School of Computer Science, UPES, Dehradun.

A

LABORATORY FILE

On

DATABASE MANAGEMENT SYSTEM (DBMS) LAB

B.TECH. -III Semester

AUG. – NOV.- 2024.

Submitted by:
Name: Harsh Vardhan Saini
SAP ID: 500120369
Roll No: R2142230056
Batch: 2

Experiment 09
Use of different SQL clauses and join

Aim:

To design and implement a relational database for managing student and sports match data, enabling analysis of student participation in various matches.

Problem Statement:

The task involves creating a database to store information about students and the matches they play in, facilitating queries to extract insights on student participation, match venues, and student demographics.

Theory:

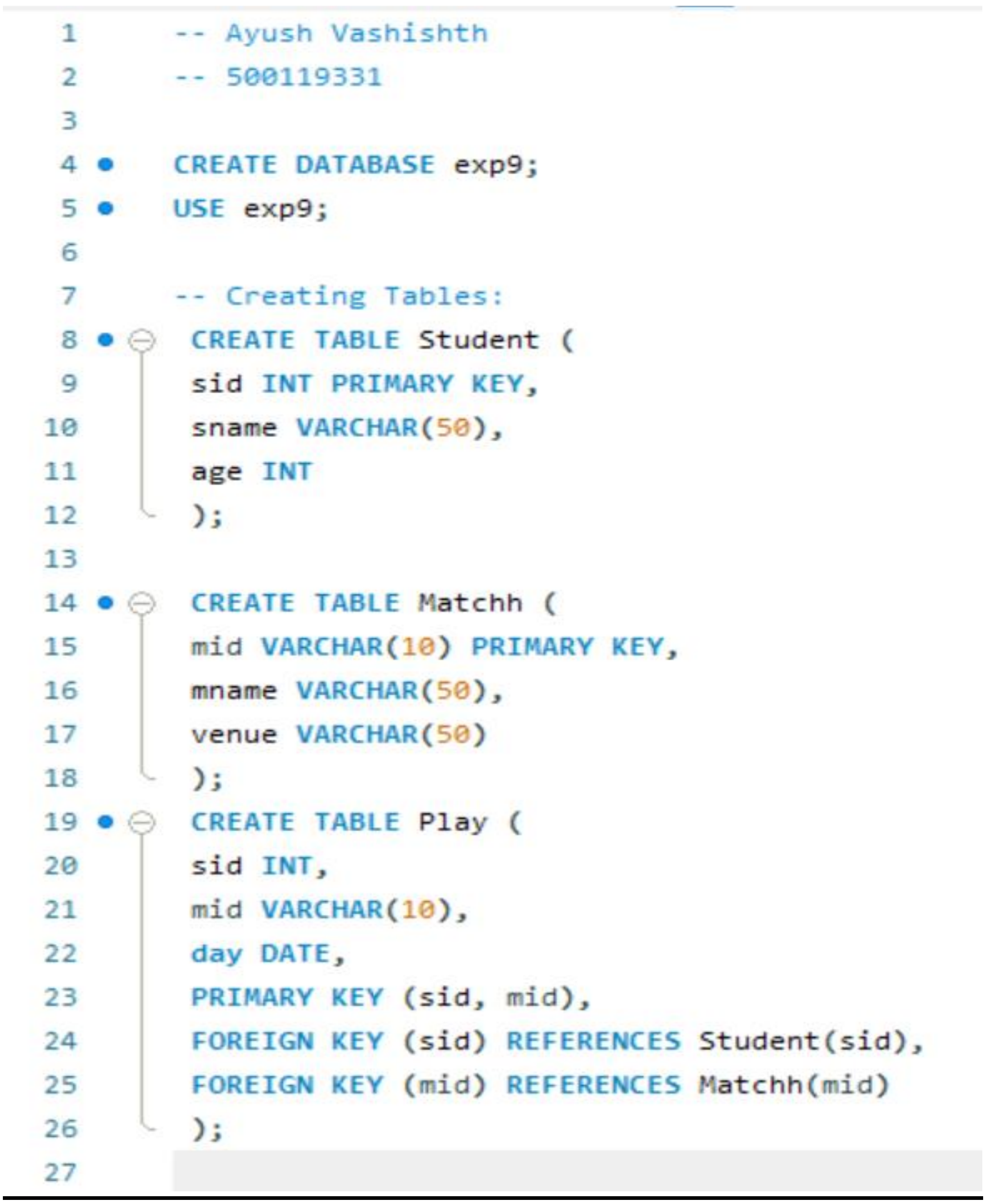
Relational databases organize data into structured tables, allowing for efficient querying and manipulation. SQL (Structured Query Language) is utilized for data definition and manipulation. Key concepts include:

- Tables: Structures to hold data in rows and columns.
- Primary Keys: Unique identifiers for records in a table, ensuring data integrity.
- Foreign Keys: References to primary keys in other tables, establishing relationships.
- JOIN Operations: Methods to combine rows from two or more tables based on related columns.
- Aggregate Functions: Functions such as `AVG()`, `COUNT()`, and `DISTINCT` to perform calculations and return summary information.

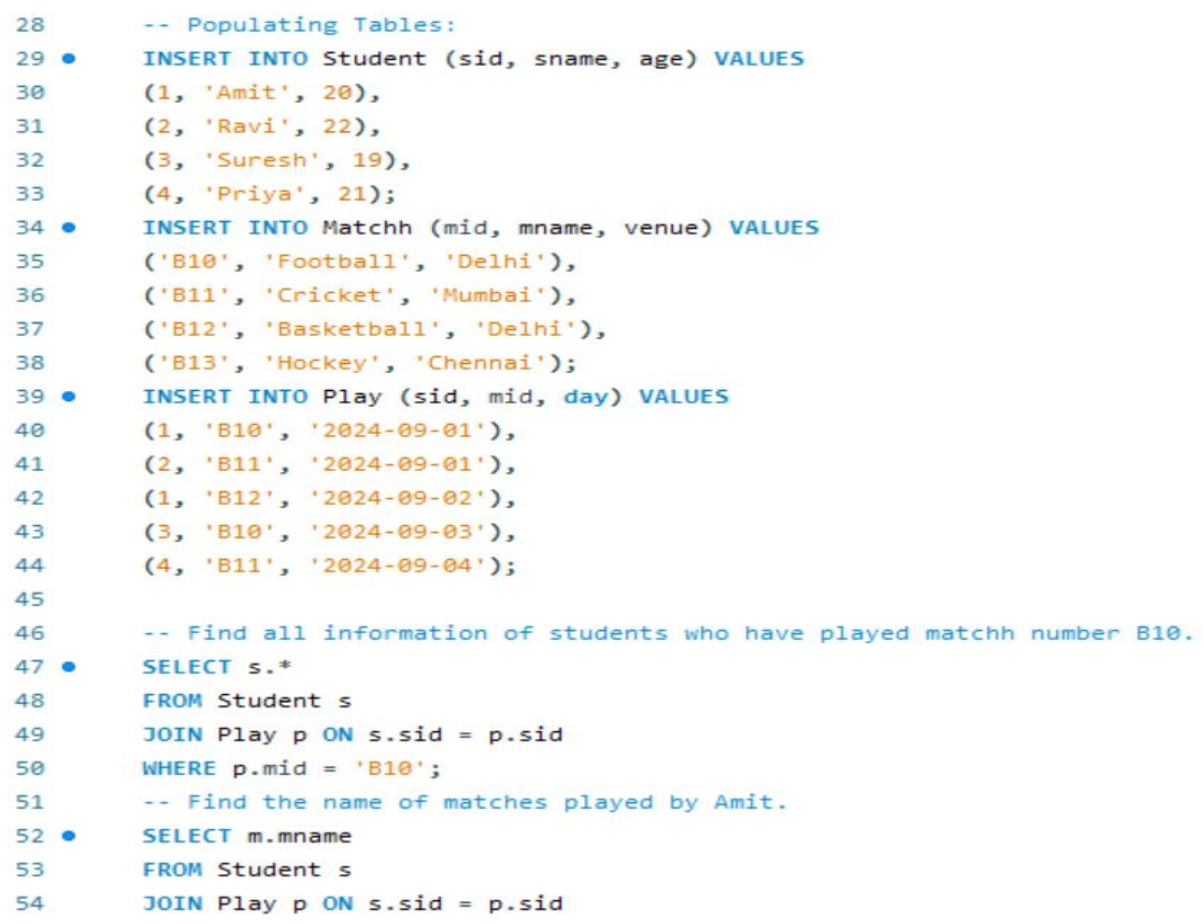
Commands Used:

1. Database and Table Creation
2. Data Insertion
3. Data Retrieval Queries

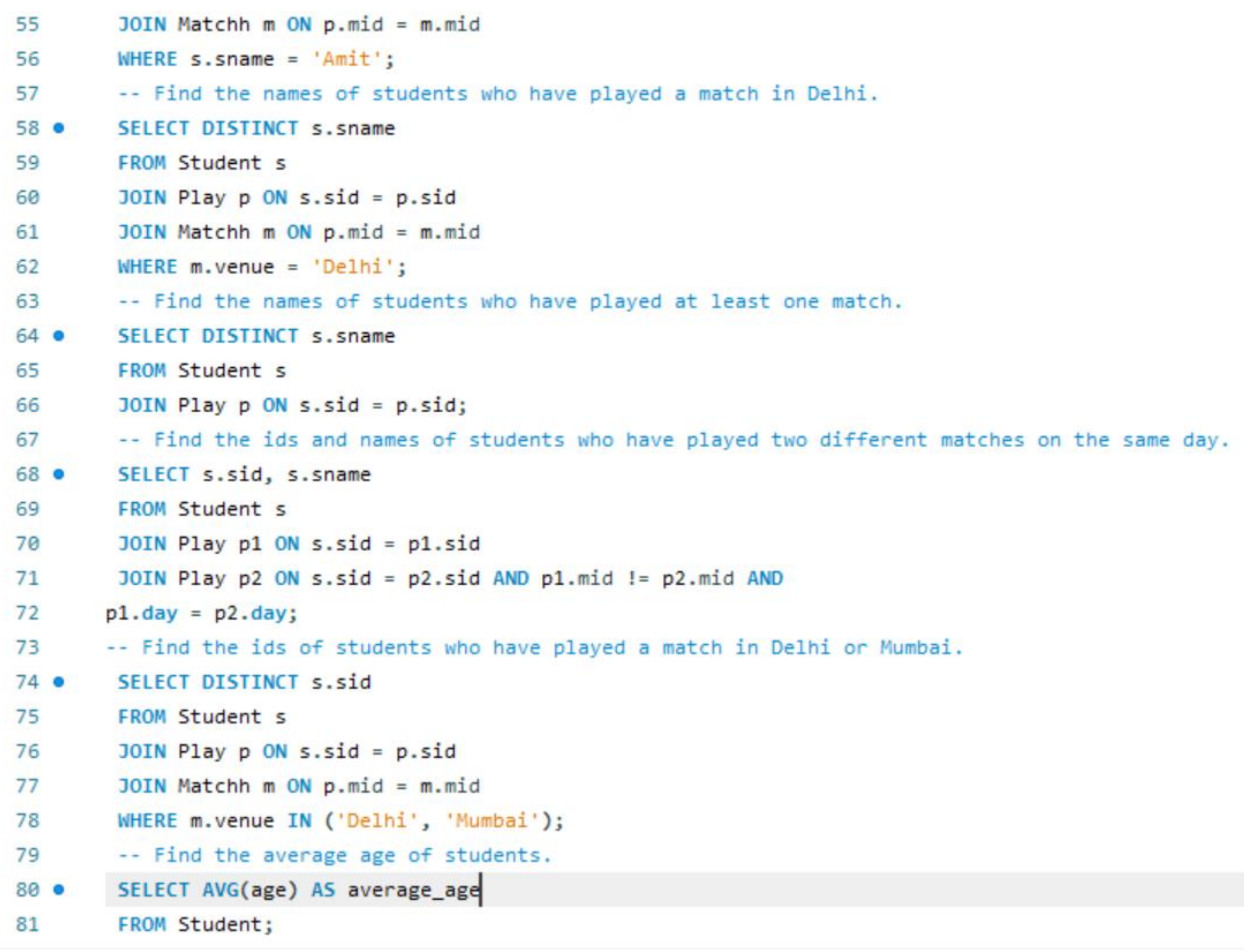
Results:



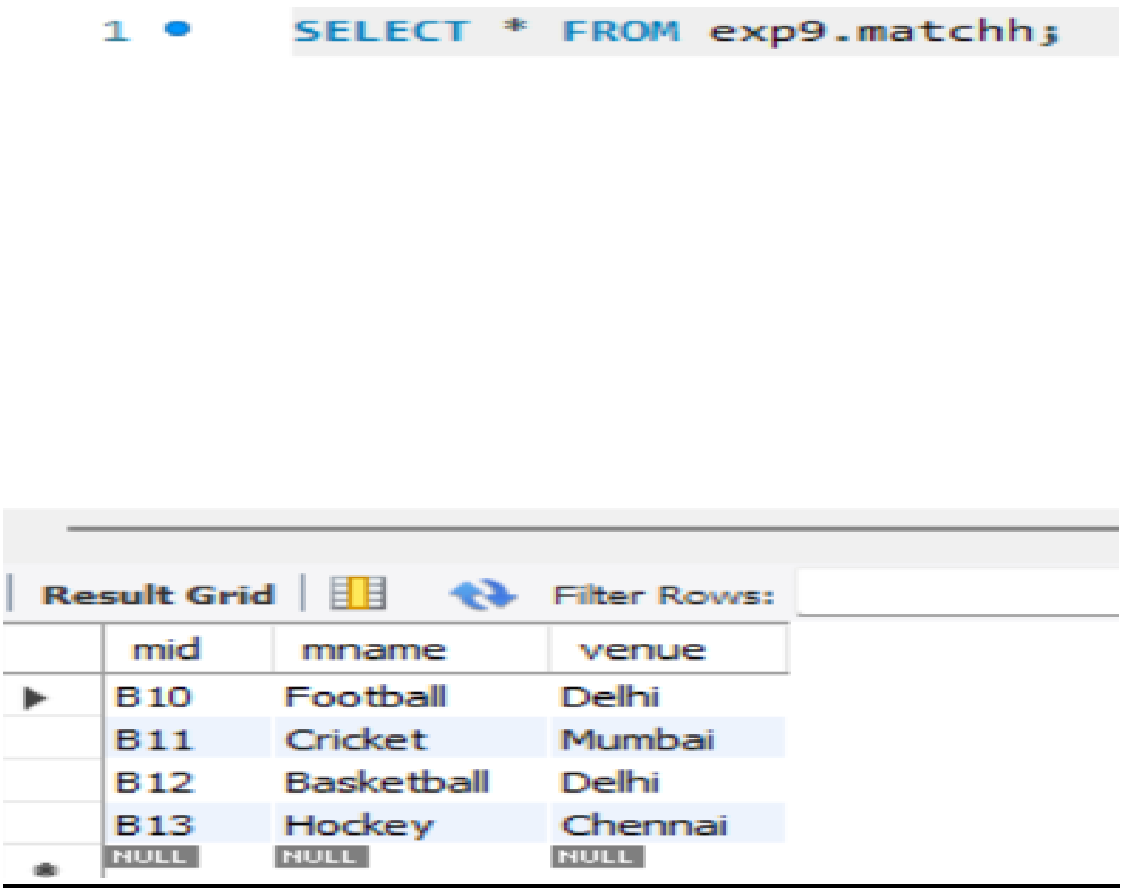
```
-- Ayush Vashishth
-- 500119331

CREATE DATABASE exp9;
USE exp9;

-- Creating Tables:
 CREATE TABLE Student (
 sid INT PRIMARY KEY,
 sname VARCHAR(50),
 age INT
 );

 CREATE TABLE Matchh (
 mid VARCHAR(10) PRIMARY KEY,
 mname VARCHAR(50),
 venue VARCHAR(50)
 );
 CREATE TABLE Play (
 sid INT,
 mid VARCHAR(10),
 day DATE,
 PRIMARY KEY (sid, mid),
 FOREIGN KEY (sid) REFERENCES Student(sid),
 FOREIGN KEY (mid) REFERENCES Matchh(mid)
 );
```



```sql
-- Populating Tables:
INSERT INTO Student (sid, sname, age) VALUES
(1, 'Amit', 20),
(2, 'Ravi', 22),
(3, 'Suresh', 19),
(4, 'Priya', 21);
INSERT INTO Matchh (mid, mname, venue) VALUES
('B10', 'Football', 'Delhi'),
('B11', 'Cricket', 'Mumbai'),
('B12', 'Basketball', 'Delhi'),
('B13', 'Hockey', 'Chennai');
INSERT INTO Play (sid, mid, day) VALUES
(1, 'B10', '2024-09-01'),
(2, 'B11', '2024-09-01'),
(1, 'B12', '2024-09-02'),
(3, 'B10', '2024-09-03'),
(4, 'B11', '2024-09-04');

-- Find all information of students who have played matchh number B10.
SELECT s.*
FROM Student s
JOIN Play p ON s.sid = p.sid
WHERE p.mid = 'B10';
-- Find the name of matches played by Amit.
SELECT m.mname
FROM Student s
JOIN Play p ON s.sid = p.sid
```



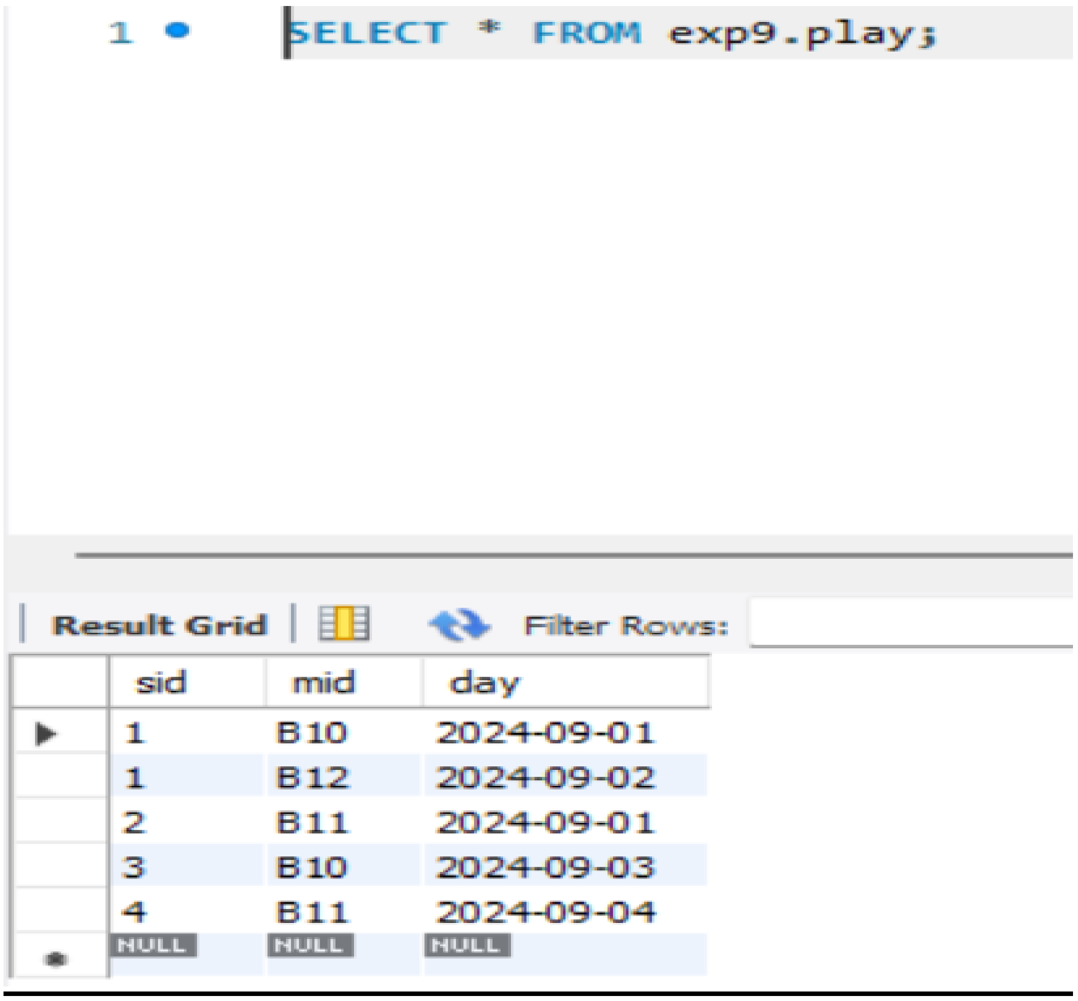
```sql
JOIN Matchh m ON p.mid = m.mid
WHERE s.sname = 'Amit';
-- Find the names of students who have played a match in Delhi.
SELECT DISTINCT s.sname
FROM Student s
JOIN Play p ON s.sid = p.sid
JOIN Matchh m ON p.mid = m.mid
WHERE m.venue = 'Delhi';
-- Find the names of students who have played at least one match.
SELECT DISTINCT s.sname
FROM Student s
JOIN Play p ON s.sid = p.sid;
-- Find the ids and names of students who have played two different matches on the same day.
SELECT s.sid, s.sname
FROM Student s
JOIN Play p1 ON s.sid = p1.sid
JOIN Play p2 ON s.sid = p2.sid AND p1.mid != p2.mid AND
p1.day = p2.day;
-- Find the ids of students who have played a match in Delhi or Mumbai.
SELECT DISTINCT s.sid
FROM Student s
JOIN Play p ON s.sid = p.sid
JOIN Matchh m ON p.mid = m.mid
WHERE m.venue IN ('Delhi', 'Mumbai');
-- Find the average age of students.
SELECT AVG(age) AS average_age
FROM Student;
```



```
SELECT * FROM exp9.matchh;
```

Result Grid | Filter Rows:

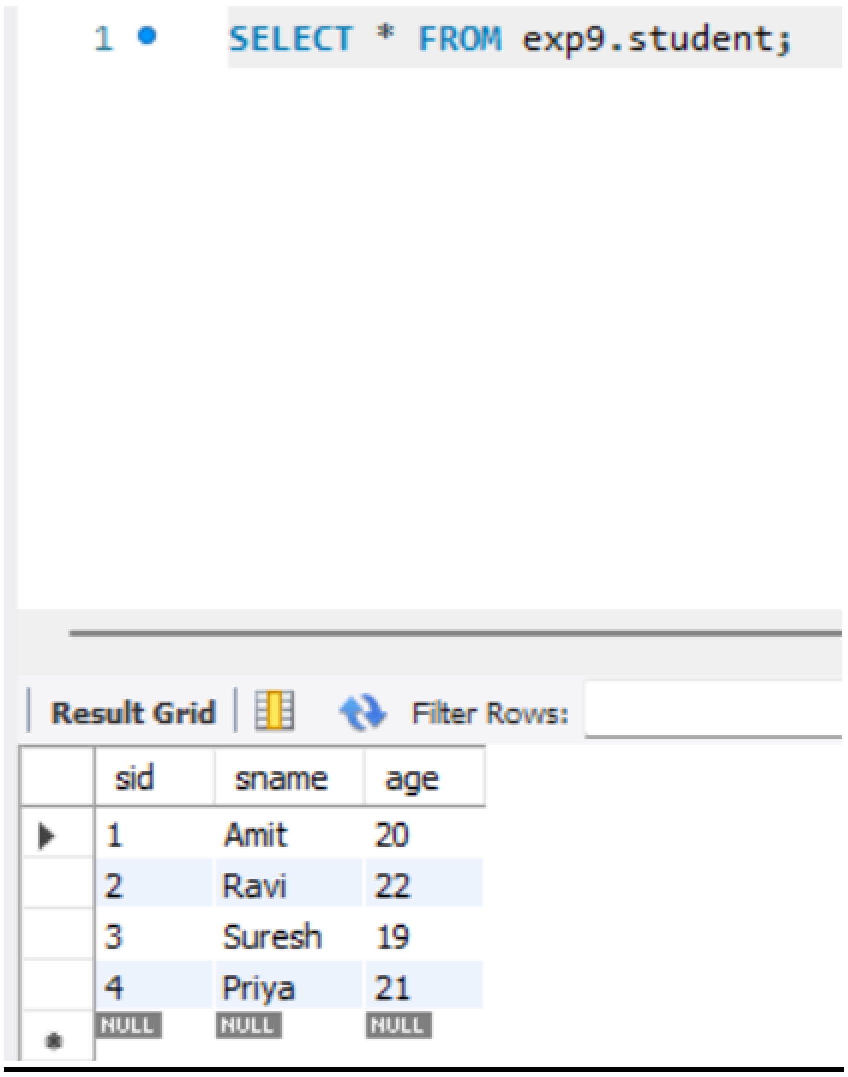
	mid	mname	venue
▶	B10	Football	Delhi
	B11	Cricket	Mumbai
	B12	Basketball	Delhi
	B13	Hockey	Chennai
	NULL	NULL	NULL



```
SELECT * FROM exp9.play;
```

Result Grid | Filter Rows:

	sid	mid	day
▶	1	B10	2024-09-01
	1	B12	2024-09-02
	2	B11	2024-09-01
	3	B10	2024-09-03
	4	B11	2024-09-04
	NULL	NULL	NULL



```sql
SELECT * FROM exp9.student;
```

Result Grid | Filter Rows:

```
  sid   sname    age
▶ 1     Amit     20
  2     Ravi     22
  3     Suresh   19
  4     Priya    21
* NULL  NULL     NULL
```

Conclusion:

The SQL code effectively establishes a relational database to manage student and match data. Various queries demonstrate the ability to analyze student participation in matches based on specific criteria, such as match location and student demographics. The design supports efficient data retrieval and can be expanded for additional analyses, such as tracking performance or more detailed demographic studies.

Future enhancements could involve adding more attributes to students and matches, such as performance statistics or additional match details, to provide deeper insights into student engagement in sports.