PROJECT NAME:

IPL 2025 REPORT

NAME: PRITESH BHARAT GAWADE

INTRODUCTION:

The Indian Premier League (IPL) has established itself as one of the most popular and competitive cricket leagues in the world, drawing attention from millions of fans globally. The IPL 2025 season promises to be even more exciting, with new players, strategies, and teams all vying for the championship. Given the vast amount of data generated during each season, analyzing and understanding this data becomes essential for sports analysts, fans, and stakeholders.

This SQL project focuses on creating a comprehensive IPL 2025 report by utilizing various database management and SQL techniques. The goal is to analyze the performance metrics of teams and players, match statistics, and other key aspects of the season. The project will involve tasks such as querying data to extract relevant insights, organizing large datasets, and visualizing the results to provide a clear understanding of team and player performance.

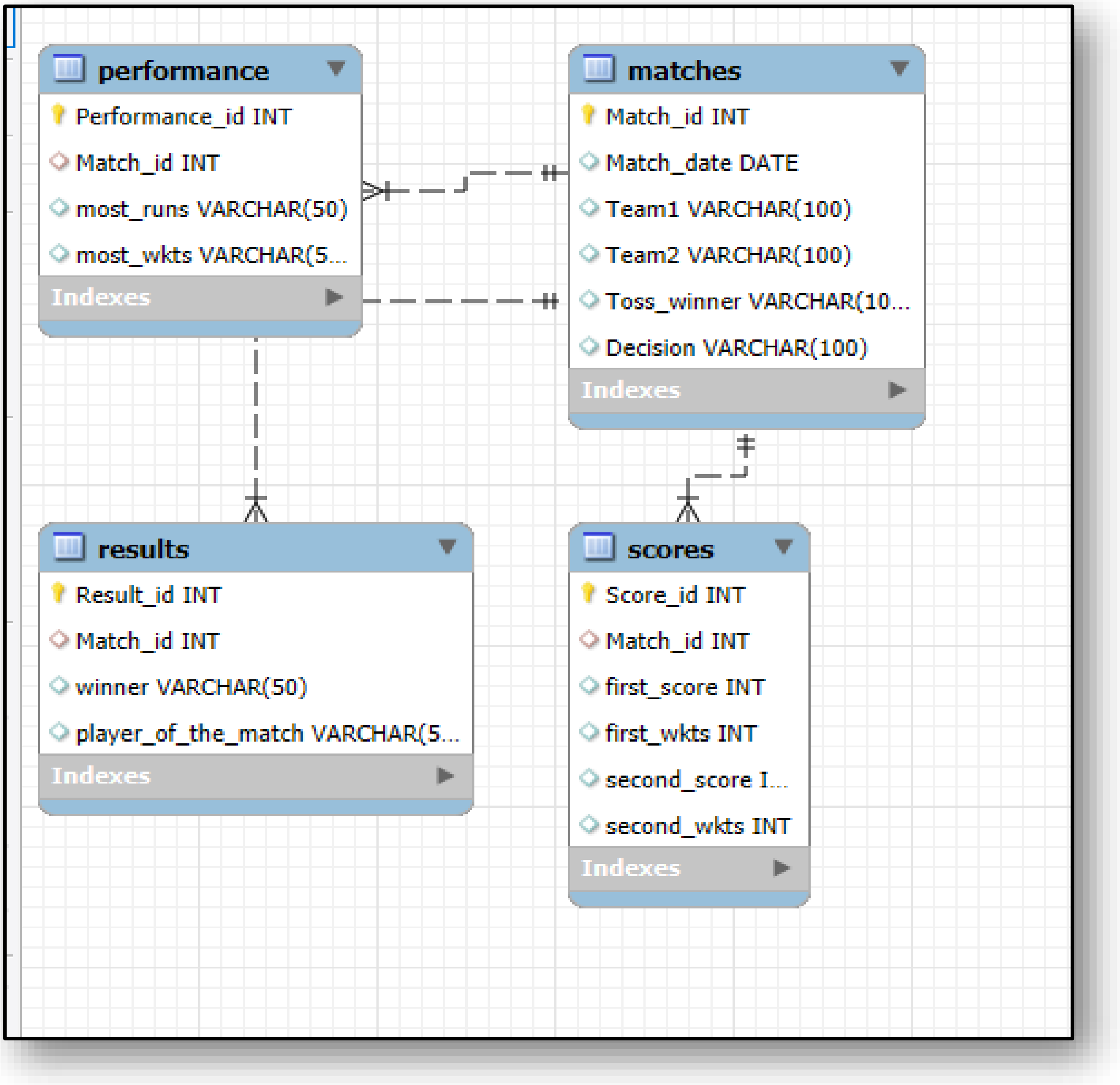
ABSTRACT:

The Indian Premier League (IPL) is one of the most celebrated cricket tournaments, drawing significant attention globally due to its competitive nature and star-studded lineup. This SQL project aims to create a detailed analytical report on the IPL 2025 season by employing structured query language (SQL) to manage, query, and analyze large datasets. The project involves extracting and processing key data points related to player performances, match statistics, team rankings, and tournament outcomes.

The primary focus is to deliver insights on critical aspects of the season, including top-performing players, win/loss ratios, batting and bowling averages, and trends in team performance. SQL queries will be designed to derive these insights from complex datasets, while also visualizing patterns that reveal the underlying dynamics of the tournament.

The IPL 2025 report aims to provide a data-driven narrative that enhances understanding of player and team performance throughout the season. This analysis is beneficial for sports analysts, enthusiasts, and stakeholders interested in deriving meaningful conclusions from large sports datasets. Through SQL's power to manipulate and analyze data efficiently, this project demonstrates the importance of database management in sports analytics.

ER-DIAGRAM: -



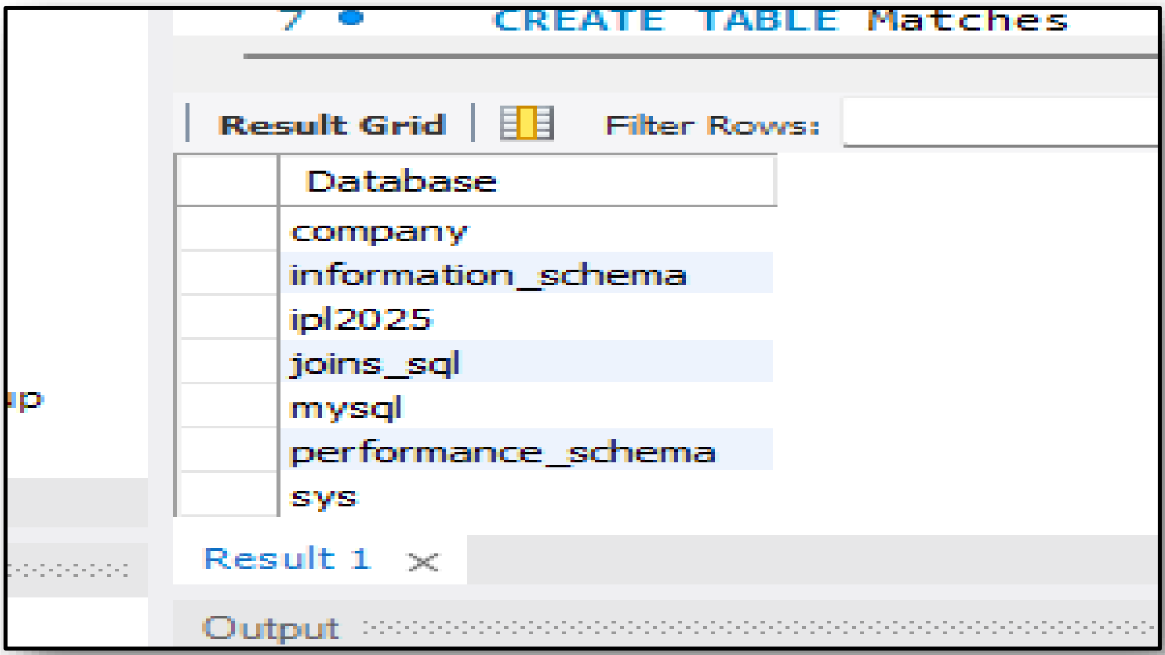
performance
Performance_id INT
Match_id INT
most_runs VARCHAR(50)
most_wkts VARCHAR(5...
Indexes
matches
Match_id INT
Match_date DATE
Team1 VARCHAR(100)
Team2 VARCHAR(100)
Toss_winner VARCHAR(10...
Decision VARCHAR(100)
Indexes
results
Result_id INT
Match_id INT
winner VARCHAR(50)
player_of_the_match VARCHAR(5...
Indexes
scores
Score_id INT
Match_id INT
first_score INT
first_wkts INT
second_score I...
second_wkts INT
Indexes

Databases:

create database ipl2025;

show databases;

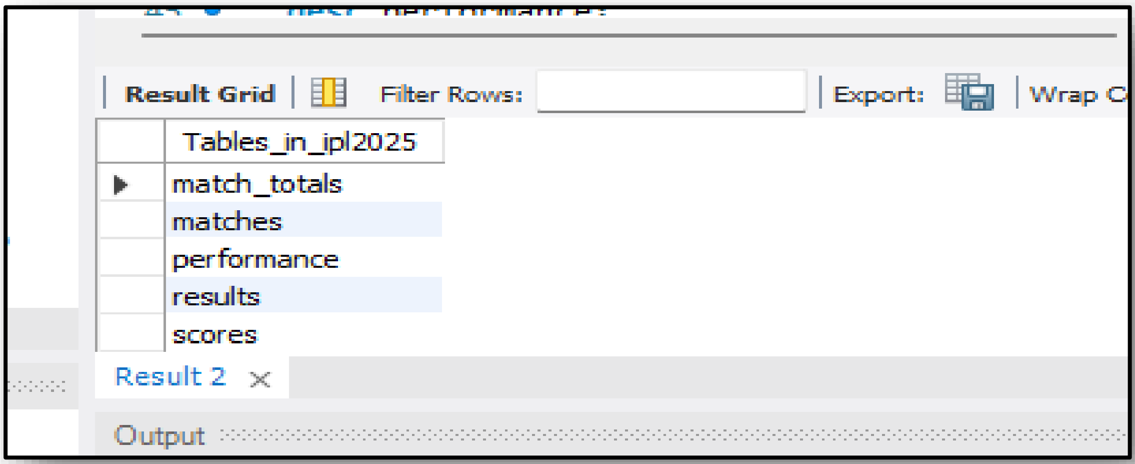
use ipl2025;



```
7 ●      CREATE TABLE Matches

Result Grid | Filter Rows:

Database
company
information_schema
ipl2025
joins_sql
mysql
performance_schema
sys

Result 1
Output
```

Tables in IPL 2025 Database:

show tables;



```
Result Grid | Filter Rows: | Export: | Wrap C

Tables_in_ipl2025
match_totals
matches
performance
results
scores

Result 2
Output
```

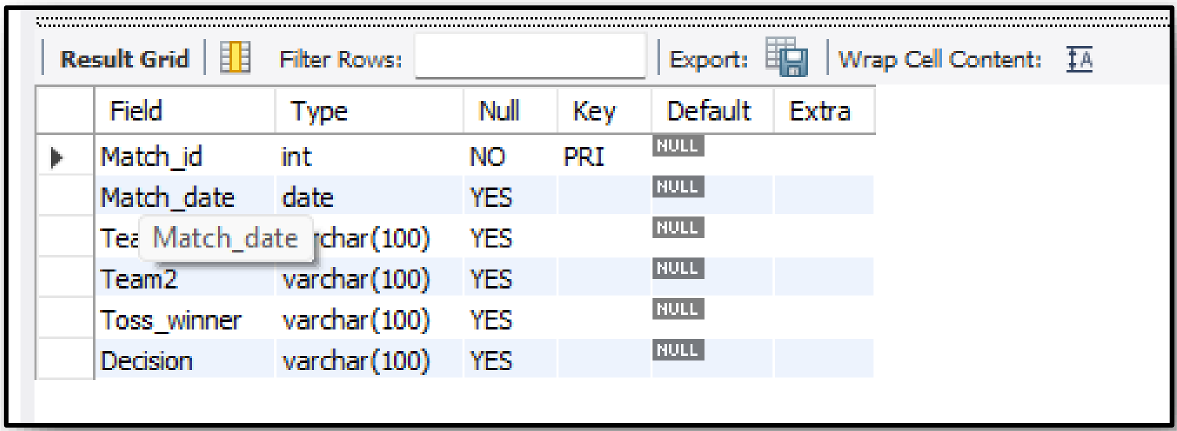
1.DATA DEFINITION LANGUAGE (DDL):

1.Creating Tables:

A) Matches

Create table Matches (Match_id int primary key, Match_date DATE, Team1 varchar(100), Team2 varchar(100), Toss_winner varchar(100), Decision varchar(100));

Desc Matches;



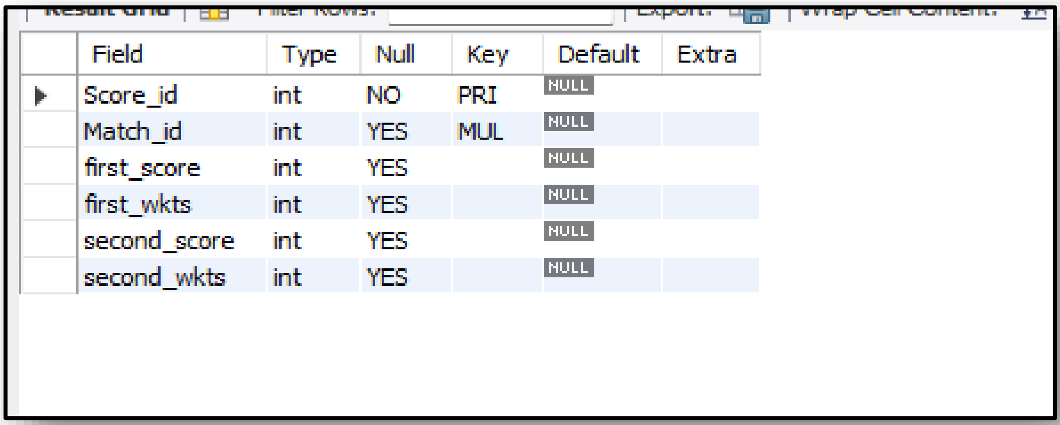
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Field	Type	Null	Key	Default	Extra
▶	Match_id	int	NO	PRI	NULL	
	Match_date	date	YES		NULL	
	Tea Match_date	rchar(100)	YES		NULL	
	Team2	varchar(100)	YES		NULL	
	Toss_winner	varchar(100)	YES		NULL	
	Decision	varchar(100)	YES		NULL	

B) Scores

Create table Scores (Score_id int primary key, Match_id int, first_score int, first_wkts int, second_score int, second_wkts int, foreign key (Match_id) reference Matches(Match_id));

Desc Score;

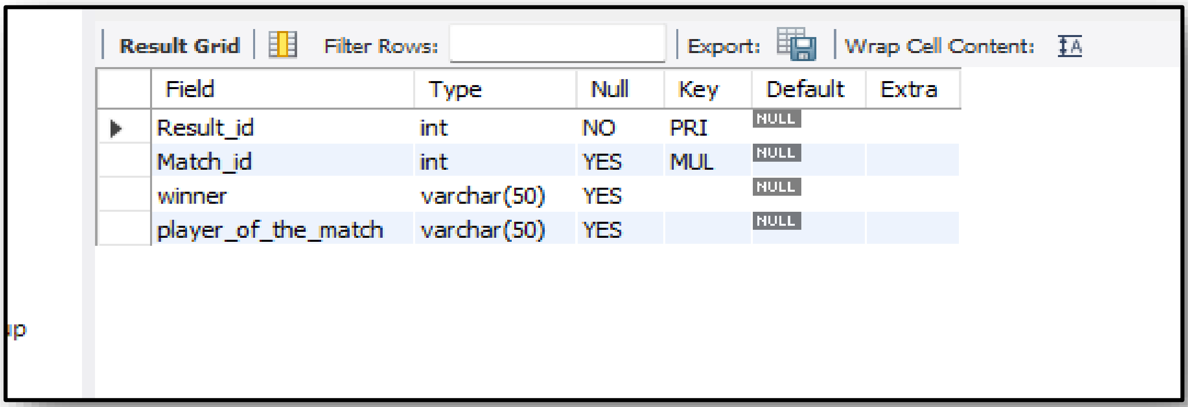


	Field	Type	Null	Key	Default	Extra
▶	Score_id	int	NO	PRI	NULL	
	Match_id	int	YES	MUL	NULL	
	first_score	int	YES		NULL	
	first_wkts	int	YES		NULL	
	second_score	int	YES		NULL	
	second_wkts	int	YES		NULL	

C) Results

Create table Results (Result_id int primary key, Match_id int, winner varchar(50), player_of_the_match varchar(50), foreign key (match_id) reference Matches(match_id));

Desc Results;



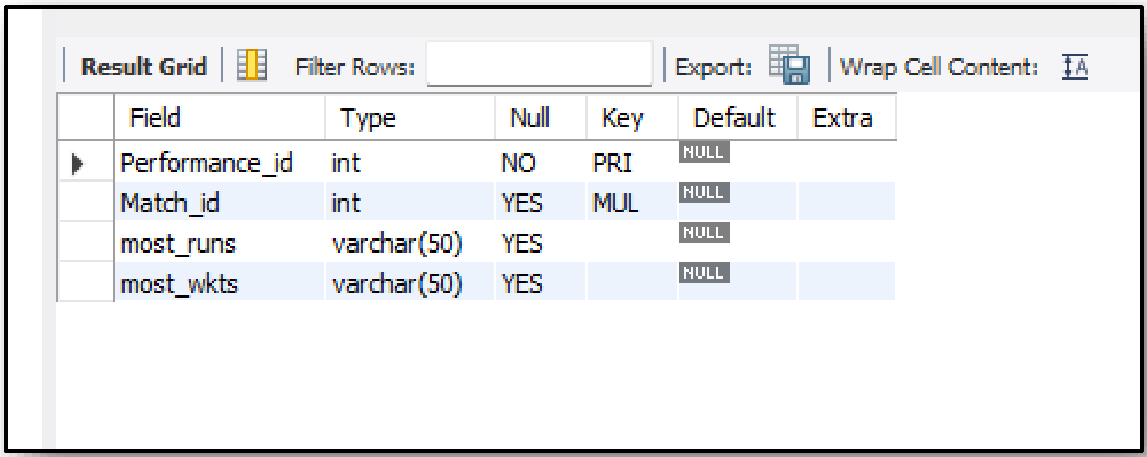
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Result_id	int	NO	PRI	NULL	
Match_id	int	YES	MUL	NULL	
winner	varchar(50)	YES		NULL	
player_of_the_match	varchar(50)	YES		NULL	

D) Performance

Create table Performance (Performance_id int primary key, Match_id int, most_runs varchar(50), most_wkts varchar(50), foreign key (Match_id) reference Matches(Match_id));

Desc Performance;

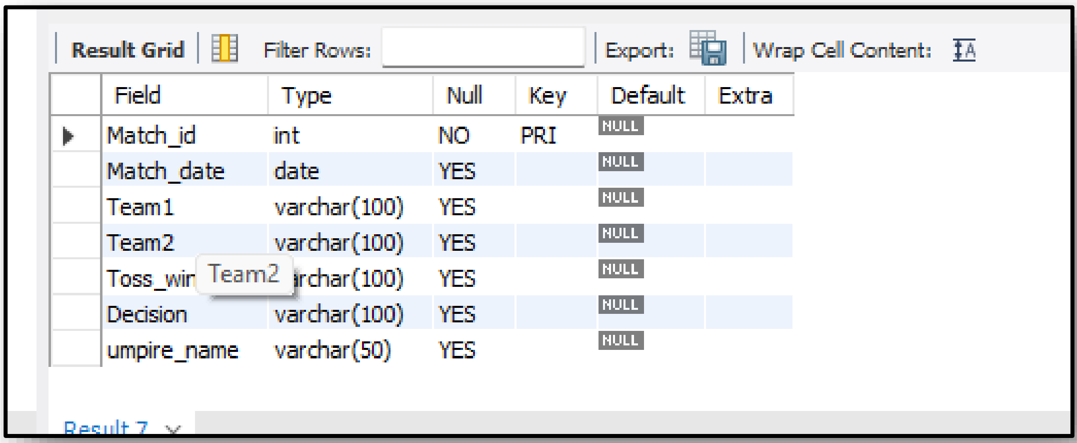


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Performance_id	int	NO	PRI	NULL	
Match_id	int	YES	MUL	NULL	
most_runs	varchar(50)	YES		NULL	
most_wkts	varchar(50)	YES		NULL	

2.Alter table:

- **Alter Table: Add column**
  alter table matches add column umpire_name varchar(50);



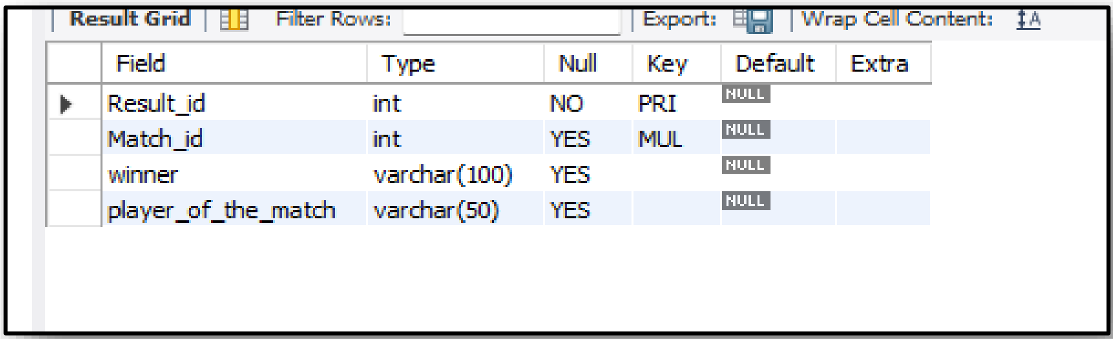
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Field	Type	Null	Key	Default	Extra
▶	Match_id	int	NO	PRI	NULL	
	Match_date	date	YES		NULL	
	Team1	varchar(100)	YES		NULL	
	Team2	varchar(100)	YES		NULL	
	Toss_wir Team2	archar(100)	YES		NULL	
	Decision	varchar(100)	YES		NULL	
	umpire_name	varchar(50)	YES		NULL	

Result 7

- **Alter Table: Modify Column**

  alter table results modify winner varchar(100);

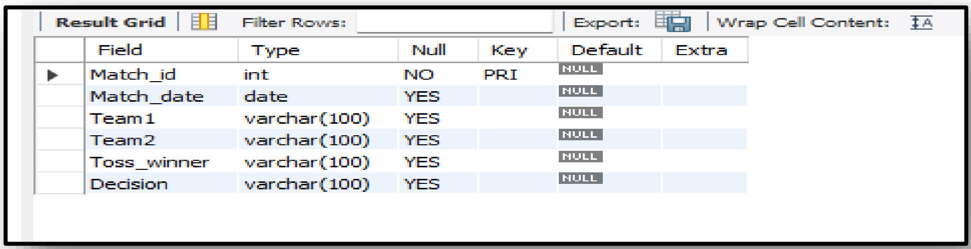


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Field	Type	Null	Key	Default	Extra
▶	Result_id	int	NO	PRI	NULL	
	Match_id	int	YES	MUL	NULL	
	winner	varchar(100)	YES		NULL	
	player_of_the_match	varchar(50)	YES		NULL	

- **Alter Table: Drop column**

  alter table matches drop column umpire_name ;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Field	Type	Null	Key	Default	Extra
▶	Match_id	int	NO	PRI	NULL	
	Match_date	date	YES		NULL	
	Team1	varchar(100)	YES		NULL	
	Team2	varchar(100)	YES		NULL	
	Toss_winner	varchar(100)	YES		NULL	
	Decision	varchar(100)	YES		NULL	

4.Drop Table:

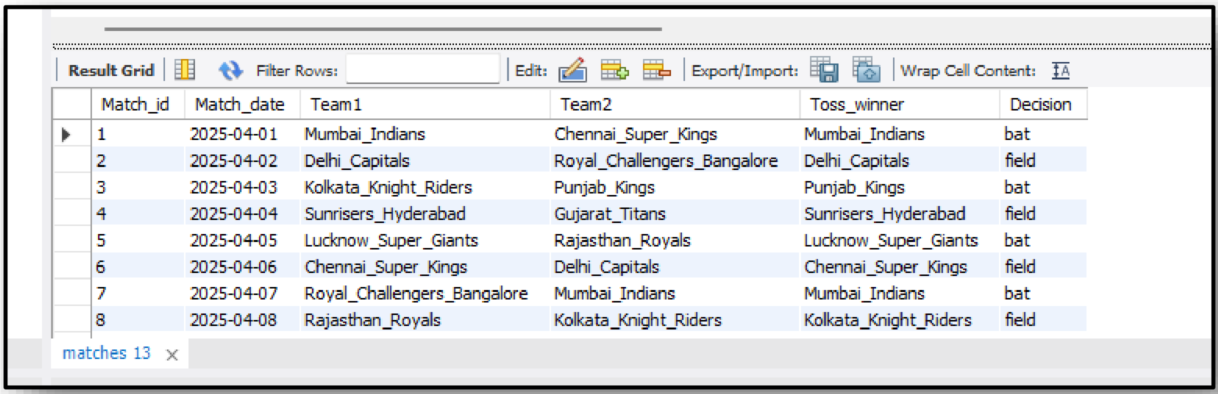
Drop table Players;

2.DATA MANIPULATION LANGUAGE (DML):

1. Insert into table:

Insert into matches values(‘1’,’2025-04-01’,’Mumbai_Indians’,’Chennai_Super_Kings’,’Mumbai_Indians’,’bat’);

Select * from Matches;



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

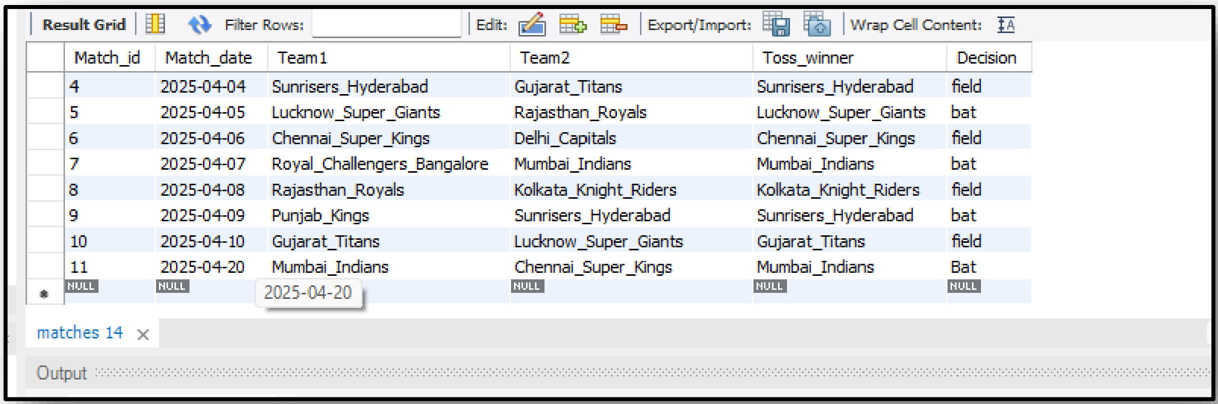
Match_id	Match_date	Team1	Team2	Toss_winner	Decision
1	2025-04-01	Mumbai_Indians	Chennai_Super_Kings	Mumbai_Indians	bat
2	2025-04-02	Delhi_Capitals	Royal_Challengers_Bangalore	Delhi_Capitals	field
3	2025-04-03	Kolkata_Knight_Riders	Punjab_Kings	Punjab_Kings	bat
4	2025-04-04	Sunrisers_Hyderabad	Gujarat_Titans	Sunrisers_Hyderabad	field
5	2025-04-05	Lucknow_Super_Giants	Rajasthan_Royals	Lucknow_Super_Giants	bat
6	2025-04-06	Chennai_Super_Kings	Delhi_Capitals	Chennai_Super_Kings	field
7	2025-04-07	Royal_Challengers_Bangalore	Mumbai_Indians	Mumbai_Indians	bat
8	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders	Kolkata_Knight_Riders	field

matches 13

2. Update into Table :

update the team1 for a match_id =11 in the Matches table to Decision'Bat' where the Toss_winner is 'team1' and the Match_date is '2025-04-16'

update Matches set team1 = 'Mumbai_Indians', Decision = 'Bat',Toss_winner ='Mumbai_Indians', Match_date = '2025-04-20' where Match_id='11';



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Match_id	Match_date	Team1	Team2	Toss_winner	Decision
4	2025-04-04	Sunrisers_Hyderabad	Gujarat_Titans	Sunrisers_Hyderabad	field
5	2025-04-05	Lucknow_Super_Giants	Rajasthan_Royals	Lucknow_Super_Giants	bat
6	2025-04-06	Chennai_Super_Kings	Delhi_Capitals	Chennai_Super_Kings	field
7	2025-04-07	Royal_Challengers_Bangalore	Mumbai_Indians	Mumbai_Indians	bat
8	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders	Kolkata_Knight_Riders	field
9	2025-04-09	Punjab_Kings	Sunrisers_Hyderabad	Sunrisers_Hyderabad	bat
10	2025-04-10	Gujarat_Titans	Lucknow_Super_Giants	Gujarat_Titans	field
11	2025-04-20	Mumbai_Indians	Chennai_Super_Kings	Mumbai_Indians	Bat
NULL	NULL	2025-04-20	NULL	NULL	NULL

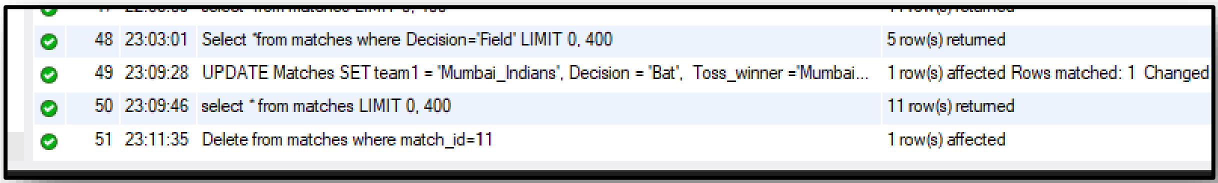
matches 14

Output

3. Delete from table:

Delete the match record from the Matches table where the Match_id is 11

Delete from matches where match_id=11;



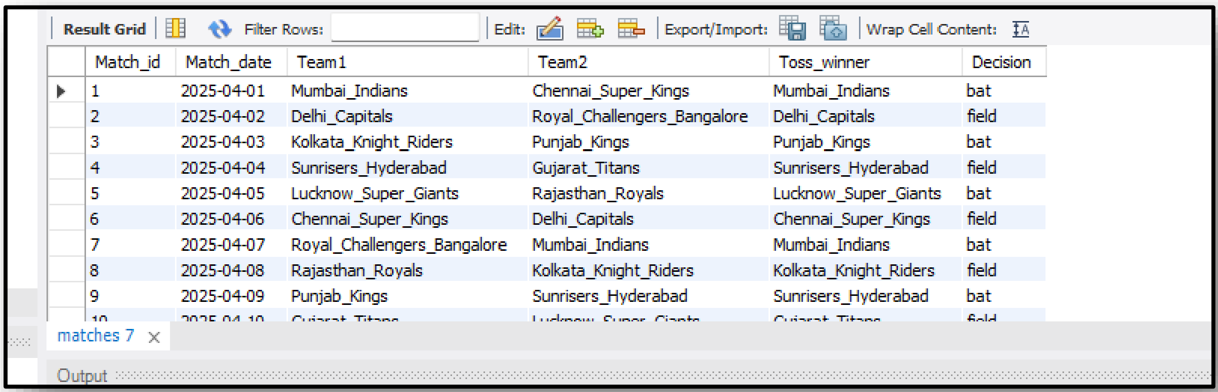
```
48  23:03:01  Select *from matches where Decision='Field' LIMIT 0, 400                                5 row(s) returned
49  23:09:28  UPDATE Matches SET team1 = 'Mumbai_Indians', Decision = 'Bat',  Toss_winner ='Mumbai...  1 row(s) affected Rows matched: 1  Changed
50  23:09:46  select * from matches LIMIT 0, 400                                                     11 row(s) returned
51  23:11:35  Delete from matches where match_id=11                                                   1 row(s) affected
```

3.DATA QUERY LANGUAGE (DQL) :

1.Select Query:

a) Select Query for entire data

select * from Matches;



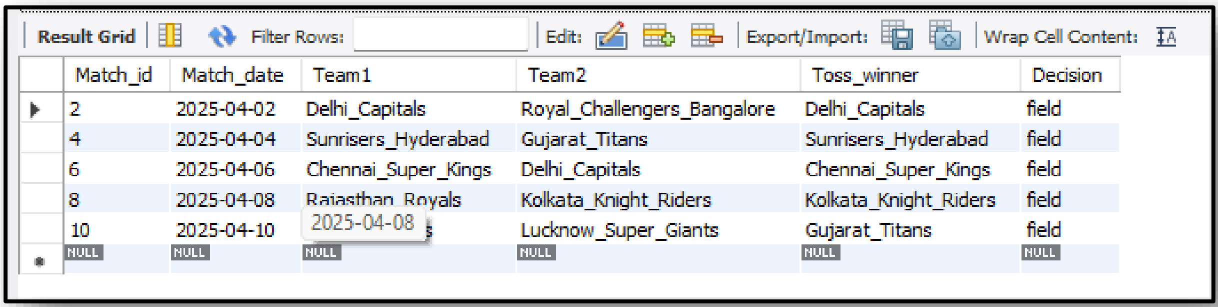
Match_id	Match_date	Team1	Team2	Toss_winner	Decision
1	2025-04-01	Mumbai_Indians	Chennai_Super_Kings	Mumbai_Indians	bat
2	2025-04-02	Delhi_Capitals	Royal_Challengers_Bangalore	Delhi_Capitals	field
3	2025-04-03	Kolkata_Knight_Riders	Punjab_Kings	Punjab_Kings	bat
4	2025-04-04	Sunrisers_Hyderabad	Gujarat_Titans	Sunrisers_Hyderabad	field
5	2025-04-05	Lucknow_Super_Giants	Rajasthan_Royals	Lucknow_Super_Giants	bat
6	2025-04-06	Chennai_Super_Kings	Delhi_Capitals	Chennai_Super_Kings	field
7	2025-04-07	Royal_Challengers_Bangalore	Mumbai_Indians	Mumbai_Indians	bat
8	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders	Kolkata_Knight_Riders	field
9	2025-04-09	Punjab_Kings	Sunrisers_Hyderabad	Sunrisers_Hyderabad	bat
10	2025-04-10	Gujarat_Titans	Lucknow_Super_Giants	Gujarat_Titans	field

matches 7

Output

b) Retrive The details of the matches where the decision after winning the toss was 'field'

Select *from matches where Decision='Field';

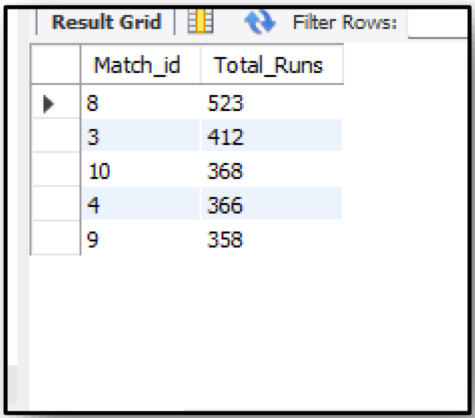


Match_id	Match_date	Team1	Team2	Toss_winner	Decision
2	2025-04-02	Delhi_Capitals	Royal_Challengers_Bangalore	Delhi_Capitals	field
4	2025-04-04	Sunrisers_Hyderabad	Gujarat_Titans	Sunrisers_Hyderabad	field
6	2025-04-06	Chennai_Super_Kings	Delhi_Capitals	Chennai_Super_Kings	field
8	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders	Kolkata_Knight_Riders	field
10	2025-04-10	2025-04-08 s	Lucknow_Super_Giants	Gujarat_Titans	field
NULL	NULL	NULL	NULL	NULL	NULL

2.Order by

Find the top 5 highest-scoring matches based on the total score of both teams

Select Match_id, (first_score + second_score) as Total_Runs from Scores order by Total_Runs desc limit 5;



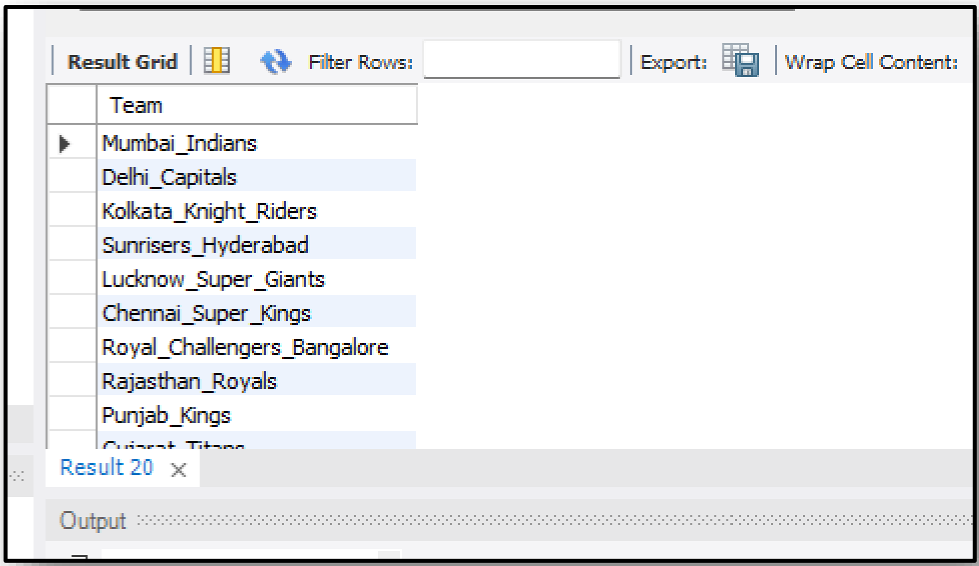
Result Grid | Filter Rows:

Match_id	Total_Runs
8	523
3	412
10	368
4	366
9	358

3.Distinct Query

Find all distinct teams that have participated in matches

Select Distinct Team1 as Team from Matches union select distinct Team2 as Team from Matches;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Team
Mumbai_Indians
Delhi_Capitals
Kolkata_Knight_Riders
Sunrisers_Hyderabad
Lucknow_Super_Giants
Chennai_Super_Kings
Royal_Challengers_Bangalore
Rajasthan_Royals
Punjab_Kings
Gujarat_Titans

Result 20

Output

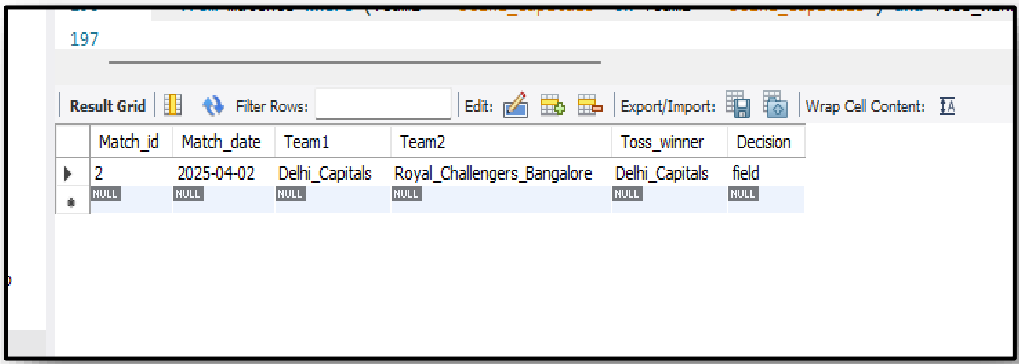
4.Where Clause:

1)Logical Operator

➢ **Using AND Operator**

Find matches where delhi capitals played and they wonn the toss

select * from matches where (Team1 = 'Delhi_Capitals' OR Team2 = 'Delhi_Capitals') and Toss_Winner = 'Delhi_Capitals';

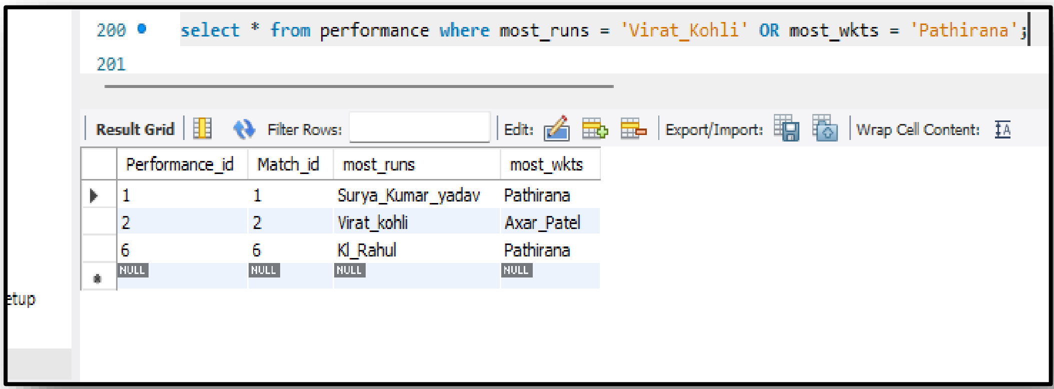


Match_id	Match_date	Team1	Team2	Toss_winner	Decision
2	2025-04-02	Delhi_Capitals	Royal_Challengers_Bangalore	Delhi_Capitals	field
NULL	NULL	NULL	NULL	NULL	NULL

➢ **Using OR Operator**

Get performance where eithr most_runs is virat kolhi or most wkts is pathiran

select * from performance where most_runs = 'Virat_Kohli' OR most_wkts = 'Pathirana';



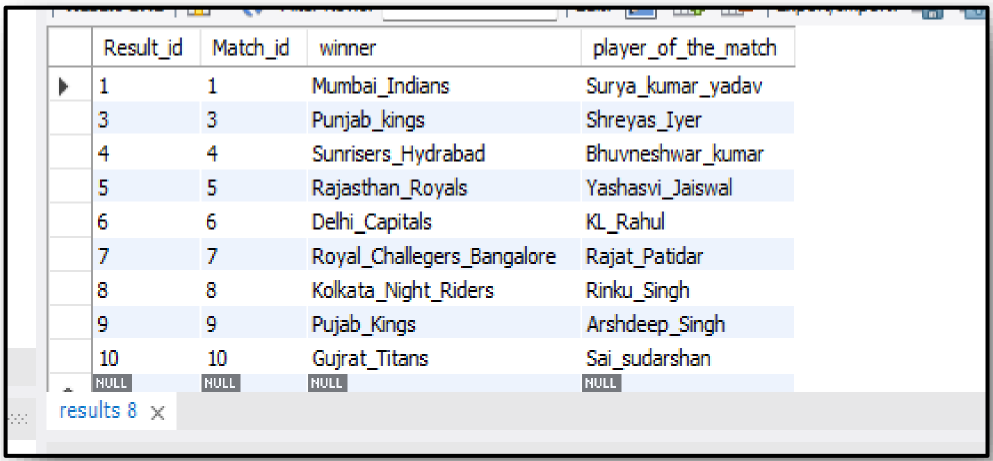
```
select * from performance where most_runs = 'Virat_Kohli' OR most_wkts = 'Pathirana';
```

Performance_id	Match_id	most_runs	most_wkts
1	1	Surya_Kumar_yadav	Pathirana
2	2	Virat_kohli	Axar_Patel
6	6	Kl_Rahul	Pathirana
NULL	NULL	NULL	NULL

➢ **Using NOT Operator**

Get all results where the player of the match is not virat kohli

select *from results where Player_of_the_match <> 'Virat_Kohli';

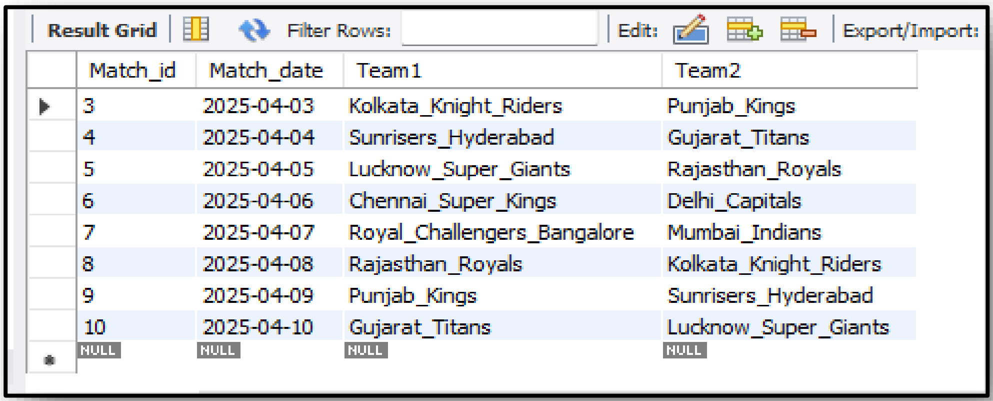


Result_id	Match_id	winner	player_of_the_match
1	1	Mumbai_Indians	Surya_kumar_yadav
3	3	Punjab_kings	Shreyas_Iyer
4	4	Sunrisers_Hydrabad	Bhuvneshwar_kumar
5	5	Rajasthan_Royals	Yashasvi_Jaiswal
6	6	Delhi_Capitals	KL_Rahul
7	7	Royal_Challegers_Bangalore	Rajat_Patidar
8	8	Kolkata_Night_Riders	Rinku_Singh
9	9	Pujab_Kings	Arshdeep_Singh
10	10	Gujrat_Titans	Sai_sudarshan
NULL	NULL	NULL	NULL

➢ **Using BETWEEN operator**

Show result where the match id is between 3 and 7

select *from results where Match_id between 3 and 7;

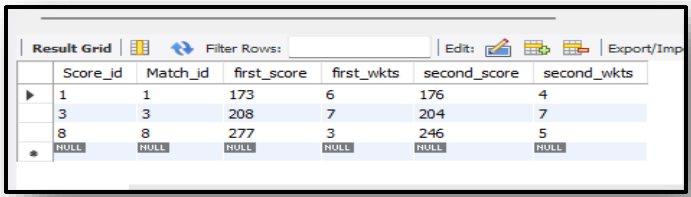


Match_id	Match_date	Team1	Team2
3	2025-04-03	Kolkata_Knight_Riders	Punjab_Kings
4	2025-04-04	Sunrisers_Hyderabad	Gujarat_Titans
5	2025-04-05	Lucknow_Super_Giants	Rajasthan_Royals
6	2025-04-06	Chennai_Super_Kings	Delhi_Capitals
7	2025-04-07	Royal_Challengers_Bangalore	Mumbai_Indians
8	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders
9	2025-04-09	Punjab_Kings	Sunrisers_Hyderabad
10	2025-04-10	Gujarat_Titans	Lucknow_Super_Giants
NULL	NULL	NULL	NULL

➢ **Using IN operator**

Show rows where the first scores is 173,208,277

select *from scores where First_Score in (173,208,277);

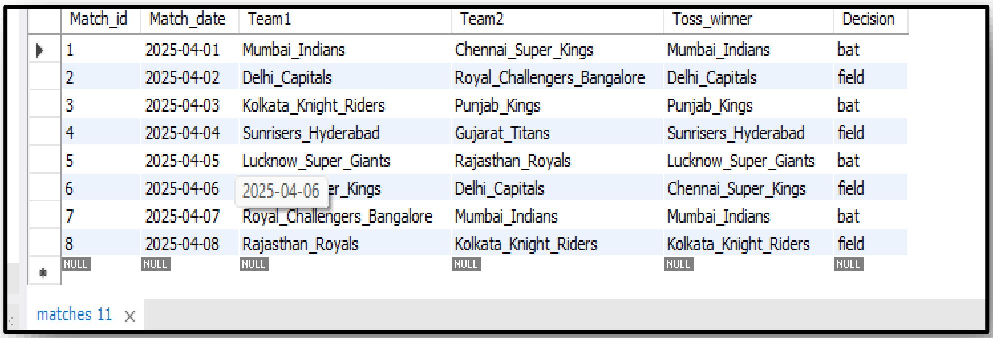


Score_id	Match_id	first_score	first_wkts	second_score	second_wkts
1	1	173	6	176	4
3	3	208	7	204	7
8	8	277	3	246	5
NULL	NULL	NULL	NULL	NULL	NULL

- **Using ANY operator**

  Find matches where match id is less than any of (7,9)

  select * from matches where Match_id < any (select val from(select 7 as val union all select 9) as t);



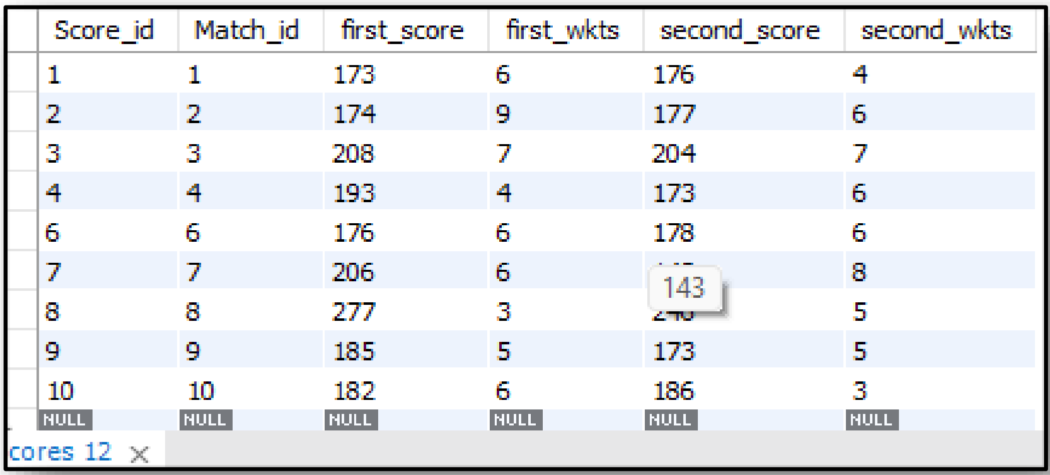
Match_id	Match_date	Team1	Team2	Toss_winner	Decision
1	2025-04-01	Mumbai_Indians	Chennai_Super_Kings	Mumbai_Indians	bat
2	2025-04-02	Delhi_Capitals	Royal_Challengers_Bangalore	Delhi_Capitals	field
3	2025-04-03	Kolkata_Knight_Riders	Punjab_Kings	Punjab_Kings	bat
4	2025-04-04	Sunrisers_Hyderabad	Gujarat_Titans	Sunrisers_Hyderabad	field
5	2025-04-05	Lucknow_Super_Giants	Rajasthan_Royals	Lucknow_Super_Giants	bat
6	2025-04-06	2025-04-06 er_Kings	Delhi_Capitals	Chennai_Super_Kings	field
7	2025-04-07	Royal_Challengers_Bangalore	Mumbai_Indians	Mumbai_Indians	bat
8	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders	Kolkata_Knight_Riders	field
NULL	NULL	NULL	NULL	NULL	NULL

matches 11 ×

- **Using ALL operator**

  Find rows where first score is greater than all of (150,160,170)

  select * from scores where First_score > all(select val from (select 150 as val union all select 160 union all select 170)as t);



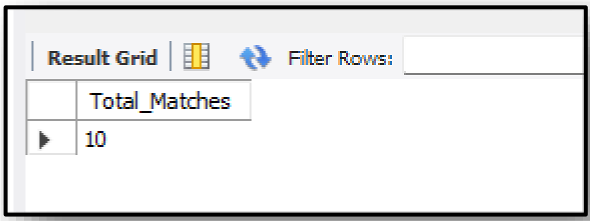
Score_id	Match_id	first_score	first_wkts	second_score	second_wkts
1	1	173	6	176	4
2	2	174	9	177	6
3	3	208	7	204	7
4	4	193	4	173	6
6	6	176	6	178	6
7	7	206	6	143	8
8	8	277	3	240	5
9	9	185	5	173	5
10	10	182	6	186	3
NULL	NULL	NULL	NULL	NULL	NULL

cores 12 ×

5.Aggregate Functions:

- **Count Function:**
  count the total number of matches played

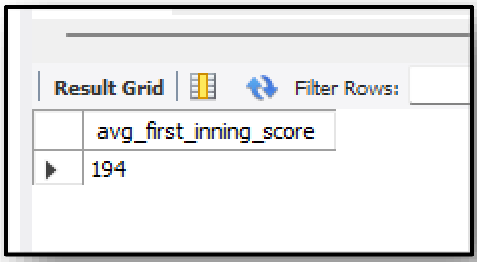
  Select count(Match_id) as Total_Matches from Matches;



```
Total_Matches
10
```

- **Average Function with round function:**
  What was the average score of teams batting first in IPL 2025

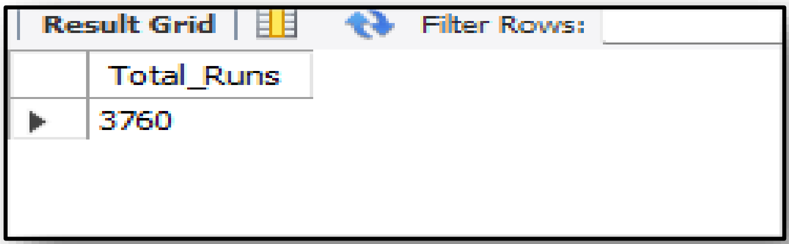
  Select round(avg(first_score)) as avg_first_inning_score from Scores join Matches on Scores.match_id = Matches.match_id;



```
avg_first_inning_score
194
```

- **Sum Function :**
  calculate the total runs scored by both teams across all matches

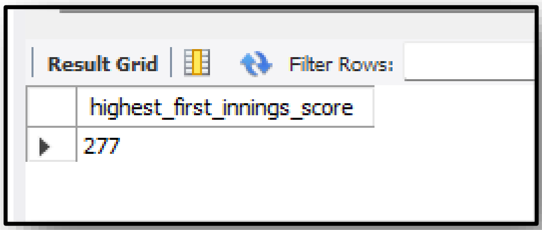
  Select sum(first_score + second_score) as Total_Runs from Scores;



```
Total_Runs
3760
```

- **Max Function:**
  What is the highest score made by any team in the first innings

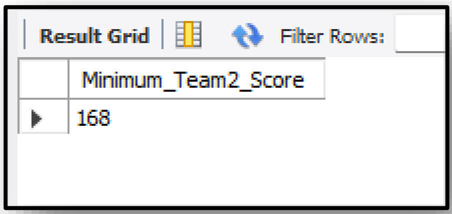
  select max(first_score) as highest_first_innings_score from Scores;



```
Result Grid | Filter Rows:
  highest_first_innings_score
▶ 277
```

- **Min Function:**
  find the minimum score by Team 2 across all matches

  Select min(first_score) as Minimum_Team2_Score from Scores where first_score>1;

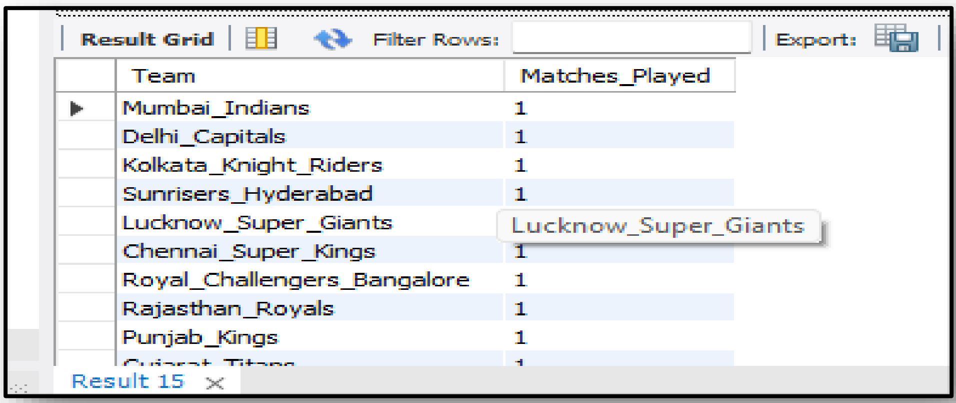


```
Result Grid | Filter Rows:
  Minimum_Team2_Score
▶ 168
```

6.Group by clause :

Find the number of matches each team has played

select Team1 as Team, count(Match_id) as Matches_Played from Matches group by Team1 union Select Team2 as Team, count(Match_id) as Matches_Played from Matches group by Team2;

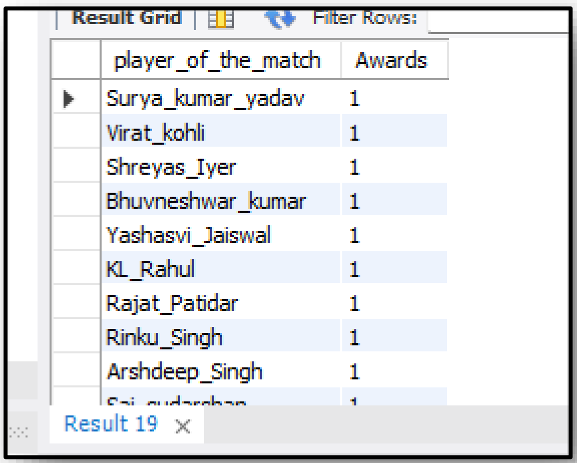


```
Result Grid | Filter Rows:                 | Export:
  Team                           Matches_Played
▶ Mumbai_Indians                 1
  Delhi_Capitals                 1
  Kolkata_Knight_Riders          1
  Sunrisers_Hyderabad            1
  Lucknow_Super_Giants           Lucknow_Super_Giants
  Chennai_Super_Kings            1
  Royal_Challengers_Bangalore    1
  Rajasthan_Royals               1
  Punjab_Kings                   1
  Gujarat_Titans                 1
Result 15 ×
```

7.Having Clause:

Find players who have won "Player of the Match" more than once

Select player_of_the_match, count(result_id) as Awards from Results group by player_of_the_match having count(result_id) > 0;

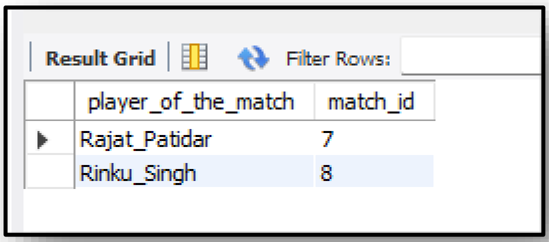


player_of_the_match	Awards
Surya_kumar_yadav	1
Virat_kohli	1
Shreyas_Iyer	1
Bhuvneshwar_kumar	1
Yashasvi_Jaiswal	1
KL_Rahul	1
Rajat_Patidar	1
Rinku_Singh	1
Arshdeep_Singh	1
Sai_sudarshan	1

8.Like Operator :

Find all players who won "Player of the Match" and have a name starting with the letter "R"

Select player_of_the_match, match_id from Results where player_of_the_match like 'R%';

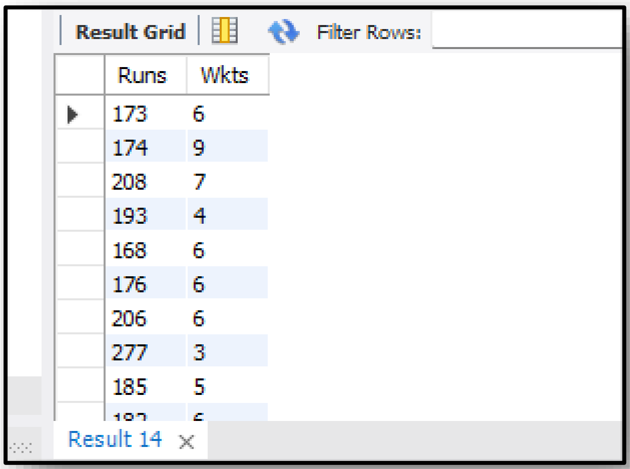


player_of_the_match	match_id
Rajat_Patidar	7
Rinku_Singh	8

9.Union :

Combine both scores and wickes together

select first_score as Runs, first_wkts as Wkts from scores union select second_score, second_wkts from scores;



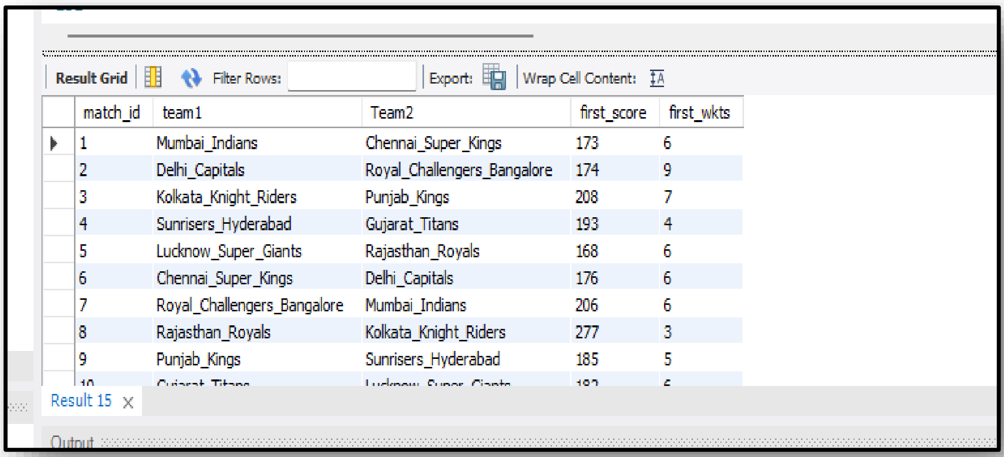
Runs	Wkts
173	6
174	9
208	7
193	4
168	6
176	6
206	6
277	3
185	5

10.Joins :

A) Inner Join:

Get match details and scores together

select m.match_id, m.team1, m.Team2, s.first_score, s.first_wkts from matches m inner join scores s on m.Match_id = s.Match_id;

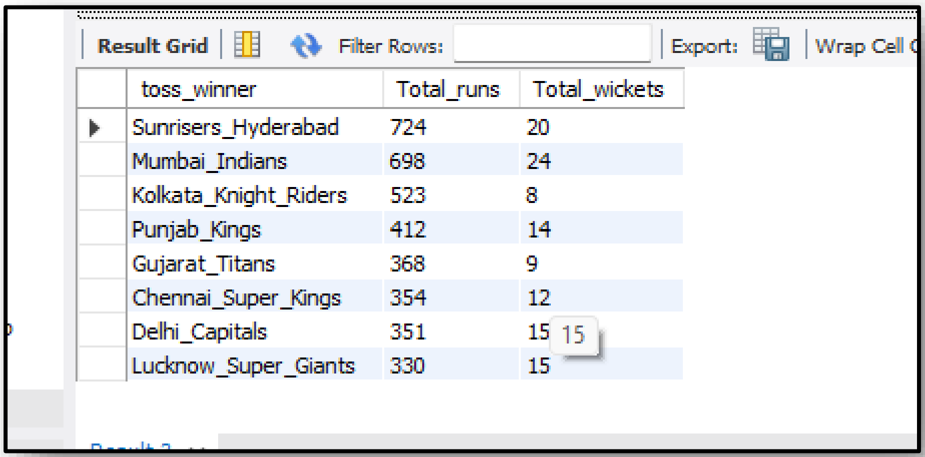


match_id	team1	Team2	first_score	first_wkts
1	Mumbai_Indians	Chennai_Super_Kings	173	6
2	Delhi_Capitals	Royal_Challengers_Bangalore	174	9
3	Kolkata_Knight_Riders	Punjab_Kings	208	7
4	Sunrisers_Hyderabad	Gujarat_Titans	193	4
5	Lucknow_Super_Giants	Rajasthan_Royals	168	6
6	Chennai_Super_Kings	Delhi_Capitals	176	6
7	Royal_Challengers_Bangalore	Mumbai_Indians	206	6
8	Rajasthan_Royals	Kolkata_Knight_Riders	277	3
9	Punjab_Kings	Sunrisers_Hyderabad	185	5

B) Left Join:

What is the performance summary (total runs, total wickets) for the toss-winning team in each match..

select Matches.toss_winner, sum(Scores.first_score + Scores.second_score) as Total_runs, sum(Scores.first_wkts + Scores.second_wkts) as Total_wickets from Matches left join Scores on Matches.match_id = Scores.match_id where year(Matches.match_date) = 2025 group by Matches.toss_winner order by Total_runs desc;

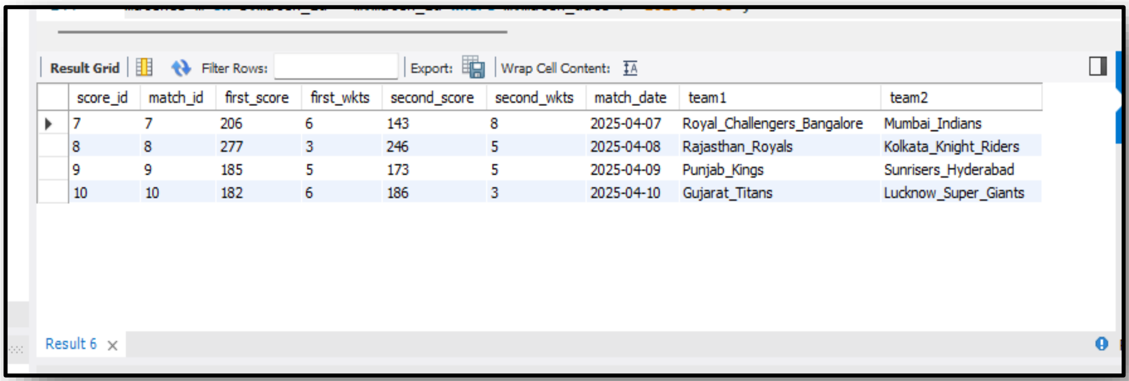


toss_winner	Total_runs	Total_wickets
Sunrisers_Hyderabad	724	20
Mumbai_Indians	698	24
Kolkata_Knight_Riders	523	8
Punjab_Kings	412	14
Gujarat_Titans	368	9
Chennai_Super_Kings	354	12
Delhi_Capitals	351	15
Lucknow_Super_Giants	330	15

C) Right Join:

Show only matches played after 2025-04-01

select s.score_id,s.match_id, s.first_score, s.first_wkts, s.second_score, s.second_wkts, m.match_date, m.team1, m.team2 from scores s right join matches m on s.match_id = m.match_id where m.match_date > '2025-04-06';



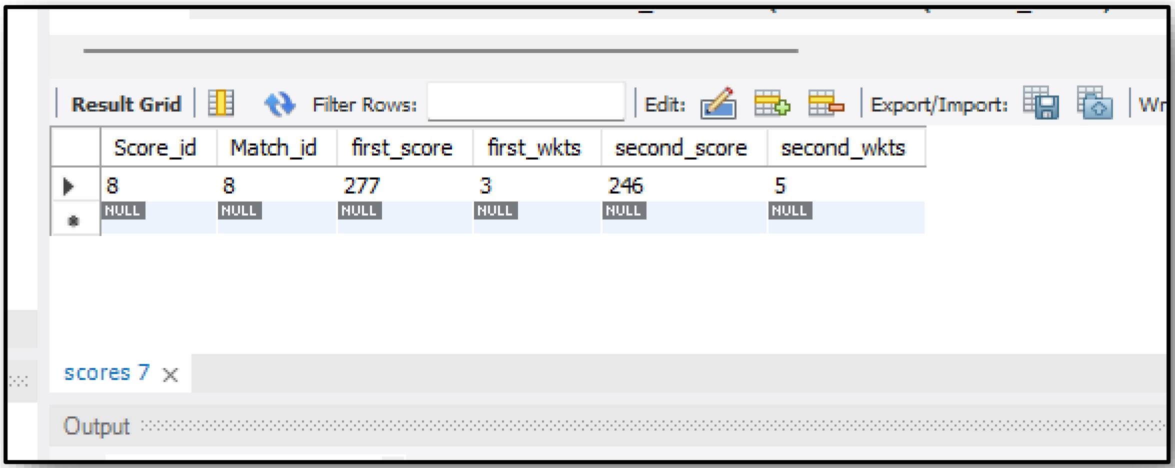
score_id	match_id	first_score	first_wkts	second_score	second_wkts	match_date	team1	team2
7	7	206	6	143	8	2025-04-07	Royal_Challengers_Bangalore	Mumbai_Indians
8	8	277	3	246	5	2025-04-08	Rajasthan_Royals	Kolkata_Knight_Riders
9	9	185	5	173	5	2025-04-09	Punjab_Kings	Sunrisers_Hyderabad
10	10	182	6	186	3	2025-04-10	Gujarat_Titans	Lucknow_Super_Giants

11.Subqueries:

1.Single row Subqueries:

Find all scores where first score is higher than the highest second score

select * from scores where first_score > (select max(second_score) from scores);



Result Grid | Filter Rows: | Edit: | Export/Import: | Wr

	Score_id	Match_id	first_score	first_wkts	second_score	second_wkts
▶	8	8	277	3	246	5
●	NULL	NULL	NULL	NULL	NULL	NULL

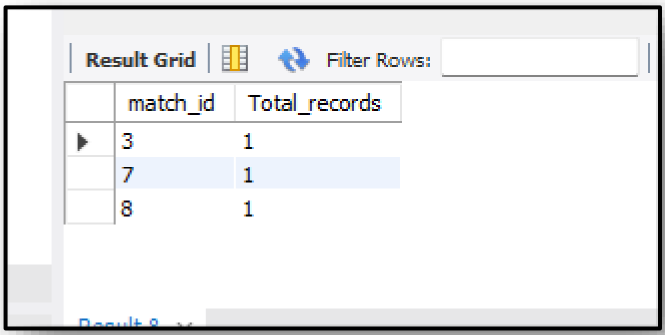
scores 7 ×

Output

2.Multiple row subquery:

Display all match id and the count of records per match , only for matches where the avg first score exceeds 200.

select match_id, count(*) as Total_records from scores group by match_id having avg(first_score) > 200;



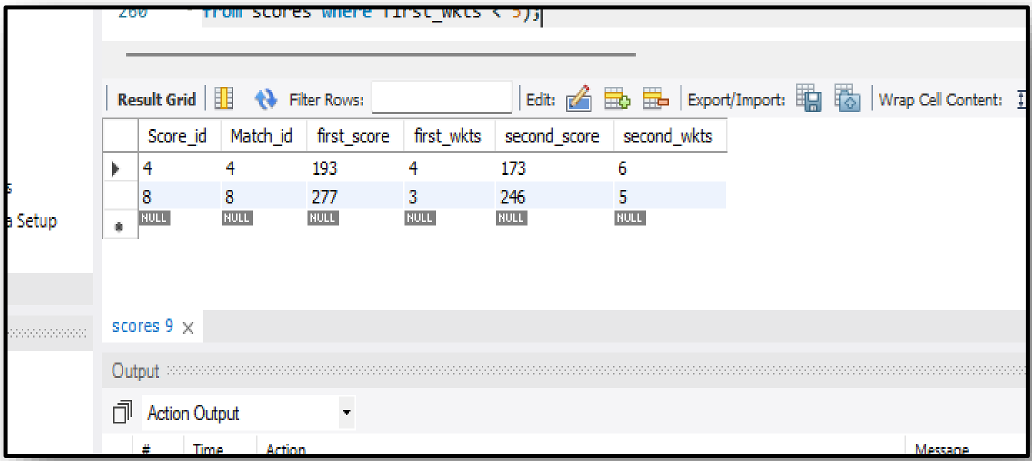
Result Grid | Filter Rows:

	match_id	Total_records
▶	3	1
	7	1
	8	1

3.Multiple column subquery :

All records whose first score, second score pair matches any first score , second score pair where first wkts is less than 5.

select * from scores where (first_score, second_score) in (select first_score, second_score from scores where first_wkts < 5);



```
from scores where first_wkts < 5);
```

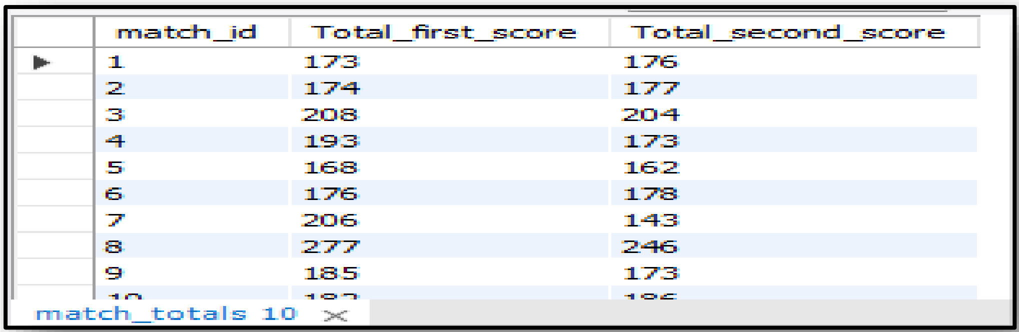
Score_id	Match_id	first_score	first_wkts	second_score	second_wkts
4	4	193	4	173	6
8	8	277	3	246	5
NULL	NULL	NULL	NULL	NULL	NULL

12.VIEW:

Create view showing each match id with total first score and total second score per match.

create view match_totals as select match_id, sum(first_score) as Total_first_score, sum(second_score) as Total_second_score from scores group by match_id;

select* from match_totals;



match_id	Total_first_score	Total_second_score
1	173	176
2	174	177
3	208	204
4	193	173
5	168	162
6	176	178
7	206	143
8	277	246
9	185	173
10	[illegible]	[illegible]

ROYAL CHALLENGERS BENGALURU
CHALLENGERS BE
QATAR AIRWAYS
NOTHING

===========================END==========================