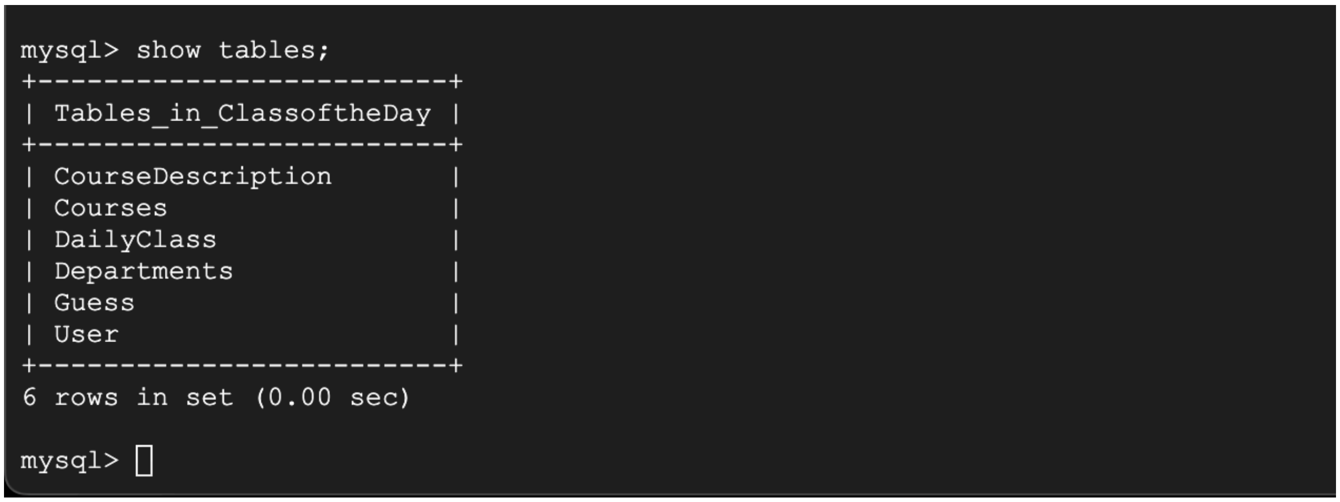
CS 411 Group 116 Project – Stage 3 Revision

Sruthi Kode (kode2), Samidha Sampat (ssamp2),
James Mallek (jmallek2), Kavya Moharana (kavyam3)

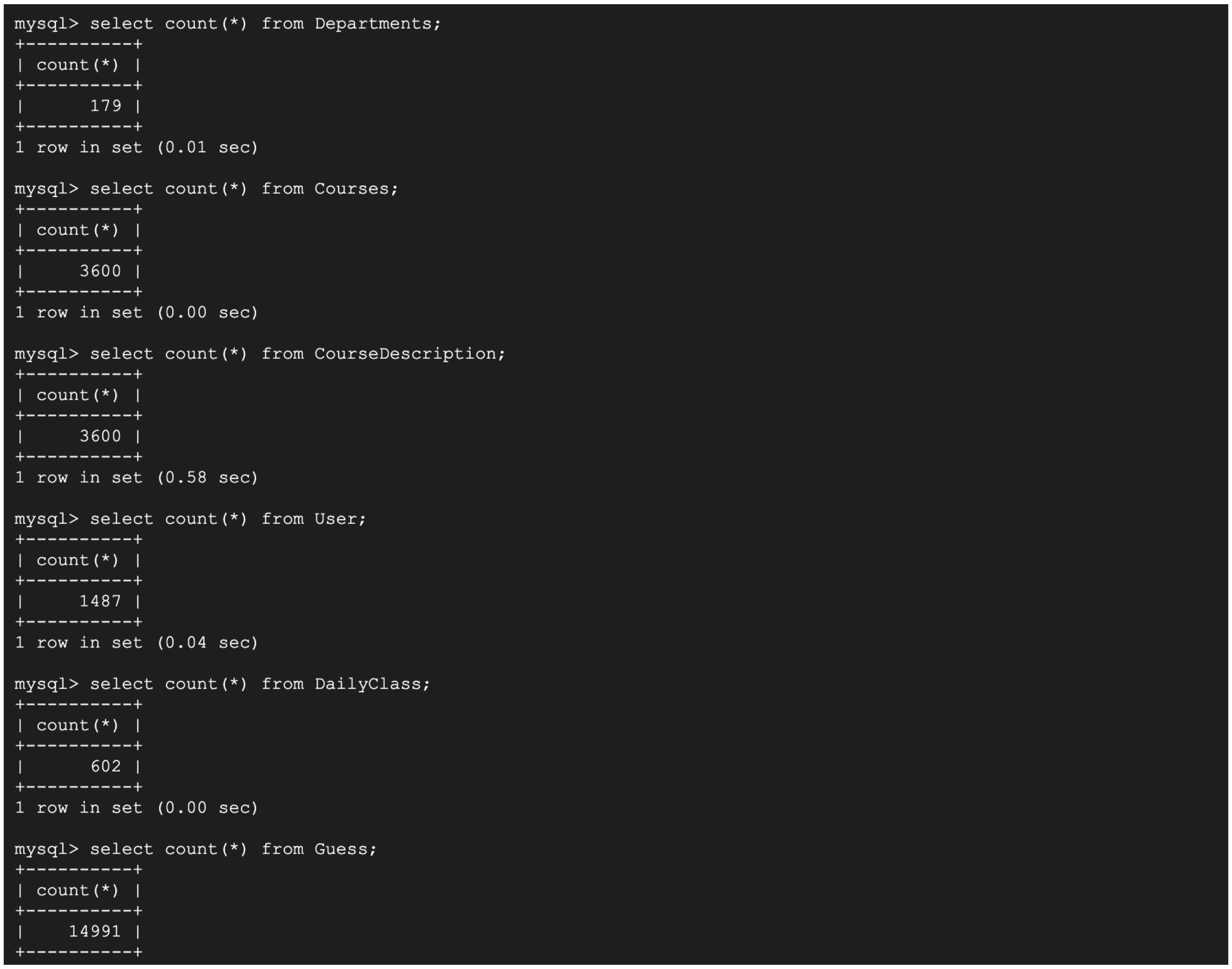
Database Implementation — DDL Commands:

```
CREATE TABLE Departments(
  DepartmentId VARCHAR(10),
  Name VARCHAR(255),
  PRIMARY KEY(DepartmentId)
);

CREATE TABLE Courses(
  CourseId Int,
  DepartmentId VARCHAR(10),
  Number Int,
  Name VARCHAR(255),
  Credits VARCHAR(50),
  GenEd VARCHAR(63),
  FOREIGN KEY(DepartmentId) REFERENCES Departments(DepartmentId),
  PRIMARY KEY(CourseId)
);

CREATE TABLE CourseDescription(
  CourseId Int,
  Description VARCHAR(1023),
  Instructors VARCHAR(255),
  FOREIGN KEY(CourseId) REFERENCES Courses(CourseId),
  PRIMARY KEY(CourseId)
);

CREATE TABLE User(
  UserId VARCHAR(31),
  Email VARCHAR(63),
  EncryptedPassword VARCHAR(255),
  DateJoined DATE,
  MostGuessed Int,
  AvgGuesses Decimal,
  FOREIGN KEY(MostGuessed) REFERENCES Courses(CourseId),
  PRIMARY KEY(UserId)
);

CREATE TABLE DailyClass(
  CurrentDate DATE,
  CorrectCourseId Int,
  MostCommonWrong Int,
  TotalAvgGuesses Decimal,
  FOREIGN KEY(CorrectCourseId) REFERENCES Courses(CourseId),
  PRIMARY KEY(CurrentDate)
);

CREATE TABLE Guess(
  GuessId Int,
  UserId VARCHAR(31),
  CurrentDate DATE,
  CourseId INT,
  FOREIGN KEY(UserId) REFERENCES User(UserId),
  FOREIGN KEY(CurrentDate) REFERENCES DailyClass(CurrentDate),
  FOREIGN KEY(CourseId) REFERENCES Courses(CourseId),
  PRIMARY KEY(GuessID)
);
```

Connection and Database Tables Screenshots:



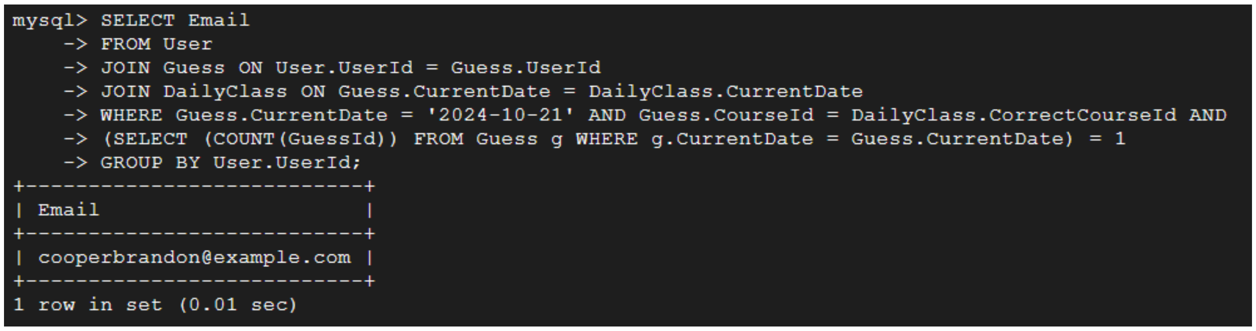
```
mysql> show tables;
+------------------------+
| Tables_in_ClassoftheDay |
+------------------------+
| CourseDescription      |
| Courses                |
| DailyClass             |
| Departments            |
| Guess                  |
| User                   |
+------------------------+
6 rows in set (0.00 sec)

mysql> 
```



```
mysql> select count(*) from Departments;
+----------+
| count(*) |
+----------+
|      179 |
+----------+
1 row in set (0.01 sec)

mysql> select count(*) from Courses;
+----------+
| count(*) |
+----------+
|     3600 |
+----------+
1 row in set (0.00 sec)

mysql> select count(*) from CourseDescription;
+----------+
| count(*) |
+----------+
|     3600 |
+----------+
1 row in set (0.58 sec)

mysql> select count(*) from User;
+----------+
| count(*) |
+----------+
|     1487 |
+----------+
1 row in set (0.04 sec)

mysql> select count(*) from DailyClass;
+----------+
| count(*) |
+----------+
|      602 |
+----------+
1 row in set (0.00 sec)

mysql> select count(*) from Guess;
+----------+
| count(*) |
+----------+
|    14991 |
+----------+
```

Query 1: Gets the emails of all users who guessed the correct course on their first try

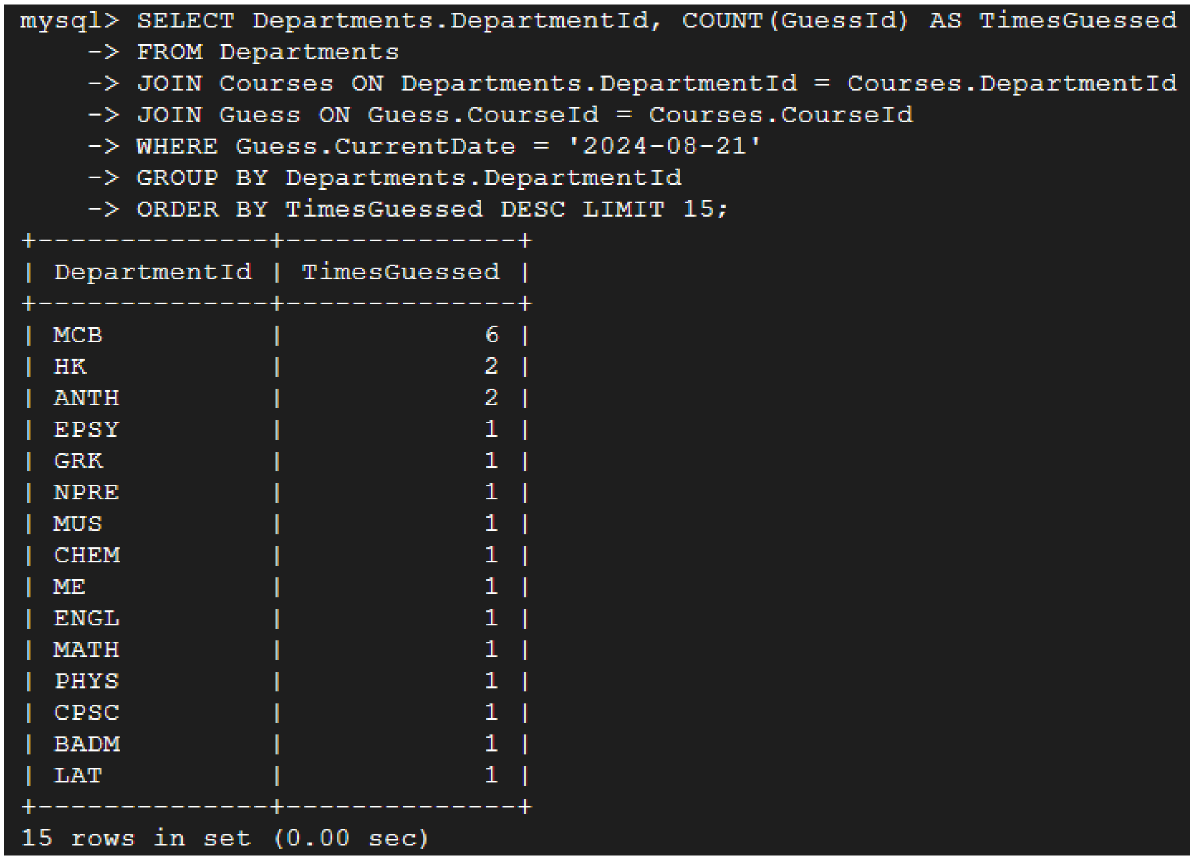
```
SELECT Email
FROM User
JOIN Guess ON User.UserId = Guess.UserId
JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
WHERE Guess.CurrentDate = '2024-10-21' AND Guess.CourseId =
DailyClass.CorrectCourseId AND
(SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate =
Guess.CurrentDate) = 1
GROUP BY User.UserId;
```


```
mysql> SELECT Email
    -> FROM User
    -> JOIN Guess ON User.UserId = Guess.UserId
    -> JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
    -> WHERE Guess.CurrentDate = '2024-10-21' AND Guess.CourseId = DailyClass.CorrectCourseId AND
    -> (SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate = Guess.CurrentDate) = 1
    -> GROUP BY User.UserId;
+--------------------------+
| Email                    |
+--------------------------+
| cooperbrandon@example.com |
+--------------------------+
1 row in set (0.01 sec)
```

There is only 1 row in the result of this query because only one user guessed the correct course in one try on this day.

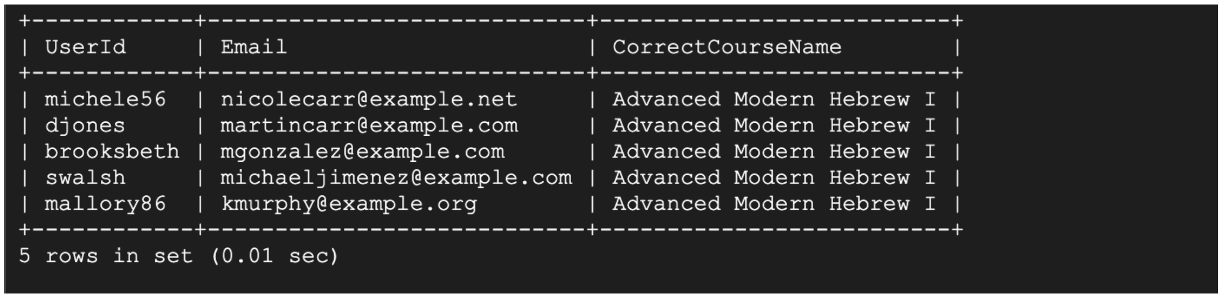
Query 2: Gets the most commonly guessed departments on a given day

```
SELECT Departments.DepartmentId, COUNT(GuessId) AS TimesGuessed
FROM Departments
      JOIN Courses ON Departments.DepartmentId =
      Courses.DepartmentId
      JOIN Guess ON Guess.CourseId = Courses.CourseId
WHERE Guess.CurrentDate = '2024-08-21'
GROUP BY Departments.DepartmentId
ORDER BY TimesGuessed DESC;
```


```
mysql> SELECT Departments.DepartmentId, COUNT(GuessId) AS TimesGuessed
    -> FROM Departments
    -> JOIN Courses ON Departments.DepartmentId = Courses.DepartmentId
    -> JOIN Guess ON Guess.CourseId = Courses.CourseId
    -> WHERE Guess.CurrentDate = '2024-08-21'
    -> GROUP BY Departments.DepartmentId
    -> ORDER BY TimesGuessed DESC LIMIT 15;
+--------------+--------------+
| DepartmentId | TimesGuessed |
+--------------+--------------+
| MCB          |            6 |
| HK           |            2 |
| ANTH         |            2 |
| EPSY         |            1 |
| GRK          |            1 |
| NPRE         |            1 |
| MUS          |            1 |
| CHEM         |            1 |
| ME           |            1 |
| ENGL         |            1 |
| MATH         |            1 |
| PHYS         |            1 |
| CPSC         |            1 |
| BADM         |            1 |
| LAT          |            1 |
+--------------+--------------+
15 rows in set (0.00 sec)
```

Query 3: Gets the users who guessed correctly on a given date

```
SELECT User.UserId, User.Email, Courses.Name AS CorrectCourseName
FROM Guess
     JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
     JOIN Courses ON DailyClass.CorrectCourseId = Courses.CourseId
     JOIN User ON Guess.UserId = User.UserId
WHERE Guess.CourseId = DailyClass.CorrectCourseId
AND Guess.CurrentDate = "2023-11-14";
```

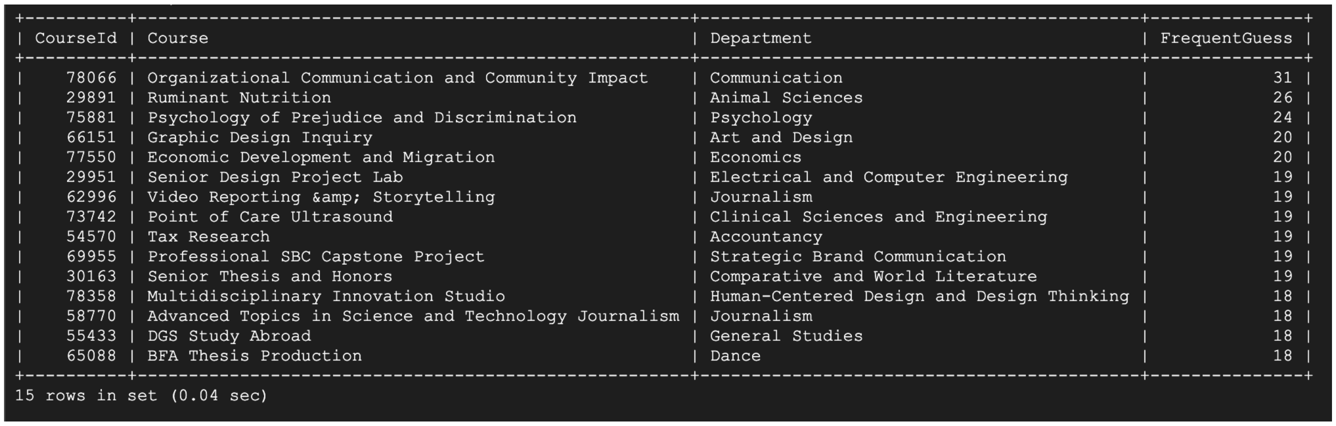


```
+------------+----------------------------+--------------------------+
| UserId     | Email                      | CorrectCourseName        |
+------------+----------------------------+--------------------------+
| michele56  | nicolecarr@example.net     | Advanced Modern Hebrew I |
| djones     | martincarr@example.com     | Advanced Modern Hebrew I |
| brooksbeth | mgonzalez@example.com      | Advanced Modern Hebrew I |
| swalsh     | michaeljimenez@example.com | Advanced Modern Hebrew I |
| mallory86  | kmurphy@example.org        | Advanced Modern Hebrew I |
+------------+----------------------------+--------------------------+
5 rows in set (0.01 sec)
```

(output is less than 15 rows)

Query 4: Gets the top 15 most frequently guessed courses overall

```
SELECT Courses.CourseId, Courses.Name AS Course, Departments.Name AS
Department, COUNT(Guess.GuessId) AS FrequentGuess
FROM Courses
     JOIN Guess ON Courses.CourseId = Guess.CourseId
     JOIN Departments ON Courses.DepartmentId =
     Departments.DepartmentId
GROUP BY Courses.CourseId, Courses.Name
ORDER BY FrequentGuess DESC
LIMIT 15;
```



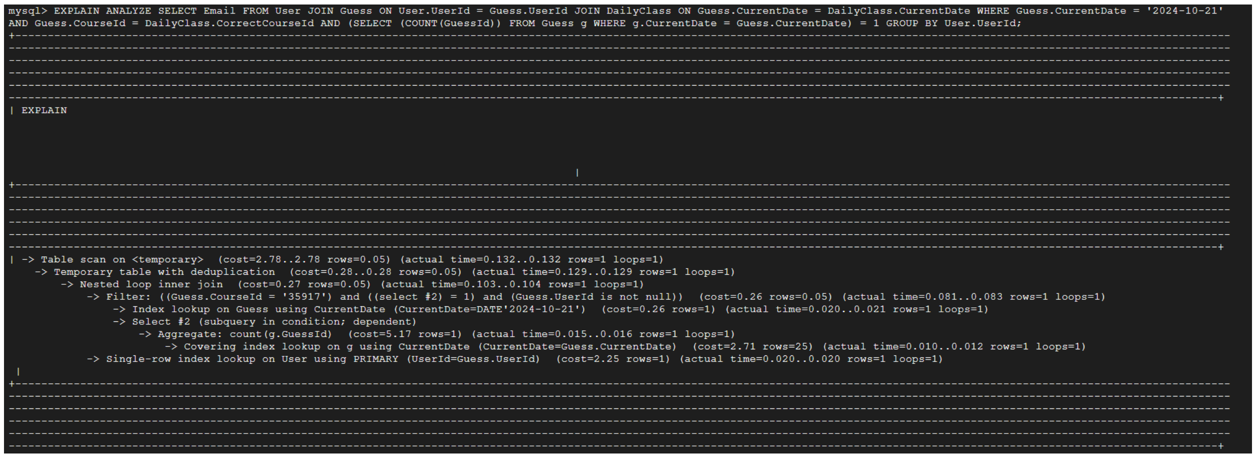
```
+----------+------------------------------------------------------+-------------------------------------------+---------------+
| CourseId | Course                                               | Department                                | FrequentGuess |
+----------+------------------------------------------------------+-------------------------------------------+---------------+
|    78066 | Organizational Communication and Community Impact    | Communication                             |            31 |
|    29891 | Ruminant Nutrition                                   | Animal Sciences                           |            26 |
|    75881 | Psychology of Prejudice and Discrimination           | Psychology                                |            24 |
|    66151 | Graphic Design Inquiry                               | Art and Design                            |            20 |
|    77550 | Economic Development and Migration                   | Economics                                 |            20 |
|    29951 | Senior Design Project Lab                            | Electrical and Computer Engineering       |            19 |
|    62996 | Video Reporting &amp; Storytelling                   | Journalism                                |            19 |
|    73742 | Point of Care Ultrasound                             | Clinical Sciences and Engineering         |            19 |
|    54570 | Tax Research                                         | Accountancy                               |            19 |
|    69955 | Professional SBC Capstone Project                    | Strategic Brand Communication             |            19 |
|    30163 | Senior Thesis and Honors                             | Comparative and World Literature          |            19 |
|    78358 | Multidisciplinary Innovation Studio                  | Human-Centered Design and Design Thinking |            18 |
|    58770 | Advanced Topics in Science and Technology Journalism | Journalism                                |            18 |
|    55433 | DGS Study Abroad                                     | General Studies                           |            18 |
|    65088 | BFA Thesis Production                                | Dance                                     |            18 |
+----------+------------------------------------------------------+-------------------------------------------+---------------+
15 rows in set (0.04 sec)
```

Indexing Analysis –

Query 1 for Attempted Indexing Designs:

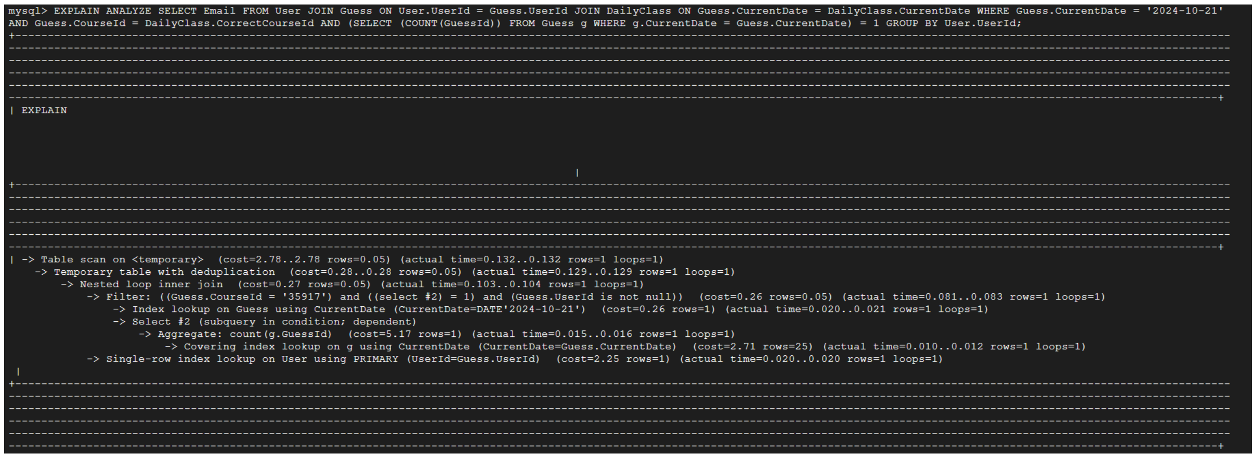
```
EXPLAIN ANALYZE SELECT Email
FROM User
JOIN Guess ON User.UserId = Guess.UserId
JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
WHERE Guess.CurrentDate = '2024-10-21' AND Guess.CourseId =
DailyClass.CorrectCourseId AND
(SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate =
Guess.CurrentDate) = 1
GROUP BY User.UserId;
```

- `CREATE INDEX Current_Date ON Guess(CurrentDate);`



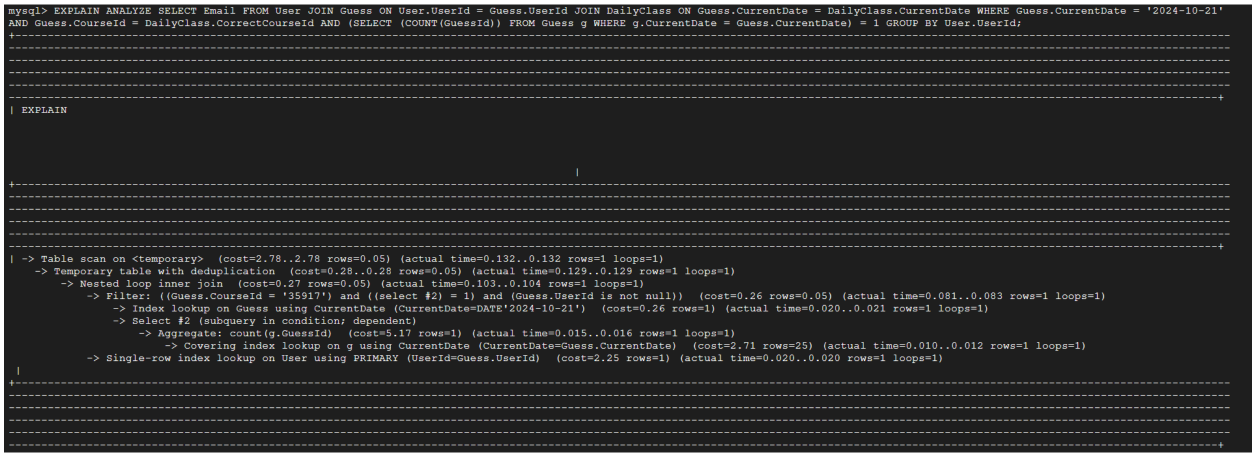
```
mysql> EXPLAIN ANALYZE SELECT Email FROM User JOIN Guess ON User.UserId = Guess.UserId JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate WHERE Guess.CurrentDate = '2024-10-21'
AND Guess.CourseId = DailyClass.CorrectCourseId AND (SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate = Guess.CurrentDate) = 1 GROUP BY User.UserId;
+--------------------------------------------------------------------------------------------------------------------+
| EXPLAIN                                                                                                            |
+--------------------------------------------------------------------------------------------------------------------+
| -> Table scan on <temporary>  (cost=2.78..2.78 rows=0.05) (actual time=0.132..0.132 rows=1 loops=1)
    -> Temporary table with deduplication  (cost=0.28..0.28 rows=0.05) (actual time=0.129..0.129 rows=1 loops=1)
        -> Nested loop inner join  (cost=0.27 rows=0.05) (actual time=0.103..0.104 rows=1 loops=1)
            -> Filter: ((Guess.CourseId = '35917') and ((select #2) = 1) and (Guess.UserId is not null))  (cost=0.26 rows=0.05) (actual time=0.081..0.083 rows=1 loops=1)
                -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-10-21')  (cost=0.26 rows=1) (actual time=0.020..0.021 rows=1 loops=1)
                -> Select #2 (subquery in condition; dependent)
                    -> Aggregate: count(g.GuessId)  (cost=5.17 rows=1) (actual time=0.015..0.016 rows=1 loops=1)
                        -> Covering index lookup on g using CurrentDate (CurrentDate=Guess.CurrentDate)  (cost=2.71 rows=25) (actual time=0.010..0.012 rows=1 loops=1)
            -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=2.25 rows=1) (actual time=0.020..0.020 rows=1 loops=1)
 |
+--------------------------------------------------------------------------------------------------------------------+
```

- `CREATE INDEX Course_Id ON Guess(CourseId);`



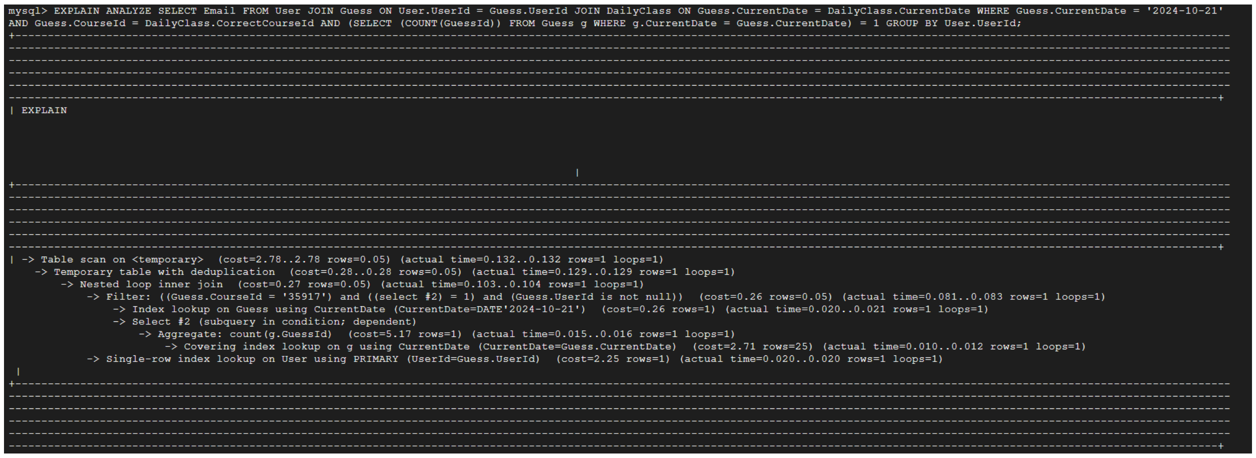
```
mysql> EXPLAIN ANALYZE SELECT Email FROM User JOIN Guess ON User.UserId = Guess.UserId JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate WHERE Guess.CurrentDate = '2024-10-21'
AND Guess.CourseId = DailyClass.CorrectCourseId AND (SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate = Guess.CurrentDate) = 1 GROUP BY User.UserId;
+--------------------------------------------------------------------------------------------------------------------+
| EXPLAIN                                                                                                            |
+--------------------------------------------------------------------------------------------------------------------+
| -> Table scan on <temporary>  (cost=2.78..2.78 rows=0.05) (actual time=0.132..0.132 rows=1 loops=1)
    -> Temporary table with deduplication  (cost=0.28..0.28 rows=0.05) (actual time=0.129..0.129 rows=1 loops=1)
        -> Nested loop inner join  (cost=0.27 rows=0.05) (actual time=0.103..0.104 rows=1 loops=1)
            -> Filter: ((Guess.CourseId = '35917') and ((select #2) = 1) and (Guess.UserId is not null))  (cost=0.26 rows=0.05) (actual time=0.081..0.083 rows=1 loops=1)
                -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-10-21')  (cost=0.26 rows=1) (actual time=0.020..0.021 rows=1 loops=1)
                -> Select #2 (subquery in condition; dependent)
                    -> Aggregate: count(g.GuessId)  (cost=5.17 rows=1) (actual time=0.015..0.016 rows=1 loops=1)
                        -> Covering index lookup on g using CurrentDate (CurrentDate=Guess.CurrentDate)  (cost=2.71 rows=25) (actual time=0.010..0.012 rows=1 loops=1)
            -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=2.25 rows=1) (actual time=0.020..0.020 rows=1 loops=1)
 |
+--------------------------------------------------------------------------------------------------------------------+
```

- `CREATE INDEX Correct_Course_Id ON DailyClass(CorrectCourseId);`



```
mysql> EXPLAIN ANALYZE SELECT Email FROM User JOIN Guess ON User.UserId = Guess.UserId JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate WHERE Guess.CurrentDate = '2024-10-21'
AND Guess.CourseId = DailyClass.CorrectCourseId AND (SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate = Guess.CurrentDate) = 1 GROUP BY User.UserId;
+------------------------------------------------------------------------------------------------------------------------------
| EXPLAIN
                                                                                          |
+------------------------------------------------------------------------------------------------------------------------------
| -> Table scan on <temporary>  (cost=2.78..2.78 rows=0.05) (actual time=0.132..0.132 rows=1 loops=1)
    -> Temporary table with deduplication  (cost=0.28..0.28 rows=0.05) (actual time=0.129..0.129 rows=1 loops=1)
        -> Nested loop inner join  (cost=0.27 rows=0.05) (actual time=0.103..0.104 rows=1 loops=1)
            -> Filter: ((Guess.CourseId = '35917') and ((select #2) = 1) and (Guess.UserId is not null))  (cost=0.26 rows=0.05) (actual time=0.081..0.083 rows=1 loops=1)
                -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-10-21')  (cost=0.26 rows=1) (actual time=0.020..0.021 rows=1 loops=1)
                -> Select #2 (subquery in condition; dependent)
                    -> Aggregate: count(g.GuessId)  (cost=5.17 rows=1) (actual time=0.015..0.016 rows=1 loops=1)
                        -> Covering index lookup on g using CurrentDate (CurrentDate=Guess.CurrentDate)  (cost=2.71 rows=25) (actual time=0.010..0.012 rows=1 loops=1)
            -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=2.25 rows=1) (actual time=0.020..0.020 rows=1 loops=1)
 |
+------------------------------------------------------------------------------------------------------------------------------+
```

- `CREATE INDEX Email ON User(Email);`



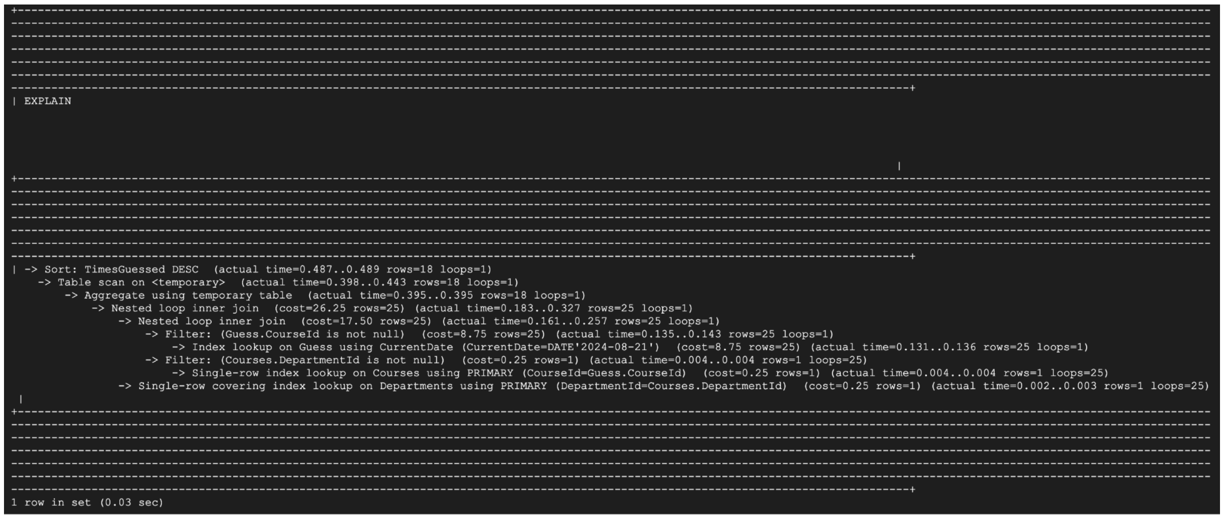
```
mysql> EXPLAIN ANALYZE SELECT Email FROM User JOIN Guess ON User.UserId = Guess.UserId JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate WHERE Guess.CurrentDate = '2024-10-21'
AND Guess.CourseId = DailyClass.CorrectCourseId AND (SELECT (COUNT(GuessId)) FROM Guess g WHERE g.CurrentDate = Guess.CurrentDate) = 1 GROUP BY User.UserId;
+------------------------------------------------------------------------------------------------------------------------------
| EXPLAIN
                                                                                          |
+------------------------------------------------------------------------------------------------------------------------------
| -> Table scan on <temporary>  (cost=2.78..2.78 rows=0.05) (actual time=0.132..0.132 rows=1 loops=1)
    -> Temporary table with deduplication  (cost=0.28..0.28 rows=0.05) (actual time=0.129..0.129 rows=1 loops=1)
        -> Nested loop inner join  (cost=0.27 rows=0.05) (actual time=0.103..0.104 rows=1 loops=1)
            -> Filter: ((Guess.CourseId = '35917') and ((select #2) = 1) and (Guess.UserId is not null))  (cost=0.26 rows=0.05) (actual time=0.081..0.083 rows=1 loops=1)
                -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-10-21')  (cost=0.26 rows=1) (actual time=0.020..0.021 rows=1 loops=1)
                -> Select #2 (subquery in condition; dependent)
                    -> Aggregate: count(g.GuessId)  (cost=5.17 rows=1) (actual time=0.015..0.016 rows=1 loops=1)
                        -> Covering index lookup on g using CurrentDate (CurrentDate=Guess.CurrentDate)  (cost=2.71 rows=25) (actual time=0.010..0.012 rows=1 loops=1)
            -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=2.25 rows=1) (actual time=0.020..0.020 rows=1 loops=1)
 |
+------------------------------------------------------------------------------------------------------------------------------+
```

In this query, indexing did not improve performance. All the queries had the exact same costs. This is likely because MySQL had already automatically indexed many of the fields used in the query. As a result, our additions did not change much.

<u>Query 2 for Attempted Indexing Designs:</u>

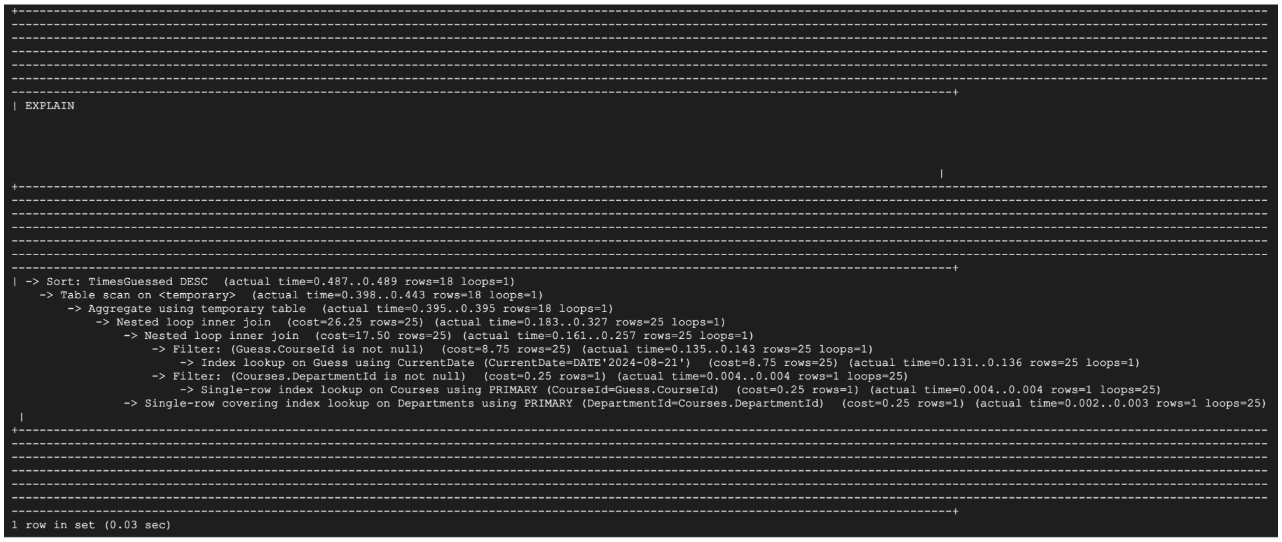
```
EXPLAIN ANALYZE SELECT Departments.DepartmentId, COUNT(GuessId) AS
TimesGuessed
FROM Departments
JOIN Courses ON Departments.DepartmentId = Courses.DepartmentId
JOIN Guess ON Guess.CourseId = Courses.CourseId
WHERE Guess.CurrentDate = '2024-08-21'
GROUP BY Departments.DepartmentId
ORDER BY TimesGuessed DESC;
```

- `CREATE INDEX Course_Id ON Guess(CourseId);`



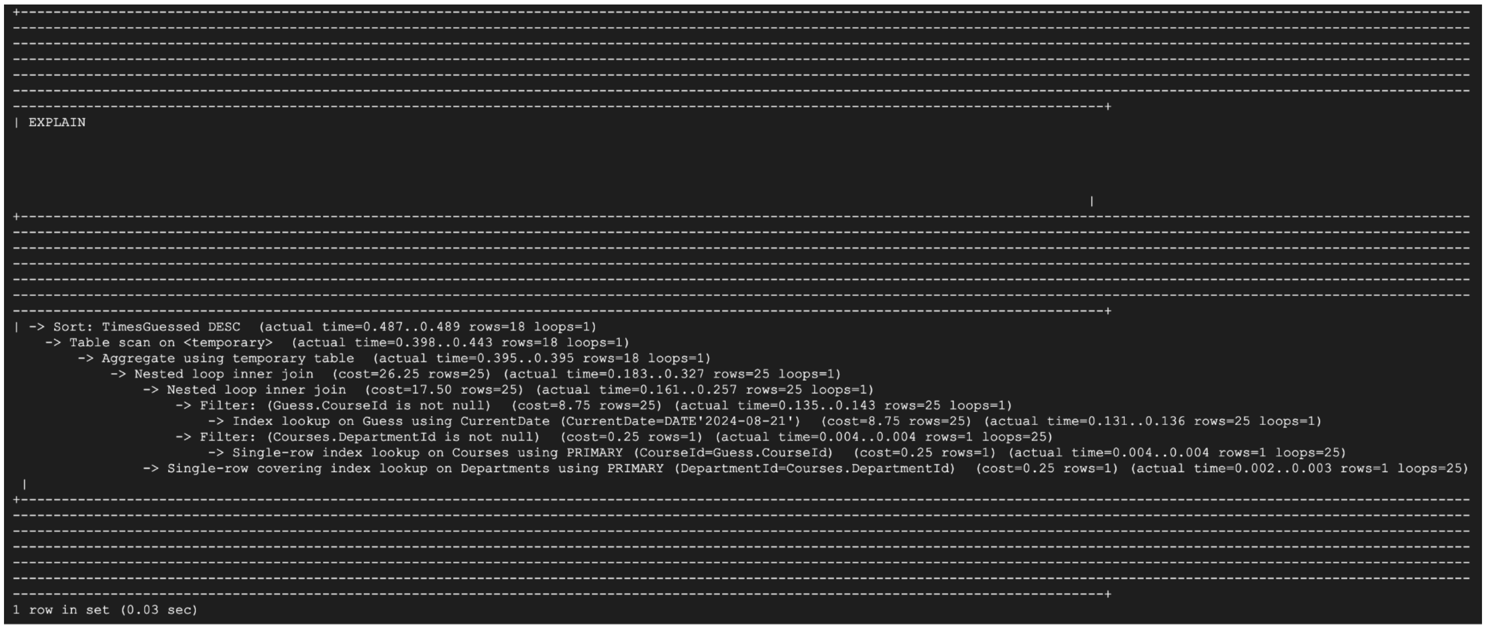
```
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
                                                                                                                          |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------+
| -> Sort: TimesGuessed DESC  (actual time=0.487..0.489 rows=18 loops=1)
    -> Table scan on <temporary>  (actual time=0.398..0.443 rows=18 loops=1)
        -> Aggregate using temporary table  (actual time=0.395..0.395 rows=18 loops=1)
            -> Nested loop inner join  (cost=26.25 rows=25) (actual time=0.183..0.327 rows=25 loops=1)
                -> Nested loop inner join  (cost=17.50 rows=25) (actual time=0.161..0.257 rows=25 loops=1)
                    -> Filter: (Guess.CourseId is not null)  (cost=8.75 rows=25) (actual time=0.135..0.143 rows=25 loops=1)
                        -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-08-21')  (cost=8.75 rows=25) (actual time=0.131..0.136 rows=25 loops=1)
                    -> Filter: (Courses.DepartmentId is not null)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=25)
                        -> Single-row index lookup on Courses using PRIMARY (CourseId=Guess.CourseId)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=25)
                -> Single-row covering index lookup on Departments using PRIMARY (DepartmentId=Courses.DepartmentId)  (cost=0.25 rows=1) (actual time=0.002..0.003 rows=1 loops=25)
 |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.03 sec)
```

- `CREATE INDEX Dept_Id ON Courses(DepartmentId);`



```
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
                                                                                                                          |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------+
| -> Sort: TimesGuessed DESC  (actual time=0.487..0.489 rows=18 loops=1)
    -> Table scan on <temporary>  (actual time=0.398..0.443 rows=18 loops=1)
        -> Aggregate using temporary table  (actual time=0.395..0.395 rows=18 loops=1)
            -> Nested loop inner join  (cost=26.25 rows=25) (actual time=0.183..0.327 rows=25 loops=1)
                -> Nested loop inner join  (cost=17.50 rows=25) (actual time=0.161..0.257 rows=25 loops=1)
                    -> Filter: (Guess.CourseId is not null)  (cost=8.75 rows=25) (actual time=0.135..0.143 rows=25 loops=1)
                        -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-08-21')  (cost=8.75 rows=25) (actual time=0.131..0.136 rows=25 loops=1)
                    -> Filter: (Courses.DepartmentId is not null)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=25)
                        -> Single-row index lookup on Courses using PRIMARY (CourseId=Guess.CourseId)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=25)
                -> Single-row covering index lookup on Departments using PRIMARY (DepartmentId=Courses.DepartmentId)  (cost=0.25 rows=1) (actual time=0.002..0.003 rows=1 loops=25)
 |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.03 sec)
```

- `CREATE INDEX Current_Date ON Guess(CurrentDate);`



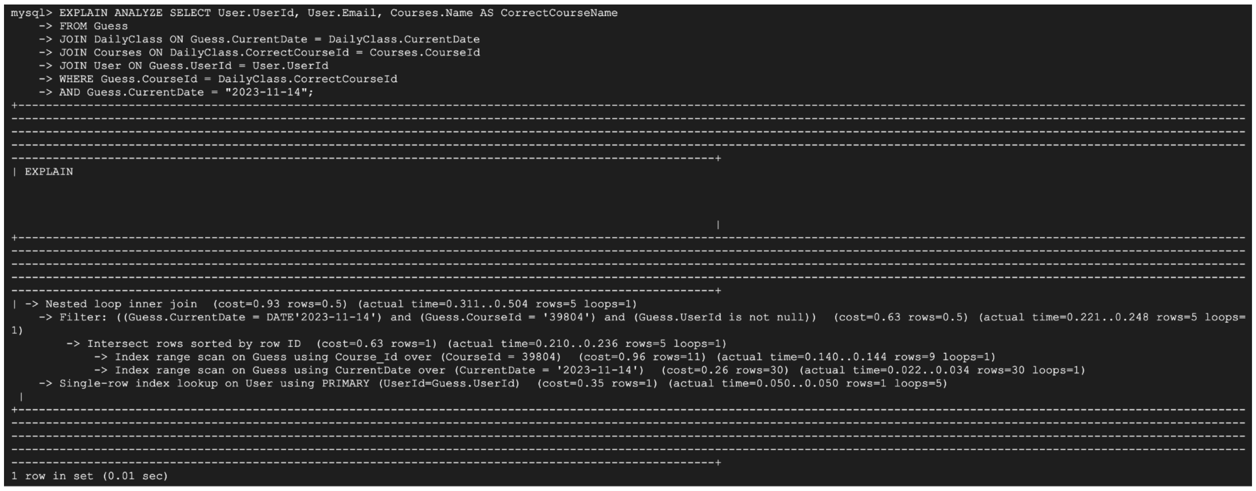
```
+--------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
| EXPLAIN




                                                                                |
+--------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
| -> Sort: TimesGuessed DESC  (actual time=0.487..0.489 rows=18 loops=1)
    -> Table scan on <temporary>  (actual time=0.398..0.443 rows=18 loops=1)
        -> Aggregate using temporary table  (actual time=0.395..0.395 rows=18 loops=1)
            -> Nested loop inner join  (cost=26.25 rows=25) (actual time=0.183..0.327 rows=25 loops=1)
                -> Nested loop inner join  (cost=17.50 rows=25) (actual time=0.161..0.257 rows=25 loops=1)
                    -> Filter: (Guess.CourseId is not null)  (cost=8.75 rows=25) (actual time=0.135..0.143 rows=25 loops=1)
                        -> Index lookup on Guess using CurrentDate (CurrentDate=DATE'2024-08-21')  (cost=8.75 rows=25) (actual time=0.131..0.136 rows=25 loops=1)
                    -> Filter: (Courses.DepartmentId is not null)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=25)
                        -> Single-row index lookup on Courses using PRIMARY (CourseId=Guess.CourseId)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=25)
                -> Single-row covering index lookup on Departments using PRIMARY (DepartmentId=Courses.DepartmentId)  (cost=0.25 rows=1) (actual time=0.002..0.003 rows=1 loops=25)
 |
+--------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
1 row in set (0.03 sec)
```

For this query, we also didn't see any performance improvement. Again, this is likely because the fields used in the query were already indexed by MySQL. We believe that the performance for this query is already optimized.

<u>Query 3 for Attempted Indexing Designs:</u>

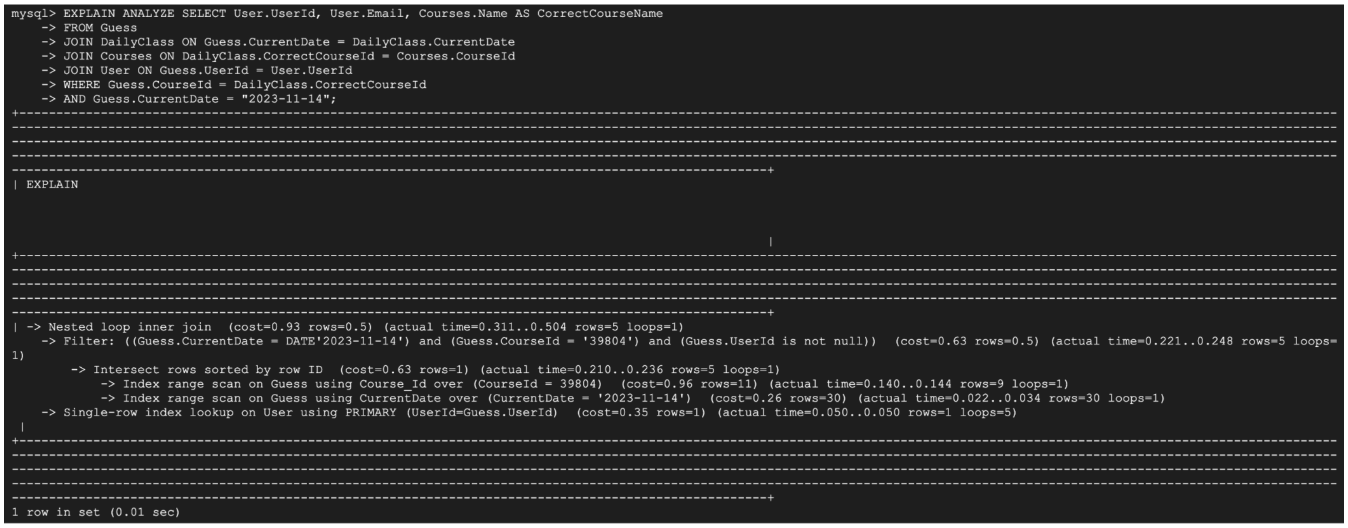
```
EXPLAIN ANALYZE SELECT User.UserId, User.Email, Courses.Name AS
CorrectCourseName
FROM Guess
      JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
      JOIN Courses ON DailyClass.CorrectCourseId = Courses.CourseId
      JOIN User ON Guess.UserId = User.UserId
WHERE Guess.CourseId = DailyClass.CorrectCourseId
AND Guess.CurrentDate = ‘2023-11-14’;
```

- `CREATE INDEX Course_Id ON Guess(CourseId);`



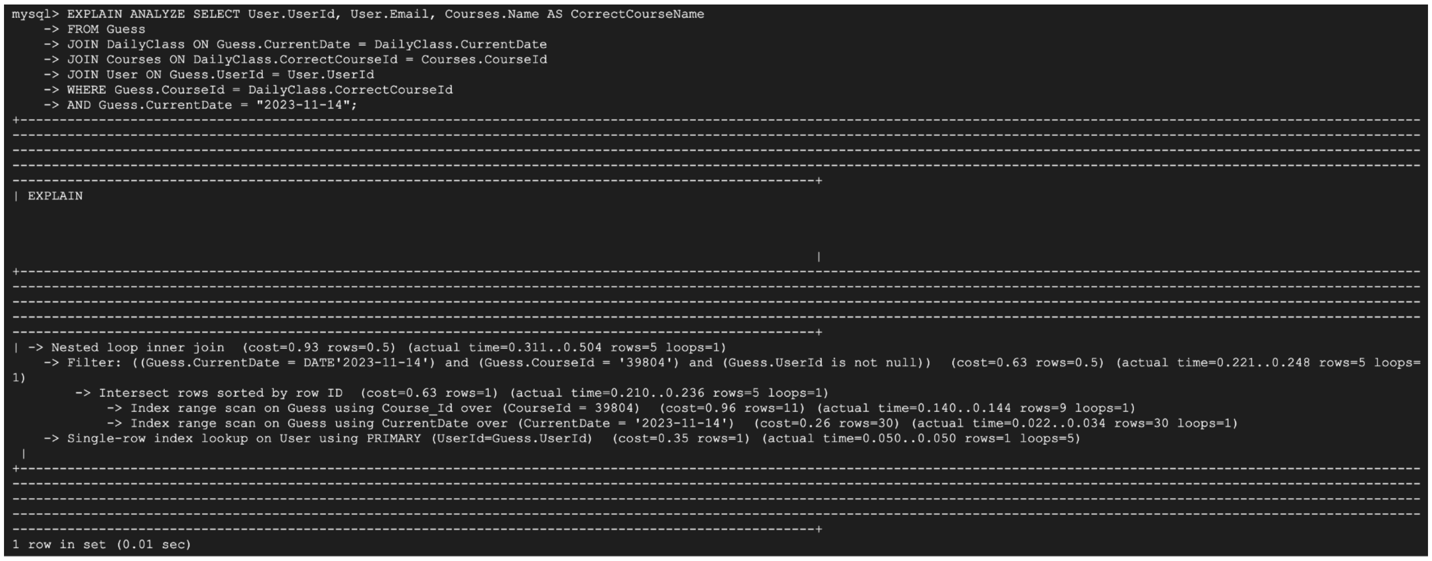
```
mysql> EXPLAIN ANALYZE SELECT User.UserId, User.Email, Courses.Name AS CorrectCourseName
    -> FROM Guess
    -> JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
    -> JOIN Courses ON DailyClass.CorrectCourseId = Courses.CourseId
    -> JOIN User ON Guess.UserId = User.UserId
    -> WHERE Guess.CourseId = DailyClass.CorrectCourseId
    -> AND Guess.CurrentDate = "2023-11-14";
+------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+
| EXPLAIN



                                                                                           |
+------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+
| -> Nested loop inner join  (cost=0.93 rows=0.5) (actual time=0.311..0.504 rows=5 loops=1)
    -> Filter: ((Guess.CurrentDate = DATE'2023-11-14') and (Guess.CourseId = '39804') and (Guess.UserId is not null))  (cost=0.63 rows=0.5) (actual time=0.221..0.248 rows=5 loops=
1)
        -> Intersect rows sorted by row ID  (cost=0.63 rows=1) (actual time=0.210..0.236 rows=5 loops=1)
            -> Index range scan on Guess using Course_Id over (CourseId = 39804)  (cost=0.96 rows=11) (actual time=0.140..0.144 rows=9 loops=1)
            -> Index range scan on Guess using CurrentDate over (CurrentDate = '2023-11-14')  (cost=0.26 rows=30) (actual time=0.022..0.034 rows=30 loops=1)
    -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=0.35 rows=1) (actual time=0.050..0.050 rows=1 loops=5)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

- CREATE INDEX Correct_Course_Id ON DailyClass(CorrectCourseId);



```
mysql> EXPLAIN ANALYZE SELECT User.UserId, User.Email, Courses.Name AS CorrectCourseName
    -> FROM Guess
    -> JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
    -> JOIN Courses ON DailyClass.CorrectCourseId = Courses.CourseId
    -> JOIN User ON Guess.UserId = User.UserId
    -> WHERE Guess.CourseId = DailyClass.CorrectCourseId
    -> AND Guess.CurrentDate = "2023-11-14";
+------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+
| EXPLAIN



                                                                                           |
+------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+
| -> Nested loop inner join  (cost=0.93 rows=0.5) (actual time=0.311..0.504 rows=5 loops=1)
    -> Filter: ((Guess.CurrentDate = DATE'2023-11-14') and (Guess.CourseId = '39804') and (Guess.UserId is not null))  (cost=0.63 rows=0.5) (actual time=0.221..0.248 rows=5 loops=
1)
        -> Intersect rows sorted by row ID  (cost=0.63 rows=1) (actual time=0.210..0.236 rows=5 loops=1)
            -> Index range scan on Guess using Course_Id over (CourseId = 39804)  (cost=0.96 rows=11) (actual time=0.140..0.144 rows=9 loops=1)
            -> Index range scan on Guess using CurrentDate over (CurrentDate = '2023-11-14')  (cost=0.26 rows=30) (actual time=0.022..0.034 rows=30 loops=1)
    -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=0.35 rows=1) (actual time=0.050..0.050 rows=1 loops=5)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

- CREATE INDEX User_Id ON Guess(UserId);



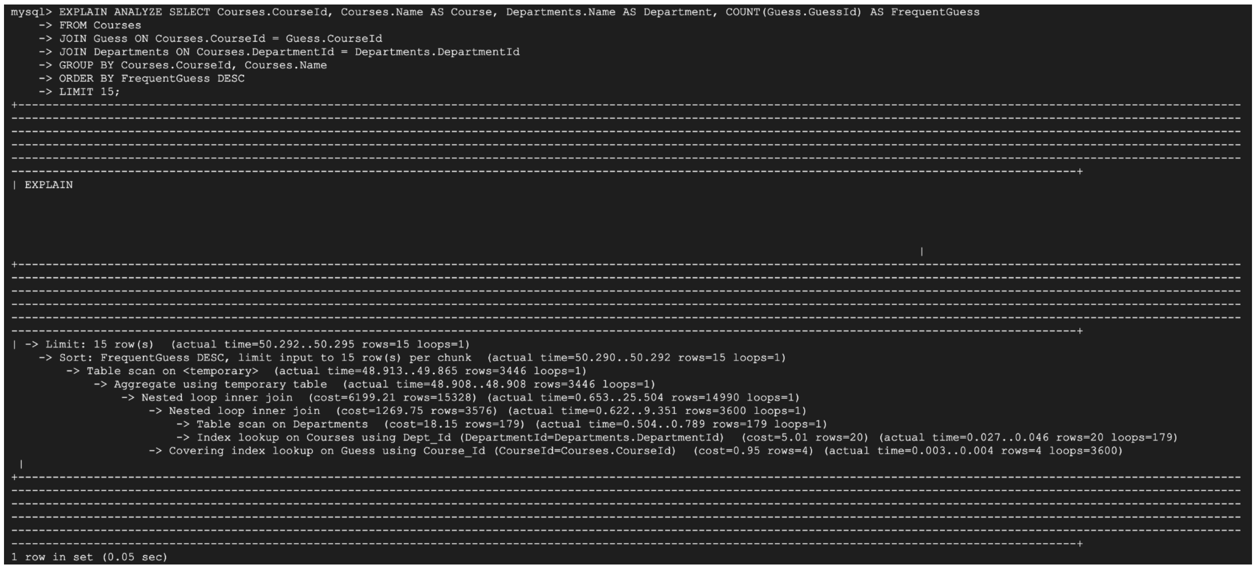
```
mysql> EXPLAIN ANALYZE SELECT User.UserId, User.Email, Courses.Name AS CorrectCourseName
    -> FROM Guess
    -> JOIN DailyClass ON Guess.CurrentDate = DailyClass.CurrentDate
    -> JOIN Courses ON DailyClass.CorrectCourseId = Courses.CourseId
    -> JOIN User ON Guess.UserId = User.UserId
    -> WHERE Guess.CourseId = DailyClass.CorrectCourseId
    -> AND Guess.CurrentDate = "2023-11-14";
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------+
| EXPLAIN



                                                                                                     |
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------+
| -> Nested loop inner join  (cost=0.93 rows=0.5) (actual time=0.311..0.504 rows=5 loops=1)
    -> Filter: ((Guess.CurrentDate = DATE'2023-11-14') and (Guess.CourseId = '39804') and (Guess.UserId is not null))  (cost=0.63 rows=0.5) (actual time=0.221..0.248 rows=5 loops=
1)
        -> Intersect rows sorted by row ID  (cost=0.63 rows=1) (actual time=0.210..0.236 rows=5 loops=1)
            -> Index range scan on Guess using Course_Id over (CourseId = 39804)  (cost=0.96 rows=11) (actual time=0.140..0.144 rows=9 loops=1)
            -> Index range scan on Guess using CurrentDate over (CurrentDate = '2023-11-14')  (cost=0.26 rows=30) (actual time=0.022..0.034 rows=30 loops=1)
    -> Single-row index lookup on User using PRIMARY (UserId=Guess.UserId)  (cost=0.35 rows=1) (actual time=0.050..0.050 rows=1 loops=5)
 |
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Again no performance improvement. We maintain that this is because the fields used in the query were already indexed by MySQL. No index we add could change how the query performs.

<u>Query 4 for Attempted Indexing Designs:</u>

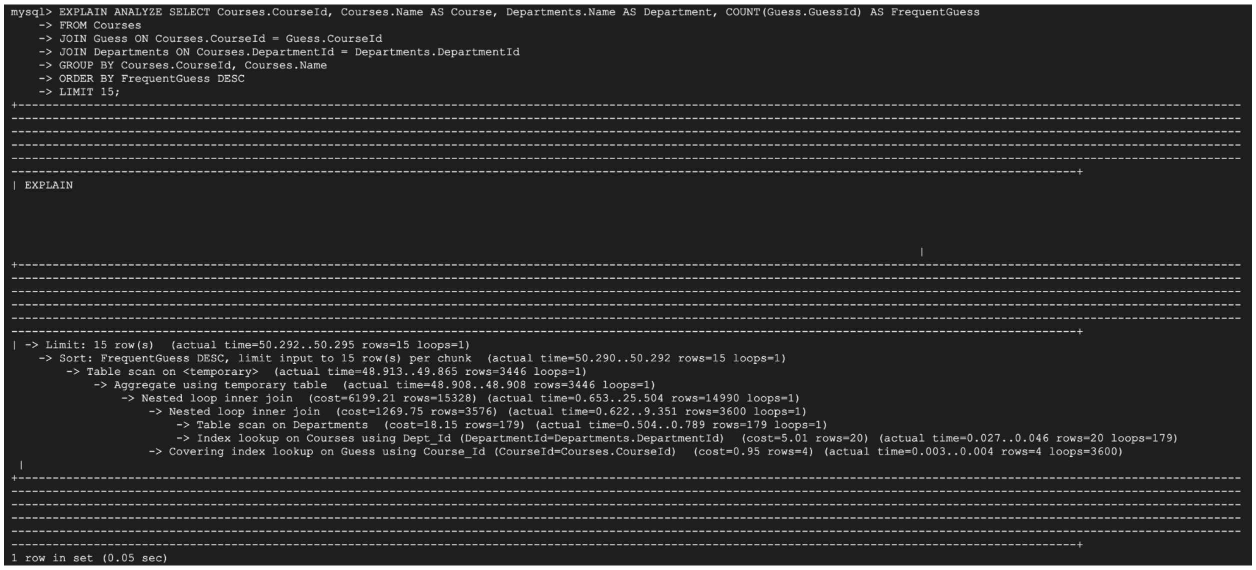
```
EXPLAIN ANALYZE SELECT Courses.CourseId, Courses.Name AS Course,
Departments.Name AS Department, COUNT(Guess.GuessId) AS FrequentGuess
FROM Courses
JOIN Guess ON Courses.CourseId = Guess.CourseId
JOIN Departments ON Courses.DepartmentId = Departments.DepartmentId
GROUP BY Courses.CourseId, Courses.Name
ORDER BY FrequentGuess DESC
LIMIT 15;
```

- `CREATE INDEX Depart_Id ON Courses(DepartmentId);`



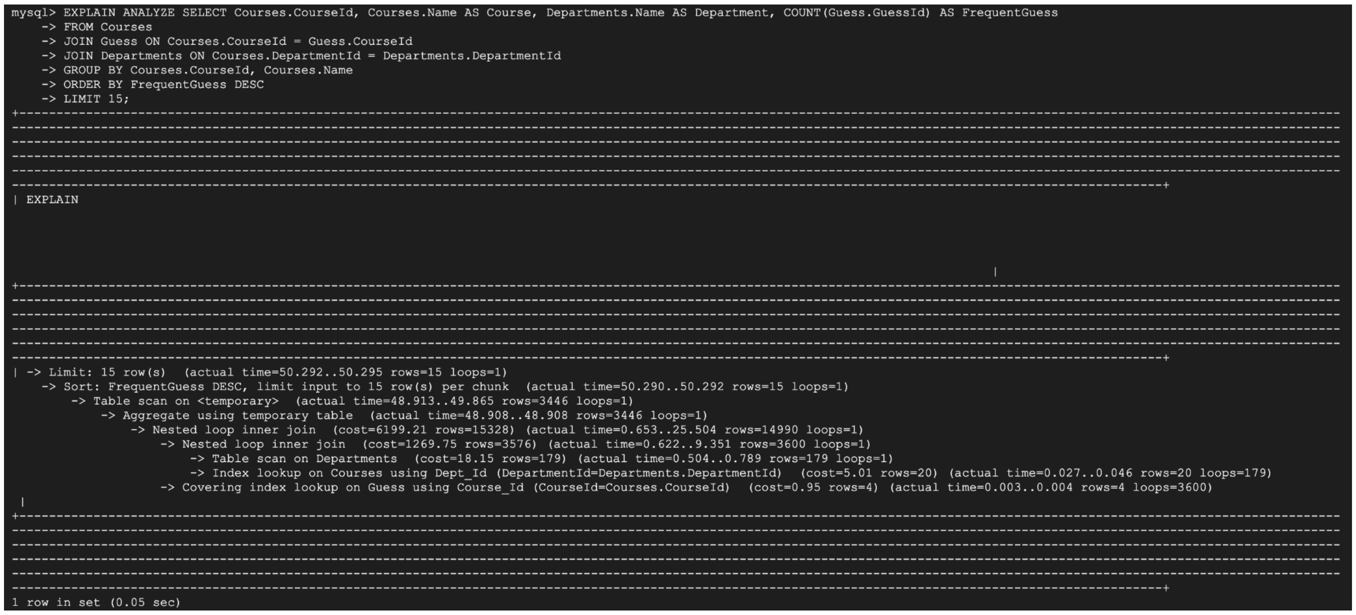
```
mysql> EXPLAIN ANALYZE SELECT Courses.CourseId, Courses.Name AS Course, Departments.Name AS Department, COUNT(Guess.GuessId) AS FrequentGuess
    -> FROM Courses
    -> JOIN Guess ON Courses.CourseId = Guess.CourseId
    -> JOIN Departments ON Courses.DepartmentId = Departments.DepartmentId
    -> GROUP BY Courses.CourseId, Courses.Name
    -> ORDER BY FrequentGuess DESC
    -> LIMIT 15;
+------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN



                                                                                                                            |
+------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=50.292..50.295 rows=15 loops=1)
    -> Sort: FrequentGuess DESC, limit input to 15 row(s) per chunk  (actual time=50.290..50.292 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=48.913..49.865 rows=3446 loops=1)
            -> Aggregate using temporary table  (actual time=48.908..48.908 rows=3446 loops=1)
                -> Nested loop inner join  (cost=6199.21 rows=15328) (actual time=0.653..25.504 rows=14990 loops=1)
                    -> Nested loop inner join  (cost=1269.75 rows=3576) (actual time=0.622..9.351 rows=3600 loops=1)
                        -> Table scan on Departments  (cost=18.15 rows=179) (actual time=0.504..0.789 rows=179 loops=1)
                        -> Index lookup on Courses using Dept_Id (DepartmentId=Departments.DepartmentId)  (cost=5.01 rows=20) (actual time=0.027..0.046 rows=20 loops=179)
                    -> Covering index lookup on Guess using Course_Id (CourseId=Courses.CourseId)  (cost=0.95 rows=4) (actual time=0.003..0.004 rows=4 loops=3600)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.05 sec)
```

- `CREATE INDEX Course_Id ON Guess(CourseId);`



```
mysql> EXPLAIN ANALYZE SELECT Courses.CourseId, Courses.Name AS Course, Departments.Name AS Department, COUNT(Guess.GuessId) AS FrequentGuess
    -> FROM Courses
    -> JOIN Guess ON Courses.CourseId = Guess.CourseId
    -> JOIN Departments ON Courses.DepartmentId = Departments.DepartmentId
    -> GROUP BY Courses.CourseId, Courses.Name
    -> ORDER BY FrequentGuess DESC
    -> LIMIT 15;
+------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN



                                                                                                                            |
+------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=50.292..50.295 rows=15 loops=1)
    -> Sort: FrequentGuess DESC, limit input to 15 row(s) per chunk  (actual time=50.290..50.292 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=48.913..49.865 rows=3446 loops=1)
            -> Aggregate using temporary table  (actual time=48.908..48.908 rows=3446 loops=1)
                -> Nested loop inner join  (cost=6199.21 rows=15328) (actual time=0.653..25.504 rows=14990 loops=1)
                    -> Nested loop inner join  (cost=1269.75 rows=3576) (actual time=0.622..9.351 rows=3600 loops=1)
                        -> Table scan on Departments  (cost=18.15 rows=179) (actual time=0.504..0.789 rows=179 loops=1)
                        -> Index lookup on Courses using Dept_Id (DepartmentId=Departments.DepartmentId)  (cost=5.01 rows=20) (actual time=0.027..0.046 rows=20 loops=179)
                    -> Covering index lookup on Guess using Course_Id (CourseId=Courses.CourseId)  (cost=0.95 rows=4) (actual time=0.003..0.004 rows=4 loops=3600)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.05 sec)
```

- `CREATE INDEX Name ON Courses(Name);`



```
mysql> EXPLAIN ANALYZE SELECT Courses.CourseId, Courses.Name AS Course, Departments.Name AS Department, COUNT(Guess.GuessId) AS FrequentGuess
    -> FROM Courses
    -> JOIN Guess ON Courses.CourseId = Guess.CourseId
    -> JOIN Departments ON Courses.DepartmentId = Departments.DepartmentId
    -> GROUP BY Courses.CourseId, Courses.Name
    -> ORDER BY FrequentGuess DESC
    -> LIMIT 15;
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN




                                                                                                                                     |
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=50.292..50.295 rows=15 loops=1)
    -> Sort: FrequentGuess DESC, limit input to 15 row(s) per chunk  (actual time=50.290..50.292 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=48.913..49.865 rows=3446 loops=1)
            -> Aggregate using temporary table  (actual time=48.908..48.908 rows=3446 loops=1)
                -> Nested loop inner join  (cost=6199.21 rows=15328) (actual time=0.653..25.504 rows=14990 loops=1)
                    -> Nested loop inner join  (cost=1269.75 rows=3576) (actual time=0.622..9.351 rows=3600 loops=1)
                        -> Table scan on Departments  (cost=18.15 rows=179) (actual time=0.504..0.789 rows=179 loops=1)
                        -> Index lookup on Courses using Dept_Id (DepartmentId=Departments.DepartmentId)  (cost=5.01 rows=20) (actual time=0.027..0.046 rows=20 loops=179)
                    -> Covering index lookup on Guess using Course_Id (CourseId=Courses.CourseId)  (cost=0.95 rows=4) (actual time=0.003..0.004 rows=4 loops=3600)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.05 sec)
```

Indexing did not improve the performance of this query. None of our added indexing changed the costs associated with the query.

Overall Analysis of Indexing:

We did not notice a difference in our results or cost when indexing for the four queries. There are likely several reasons why the change in indexing did not bring a better effect on our queries' performances. Most likely, the indexing was redundant due to MySQL's automatic indexing. Our dataset also might have low-selectivity where scanning or filtering would be faster than accessing the index. This means variables like CurrentDate may have had many repeated values, leading to limited performance gains. Additionally, more complex queries such as these (with nested joins) can render indexing to be ineffective. Overall, for an index to improve query performance it should match the query structure and be used on high-selectivity columns. Creating queries that use different attributes could avoid the automatic indexing issue, however this would not improve overall performance.