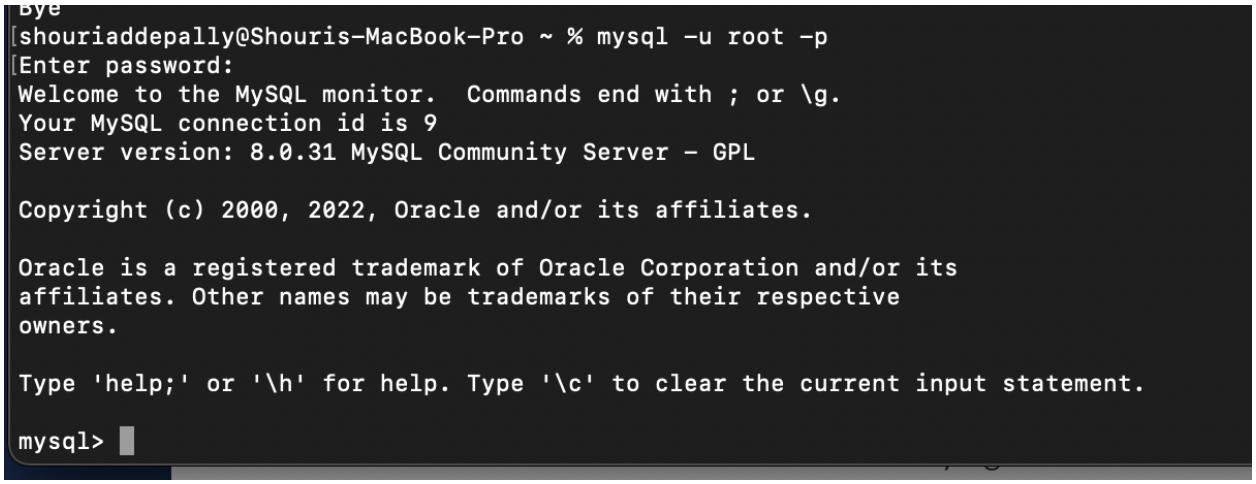
Establishing MySQL Connection:



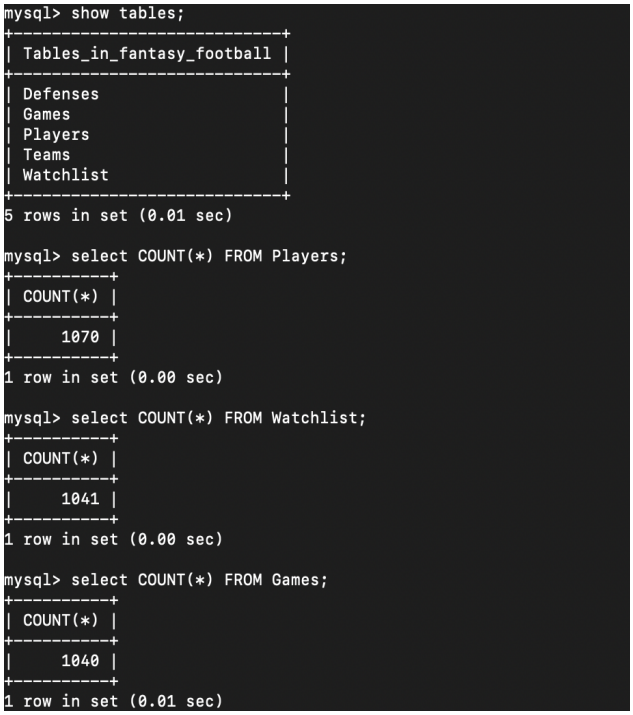
Creating Tables Using DDL Commands:

```
CREATE TABLE Teams (
PlayerID INT,
DefenseID INT,
Division VARCHAR(50),
Record VARCHAR(50),
Schedule VARCHAR(50),
Team VARCHAR(10)
);

CREATE TABLE Defenses (
Team VARCHAR(10),
Sacks INT,
Interceptions INT,
Fumbles_Recovered INT,
Touchdowns INT,
Points_Allowed INT,
Yards_Allowed INT,
Avg_Fantasy_Points INT
);

CREATE TABLE Games (
GameID INT,
Date VARCHAR(50),
Week INT,
Home_Team VARCHAR(5),
Home_Score INT,
Away_Score INT,
Away_Team VARCHAR(5)
);
```

```
CREATE TABLE Watchlist (
   WatchlistID INT,
   UserID INT,
   Name VARCHAR(100),
   Notes VARCHAR(200)
);

CREATE TABLE Players (
   PlayerID INT,
   Player_Name VARCHAR(50),
   Team VARCHAR(5),
   Position VARCHAR(50),
   Age INT,
   Avg_Fantasy_Points REAL
);
```

Inserting Data Into Tables:

For our database we have 1000 data-entries in the Players, Watchlist, and Games tables. For these tables, we filled in our column values based on the data we gathered on fantasy players and their fantasy outputs using a simple python script. We also followed a similar approach for the Games table which was filled out using previous years' data using home and away team scores and using a similar python script as the Players table. The Watchlist also contains players and notes that correspond to fantasy players which have generated or have potential to generate a high amount of fantasy points from week to week. Below is an image of all the data counts for the three tables which each contain at least 1000 rows.



```
mysql> show tables;
+----------------------------+
| Tables_in_fantasy_football |
+----------------------------+
| Defenses                   |
| Games                      |
| Players                    |
| Teams                      |
| Watchlist                  |
+----------------------------+
5 rows in set (0.01 sec)

mysql> select COUNT(*) FROM Players;
+----------+
| COUNT(*) |
+----------+
|     1070 |
+----------+
1 row in set (0.00 sec)

mysql> select COUNT(*) FROM Watchlist;
+----------+
| COUNT(*) |
+----------+
|     1041 |
+----------+
1 row in set (0.00 sec)

mysql> select COUNT(*) FROM Games;
+----------+
| COUNT(*) |
+----------+
|     1040 |
+----------+
1 row in set (0.01 sec)
```

Advanced SQL Queries:

```
SELECT ROUND(Total_Fantasy_Points / Total_Players, 2) AS Average_Fantasy_Points, P.Team
FROM
(SELECT SUM(Avg_Fantasy_Points) AS Total_Fantasy_Points, Team FROM Players GROUP BY Team)
AS T
JOIN
(SELECT COUNT(PlayerID) AS Total_Players, Team
FROM Players GROUP BY Team) AS P
ON
(P.Team = T.Team)
ORDER BY Average_Fantasy_Points DESC
LIMIT 15;
```

```
mysql> SELECT ROUND(Total_Fantasy_Points / Total_Players, 2) AS Average_Fantasy_Points, P.Team
    -> FROM
    -> (SELECT SUM(Avg_Fantasy_Points) AS Total_Fantasy_Points, Team FROM Players GROUP BY Team) AS T
    -> JOIN
    -> (SELECT COUNT(PlayerID) AS Total_Players, Team
    -> FROM Players GROUP BY Team) AS P
    -> ON
    -> (P.Team = T.Team)
    -> ORDER BY Average_Fantasy_Points DESC
    -> LIMIT 15;
+-----------------------+------+
| Average_Fantasy_Points | Team |
+-----------------------+------+
|                  7.86 | LAR  |
|                  7.67 | BAL  |
|                  7.45 | TAM  |
|                  7.19 | BUF  |
|                  7.16 | PHI  |
|                   7.1 | SEA  |
|                  7.06 | DEN  |
|                  6.97 | ARI  |
|                  6.96 | DAL  |
|                  6.93 | LAC  |
|                  6.82 | JAX  |
|                  6.79 | PIT  |
|                  6.79 | GNB  |
|                  6.75 | LVR  |
|                  6.74 | MIN  |
+-----------------------+------+
15 rows in set (0.02 sec)
```

```
(SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points
 FROM Players p JOIN Teams t ON(p.Team = t.Team)
WHERE t.Division Like('NFC West') AND p.Avg_Fantasy_Points > 13
LIMIT 7)
UNION
(SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points
FROM Players p JOIN Teams t ON(p.Team = t.Team)
WHERE t.Division Like ('AFC West') and p.Avg_Fantasy_Points > 13
LIMIT 8);
```

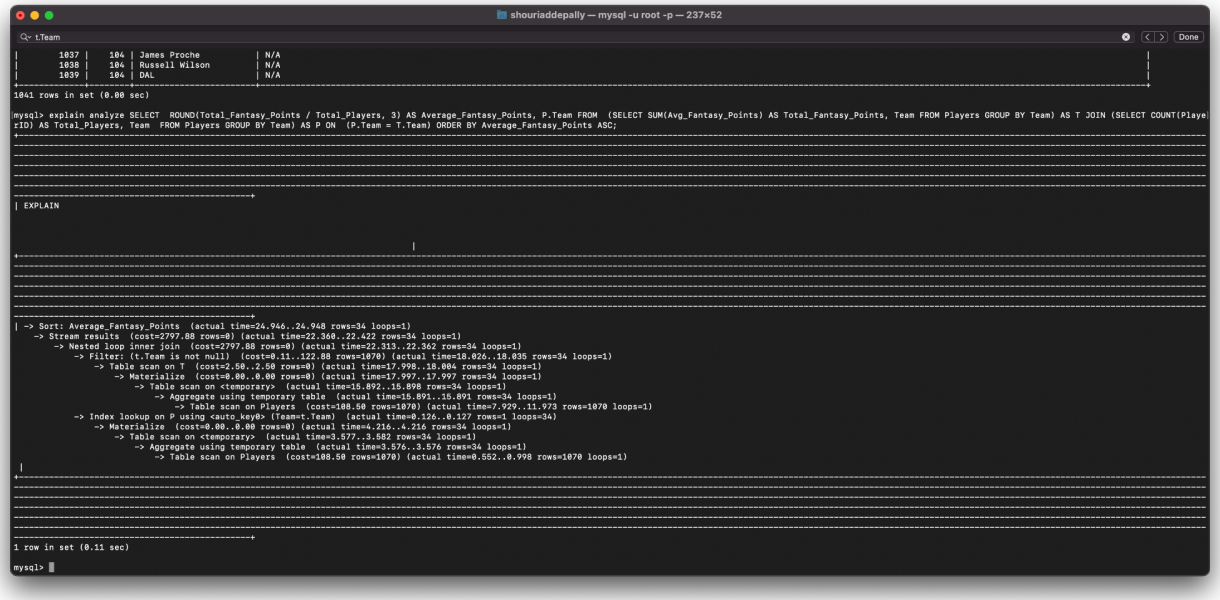
```
mysql> (SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points
    ->  FROM Players p JOIN Teams t ON(p.Team = t.Team)
    -> WHERE t.Division Like('NFC West') AND p.Avg_Fantasy_Points > 13
    -> LIMIT 7)
    -> UNION
    -> (SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points
    -> FROM Players p JOIN Teams t ON(p.Team = t.Team)
    -> WHERE t.Division Like ('AFC West') and p.Avg_Fantasy_Points > 13
    -> LIMIT 8);
+------------------+------+----------+--------------------+
| Player_Name      | Team | Division | Avg_Fantasy_Points |
+------------------+------+----------+--------------------+
| Cooper Kupp      | LAR  | NFC West |              17.35 |
| Deebo Samuel     | SFO  | NFC West |              16.38 |
| James Conner     | ARI  | NFC West |              14.73 |
| Matthew Stafford | LAR  | NFC West |              19.41 |
| Kyler Murray     | ARI  | NFC West |              21.43 |
| Elijah Mitchell  | SFO  | NFC West |              13.27 |
| Russell Wilson   | SEA  | NFC West |              17.36 |
| Austin Ekeler    | LAC  | AFC West |              17.12 |
| Justin Herbert   | LAC  | AFC West |              22.41 |
| Patrick Mahomes  | KAN  | AFC West |              21.29 |
| Derek Carr       | LVR  | AFC West |              15.12 |
| Teddy Bridgewater | DEN | AFC West |               14.5 |
| Marcus Kemp      | KAN  | AFC West |                 16 |
| Raheem Gallman   | LAC  | AFC West |              13.41 |
| Ben Dalton       | DEN  | AFC West |              13.67 |
+------------------+------+----------+--------------------+
15 rows in set (0.01 sec)
```

Indexing Analysis:

Query 1:

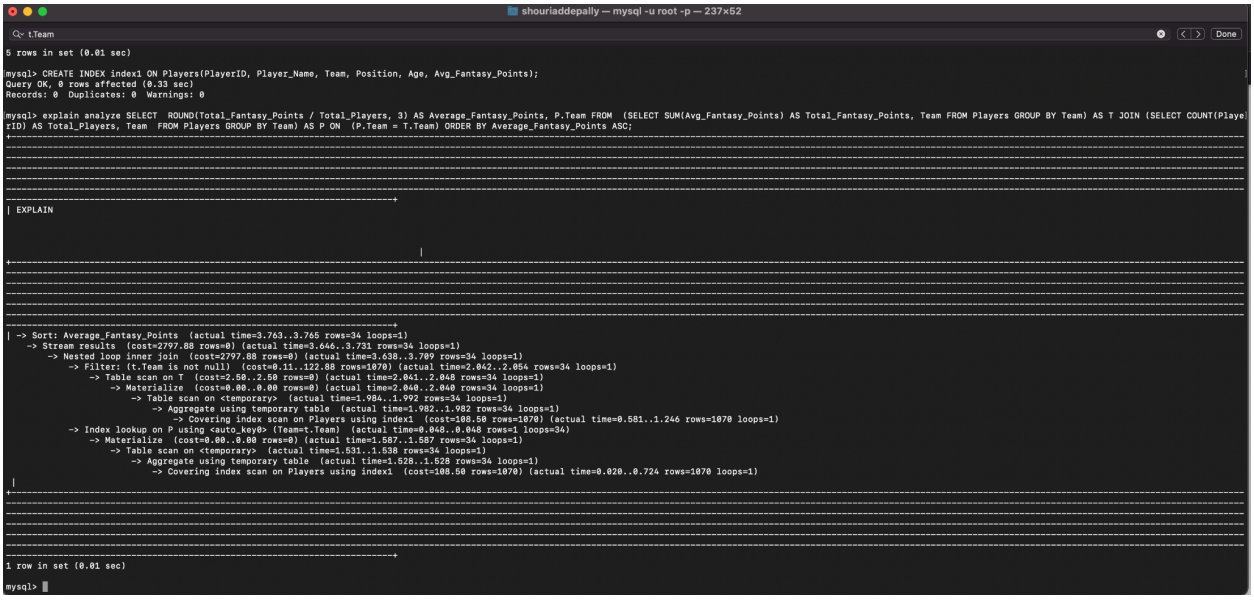
```
SELECT ROUND(Total_Fantasy_Points / Total_Players, 2) AS Average_Fantasy_Points, P.Team
FROM
(SELECT SUM(Avg_Fantasy_Points) AS Total_Fantasy_Points, Team FROM Players GROUP BY Team)
AS T
JOIN
(SELECT COUNT(PlayerID) AS Total_Players, Team
FROM Players GROUP BY Team) AS P
ON
(P.Team = T.Team)
ORDER BY Average_Fantasy_Points DESC
LIMIT 15;
```

Indexing Design #0 - Default Index:



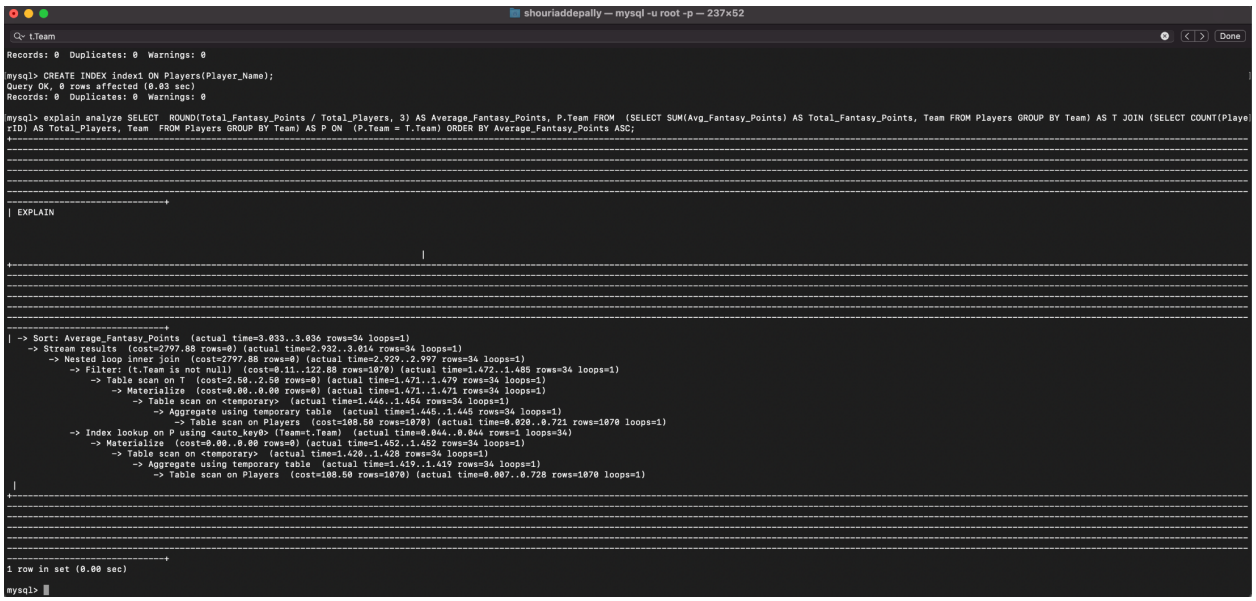
Indexing Design #1 - Index on the players table:

CREATE INDEX index1 ON Players(PlayerID, Player_Name, Team, Position, Age, Avg_Fantasy_Points);



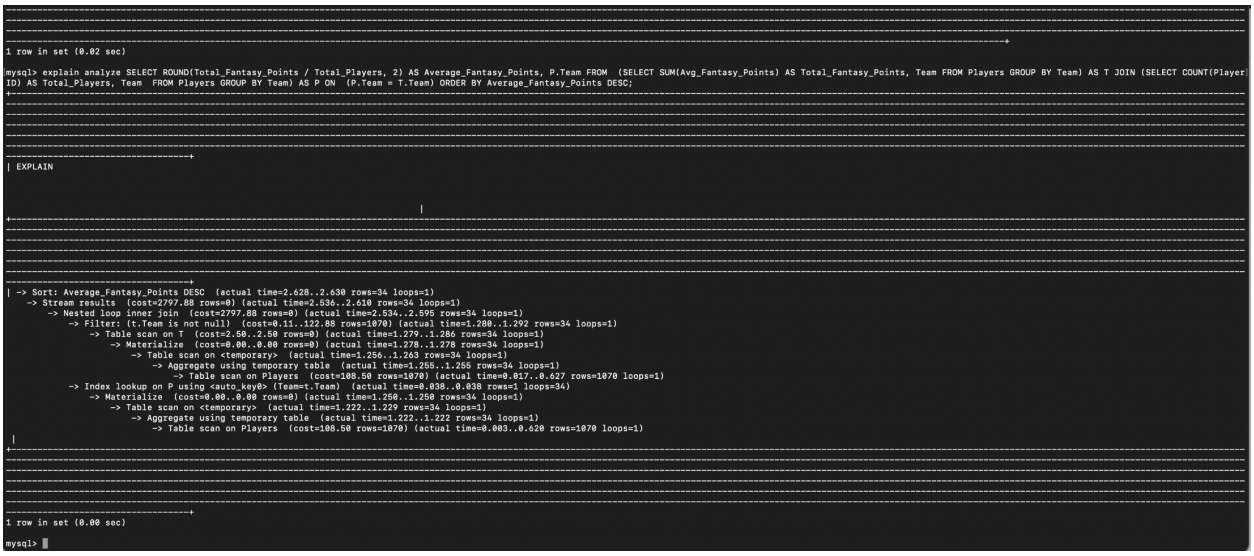
Indexing Design #2 - Index on Player_Name column in Players Table:

CREATE INDEX index1 ON Players(Player_Name);



Indexing Design #3 - Unique Index on PlayerID in Players Table:

CREATE UNIQUE INDEX index1 ON Players(PlayerID);

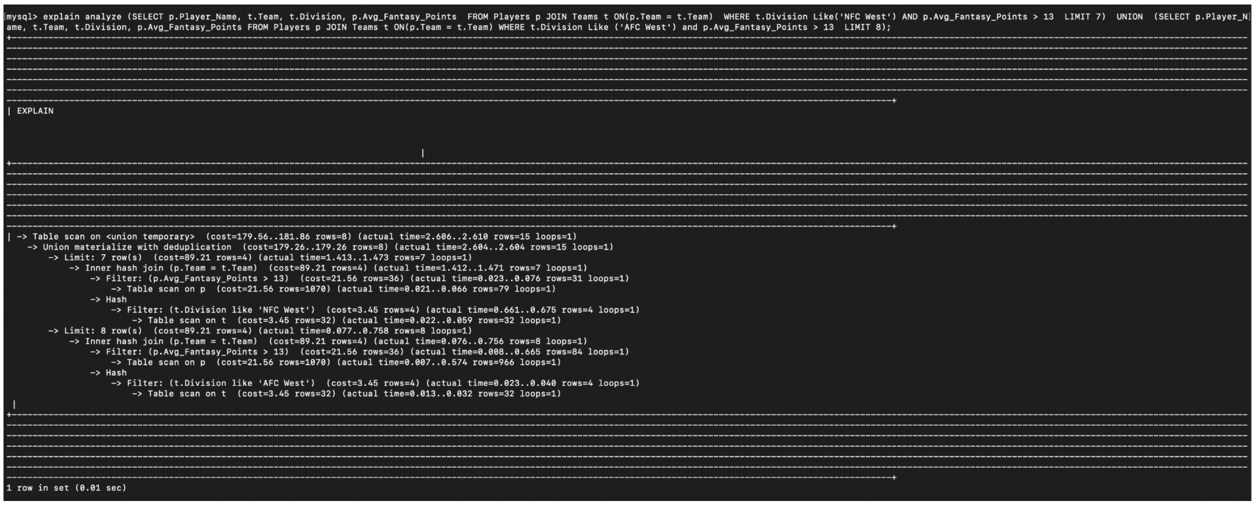


Final Indexing Analysis Report - Query 1:

For our first query, we selected our third indexing design as the best index design. In this indexing design, we create an index on the PlayerID on the Players table compared to previous designs where we indexed on the entire Players table, as well as the Player_Name column in the Players table. The reason why we chose this indexing design as our best design was because its start up time at the first leaf level is faster than either of the indexing designs mentioned previously - measuring a range of with a minimal range of 0.0017 - 0.627 time units. In addition, this third design is also faster than the previous designs when performing an index lookup on the second subquery statement, taking roughly 0.003-0.620 time units as its start up time at the second leaf level.

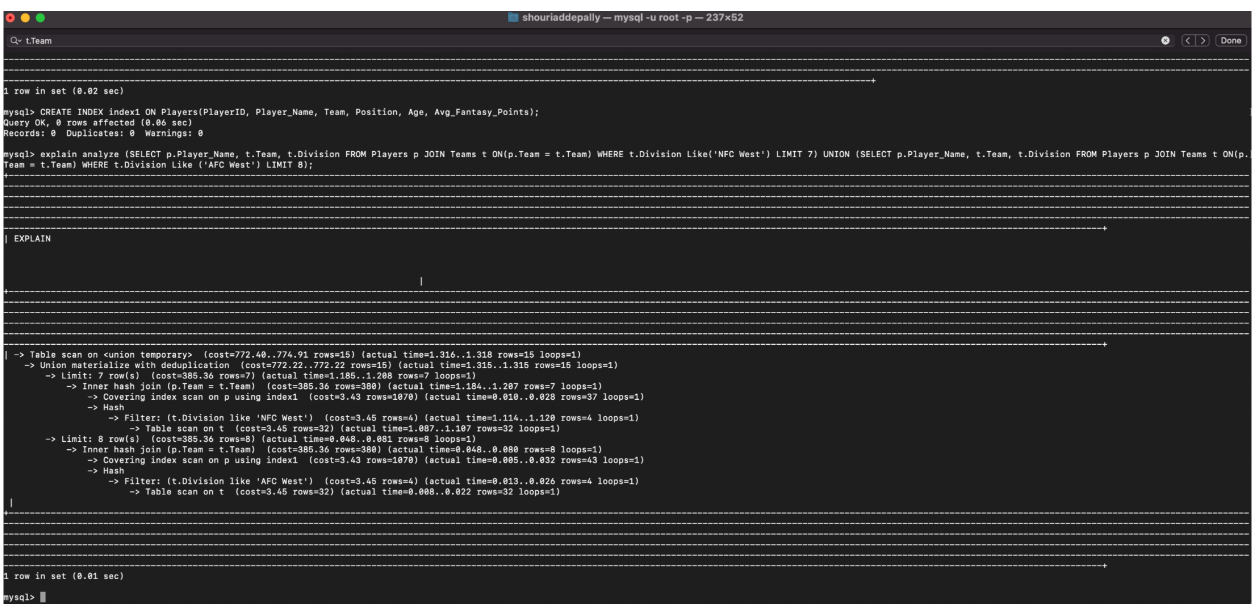
Query 2:
(SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points
FROM Players p JOIN Teams t ON(p.Team = t.Team)
WHERE t.Division Like('NFC West') AND p.Avg_Fantasy_Points > 13
LIMIT 7)
UNION
(SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points
FROM Players p JOIN Teams t ON(p.Team = t.Team)
WHERE t.Division Like ('AFC West') and p.Avg_Fantasy_Points > 13
LIMIT 8);

Indexing Design #0 - Default Index:



```
mysql> explain analyze (SELECT p.Player_Name, t.Team, t.Division, p.Avg_Fantasy_Points  FROM Players p JOIN Teams t ON(p.Team = t.Team)  WHERE t.Division Like('NFC West') AND p.Avg_Fantasy_Points > 13  LIMIT 7)  UNION  (SELECT p.Player_N
ame, t.Team, t.Division, p.Avg_Fantasy_Points FROM Players p JOIN Teams t ON(p.Team = t.Team) WHERE t.Division Like ('AFC West') and p.Avg_Fantasy_Points > 13  LIMIT 8);
| EXPLAIN
| -> Table scan on <union temporary>  (cost=179.56..181.86 rows=8) (actual time=2.606..2.610 rows=15 loops=1)
    -> Union materialize with deduplication  (cost=179.26..179.26 rows=8) (actual time=2.604..2.604 rows=15 loops=1)
        -> Limit: 7 row(s)  (cost=89.21 rows=4) (actual time=1.413..1.473 rows=7 loops=1)
            -> Inner hash join (p.Team = t.Team)  (cost=89.21 rows=4) (actual time=1.412..1.471 rows=7 loops=1)
                -> Filter: (p.Avg_Fantasy_Points > 13)  (cost=21.56 rows=36) (actual time=0.023..0.076 rows=31 loops=1)
                    -> Table scan on p  (cost=21.56 rows=1070) (actual time=0.021..0.066 rows=79 loops=1)
                -> Hash
                    -> Filter: (t.Division like 'NFC West')  (cost=3.45 rows=4) (actual time=0.661..0.675 rows=4 loops=1)
                        -> Table scan on t  (cost=3.45 rows=32) (actual time=0.022..0.059 rows=32 loops=1)
        -> Limit: 8 row(s)  (cost=89.21 rows=4) (actual time=0.077..0.758 rows=8 loops=1)
            -> Inner hash join (p.Team = t.Team)  (cost=89.21 rows=4) (actual time=0.076..0.756 rows=8 loops=1)
                -> Filter: (p.Avg_Fantasy_Points > 13)  (cost=21.56 rows=36) (actual time=0.008..0.665 rows=84 loops=1)
                    -> Table scan on p  (cost=21.56 rows=1070) (actual time=0.007..0.574 rows=966 loops=1)
                -> Hash
                    -> Filter: (t.Division like 'AFC West')  (cost=3.45 rows=4) (actual time=0.023..0.040 rows=4 loops=1)
                        -> Table scan on t  (cost=3.45 rows=32) (actual time=0.013..0.032 rows=32 loops=1)
 |
1 row in set (0.01 sec)
```

Indexing Design #1 - Index on the players table:

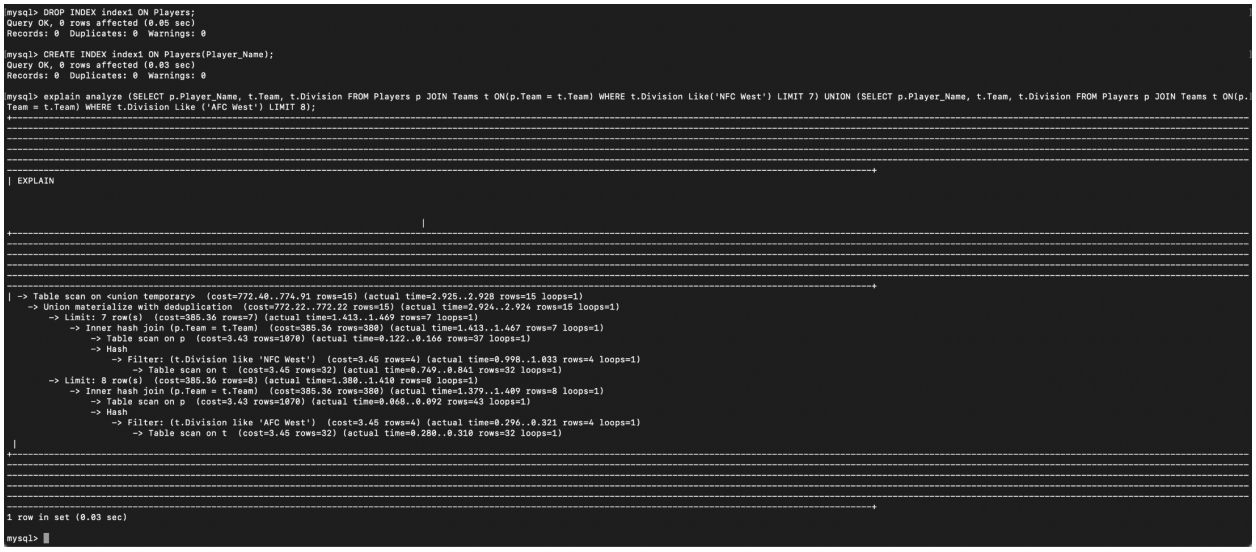


```
shouriaddepally — mysql -u root -p — 237×52
t.Team   Done
1 row in set (0.02 sec)

mysql> CREATE INDEX index1 ON Players(PlayerID, Player_Name, Team, Position, Age, Avg_Fantasy_Points);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> explain analyze (SELECT p.Player_Name, t.Team, t.Division FROM Players p JOIN Teams t ON(p.Team = t.Team) WHERE t.Division Like('NFC West') LIMIT 7) UNION (SELECT p.Player_Name, t.Team, t.Division FROM Players p JOIN Teams t ON(p.
Team = t.Team) WHERE t.Division Like ('AFC West') LIMIT 8);
| EXPLAIN
| -> Table scan on <union temporary>  (cost=772.40..774.91 rows=15) (actual time=1.316..1.318 rows=15 loops=1)
    -> Union materialize with deduplication  (cost=772.22..772.22 rows=15) (actual time=1.315..1.315 rows=15 loops=1)
        -> Limit: 7 row(s)  (cost=385.36 rows=7) (actual time=1.185..1.208 rows=7 loops=1)
            -> Inner hash join (p.Team = t.Team)  (cost=385.36 rows=380) (actual time=1.184..1.207 rows=7 loops=1)
                -> Covering index scan on p using index1  (cost=3.43 rows=1070) (actual time=0.010..0.028 rows=37 loops=1)
                -> Hash
                    -> Filter: (t.Division like 'NFC West')  (cost=3.45 rows=4) (actual time=1.114..1.120 rows=4 loops=1)
                        -> Table scan on t  (cost=3.45 rows=32) (actual time=1.087..1.107 rows=32 loops=1)
        -> Limit: 8 row(s)  (cost=385.36 rows=8) (actual time=0.048..0.081 rows=8 loops=1)
            -> Inner hash join (p.Team = t.Team)  (cost=385.36 rows=380) (actual time=0.048..0.080 rows=8 loops=1)
                -> Covering index scan on p using index1  (cost=3.43 rows=1070) (actual time=0.005..0.032 rows=43 loops=1)
                -> Hash
                    -> Filter: (t.Division like 'AFC West')  (cost=3.45 rows=4) (actual time=0.013..0.026 rows=4 loops=1)
                        -> Table scan on t  (cost=3.45 rows=32) (actual time=0.008..0.022 rows=32 loops=1)
 |
1 row in set (0.01 sec)

mysql>
```

CREATE INDEX index1 ON Players(PlayerID, Player_Name, Team, Position, Age, Avg_Fantasy_Points);

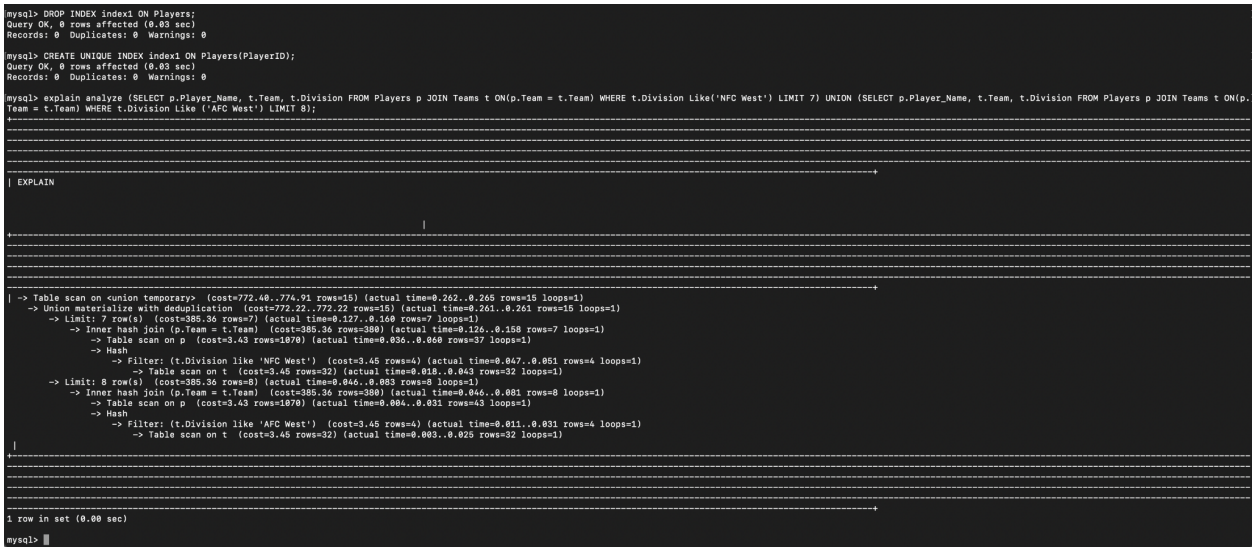
Indexing Design #2 - Index on Player_Name column in Players Table:

CREATE INDEX index1 ON Players(Player_Name);



Indexing Design #3 - Unique Index on PlayerID in Players Table:

CREATE UNIQUE INDEX index1 ON Players(PlayerID);



Final Indexing Analysis Report - Query 2:

For our second query, we selected our third indexing design as the best index design. In this indexing design, we create an index on the PlayerID on the Players table compared to previous designs where we indexed on the entire Players table, as well as the Player_Name column in the Players table. In our query, we use the UNION set operation to join the players who are on teams that belong either in the AFC West or in the NFC West and had an average fantasy points score greater than 13 points per week. The first portion of our UNION operation selects attributes from the Players table as well as the Teams table, which are then joined based on

equivalent Team values across the Players and Teams tables. The filtering table scan on this first leaf level is faster than any of the previous approaches, measuring with the minimal time range of 0.018-0.043 time units. In addition, the second table scan also measures the fastest second leaf level startup time range at 0.003 - 0.025 time units. These two minimal measurements for leaf level scans is why we chose this Indexing Design #3 as the best indexing approach for this query.