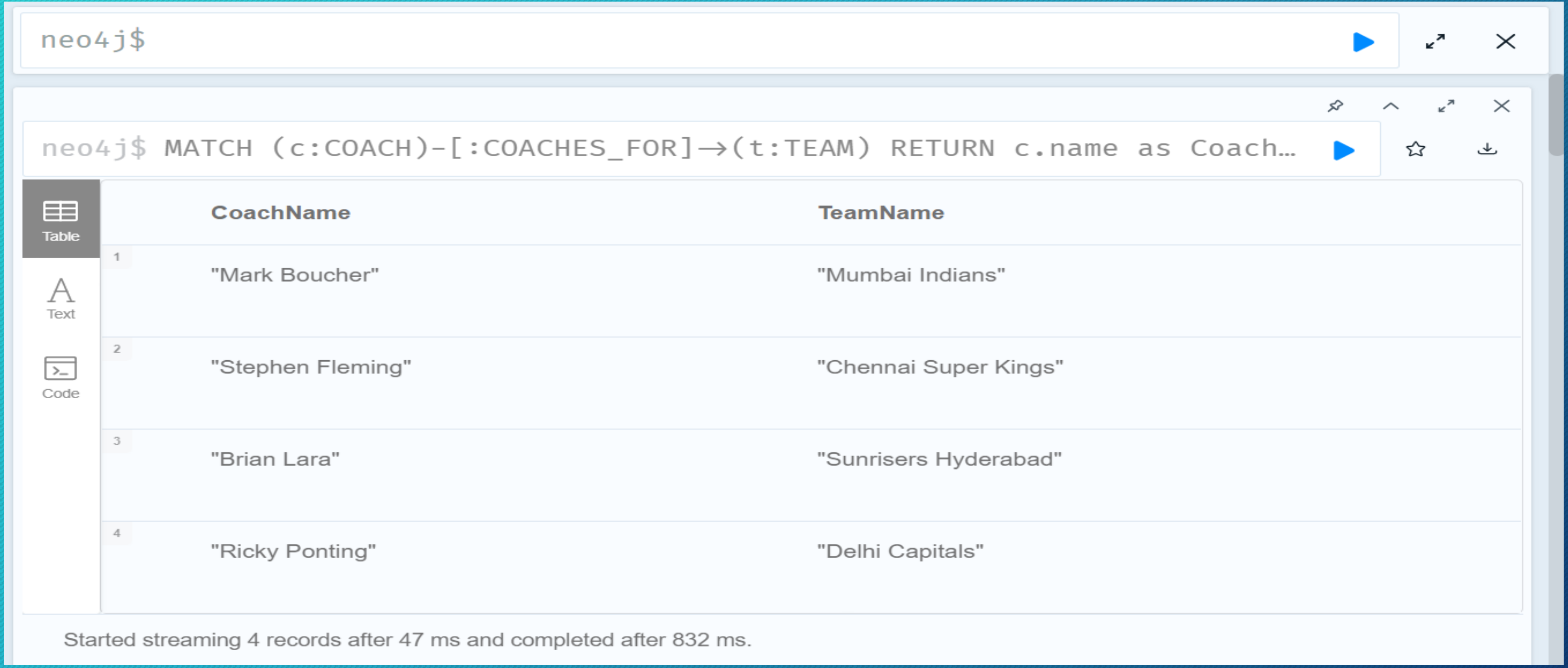
```
neo4j$ MATCH (p:PLAYER) RETURN p.name as PlayerName, p.sr as StrikeRat…
```

	PlayerName	StrikeRate
1	"Prithvi Shaw"	146.37
2	"Umran Malik"	143.75
3	"David Warner"	139.41

Started streaming 3 records after 40 ms and completed after 73 ms.

Find all the coaches and the teams they coach:

```
MATCH (c:COACH)-[:COACHES_FOR]->(t:TEAM)RETURN
c.name as CoachName, t.name as TeamName;
```



```
neo4j$ MATCH (c:COACH)-[:COACHES_FOR]→(t:TEAM) RETURN c.name as Coach…
```

```
   CoachName            TeamName
1  "Mark Boucher"       "Mumbai Indians"
2  "Stephen Fleming"    "Chennai Super Kings"
3  "Brian Lara"         "Sunrisers Hyderabad"
4  "Ricky Ponting"      "Delhi Capitals"
```

Started streaming 4 records after 47 ms and completed after 832 ms.

CONCLUSION

- In this project, we leveraged the power of Neo4j, a graph database management system, to model and analyze the Indian Premier League (IPL) statistics. Through our graph database, we were able to efficiently store and represent complex relationships between various entities such as players, teams, and coaches. This enabled us to perform in-depth analyses and gain valuable insights into the performance of IPL teams and players over the years.

- By utilizing Neo4j's powerful query language, Cypher, we executed a wide range of queries that showcased the flexibility and efficiency of graph databases in handling interconnected data. These queries included aggregations, filtering, and traversal of relationships, demonstrating the power of graph databases in dealing with complex data structures.

- Our IPL database provides a foundation for further analysis and exploration, including examining player performance trends, identifying key factors that contribute to a team's success, and uncovering hidden patterns in the data. This project showcases the potential of using graph databases in sports analytics and paves the way for more advanced analytics and insights in the future.

TVS EUROGRIP TYRES
SNJ

ThANk you