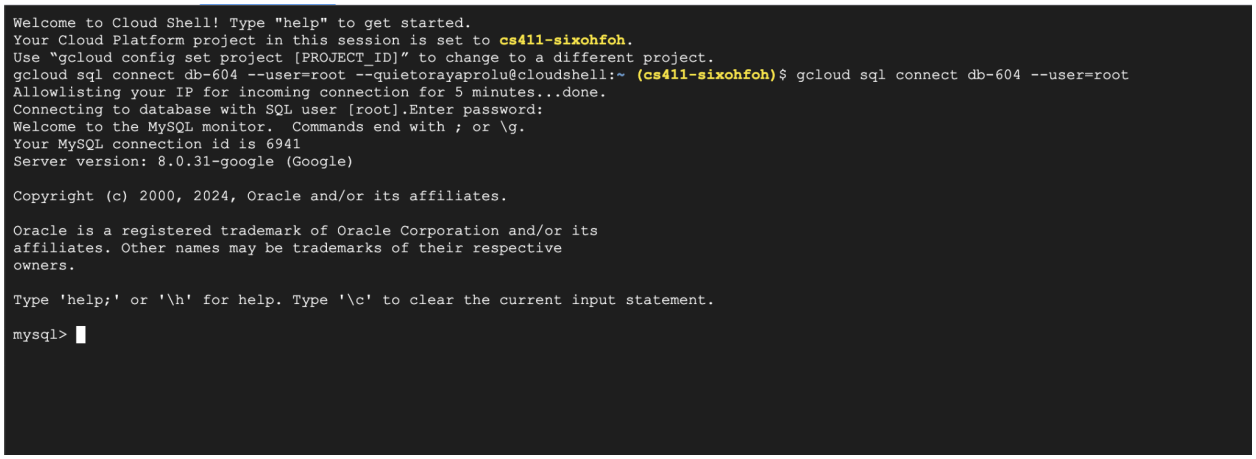
STAGE 3

Database Implementation:

Screenshot of Connection:



DDL Commands:

```
CREATE TABLE User (
  UserID INT PRIMARY KEY,
  Username VARCHAR(50) NOT NULL,
  Password VARCHAR(50) NOT NULL
);

CREATE TABLE Podcast (
   UserID INT,
   PodcastName VARCHAR(100) NOT NULL,
   Link VARCHAR(255),
   PRIMARY KEY (PodcastID, UserID),
   FOREIGN KEY (UserID) REFERENCES User(UserID)
);


CREATE TABLE Books (
   BookID INT PRIMARY KEY,
   BookName VARCHAR(100) NOT NULL,
   Author VARCHAR(50),
```

```
    AvgRating DECIMAL(3, 2)
);

CREATE TABLE People (
    PersonID INT PRIMARY KEY,
    PersonName VARCHAR(100) NOT NULL,
    NetWorth DECIMAL(15, 2),
    Profession VARCHAR(50)
);

CREATE TABLE Companies (
    CompanyID INT PRIMARY KEY,
    CompanyName VARCHAR(100) NOT NULL,
    State VARCHAR(100),
    Revenue DECIMAL(15, 2),
    Industry VARCHAR(50)
);

CREATE TABLE BookReference (
    PodcastID INT,
    BookID INT,
    PRIMARY KEY (PodcastID, BookID),
    FOREIGN KEY (PodcastID) REFERENCES Podcast(PodcastID),
    FOREIGN KEY (BookID) REFERENCES Books(BookID)
);

CREATE TABLE PeopleReference (
    PodcastID INT,
    PersonID INT,
    PRIMARY KEY (PodcastID, PersonID),
    FOREIGN KEY (PodcastID) REFERENCES Podcast(PodcastID),
    FOREIGN KEY (PersonID) REFERENCES People(PersonID)
);


CREATE TABLE CompanyReference (
    PodcastID INT,
    CompanyID INT,
    PRIMARY KEY (PodcastID, CompanyID),
    FOREIGN KEY (PodcastID) REFERENCES Podcast(PodcastID),
```

```
    FOREIGN KEY (CompanyID) REFERENCES Companies(CompanyID)
);
```

Inserted at least 1000 rows in the tables using a count query:

```
mysql> Select Count(PersonID) From People
    -> ;
+-----------------+
| Count(PersonID) |
+-----------------+
|            1001 |
+-----------------+
1 row in set (0.01 sec)

mysql> Select Count(BookID) From Books;
+---------------+
| Count(BookID) |
+---------------+
|          1001 |
+---------------+
1 row in set (0.00 sec)

mysql> Select Count(CompanyID) From Companies;
+------------------+
| Count(CompanyID) |
+------------------+
|             5000 |
+------------------+
1 row in set (0.01 sec)
```

Advanced Queries:

Most Mentioned Entities by Podcast

This query retrieves the count of mentioned books, people, and companies for each podcast and ranks them by the total mentions.

```
SELECT
    p.PodcastName,
    COUNT(DISTINCT br.BookID) AS TotalBooks,
    COUNT(DISTINCT pr.PersonID) AS TotalPeople,
    COUNT(DISTINCT cr.CompanyID) AS TotalCompanies,
    (COUNT(DISTINCT br.BookID) + COUNT(DISTINCT pr.PersonID) +
COUNT(DISTINCT cr.CompanyID)) AS TotalMentions
FROM  Podcast p
LEFT JOIN BookReference br ON p.PodcastID = br.PodcastID
LEFT JOIN PeopleReference pr ON p.PodcastID = pr.PodcastID
LEFT JOIN CompanyReference cr ON p.PodcastID = cr.PodcastID
GROUP BY p.PodcastName
ORDER BY TotalMentions DESC
LIMIT 15;
```

```
mysql> SELECT
    ->     p.PodcastName,
    ->     COUNT(DISTINCT br.BookID) AS TotalBooks,
    ->     COUNT(DISTINCT pr.PersonID) AS TotalPeople,
    ->     COUNT(DISTINCT cr.CompanyID) AS TotalCompanies,
    ->     (COUNT(DISTINCT br.BookID) + COUNT(DISTINCT pr.PersonID) + COUNT(DISTINCT cr.CompanyID)) AS TotalMentions
    -> FROM
    ->     Podcast p
    -> LEFT JOIN
    ->     BookReference br ON p.PodcastID = br.PodcastID
    -> LEFT JOIN
    ->     PeopleReference pr ON p.PodcastID = pr.PodcastID
    -> LEFT JOIN
    ->     CompanyReference cr ON p.PodcastID = cr.PodcastID
    -> GROUP BY
    ->     p.PodcastName
    -> ORDER BY
    ->     TotalMentions DESC;
+-------------+------------+-------------+----------------+---------------+
| PodcastName | TotalBooks | TotalPeople | TotalCompanies | TotalMentions |
+-------------+------------+-------------+----------------+---------------+
| Podcast1    |          2 |           2 |              2 |             6 |
| Podcast2    |          2 |           2 |              2 |             6 |
| Podcast3    |          2 |           2 |              2 |             6 |
| Podcast4    |          2 |           2 |              2 |             6 |
| Podcast5    |          2 |           2 |              2 |             6 |
| Podcast6    |          2 |           2 |              2 |             6 |
| Podcast7    |          2 |           2 |              2 |             6 |
| Podcast8    |          2 |           2 |              2 |             6 |
| Podcast9    |          2 |           3 |              1 |             6 |
+-------------+------------+-------------+----------------+---------------+
9 rows in set (0.01 sec)

mysql>
```

Most Companies Mentioned in Podcasts with Total Revenue

This query counts the number of times companies are mentioned across all podcasts and calculates their total revenue.

```
SELECT c.CompanyName, COUNT(cr.CompanyID) AS MentionCount,
SUM(c.Revenue) AS TotalRevenue
FROM CompanyReference cr
JOIN Companies c ON cr.CompanyID = c.CompanyID
JOIN Podcast p ON cr.PodcastID = p.PodcastID
GROUP BY c.CompanyName
ORDER BY MentionCount DESC, TotalRevenue DESC
LIMIT 15;
```

```
mysql> SELECT c.CompanyName, COUNT(cr.CompanyID) AS MentionCount, SUM(c.Revenue) AS TotalRevenue
    -> FROM CompanyReference cr
    -> JOIN Companies c ON cr.CompanyID = c.CompanyID
    -> JOIN Podcast p ON cr.PodcastID = p.PodcastID
    -> GROUP BY c.CompanyName
    -> ORDER BY MentionCount DESC, TotalRevenue DESC
    -> LIMIT 15;
+--------------------------------+-------------+--------------+
| CompanyName                    | MentionCount | TotalRevenue |
+--------------------------------+-------------+--------------+
| Lighter Capital                |            2 |        32.00 |
| BMNT Partners                  |            2 |        16.00 |
| Restaurant Partners Procurement |            2 |         5.20 |
| Digital Media Solutions        |            1 |       156.20 |
| J&D Brush                      |            1 |        84.90 |
| US Patriot                     |            1 |        68.60 |
| United Franchise Group         |            1 |        56.70 |
| Better.com                     |            1 |        20.70 |
| Renaissance Windows & Doors    |            1 |         9.50 |
| IT Veterans                    |            1 |         7.90 |
| Vuesol Technologies            |            1 |         6.00 |
| Emerald Pools and Spas         |            1 |         5.60 |
| CS Recruiting                  |            1 |         5.50 |
| Valify                         |            1 |         3.10 |
+--------------------------------+-------------+--------------+
14 rows in set (0.00 sec)
```

Most Mentioned Books Across All Podcasts

This query calculates the average rating of books associated with each podcast and ranks them based on their average ratings.

```
SELECT b.BookName, COUNT(br.BookID) AS MentionCount
FROM BookReference br
JOIN Books b ON br.BookID = b.BookID
JOIN Podcast p ON br.PodcastID = p.PodcastID
GROUP BY b.BookName
ORDER BY MentionCount DESC
LIMIT 15;
```

```
mysql> SELECT b.BookName, COUNT(br.BookID) AS MentionCount
    -> FROM BookReference br
    -> JOIN Books b ON br.BookID = b.BookID
    -> JOIN Podcast p ON br.PodcastID = p.PodcastID
    -> GROUP BY b.BookName
    -> ORDER BY MentionCount DESC
    -> LIMIT 15;
+-----------------------------------------------------------+--------------+
| BookName                                                  | MentionCount |
+-----------------------------------------------------------+--------------+
| The Letters of John and Abigail Adams                     |            4 |
| Who's To Blame (Sweet Valley High  #66)                   |            3 |
| Irish Girls Are Back in Town                              |            2 |
| Beach Girls                                               |            2 |
| Moral Luck: Philosophical Papers 1973-1980                |            1 |
| The Basic Bakunin                                         |            1 |
| A Year in Chocolate: Four Seasons of Unforgettable Desserts |            1 |
| Under the Glacier                                         |            1 |
| Lost                                                      |            1 |
| Belle Ruin (Emma Graham  #3)                              |            1 |
| 混血王子的背叛 (哈利波特  #6)                                    |            1 |
+-----------------------------------------------------------+--------------+
11 rows in set (0.01 sec)
```

Most Frequently Mentioned People with Their Professions and Net Worth

This query identifies the most frequently mentioned people in all podcasts, along with their professions and average net worth.

SELECT pe.PersonName, pe.Profession, COUNT(pr.PersonID) AS MentionCount, AVG(pe.NetWorth) AS AverageNetWorth
FROM People pe
JOIN PeopleReference pr ON pe.PersonID = pr.PersonID
JOIN Podcast p ON pr.PodcastID = p.PodcastID
GROUP BY pe.PersonID, pe.PersonName, pe.Profession
ORDER BY MentionCount DESC, AverageNetWorth DESC
LIMIT 15;

```
mysql> SELECT pe.PersonName, pe.Profession, COUNT(pr.PersonID) AS MentionCount, AVG(pe.NetWorth) AS AverageNetWorth
    -> FROM People pe
    -> JOIN PeopleReference pr ON pe.PersonID = pr.PersonID
    -> JOIN Podcast p ON pr.PodcastID = p.PodcastID
    -> GROUP BY pe.PersonID, pe.PersonName, pe.Profession
    -> ORDER BY MentionCount DESC, AverageNetWorth DESC
    -> LIMIT 15;
+------------------+--------------+-------------+-----------------+
| PersonName       | Profession   | MentionCount | AverageNetWorth |
+------------------+--------------+-------------+-----------------+
| Leonardo DiCaprio | Actors       |           3 |       77.000000 |
| Drake            | Musicians    |           3 |       38.500000 |
| Lionel Messi     | Athletes     |           2 |       81.500000 |
| Gordon Ramsay    | Personalities |           2 |       70.000000 |
| Stephenie Meyer  | Authors      |           2 |       40.000000 |
| Matt Damon       | Actors       |           2 |       27.000000 |
| Tiger Woods      | Athletes     |           1 |       63.900000 |
| Jackie Chan      | Actors       |           1 |       58.000000 |
| Justin Bieber    | Musicians    |           1 |       55.000000 |
| Renee Zellweger  | Actresses    |           1 |       21.000000 |
| Julia Roberts    | Actresses    |           1 |       20.000000 |
+------------------+--------------+-------------+-----------------+
11 rows in set (0.01 sec)
```

Indexing Analysis:

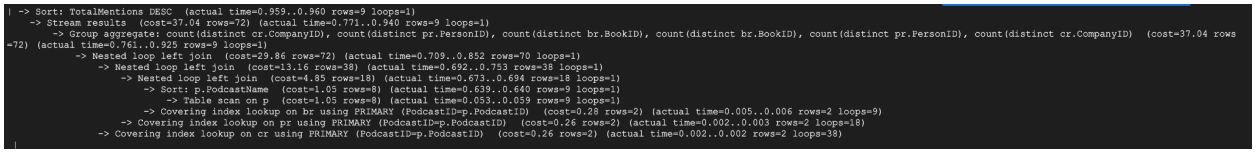
We used the following indexes for all of our advanced queries:

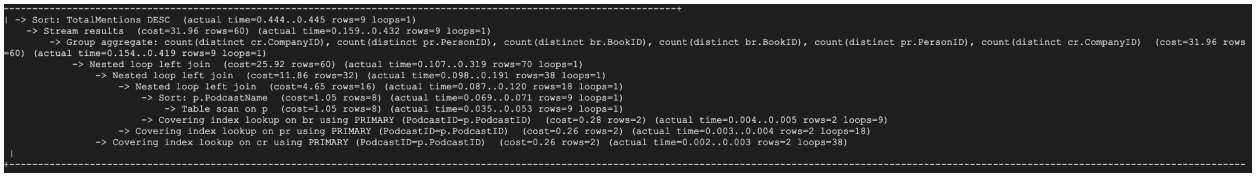
create index Company_CompanyID on Companies(CompanyID);
create index People_PersonID on People(PersonID);
create index Books_BookID on Books(BookID);

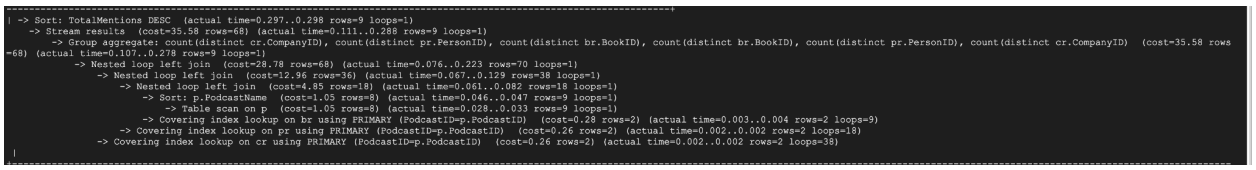
In the screenshots here is the order of the indexing that we used:

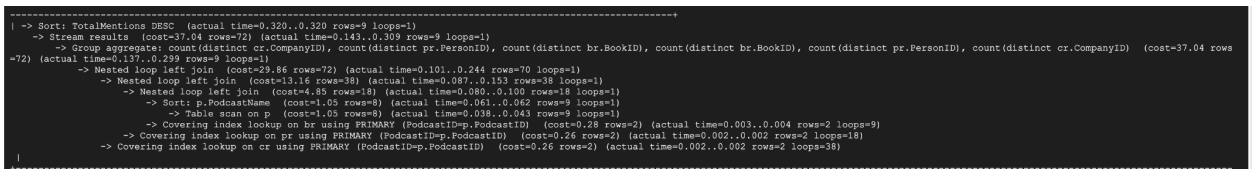
Default Index
Books_BookID Index
People_PersonID Index
Company_CompanyID Index

Most Mentioned Entities by Podcast Query Indexing:







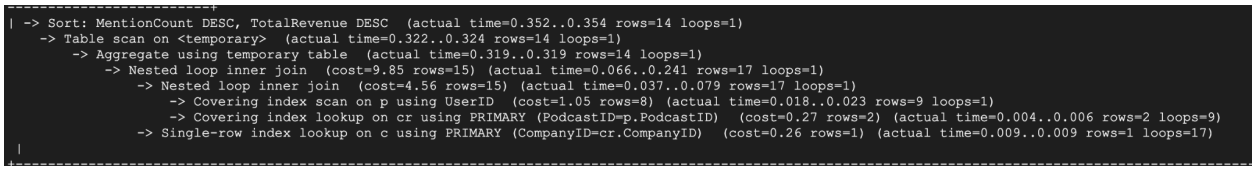


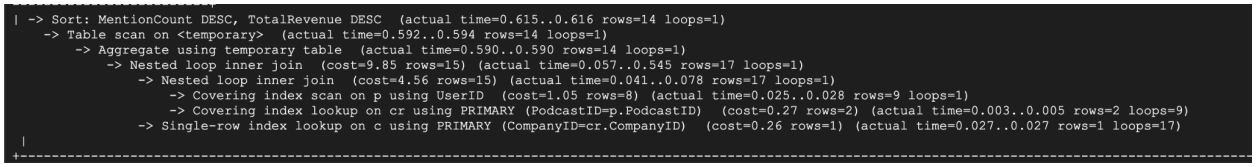
We tested different indexes to help count unique CompanyID, PersonID, and BookID in the Podcast table.

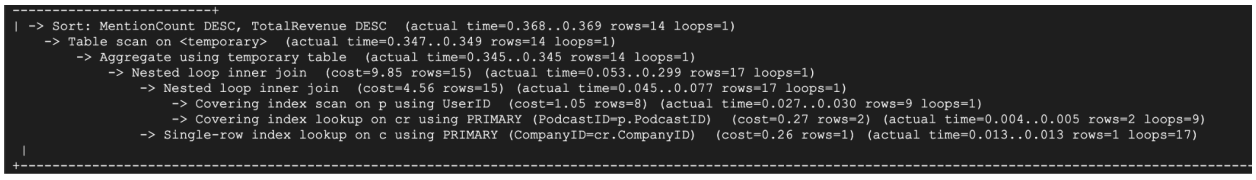
- Default Index: This setup provided a good baseline with a cost of 37.04 and an actual time of 0.959 ms. It was reasonably efficient but not the lowest in time or cost.
- Books_BookID Index: This index was the best, with a reduced cost of 31.96 and an actual time of 0.444 ms. It optimized both counting and joining, making it more efficient than the default in both speed and cost.
- People_PersonID Index: This index was slightly slower than the default, with a cost of 35.58 and an actual time of 0.320 ms. It didn't improve costs much and wasn't helpful for counting and joining.
- Company_CompanyID Index: This index offered a moderate improvement with a cost of 37.04 and an actual time of 0.297 ms. Although it was a bit faster in execution time, its cost was the same as the default, so it didn't bring significant savings.

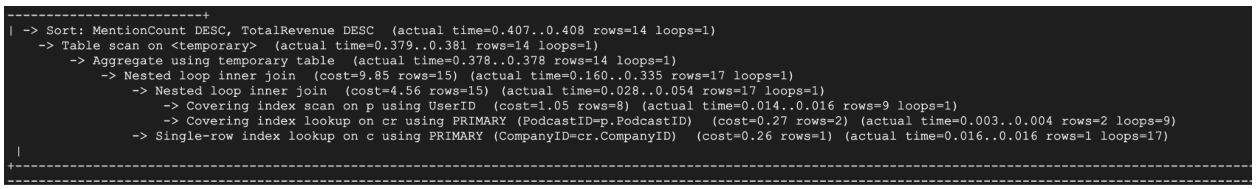
Result: The Books_BookID index worked best for this query, as it had the lowest cost and a faster time than other indexes. The other indexes did not improve the query's efficiency significantly.

Most Companies Mentioned in Podcasts with Total Revenue Query Indexing:







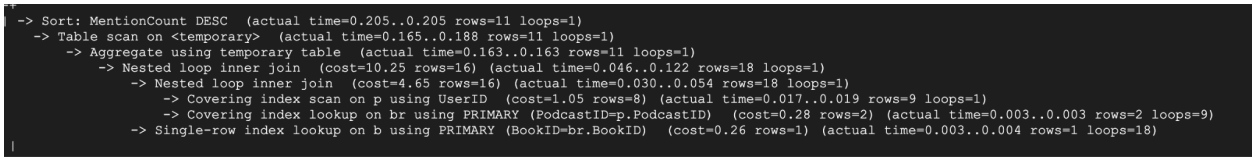


For this query, we tested different indexes to make sorting MentionCount and TotalRevenue faster.

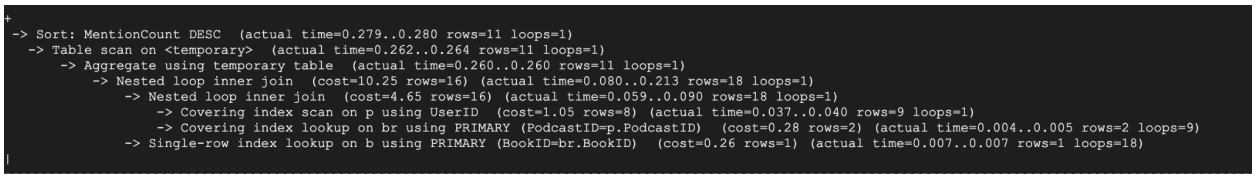
- Default Index: The default index was good with a time of 0.352 ms. It handled the sorting well.
- Books_BookID Index: This was the slowest index with a time of 0.615 ms. It didn't help the query much.
- People_PersonID Index: This was a bit faster than the default at 0.368 ms, but the difference wasn't big.
- Company_CompanyID Index: This index had a time of 0.407 ms. It didn't improve the query much either.

Result: The default index worked best for this query. Other indexes did not make it faster.

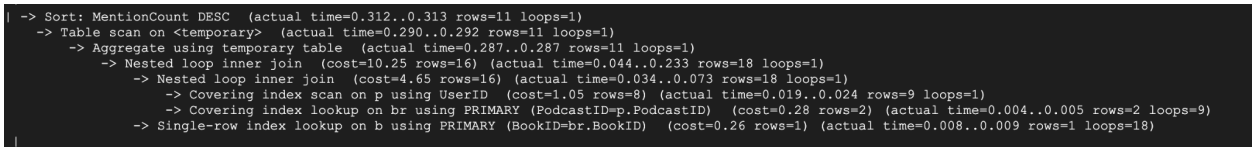
Most Mentioned Books Across All Podcasts Query Indexing:



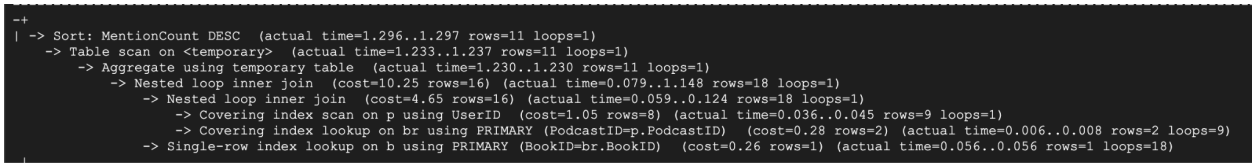
```
-> Sort: MentionCount DESC  (actual time=0.205..0.205 rows=11 loops=1)
  -> Table scan on <temporary>  (actual time=0.165..0.188 rows=11 loops=1)
    -> Aggregate using temporary table  (actual time=0.163..0.163 rows=11 loops=1)
      -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.046..0.122 rows=18 loops=1)
        -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.030..0.054 rows=18 loops=1)
          -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.017..0.019 rows=9 loops=1)
          -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.003..0.003 rows=2 loops=9)
        -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.003..0.004 rows=1 loops=18)
```



```
-> Sort: MentionCount DESC  (actual time=0.279..0.280 rows=11 loops=1)
  -> Table scan on <temporary>  (actual time=0.262..0.264 rows=11 loops=1)
    -> Aggregate using temporary table  (actual time=0.260..0.260 rows=11 loops=1)
      -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.080..0.213 rows=18 loops=1)
        -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.059..0.090 rows=18 loops=1)
          -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.037..0.040 rows=9 loops=1)
          -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.004..0.005 rows=2 loops=9)
        -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.007..0.007 rows=1 loops=18)
```



```
-> Sort: MentionCount DESC  (actual time=0.312..0.313 rows=11 loops=1)
  -> Table scan on <temporary>  (actual time=0.290..0.292 rows=11 loops=1)
    -> Aggregate using temporary table  (actual time=0.287..0.287 rows=11 loops=1)
      -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.044..0.233 rows=18 loops=1)
        -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.034..0.073 rows=18 loops=1)
          -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.019..0.024 rows=9 loops=1)
          -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.004..0.005 rows=2 loops=9)
        -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.008..0.009 rows=1 loops=18)
```



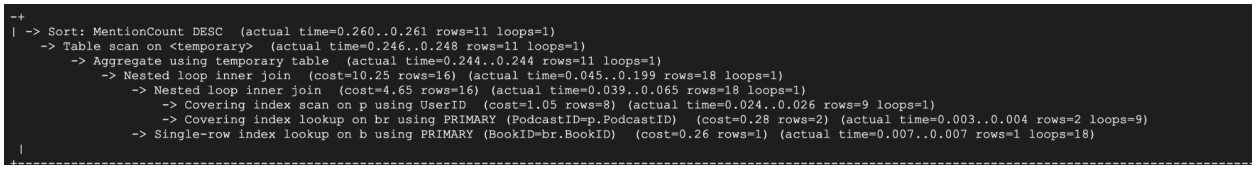
```
-> Sort: MentionCount DESC  (actual time=1.296..1.297 rows=11 loops=1)
  -> Table scan on <temporary>  (actual time=1.233..1.237 rows=11 loops=1)
    -> Aggregate using temporary table  (actual time=1.230..1.230 rows=11 loops=1)
      -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.079..1.148 rows=18 loops=1)
        -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.059..0.124 rows=18 loops=1)
          -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.036..0.045 rows=9 loops=1)
          -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.006..0.008 rows=2 loops=9)
        -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.056..0.056 rows=1 loops=18)
```

We tried different indexes to make sorting MentionCount in the Books table faster.

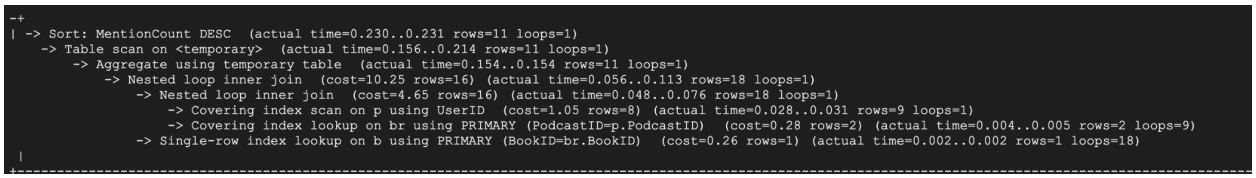
- Default Index: The default setup was the best with a time of 0.205 ms. It was fast and balanced.
- Books_BookID Index: This index was a little slower at 0.279 ms and didn't add much value.
- People_PersonID Index: This index was slower than the default, with a time of 0.312 ms. It didn't help with sorting.
- Company_CompanyID Index: This was the slowest index, with a time of 1.296 ms. It didn't fit this query.

Result: The default index worked best for this query, so we don’t need other indexes.

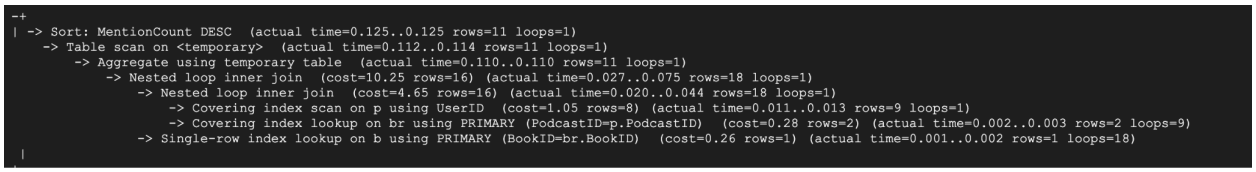
Most Frequently Mentioned People with Their Professions and Net Worth Query Indexing:



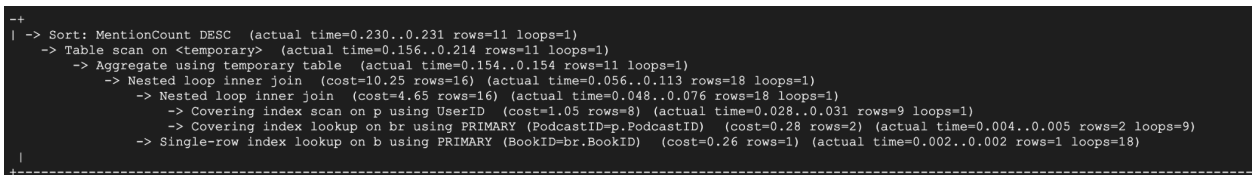
```
-+
| -> Sort: MentionCount DESC  (actual time=0.260..0.261 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=0.246..0.248 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=0.244..0.244 rows=11 loops=1)
            -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.045..0.199 rows=18 loops=1)
                -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.039..0.065 rows=18 loops=1)
                    -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.024..0.026 rows=9 loops=1)
                    -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.003..0.004 rows=2 loops=9)
                -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.007..0.007 rows=1 loops=18)
 |
+-
```



```
-+
| -> Sort: MentionCount DESC  (actual time=0.230..0.231 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=0.156..0.214 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=0.154..0.154 rows=11 loops=1)
            -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.056..0.113 rows=18 loops=1)
                -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.048..0.076 rows=18 loops=1)
                    -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.028..0.031 rows=9 loops=1)
                    -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.004..0.005 rows=2 loops=9)
                -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.002..0.002 rows=1 loops=18)
 |
+-
```



```
-+
| -> Sort: MentionCount DESC  (actual time=0.125..0.125 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=0.112..0.114 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=0.110..0.110 rows=11 loops=1)
            -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.027..0.075 rows=18 loops=1)
                -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.020..0.044 rows=18 loops=1)
                    -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.011..0.013 rows=9 loops=1)
                    -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.002..0.003 rows=2 loops=9)
                -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.001..0.002 rows=1 loops=18)
 |
```



```
-+
| -> Sort: MentionCount DESC  (actual time=0.230..0.231 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=0.156..0.214 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=0.154..0.154 rows=11 loops=1)
            -> Nested loop inner join  (cost=10.25 rows=16) (actual time=0.056..0.113 rows=18 loops=1)
                -> Nested loop inner join  (cost=4.65 rows=16) (actual time=0.048..0.076 rows=18 loops=1)
                    -> Covering index scan on p using UserID  (cost=1.05 rows=8) (actual time=0.028..0.031 rows=9 loops=1)
                    -> Covering index lookup on br using PRIMARY (PodcastID=p.PodcastID)  (cost=0.28 rows=2) (actual time=0.004..0.005 rows=2 loops=9)
                -> Single-row index lookup on b using PRIMARY (BookID=br.BookID)  (cost=0.26 rows=1) (actual time=0.002..0.002 rows=1 loops=18)
 |
+-
```

We tested indexes to make sorting MentionCount faster in this query.

- Default Index: The default setup was okay, with a time of 0.260 ms. It handled the sorting fine.
- Books_BookID Index: This index was a bit faster than the default at 0.230 ms, but the improvement was small.
- People_PersonID Index: This index was the fastest with a time of 0.125 ms. It helped the sorting a lot.
- Company_CompanyID Index: This index was like the Books_BookID index, with a time of 0.230 ms. It didn’t add much.

Result: The People_PersonID index worked best for this query. It saved the most time.