DDL Commands

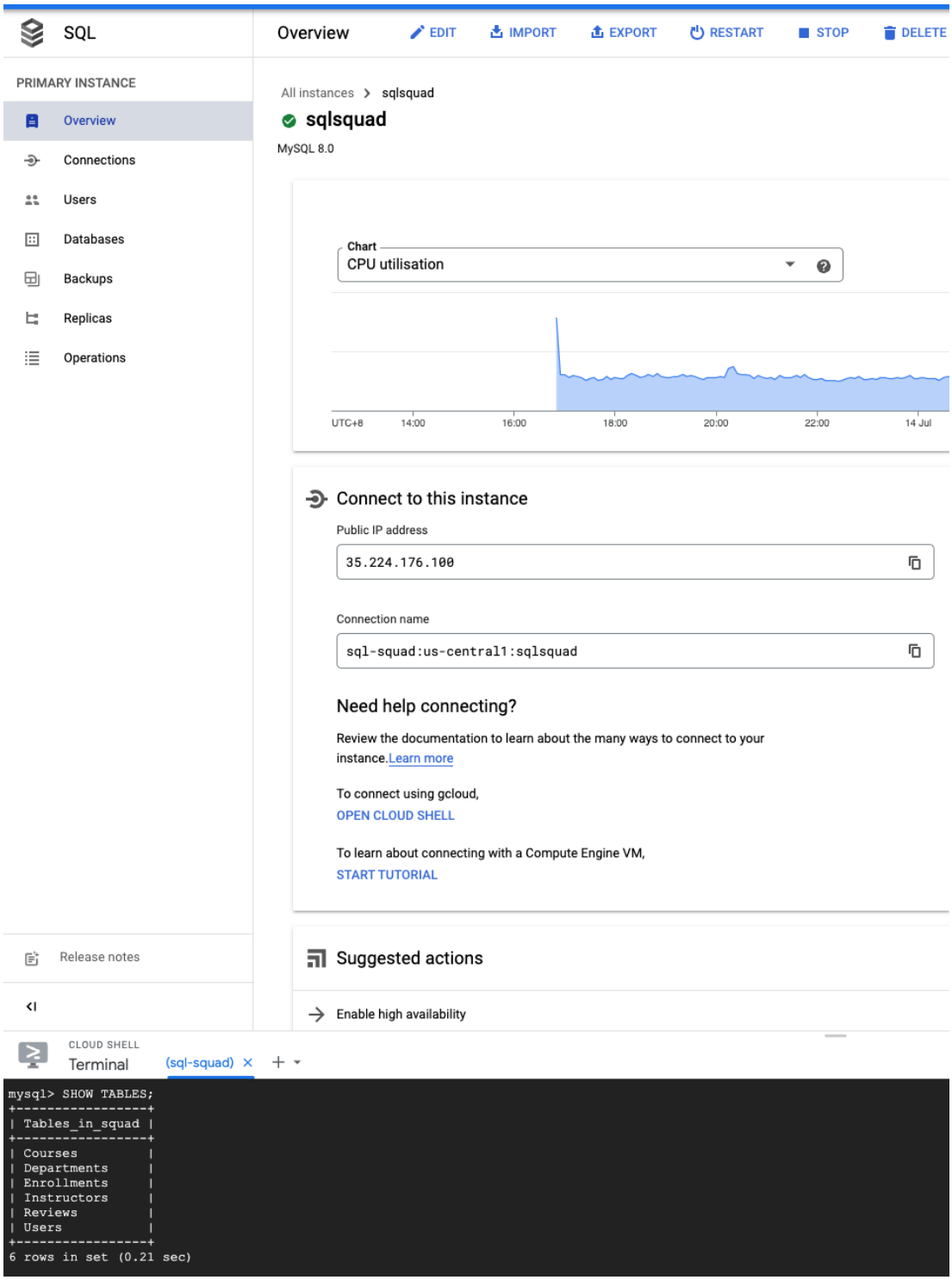
```
CREATE TABLE Users(
        Username VARCHAR(100),
        NetID VARCHAR(50),
        Password VARCHAR(100),
        IsGuest BOOLEAN,

        PRIMARY KEY(Username));

CREATE TABLE Reviews(
        ReviewID VARCHAR(50),
        Rating REAL,
        Comment VARCHAR(500),
        IsRecommended BOOLEAN,
        RequiresTextbook BOOLEAN,
        Upvotes INT,
        Downvotes INT,
        UserNetID VARCHAR(50),
        CRN VARCHAR(6),
        InstructorNetID VARCHAR(50),

        PRIMARY KEY(ReviewID),
        FOREIGN KEY(UserNetID) REFERENCES Users(NetID),
        FOREIGN KEY(InstructorNetID) REFERENCES Instructors(NetID),
        FOREIGN KEY(CRN) REFERENCES Courses(CRN));

CREATE TABLE Courses(
        CRN VARCHAR(6),
        CourseName VARCHAR(50),
        CourseNumber VARCHAR(3),
        Description VARCHAR(500),
        DeptAbv VARCHAR(4),

        PRIMARY KEY(CRN),
        FOREIGN KEY(DeptAbv) REFERENCES Departments(DeptAbv));

CREATE TABLE Instructors(
        NetID VARCHAR(50),
        Name VARCHAR(100),
        DeptAbv VARCHAR(4),

        PRIMARY KEY (NetID),
```

```
        FOREIGN KEY (DeptAbv) REFERENCES Departments(DeptAbv));

CREATE TABLE Departments(
        DeptAbv VARCHAR(4),
        DeptName VARCHAR(50),
        Location VARCHAR(100),
        URL VARCHAR(50),
        PhoneNumber VARCHAR(20),

        PRIMARY KEY (DeptAbv));
```

Relationship Tables (for applicable relationships):

```
CREATE TABLE Enrolled(
        UserNetID VARCHAR(50),
        CRN VARCHAR(6),
        InstructorNetID VARCHAR(50),
        YearTerm VARCHAR(10),
        GPA REAL,

        PRIMARY KEY(UserNetID, CRN, InstructorNetID, YearTerm),
        FOREIGN KEY(UserNetID) REFERENCES Users(NetID),
        FOREIGN KEY(CRN) REFERENCES Courses(CRN),
        FOREIGN KEY(InstructorNetID) REFERENCES Instructors(NetID));
```

Proof of Implementation / Terminal Connection on GCP:



Proof of at least 1000 entries in 3 tables:

```
mysql> SELECT COUNT(*) FROM Courses;
+----------+
| COUNT(*) |
+----------+
|    12212 |
+----------+
1 row in set (0.24 sec)
```

```
mysql> SELECT COUNT(*) FROM Reviews;
+----------+
| COUNT(*) |
+----------+
|     1000 |
+----------+
1 row in set (0.22 sec)
```

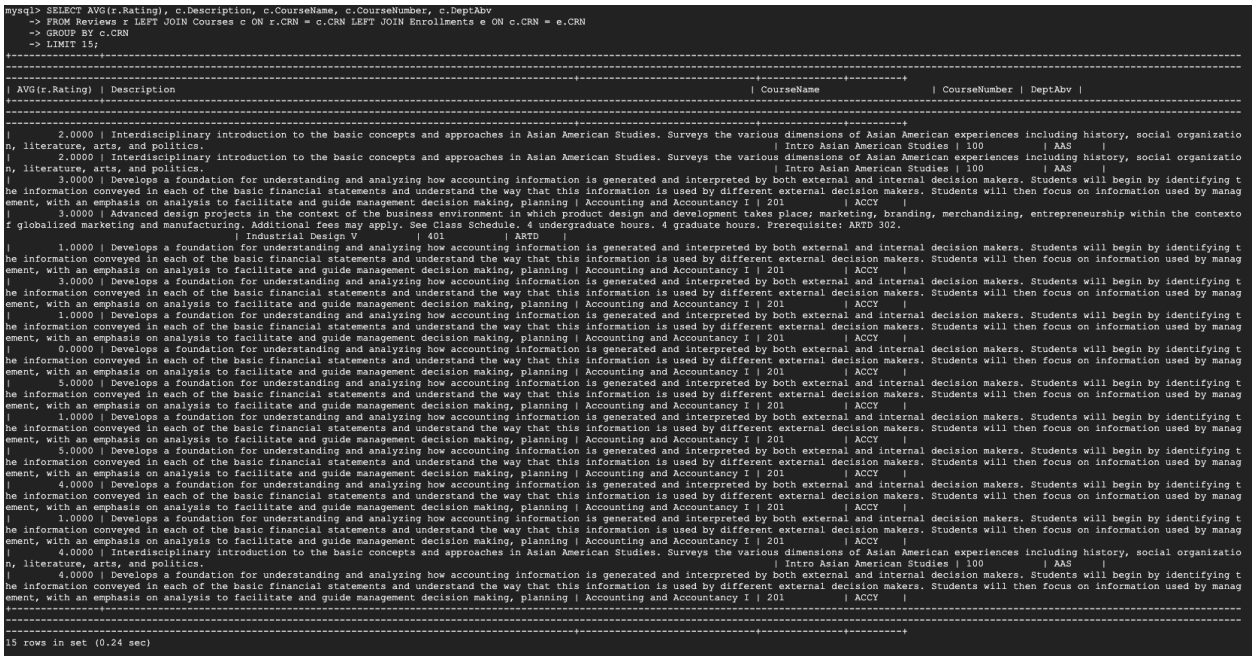
```
mysql> SELECT COUNT(*) FROM Instructors;
+----------+
| COUNT(*) |
+----------+
|     3512 |
+----------+
1 row in set (0.21 sec)
```

2 Different Advanced SQL Queries:

Query 1: Search for Course: Average Rating, and relevant Course Details

purpose: This allows users to see the general rating and relevant details for courses, specific to the CRN, that they may want to take. When sorted, this allows the user to see courses with the highest ratings in UIUC.

```
SELECT AVG(r.Rating), c.Description, c.CourseName, c.CourseNumber, c.DeptAbv
FROM Reviews r LEFT JOIN Courses c ON r.CRN = c.CRN LEFT JOIN Enrollments e ON
c.CRN = e.CRN
GROUP BY c.CRN
LIMIT 15;
```



```
mysql> SELECT AVG(r.Rating), c.Description, c.CourseName, c.CourseNumber, c.DeptAbv
    -> FROM Reviews r LEFT JOIN Courses c ON r.CRN = c.CRN LEFT JOIN Enrollments e ON c.CRN = e.CRN
    -> GROUP BY c.CRN
    -> LIMIT 15;
+-------------+-------------+------------+--------------+---------+
| AVG(r.Rating) | Description | CourseName | CourseNumber | DeptAbv |
+-------------+-------------+------------+--------------+---------+
|        2.0000 | Interdisciplinary introduction to the basic concepts and approaches in Asian American Studies. Surveys the various dimensions of Asian American experiences including history, social organizatio
n, literature, arts, and politics.                                                                                                                                                                    | Intro Asian American Studies | 100          | AAS     |
|        2.0000 | Interdisciplinary introduction to the basic concepts and approaches in Asian American Studies. Surveys the various dimensions of Asian American experiences including history, social organizatio
n, literature, arts, and politics.                                                                                                                                                                    | Intro Asian American Studies | 100          | AAS     |
|        3.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        3.0000 | Advanced design projects in the context of the business environment in which product design and development takes place; marketing, branding, merchandizing, entrepreneurship within the contexto
f globalized marketing and manufacturing. Additional fees may apply. See Class Schedule. 4 undergraduate hours. 4 graduate hours. Prerequisite: ARTD 302.                                                | Industrial Design V          | 401          | ARTD    |
|        1.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        3.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        1.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        0.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        5.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        1.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        5.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        4.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        1.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
|        4.0000 | Interdisciplinary introduction to the basic concepts and approaches in Asian American Studies. Surveys the various dimensions of Asian American experiences including history, social organizatio
n, literature, arts, and politics.                                                                                                                                                                    | Intro Asian American Studies | 100          | AAS     |
|        4.0000 | Develops a foundation for understanding and analyzing how accounting information is generated and interpreted by both external and internal decision makers. Students will begin by identifying t
he information conveyed in each of the basic financial statements and understand the way that this information is used by different external decision makers. Students will then focus on information used by manag
ement, with an emphasis on analysis to facilitate and guide management decision making, planning | Accounting and Accountancy I | 201          | ACCY    |
+-------------+-------------+------------+--------------+---------+
15 rows in set (0.24 sec)
```

Query 2: Instructor with the Highest Rating grouped by DeptAbv and CourseNumber

Purpose: This query allows users to filter for instructors that have the highest ratings for the courses that they might potentially want to take, in order to make a better choice when picking instructors. This is specific to the Department Abbreviation, for example ‘CS’, and the Course Number, so it finds the professor or professors with the highest average rating for ‘CS100’ for example.

SELECT DISTINCT i.Name, s2.DeptAbv, s2.CourseNumber, s2.maxRating
FROM (SELECT MAX(s1.avgRating) as maxRating, s1.DeptAbv, s1.CourseNumber
FROM (SELECT AVG(r.Rating) as avgRating, c.DeptAbv, c.CourseNumber, r.InstructorNetID
FROM Reviews r LEFT JOIN Courses c ON r.CRN = c.CRN GROUP BY c.DeptAbv,
c.CourseNumber, r.InstructorNetID) AS s1 GROUP BY s1.DeptAbv, s1.CourseNumber) AS s2
LEFT JOIN (SELECT AVG(r2.Rating) as avgRating, c2.DeptAbv, c2.CourseNumber,
r2.InstructorNetID FROM Reviews r2 LEFT JOIN Courses c2 ON r2.CRN = c2.CRN GROUP BY
c2.DeptAbv, c2.CourseNumber, r2.InstructorNetID) AS s3 ON s2.maxRating = s3.avgRating
AND s2.DeptAbv = s3.DeptAbv AND s2.CourseNumber = s3.CourseNumber LEFT JOIN
Instructors i ON s3.InstructorNetID = i.NetID
LIMIT 15;

```
mysql> SELECT DISTINCT i.Name, s2.DeptAbv, s2.CourseNumber, s2.maxRating
    -> FROM (SELECT MAX(s1.avgRating) as maxRating, s1.DeptAbv, s1.CourseNumber
    -> FROM (SELECT AVG(r.Rating) as avgRating, c.DeptAbv, c.CourseNumber, r.InstructorNetID FROM Reviews r LEFT JOIN Courses c ON r.CRN = c.CRN GROUP BY c.DeptAbv, c.CourseNumber, r.InstructorNetID) AS s1 GROUP
 BY s1.DeptAbv, s1.CourseNumber) AS s2 LEFT JOIN (SELECT AVG(r2.Rating) as avgRating, c2.DeptAbv, c2.CourseNumber, r2.InstructorNetID FROM Reviews r2 LEFT JOIN Courses c2 ON r2.CRN = c2.CRN GROUP BY c2.DeptAbv,
c2.CourseNumber, r2.InstructorNetID) AS s3 ON s2.maxRating = s3.avgRating AND s2.DeptAbv = s3.DeptAbv AND s2.CourseNumber = s3.CourseNumber LEFT JOIN Instructors i ON s3.InstructorNetID = i.NetID
    -> LIMIT 15;
+---------------+---------+-------------+-----------+
| Name          | DeptAbv | CourseNumber | maxRating |
+---------------+---------+-------------+-----------+
| Kwon, S       | AAS     | 100          |    4.0000 |
| Chen, Y       | ACCY    | 201          |    2.5000 |
| Sethi, S      | ARTD    | 401          |    5.0000 |
| Avery, M      | ARTD    | 401          |    5.0000 |
| Minor, J      | ACCY    | 202          |    4.5000 |
| Sharif, L     | AAS     | 105          |    1.0000 |
| Shapland, J   | ACCY    | 290          |    2.0000 |
| Davis, G      | ACCY    | 301          |    2.7500 |
| Tabares, L    | AAS     | 120          |    1.0000 |
| Curtis, S     | ACCY    | 302          |    3.6000 |
| Tabares, L    | AAS     | 200          |    2.0000 |
| Hamm, B       | ACCY    | 303          |    2.0000 |
| Pryor, M      | AAS     | 201          |    5.0000 |
| Moreira, J    | AAS     | 201          |    5.0000 |
| Williamson, M | ACCY    | 304          |    3.0000 |
+---------------+---------+-------------+-----------+
15 rows in set (0.23 sec)
```

Indexing Analysis

Query 1

Query Details:

```
SELECT AVG(r.Rating), c.Description, c.CourseName, c.CourseNumber,
c.DeptAbv
FROM Reviews r LEFT JOIN Courses c ON r.CRN = c.CRN LEFT JOIN Enrollments e
ON c.CRN = e.CRN
GROUP BY c.CRN
LIMIT 15;
```

EXPLAIN ANALYZE results:

No indexes:

| EXPLAIN
| -> Limit: 15 row(s) (actual time=8.779..8.783 rows=15 loops=1)
    -> Table scan on <temporary> (actual time=0.002..0.005 rows=15 loops=1)
        -> Aggregate using temporary table (actual time=8.777..8.781 rows=15 loops=1)
            -> Nested loop left join (cost=801.75 rows=1000) (actual time=0.071..5.774 rows=1177 loops=1)
                -> Nested loop left join (cost=451.75 rows=1000) (actual time=0.058..2.536 rows=1000 loops=1)
                    -> Table scan on r (cost=101.75 rows=1000) (actual time=0.040..0.355 rows=1000 loops=1)
                    -> Single-row index lookup on c using PRIMARY (CRN=r.CRN) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)

```
            -> Index lookup on e using CRN (CRN=c.CRN)  (cost=0.25 rows=1) (actual
time=0.002..0.003 rows=1 loops=1000)
 |
1 row in set (0.21 sec)
```

3 Different Indexing Designs:

Indexing on c.DeptAbv results:

```
| EXPLAIN
| -> Limit: 15 row(s)  (actual time=9.361..9.366 rows=15 loops=1)
    -> Table scan on <temporary>  (actual time=0.002..0.005 rows=15 loops=1)
       -> Aggregate using temporary table  (actual time=9.360..9.364 rows=15 loops=1)
          -> Nested loop left join  (cost=825.37 rows=1185) (actual time=0.071..6.242 rows=1177
loops=1)
             -> Nested loop left join  (cost=451.75 rows=1000) (actual time=0.061..2.926
rows=1000 loops=1)
                -> Table scan on r  (cost=101.75 rows=1000) (actual time=0.044..0.375 rows=1000
loops=1)
                -> Single-row index lookup on c using PRIMARY (CRN=r.CRN)  (cost=0.25 rows=1)
(actual time=0.002..0.002 rows=1 loops=1000)
             -> Index lookup on e using CRN (CRN=c.CRN)  (cost=0.26 rows=1) (actual
time=0.002..0.003 rows=1 loops=1000)
 |
1 row in set (0.22 sec) (this is poorer in performance compared to the raw one, 0.22 > 0.21)
```

Indexing on c.CourseNumber:

```
| EXPLAIN
| -> Limit: 15 row(s)  (actual time=9.357..9.362 rows=15 loops=1)
    -> Table scan on <temporary>  (actual time=0.002..0.005 rows=15 loops=1)
       -> Aggregate using temporary table  (actual time=9.357..9.360 rows=15 loops=1)
          -> Nested loop left join  (cost=825.37 rows=1185) (actual time=0.105..6.218 rows=1177
loops=1)
             -> Nested loop left join  (cost=451.75 rows=1000) (actual time=0.075..3.035
rows=1000 loops=1)
                -> Table scan on r  (cost=101.75 rows=1000) (actual time=0.053..0.419 rows=1000
loops=1)
                -> Single-row index lookup on c using PRIMARY (CRN=r.CRN)  (cost=0.25 rows=1)
(actual time=0.002..0.002 rows=1 loops=1000)
             -> Index lookup on e using CRN (CRN=c.CRN)  (cost=0.26 rows=1) (actual
time=0.002..0.003 rows=1 loops=1000)
 |
1 row in set (0.22 sec) (this is poorer in performance compared to the raw one, 0.22 > 0.21)
```

Indexing on r.Rating:

```
| EXPLAIN
| -> Limit: 15 row(s)  (actual time=8.617..8.621 rows=15 loops=1)
    -> Table scan on <temporary>  (actual time=0.002..0.004 rows=15 loops=1)
        -> Aggregate using temporary table  (actual time=8.616..8.619 rows=15 loops=1)
            -> Nested loop left join  (cost=825.37 rows=1185) (actual time=0.060..5.598 rows=1177
loops=1)
                -> Nested loop left join  (cost=451.75 rows=1000) (actual time=0.051..2.767
rows=1000 loops=1)
                    -> Table scan on r  (cost=101.75 rows=1000) (actual time=0.034..0.370 rows=1000
loops=1)
                    -> Single-row index lookup on c using PRIMARY (CRN=r.CRN)  (cost=0.25 rows=1)
(actual time=0.002..0.002 rows=1 loops=1000)
                -> Index lookup on e using CRN (CRN=c.CRN)  (cost=0.26 rows=1) (actual
time=0.002..0.003 rows=1 loops=1000)
|
1 row in set (0.22 sec)
```

Conclusion

As we can tell from the above EXPLAIN ANALYZE results, different indexing designs did nothing to speed up the queries compared to the base case of no indexes. It is unlikely that indexing will help here since we are finding the average rating for each separate CRN, hence we are going to go through each and every row. So indexes do not speed up this query at all.

Query 2

Query Details:

```
SELECT DISTINCT i.Name, s2.DeptAbv, s2.CourseNumber, s2.maxRating
FROM (SELECT MAX(s1.avgRating) as maxRating, s1.DeptAbv, s1.CourseNumber
FROM (SELECT AVG(r.Rating) as avgRating, c.DeptAbv, c.CourseNumber, r.InstructorNetID
FROM Reviews r LEFT JOIN Courses c ON r.CRN = c.CRN GROUP BY c.DeptAbv,
c.CourseNumber, r.InstructorNetID) AS s1 GROUP BY s1.DeptAbv, s1.CourseNumber) AS s2
LEFT JOIN (SELECT AVG(r2.Rating) as avgRating, c2.DeptAbv, c2.CourseNumber,
r2.InstructorNetID FROM Reviews r2 LEFT JOIN Courses c2 ON r2.CRN = c2.CRN GROUP BY
c2.DeptAbv, c2.CourseNumber, r2.InstructorNetID) AS s3 ON s2.maxRating = s3.avgRating
AND s2.DeptAbv = s3.DeptAbv AND s2.CourseNumber = s3.CourseNumber LEFT JOIN
Instructors i ON s3.InstructorNetID = i.NetID
LIMIT 15;
```

EXPLAIN ANALYZE Results:

No Indexes:

```
| EXPLAIN
| -> Limit: 15 row(s)  (cost=5346.01..5346.01 rows=0) (actual time=11.744..11.747 rows=15 loops=1)
    -> Table scan on <temporary>  (cost=2.50..2.50 rows=0) (actual time=0.000..0.002 rows=15 loops=1)
        -> Temporary table with deduplication  (cost=5348.51..5348.51 rows=0) (actual time=11.742..11.744 rows=15 loops=1)
            -> Limit table size: 15 unique row(s)
                -> Nested loop left join  (cost=5346.01 rows=0) (actual time=11.645..11.709 rows=15 loops=1)
                    -> Nested loop left join  (cost=2674.26 rows=0) (actual time=11.631..11.657 rows=15 loops=1)
                        -> Table scan on s2  (cost=2.50..2.50 rows=0) (actual time=0.001..0.003 rows=13 loops=1)
                        -> Materialize  (cost=2.50..2.50 rows=0) (actual time=6.001..6.005 rows=383 loops=1)
                            -> Table scan on <temporary>  (actual time=0.000..0.026 rows=383 loops=1)
                                -> Aggregate using temporary table  (actual time=5.851..5.899 rows=383 loops=1)
                                    -> Table scan on s1  (cost=2.50..2.50 rows=0) (actual time=0.001..0.043 rows=575 loops=1)
                                        -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5.190..5.275 rows=575 loops=1)
                                            -> Table scan on <temporary>  (actual time=0.001..0.052 rows=575 loops=1)
                                                -> Aggregate using temporary table  (actual time=4.804..4.890 rows=575 loops=1)
                                                    -> Nested loop left join  (cost=451.75 rows=1000) (actual time=0.079..3.261 rows=1000 loops=1)
                                                        -> Table scan on r  (cost=101.75 rows=1000) (actual time=0.057..0.426 rows=1000 loops=1)
                                                        -> Single-row index lookup on c using PRIMARY (CRN=r.CRN)  (cost=0.25 rows=1) (actual time=0.003..0.003 rows=1 loops=1000)
                    -> Index lookup on s3 using <auto_key1> (avgRating=s2.maxRating, DeptAbv=s2.DeptAbv, CourseNumber=s2.CourseNumber)  (actual time=0.001..0.002 rows=1 loops=13)
                -> Materialize  (cost=0.00..0.00 rows=0) (actual time=5.644..5.649 rows=575 loops=1)
                    -> Table scan on <temporary>  (actual time=0.001..0.086 rows=575 loops=1)
```

-> Aggregate using temporary table (actual time=4.627..4.763 rows=575 loops=1)
-> Nested loop left join (cost=451.75 rows=1000) (actual time=0.068..2.726 rows=1000 loops=1)
-> Table scan on r2 (cost=101.75 rows=1000) (actual time=0.058..0.431 rows=1000 loops=1)
-> Single-row index lookup on c2 using PRIMARY (CRN=r2.CRN) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
-> Single-row index lookup on i using PRIMARY (NetID=s3.InstructorNetID) (cost=0.25 rows=1) (actual time=0.003..0.003 rows=1 loops=15)
|
1 row in set **(0.22 sec)**

3 Different Indexing Designs:

Indexing on Reviews(Rating):

| EXPLAIN
| -> Limit: 15 row(s) (cost=5346.01..5346.01 rows=0) (actual time=10.007..10.010 rows=15 loops=1)
-> Table scan on <temporary> (cost=2.50..2.50 rows=0) (actual time=0.000..0.002 rows=15 loops=1)
-> Temporary table with deduplication (cost=5348.51..5348.51 rows=0) (actual time=10.006..10.008 rows=15 loops=1)
-> Limit table size: 15 unique row(s)
-> Nested loop left join (cost=5346.01 rows=0) (actual time=9.921..9.975 rows=15 loops=1)
-> Nested loop left join (cost=2674.26 rows=0) (actual time=9.910..9.937 rows=15 loops=1)
-> Table scan on s2 (cost=2.50..2.50 rows=0) (actual time=0.000..0.002 rows=13 loops=1)
-> Materialize (cost=2.50..2.50 rows=0) (actual time=5.338..5.341 rows=383 loops=1)
-> Table scan on <temporary> (actual time=0.000..0.026 rows=383 loops=1)
-> Aggregate using temporary table (actual time=5.189..5.236 rows=383 loops=1)
-> Table scan on s1 (cost=2.50..2.50 rows=0) (actual time=0.001..0.046 rows=575 loops=1)
-> Materialize (cost=2.50..2.50 rows=0) (actual time=4.563..4.641 rows=575 loops=1)
-> Table scan on <temporary> (actual time=0.001..0.051 rows=575 loops=1)

-> Aggregate using temporary table (actual time=4.158..4.241 rows=575 loops=1)
-> Nested loop left join (cost=451.75 rows=1000) (actual time=0.076..2.654 rows=1000 loops=1)
-> Table scan on r (cost=101.75 rows=1000) (actual time=0.054..0.404 rows=1000 loops=1)
-> Single-row index lookup on c using PRIMARY (CRN=r.CRN) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
-> Index lookup on s3 using <auto_key1> (avgRating=s2.maxRating, DeptAbv=s2.DeptAbv, CourseNumber=s2.CourseNumber) (actual time=0.002..0.002 rows=1 loops=13)
-> Materialize (cost=0.00..0.00 rows=0) (actual time=4.587..4.592 rows=575 loops=1)
-> Table scan on <temporary> (actual time=0.001..0.045 rows=575 loops=1)
-> Aggregate using temporary table (actual time=3.747..3.824 rows=575 loops=1)
-> Nested loop left join (cost=451.75 rows=1000) (actual time=0.045..2.279 rows=1000 loops=1)
-> Table scan on r2 (cost=101.75 rows=1000) (actual time=0.034..0.373 rows=1000 loops=1)
-> Single-row index lookup on c2 using PRIMARY (CRN=r2.CRN) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
-> Single-row index lookup on i using PRIMARY (NetID=s3.InstructorNetID) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=15)
|
1 row in set **(0.22 sec)**

Indexing on Instructors(NetID)

| EXPLAIN
| -> Limit: 15 row(s) (cost=5346.01..5346.01 rows=0) (actual time=10.903..10.905 rows=15 loops=1)
-> Table scan on <temporary> (cost=2.50..2.50 rows=0) (actual time=0.001..0.002 rows=15 loops=1)
-> Temporary table with deduplication (cost=5348.51..5348.51 rows=0) (actual time=10.902..10.904 rows=15 loops=1)
-> Limit table size: 15 unique row(s)
-> Nested loop left join (cost=5346.01 rows=0) (actual time=10.806..10.866 rows=15 loops=1)
-> Nested loop left join (cost=2674.26 rows=0) (actual time=10.793..10.826 rows=15 loops=1)
-> Table scan on s2 (cost=2.50..2.50 rows=0) (actual time=0.000..0.003 rows=13 loops=1)

```
                    -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5.457..5.460 rows=383
loops=1)
                        -> Table scan on <temporary>  (actual time=0.000..0.024 rows=383
loops=1)
                            -> Aggregate using temporary table  (actual time=5.296..5.343 rows=383
loops=1)
                                -> Table scan on s1  (cost=2.50..2.50 rows=0) (actual
time=0.001..0.041 rows=575 loops=1)
                                    -> Materialize  (cost=2.50..2.50 rows=0) (actual time=4.647..4.725
rows=575 loops=1)
                                        -> Table scan on <temporary>  (actual time=0.001..0.048
rows=575 loops=1)
                                            -> Aggregate using temporary table  (actual time=4.247..4.326
rows=575 loops=1)
                                                -> Nested loop left join  (cost=451.75 rows=1000) (actual
time=0.064..2.702 rows=1000 loops=1)
                                                    -> Table scan on r  (cost=101.75 rows=1000) (actual
time=0.043..0.391 rows=1000 loops=1)
                                                    -> Single-row index lookup on c using PRIMARY
(CRN=r.CRN)  (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
                    -> Index lookup on s3 using <auto_key1> (avgRating=s2.maxRating,
DeptAbv=s2.DeptAbv, CourseNumber=s2.CourseNumber)  (actual time=0.002..0.002 rows=1
loops=13)
                    -> Materialize  (cost=0.00..0.00 rows=0) (actual time=5.358..5.362 rows=575
loops=1)
                        -> Table scan on <temporary>  (actual time=0.001..0.055 rows=575
loops=1)
                            -> Aggregate using temporary table  (actual time=4.477..4.580 rows=575
loops=1)
                                -> Nested loop left join  (cost=451.75 rows=1000) (actual
time=0.641..2.958 rows=1000 loops=1)
                                    -> Table scan on r2  (cost=101.75 rows=1000) (actual
time=0.621..0.965 rows=1000 loops=1)
                                    -> Single-row index lookup on c2 using PRIMARY (CRN=r2.CRN)
(cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
                -> Single-row index lookup on i using PRIMARY (NetID=s3.InstructorNetID)
(cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=15)

1 row in set (0.22 sec)
```

Indexing on Courses(DeptAbv) and Courses(CourseNumber):

```
| EXPLAIN
```

```
| -> Limit: 15 row(s)  (cost=5346.01..5346.01 rows=0) (actual time=9.767..9.770 rows=15
loops=1)
    -> Table scan on <temporary>  (cost=2.50..2.50 rows=0) (actual time=0.001..0.002 rows=15
loops=1)
        -> Temporary table with deduplication  (cost=5348.51..5348.51 rows=0) (actual
time=9.766..9.768 rows=15 loops=1)
            -> Limit table size: 15 unique row(s)
                -> Nested loop left join  (cost=5346.01 rows=0) (actual time=9.679..9.734 rows=15
loops=1)
                    -> Nested loop left join  (cost=2674.26 rows=0) (actual time=9.668..9.695 rows=15
loops=1)
                        -> Table scan on s2  (cost=2.50..2.50 rows=0) (actual time=0.000..0.002
rows=13 loops=1)
                        -> Materialize  (cost=2.50..2.50 rows=0) (actual time=5.134..5.137 rows=383
loops=1)
                            -> Table scan on <temporary>  (actual time=0.000..0.024 rows=383
loops=1)
                                -> Aggregate using temporary table  (actual time=4.985..5.031 rows=383
loops=1)
                                    -> Table scan on s1  (cost=2.50..2.50 rows=0) (actual
time=0.001..0.042 rows=575 loops=1)
                                    -> Materialize  (cost=2.50..2.50 rows=0) (actual time=4.357..4.431
rows=575 loops=1)
                                        -> Table scan on <temporary>  (actual time=0.001..0.046
rows=575 loops=1)
                                            -> Aggregate using temporary table  (actual time=3.975..4.054
rows=575 loops=1)
                                                -> Nested loop left join  (cost=451.75 rows=1000) (actual
time=0.065..2.505 rows=1000 loops=1)
                                                    -> Table scan on r  (cost=101.75 rows=1000) (actual
time=0.045..0.382 rows=1000 loops=1)
                                                    -> Single-row index lookup on c using PRIMARY
(CRN=r.CRN)  (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
                    -> Index lookup on s3 using <auto_key1> (avgRating=s2.maxRating,
DeptAbv=s2.DeptAbv, CourseNumber=s2.CourseNumber)  (actual time=0.002..0.002 rows=1
loops=13)
                        -> Materialize  (cost=0.00..0.00 rows=0) (actual time=4.549..4.554 rows=575
loops=1)
                            -> Table scan on <temporary>  (actual time=0.001..0.048 rows=575
loops=1)
                                -> Aggregate using temporary table  (actual time=3.711..3.791 rows=575
loops=1)
                                    -> Nested loop left join  (cost=451.75 rows=1000) (actual
time=0.048..2.250 rows=1000 loops=1)
```

-> Table scan on r2 (cost=101.75 rows=1000) (actual time=0.039..0.380 rows=1000 loops=1)
-> Single-row index lookup on c2 using PRIMARY (CRN=r2.CRN) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
-> Single-row index lookup on i using PRIMARY (NetID=s3.InstructorNetID) (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=15)
|
1 row in set **(0.22 sec)**

Conclusion:
As we can tell from the results above, the performance of the query does not change even with the different indexes. Trying out different indexing designs has had no effect on the speed of the query. We think that this is because, like the above query, the query needs to calculate the average rating of each course, which means that each and every tuple in the Reviews relation will need to be processed, hence indexing does not speed up the query in any way, since there isn't a more efficient way to identify which row to start at, since every row needs to be processed.