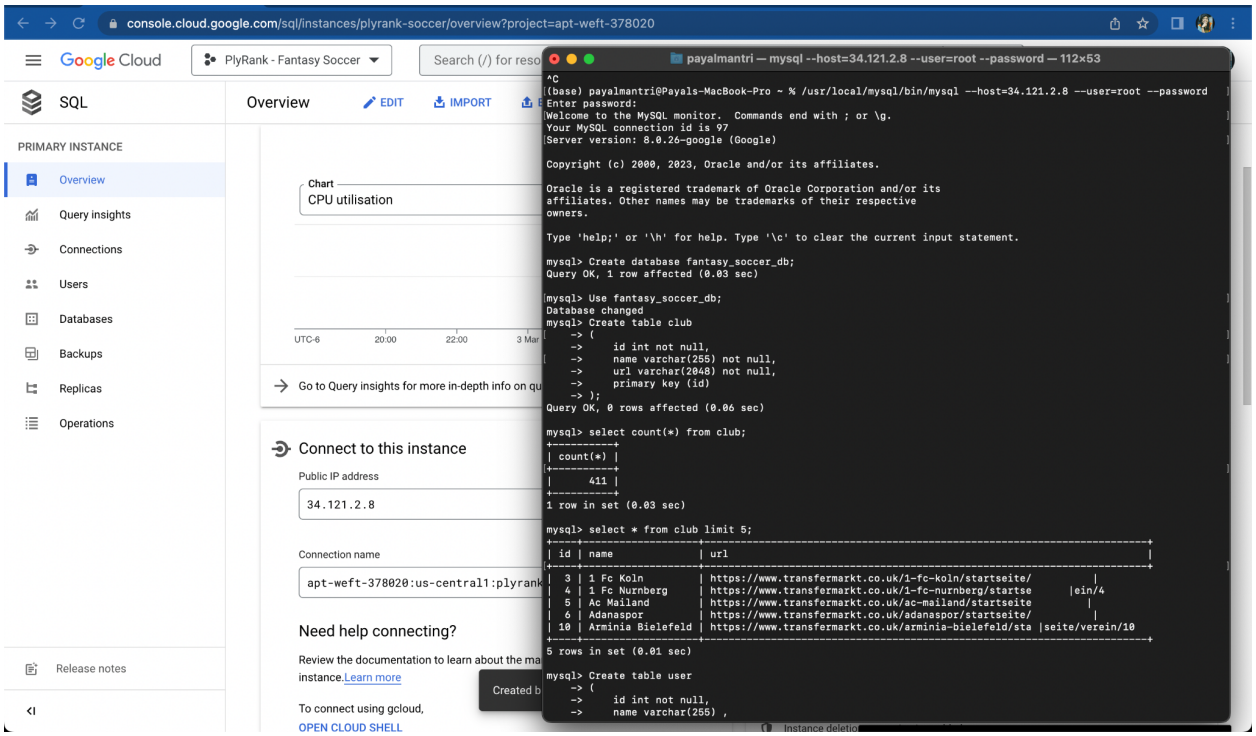
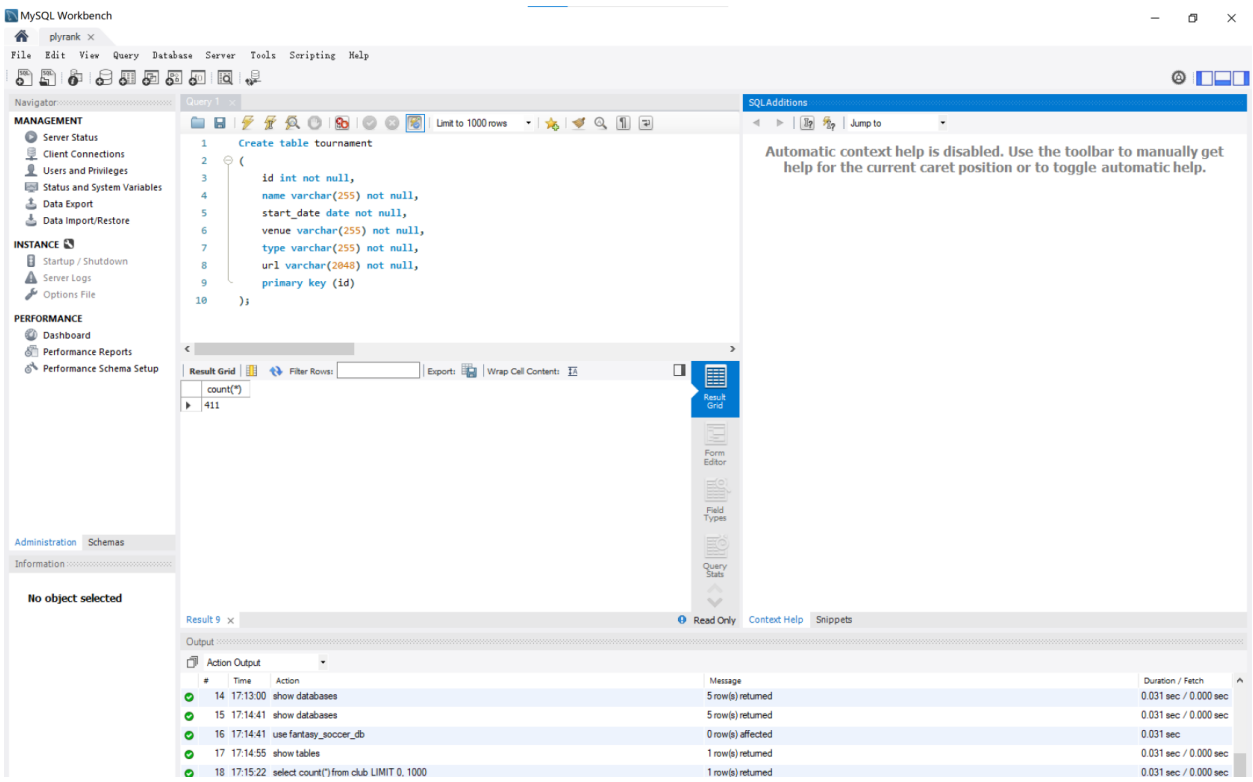
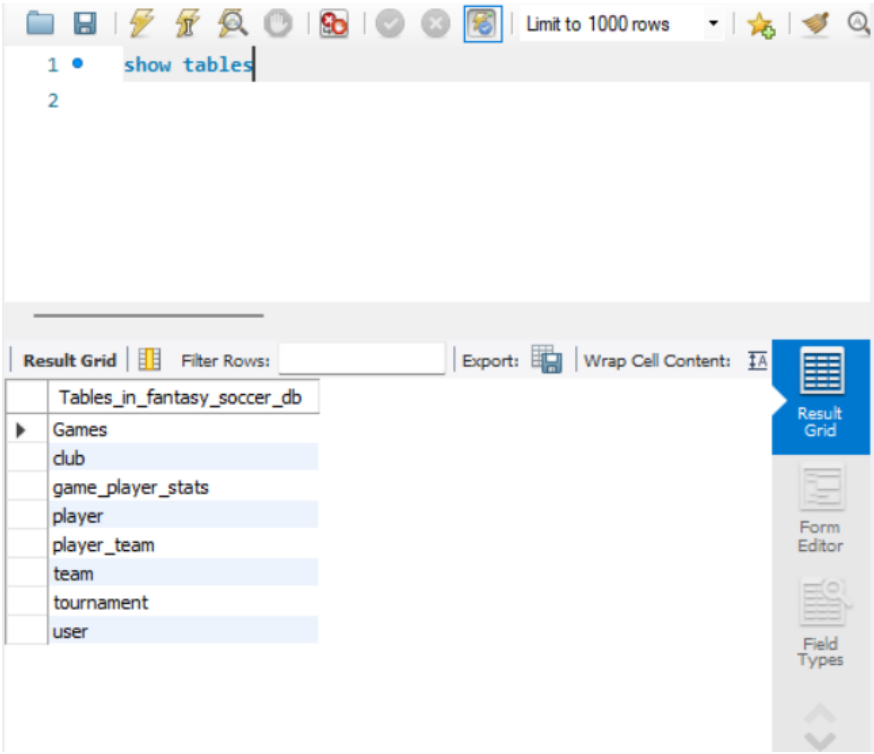
1. Screenshot of connecting to GCP

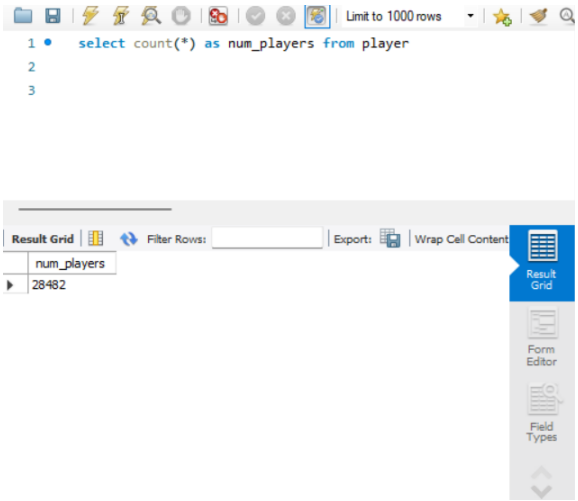


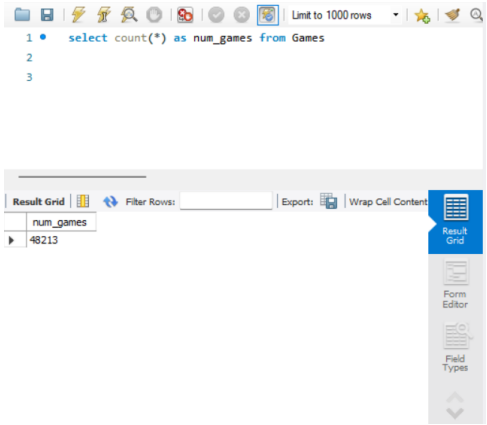


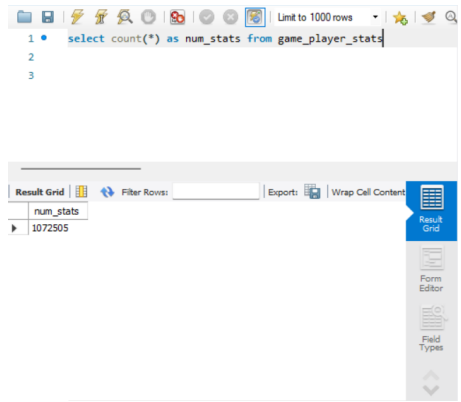
2. Screenshot of implemented tables



3. Screenshot for data insertion in GCP SQL
   a.

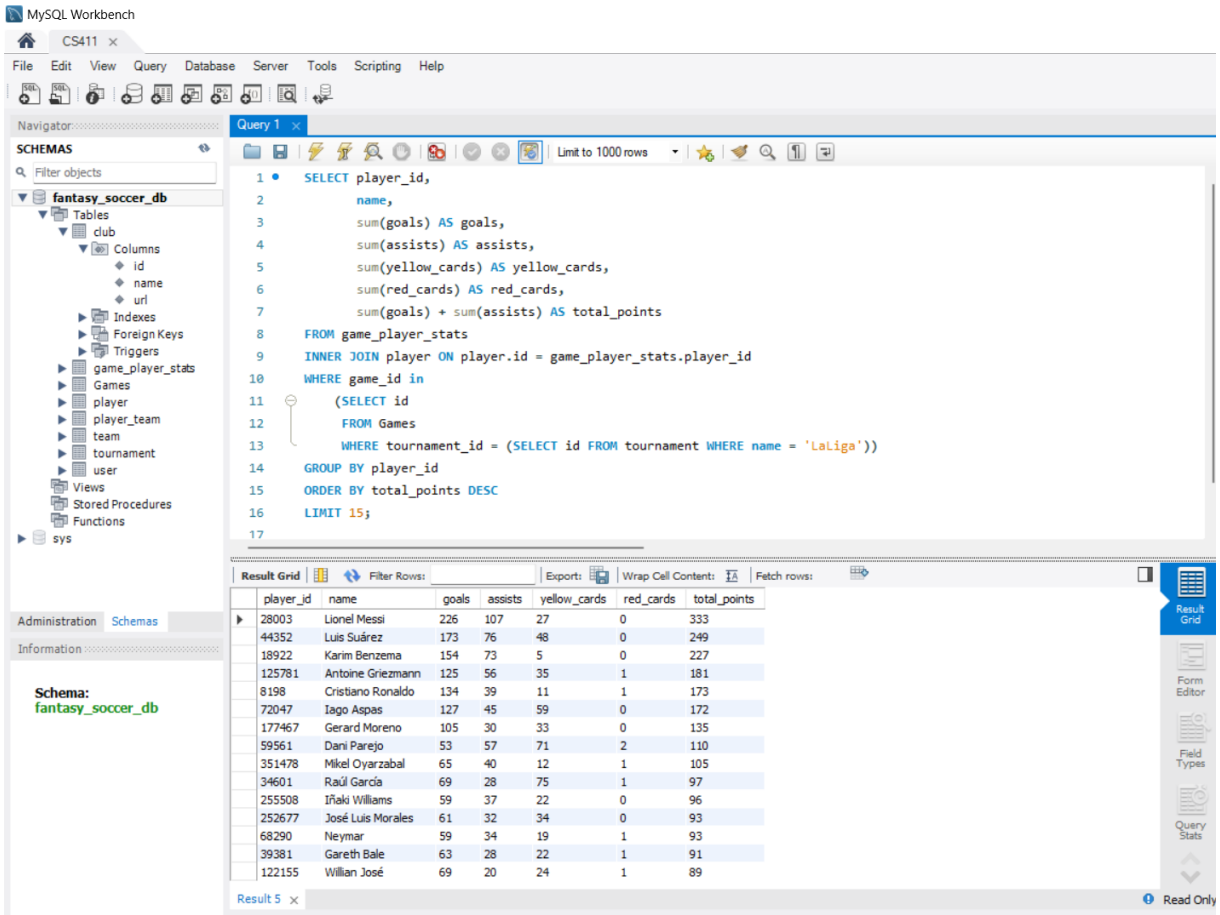






4. Advanced Queries and Results
    a. Given a tournament name , find all the stats of players in the tournament

```
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE game_id in
    (SELECT id
     FROM Games
     WHERE tournament_id = (SELECT id FROM tournament WHERE name =
'LaLiga'))
GROUP BY player_id
ORDER BY total_points DESC
LIMIT 15;
```



b. Fetch lifetime stats of a player

```
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE player.name = 'Lionel Messi'
GROUP BY player_id
ORDER BY total_points DESC;
```

```sql
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE player.name = 'Lionel Messi'
GROUP BY player_id
ORDER BY total_points DESC;
```

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

player_id	name	goals	assists	yellow_cards	red_cards	total_points
28003	Lionel Messi	329	178	44	1	507

c. Fetch top 15 players based on goals and assists for a club

```
Select player_id, player.name,club.name, sum(goals) as goals, sum(assists) as
assists
from game_player_stats
        inner join player on player.id = game_player_stats.player_id
        inner join club on club.id = player.club_id
where club.name = 'Fc Barcelona'
group by player_id
order by goals desc, assists desc
limit 15;
```

15 rows CSV

	player_id	name	name	goals	assists
1	38253	Robert Lewandowski	Fc Barcelona	336	71
2	26399	Sergio Agüero	Fc Barcelona	170	39
3	411295	Raphinha	Fc Barcelona	56	35
4	288230	Ousmane Dembélé	Fc Barcelona	51	55
5	294808	Franck Kessié	Fc Barcelona	43	16
6	112515	Marcos Alonso	Fc Barcelona	34	29
7	398184	Ferran Torres	Fc Barcelona	30	19
8	18944	Gerard Piqué	Fc Barcelona	30	7
9	69751	Jordi Alba	Fc Barcelona	20	78
10	326330	Frenkie de Jong	Fc Barcelona	19	27
11	683840	Pedri	Fc Barcelona	15	5
12	466810	Ansu Fati	Fc Barcelona	14	7
13	85370	Sergi Roberto	Fc Barcelona	12	34
14	15951	Dani Alves	Fc Barcelona	11	42
15	411975	Jules Kounde	Fc Barcelona	10	8

5. Indexing analysis

a. Fetch lifetime stats of a player

No Index:

```
explain analyze
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE player.name = 'Lionel Messi'
GROUP BY player_id
ORDER BY total_points DESC;
```

Form Editor | Navigate: 1 / 1

EXPLAIN:

```
-> Sort: (sum(game_player_stats.goals) + sum(game_player_stats.assists)) DESC  (actual
time=11.951..11.951 rows=1 loops=1)
    -> Table scan on <temporary>  (actual time=0.003..0.003 rows=1 loops=1)
        -> Aggregate using temporary table  (actual time=11.922..11.922 rows=1 loops=1)
```

Index on player(name):

```sql
create index player_name on player(name);
explain analyze
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE player.name = 'Lionel Messi'
GROUP BY player_id
ORDER BY total_points DESC;
```

Form Editor | Navigate:

EXPLAIN:

```
-> Sort: (sum(game_player_stats.goals) + sum(game_player_stats.assists)) DESC  (actual time=0.890..0.890 rows=1 loops=1)
    -> Table scan on <temporary>  (actual time=0.001..0.001 rows=1 loops=1)
        -> Aggregate using temporary table  (actual time=0.845..0.845 rows=1 loops=1)
```

Index on game_player_stats(red_cards):

```sql
create index goals_idx on game_player_stats(goals);
explain analyze
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE player.name = 'Lionel Messi'
GROUP BY player_id
ORDER BY total_points DESC;
```

Form Editor | Navigate: 1 / 1

EXPLAIN:

```
-> Sort: (sum(game_player_stats.goals) + sum(game_player_stats.assists)) DESC  (actual time=12.801..12.801 rows=1 loops=1)
    -> Table scan on <temporary>  (actual time=0.004..0.004 rows=1 loops=1)
        -> Aggregate using temporary table  (actual time=12.737..12.737 rows=1 loops=1)
            -> Nested loop inner join  (cost=19779.21 rows=158487) (actual time=1.074..12.206 rows=395 loops=1)
```

Index on game_player_stats(goals):

```sql
create index goals_idx on game_player_stats(goals);
explain analyze
SELECT player_id,
       name,
       sum(goals) AS goals,
       sum(assists) AS assists,
       sum(yellow_cards) AS yellow_cards,
       sum(red_cards) AS red_cards,
       sum(goals) + sum(assists) AS total_points
FROM game_player_stats
INNER JOIN player ON player.id = game_player_stats.player_id
WHERE player.name = 'Lionel Messi'
GROUP BY player_id
ORDER BY total_points DESC;
```

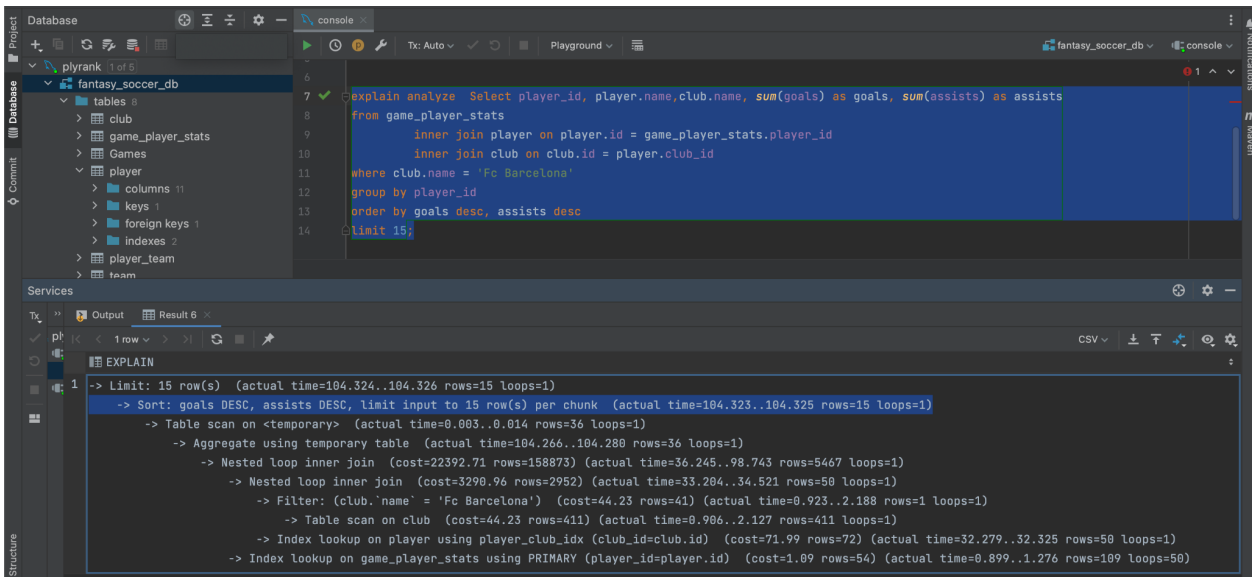
Form Editor | Navigate: 1 / 1

EXPLAIN:

```
-> Sort: (sum(game_player_stats.goals) + sum(game_player_stats.assists)) DESC  (actual time=12.801..12.801 rows=1 loops=1)
    -> Table scan on <temporary>  (actual time=0.004..0.004 rows=1 loops=1)
        -> Aggregate using temporary table  (actual time=12.737..12.737 rows=1 loops=1)
            -> Nested loop inner join  (cost=19779.21 rows=158487) (actual time=1.074..12.206 rows=395 loops=1)
```

Analysis: For the lifetime stats of a player query, we chose the index on player(name) because the time spent is significantly shorter than without using any indices. The reduction in time is likely due to the WHERE command for the player 'Lionel Messi,' since we do not search the entire table for instances with the player name 'Lionel Messi' and instead use the index. The game_player_stats(red_cards) and game_player_stats(goals) are not a part of the WHERE command and have little effect on the time spent, and performance of the query .
Hence we would select adding index on name field in player table - **player(name)** for this query.

b. Fetch players based on goals and assists for a given club
No Index



Adding Index on goals field in game_palyer_stats : This helps reduce the sort time from 114ms to 7.781 ms

```
create index goals on game_player_stats(goals);
explain analyze  Select player_id, player.name,club.name, sum(goals) as goals, sum(assists) as assists
from game_player_stats
        inner join player on player.id = game_player_stats.player_id
        inner join club on club.id = player.club_id
where club.name = 'Fc Barcelona'
group by player_id
order by goals desc, assists desc
limit 15;
```

```
-> Limit: 15 row(s)  (actual time=7.782..7.785 rows=15 loops=1)
    -> Sort: goals DESC, assists DESC, limit input to 15 row(s) per chunk  (actual time=7.781..7.783 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.002..0.010 rows=36 loops=1)
            -> Aggregate using temporary table  (actual time=7.734..7.745 rows=36 loops=1)
                -> Nested loop inner join  (cost=19846.41 rows=158873) (actual time=0.276..2.833 rows=5467 loops=1)
                    -> Nested loop inner join  (cost=3155.24 rows=2952) (actual time=0.233..0.400 rows=50 loops=1)
                        -> Filter: (club.`name` = 'Fc Barcelona')  (cost=42.35 rows=41) (actual time=0.093..0.240 rows=1 loops=1)
                            -> Table scan on club  (cost=42.35 rows=411) (actual time=0.078..0.191 rows=411 loops=1)
                        -> Index lookup on player using player_club_idx (club_id=club.id)  (cost=68.73 rows=72) (actual time=0.138..0.155 rows=50 loops=1)
                    -> Index lookup on game_player_stats using PRIMARY (player_id=player.id)  (cost=0.27 rows=54) (actual time=0.011..0.042 rows=109 loops=50)
```

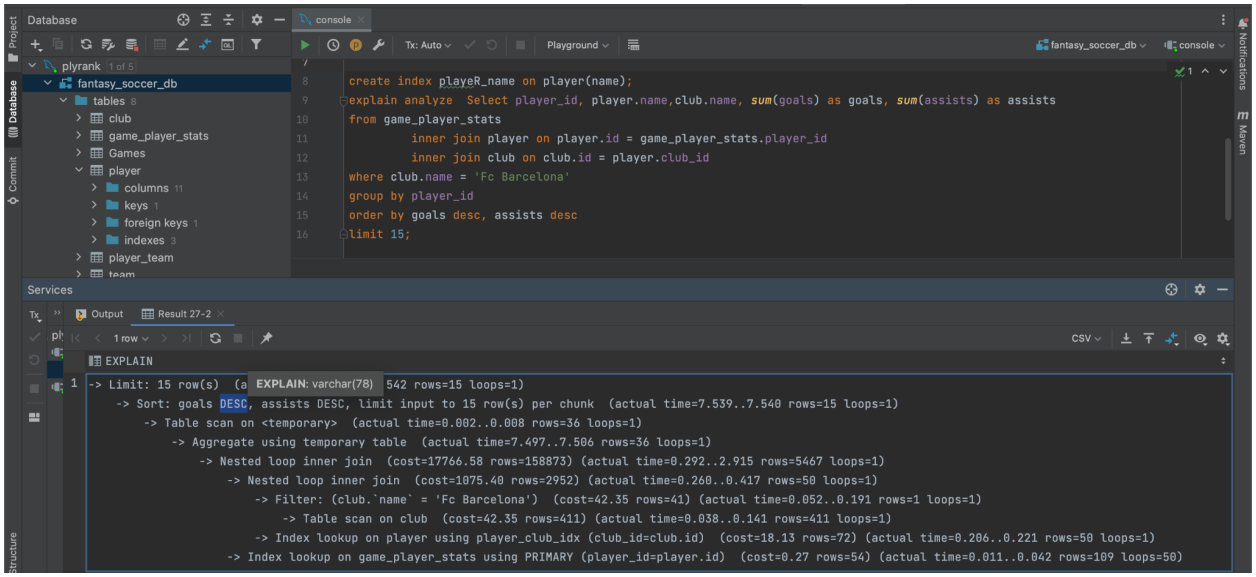
Adding Index on club name

It helps in reducing the cost on club table lookup . Now the lookup is for only one row compared to 411 rows without index.

```
create index club_name on club(name);
explain analyze  Select player_id, player.name,club.name, sum(goals) as goals, sum(assists) as assists
from game_player_stats
        inner join player on player.id = game_player_stats.player_id
        inner join club on club.id = player.club_id
where club.name = 'Fc Barcelona'
group by player_id
order by goals desc, assists desc
limit 15;
```

```
-> Limit: 15 row(s)  (actual time=8.397..8.400 rows=15 loops=1)
    -> Sort: goals DESC, assists DESC, limit input to 15 row(s) per chunk  (actual time=8.396..8.398 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.003..0.015 rows=36 loops=1)
            -> Aggregate using temporary table  (actual time=8.326..8.341 rows=36 loops=1)
                -> Nested loop inner join  (cost=482.20 rows=3866) (actual time=0.571..3.266 rows=5467 loops=1)
                    -> Nested loop inner join  (cost=76.09 rows=72) (actual time=0.533..0.563 rows=50 loops=1)
                        -> Index lookup on club using club_name (name='Fc Barcelona')  (cost=0.35 rows=1) (actual time=0.009..0.017 rows=1 loops=1)
                        -> Index lookup on player using player_club_idx (club_id=club.id)  (cost=75.74 rows=72) (actual time=0.522..0.541 rows=50 loops=1)
                    -> Index lookup on game_player_stats using PRIMARY (player_id=player.id)  (cost=0.35 rows=54) (actual time=0.012..0.046 rows=109 loops=50)
```

Adding index on name in player table - player.name - This doesn't help much



Analysis : We see that when we add index on club.name , the row search is reduced to only 1 row from 411 rows.
Adding an index on club.name helps because it enables the database to quickly locate the rows in the club table that match the WHERE clause club.name = 'Fc Barcelona'. Without this index, the database would need to scan the entire club table to find the matching rows, which could be very slow if the table is large.
Note that the ORDER BY clause also affects the performance of the query. We can see that adding indexes on the goals column improves the sorting performance by a huge amount. However, adding indexes on other attributes doesn't not make a significant difference for this particular query, since the JOIN conditions and the WHERE clause involve only the club and player tables, and the player_id attribute is already the primary key of the game_player_stats table

We would select a combination of index - First on **club.name** field and **game_player_stats.goals** field .