MGIS 725
Group Project Draft
Student Evaluation System
11/27/2022

Group Members:
Sanjeev Hirudayaraj, Sai Karan Uppugalla, Sachi Nutulapati, Param Anand

Preface

This project aims to create a student evaluation system using Oracle SQL to create the data structures and Oracle Apex to create applications to promote ease of access. The primary and bridge tables are created using SQL queries with their respective primary and functional keys. The application is made so that the student can input their evaluation, which then leads the administrators, instructors, and students to the ratings of professors. The information retrieved from these forms would enable administrators to appraise the professor's salary while allowing the administrator to improve their performance. The application system would assist the students in enrolling in their desired courses based on the ratings.

This document goes through the ER-diagram used to convey the overall structure of the database, followed by the SQL queries used to create the database and then the screenshots of the application pages created in Oracle Apex along with the SQL queries.

Table Names

Application Name - NAUH_Student_Evaluation_System

Evaluation_NAUH;

Course_Student_NAUH;

Course_Instructor_NAUH;

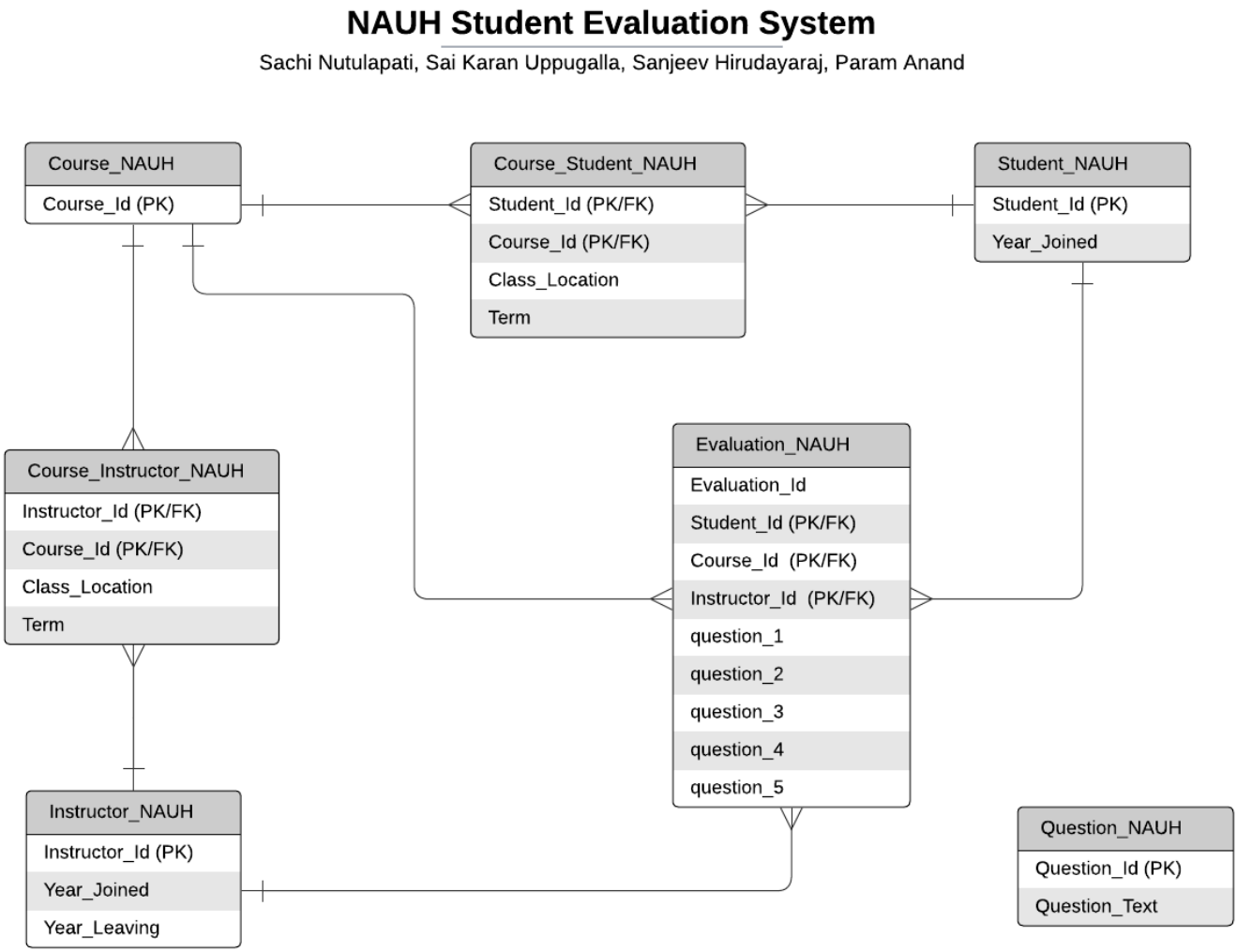
Student_NAUH;

Course_NAUH;

Instructor_NAUH;

Question_NAU

ER Diagram


NAUH Student Evaluation System
Sachi Nutulapati, Sai Karan Uppugalla, Sanjeev Hirudayaraj, Param Anand
Course_NAUH
Course_Id (PK)
Course_Student_NAUH
Student_Id (PK/FK)
Course_Id (PK/FK)
Class_Location
Term
Student_NAUH
Student_Id (PK)
Year_Joined
Course_Instructor_NAUH
Instructor_Id (PK/FK)
Course_Id (PK/FK)
Class_Location
Term
Evaluation_NAUH
Evaluation_Id
Student_Id (PK/FK)
Course_Id (PK/FK)
Instructor_Id (PK/FK)
question_1
question_2
question_3
question_4
question_5
Instructor_NAUH
Instructor_Id (PK)
Year_Joined
Year_Leaving
Question_NAUH
Question_Id (PK)
Question_Text

DDL - SQL Script

```
-- SELECT * STATEMENTS
SELECT * FROM Evaluation_NAUH
SELECT * FROM Student_NAUH
SELECT * FROM Course_NAUH
SELECT * FROM Instructor_NAUH
SELECT * FROM Course_Student_NAUH
SELECT * FROM Course_Instructor_NAUH

-- DROP STATEMENTS
DROP TABLE Evaluation_NAUH;
DROP TABLE Course_Student_NAUH;
DROP TABLE Course_Instructor_NAUH;
DROP TABLE Student_NAUH;
DROP TABLE Course_NAUH;
DROP TABLE Instructor_NAUH;

-- CREATE STATEMENTS
CREATE TABLE Evaluation_NAUH (
  evaluation_id NUMBER GENERATED BY DEFAULT AS IDENTITY,
  course_id VARCHAR(7) NOT NULL,
  student_id NUMBER(9,0) NOT NULL,
  instructor_id NUMBER(5,0) NOT NULL,
  question_1 NUMBER(1,0) NOT NULL,
  question_2 NUMBER(1,0) NOT NULL,
  question_3 NUMBER(1,0) NOT NULL,
  question_4 NUMBER(1,0) NOT NULL,
  question_5 NUMBER(1,0) NOT NULL,
  CONSTRAINT PK_Evaluation_NAUH PRIMARY KEY (evaluation_id),
  CONSTRAINT fk_student_id_bridge FOREIGN KEY (student_id) REFERENCES
Student_NAUH (student_id),
  CONSTRAINT fk_course_id_bridge FOREIGN KEY (course_id) REFERENCES
Course_NAUH (course_id),
  CONSTRAINT fk_instructor_id_bridge FOREIGN KEY (instructor_id) REFERENCES
Instructor_NAUH (instructor_id)
);


CREATE TABLE Course_Instructor_NAUH (
  course_id VARCHAR(7) NOT NULL,
  instructor_id NUMBER(5,0) NOT NULL,
  term VARCHAR(10),
  class_location VARCHAR(30),
  CONSTRAINT PK_Course_Instructor_NAUH PRIMARY KEY (course_id, instructor_id),
  CONSTRAINT fk_instructor_course_id FOREIGN KEY (instructor_id) REFERENCES
Instructor_NAUH (instructor_id),
```

```
  CONSTRAINT fk_course_instructor_id FOREIGN KEY (course_id) REFERENCES
Course_NAUH (course_id)
);

CREATE TABLE Course_Student_NAUH (
  course_id VARCHAR(7) NOT NULL,
  student_id NUMBER(9,0) NOT NULL,
  term VARCHAR(10),
  class_location VARCHAR(30),
  CONSTRAINT PK_Course_Student_NAUH PRIMARY KEY (course_id, student_id),
  CONSTRAINT fk_student_id FOREIGN KEY (student_id) REFERENCES Student_NAUH
(student_id),
  CONSTRAINT fk_course_id FOREIGN KEY (course_id) REFERENCES Course_NAUH
(course_id)
);

CREATE TABLE Student_NAUH (
  student_id NUMBER(9,0) NOT NULL,
  date_joined DATE,
  CONSTRAINT PK_Student_NAUH PRIMARY KEY (student_id)
);

CREATE TABLE Question_NAUH (
  question_id NUMBER(1,0) NOT NULL,
  question_text VARCHAR (100) NOT NULL,
  CONSTRAINT PK_Student_NAUH PRIMARY KEY (question_id)
);

CREATE TABLE Course_NAUH (
  course_id VARCHAR(7) NOT NULL,
  CONSTRAINT PK_Course_NAUH PRIMARY KEY (course_id)
);

CREATE TABLE Instructor_NAUH (
  instructor_id NUMBER(5,0) NOT NULL,
  date_joined DATE,
  date_leaving DATE,
  CONSTRAINT PK_Instructor_NAUH PRIMARY KEY (instructor_id)
);
-- INSERT STATEMENTS
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000001, '10-01-
2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000002, '10-02-
2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000003, '10-03-
2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000004, '10-23-
2002');
```

```
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000005, '10-24-2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000006, '09-15-2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000007, '09-16-2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000008, '09-17-2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000009, '09-20-2002');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000010, '01-11-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000011, '01-13-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000012, '01-14-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000013, '01-16-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000014, '08-15-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000015, '08-12-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000016, '08-19-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000017, '08-21-2003');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000018, '01-01-2004');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000019, '01-26-2004');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000020, '08-09-2004');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000021, '08-10-2004');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000022, '01-10-2005');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000023, '01-18-2005');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000024, '01-19-2006');
INSERT INTO Student_NAUH (student_id, date_joined) VALUES (100000025, '08-10-2006');

INSERT INTO Question_NAUH (question_id, question_text) VALUES (1, ‘Q1. The instructor enhanced my interest in this Course’);
INSERT INTO Question_NAUH (question_id, question_text) VALUES (2, ‘Q2. The instructor presented the course material in an organized manner.‘);
INSERT INTO Question_NAUH (question_id, question_text) VALUES (3, ‘Q3. The instructor communicated the course material clearly.‘);
```

```
INSERT INTO Question_NAUH (question_id, question_text) VALUES (4, ‘Q4. The
instructor established a positive learning environment.’);
INSERT INTO Question_NAUH (question_id, question_text) VALUES (5,
Q5. The instructor provided helpful feedback about my work in this course.‘);

INSERT INTO Question_NAUH (cours
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('BANA680', 100000024, '2021', 'LBR');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('BANA680', 100000022, '2021', 'LBR');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('BANA680', 100000021, '2021', 'LBR');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('STAT745', 100000001, '2021', 'GNT');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('STAT745', 100000002, '2021', 'GNT');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('STAT745', 100000007, '2021', 'GNT');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('STAT745', 100000004, '2021', 'GNT');
INSERT INTO Course_Student_NAUH (course_id, student_id, term, class_location)
VALUES ('STAT745', 100000011, '2021', 'GNT');

INSERT INTO Instructor_NAUH (instructor_id, date_joined, date_leaving) VALUES (10001,
'09-08-2000', '12-31-2030');
INSERT INTO Instructor_NAUH (instructor_id, date_joined, date_leaving) VALUES (10002,
'05-12-2000', '12-31-2030');
INSERT INTO Instructor_NAUH (instructor_id, date_joined, date_leaving) VALUES (10003,
'04-08-2001', '12-31-2031');
INSERT INTO Instructor_NAUH (instructor_id, date_joined, date_leaving) VALUES (10004,
'09-19-2001', '12-31-2031');
INSERT INTO Instructor_NAUH (instructor_id, date_joined, date_leaving) VALUES (10005,
'09-21-2001', '12-31-2031');


INSERT INTO Course_NAUH (course_id) VALUES ('MKTG728');
INSERT INTO Course_NAUH (course_id) VALUES ('SWEN610');
INSERT INTO Course_NAUH (course_id) VALUES ('MGIS725');
INSERT INTO Course_NAUH (course_id) VALUES ('BANA680');
INSERT INTO Course_NAUH (course_id) VALUES ('STAT745');
INSERT INTO Course_NAUH (course_id) VALUES ('ISTE782');
INSERT INTO Course_NAUH (course_id) VALUES ('ISEE750');
INSERT INTO Course_NAUH (course_id) VALUES ('DSCI633');
INSERT INTO Course_NAUH (course_id) VALUES ('MATH252');

INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MKTG728',          10001,2021,  'SAU');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('SWEN610',          10002,2021,  'GNT');
```

```
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MGIS725', 10004,2021,  'WLC');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('BANA680',10001,2021,  'LBR');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MGIS650', 10005,2021,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('STAT745', 10003,2021,  'WLC');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('ISTE782',  10004,2021,  'LBR');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('ISEE750',  10002,2021,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('DSCI633', 10003,2021,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MATH252',          10005,2021,  'SAU');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MKTG728',          10001,2022,  'SAU');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('SWEN610',          10002,2022,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MGIS725', 10004,2022,  'WLC');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('BANA680',10001,2022,  'LBR');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MGIS650', 10005,2022,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('STAT745', 10003,2022,  'WLC');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('ISTE782',  10004,2022,  'LBR');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('ISEE750',  10002,2022,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('DSCI633', 10003,2022,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MATH252',          10005,2022,  'SAU');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MKTG728',          10001,2023,  'SAU');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('SWEN610',          10002,2023,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MGIS725', 10004,2023,  'WLC');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('BANA680',10001,2023,  'LBR');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MGIS650', 10005,2023,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('STAT745', 10003,2023,  'WLC');
```

```
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('ISTE782',  10004,2023,  'LBR');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('ISEE750',  10002,2023,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('DSCI633', 10003,2023,  'GNT');
INSERT INTO Course_Instructor_NAUH (course_id,instructor_id,term,class_location)
VALUES('MATH252',          10005,2023,  'SAU');
```

DML and Screenshots of Application

Evaluation Form

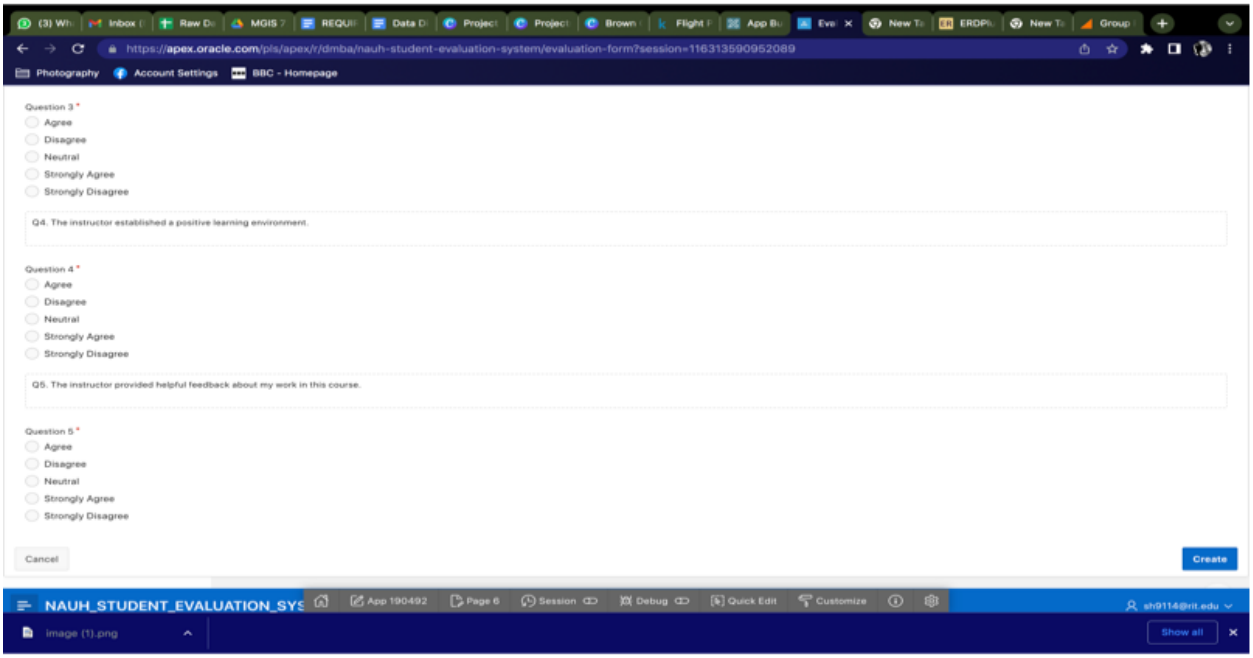


Figure.1 Evaluation Form

Description

The above figure 1 is the first page of our application which is the evaluation containing the course id, student id, instructor_id, and the set of 5 questions. A student can submit one evaluation for one course which satisfies the criteria defined in the ER diagram - a course can be taken by many students and a student can take many courses or more than one course.

Course Offerings

Description:

The figure below shows the details which helps students, administrators, and instructors view a report on which courses are offered every year as well as review at least the last year and upcoming year schedules. The application was created using the Course_Instructor_NAUH table.

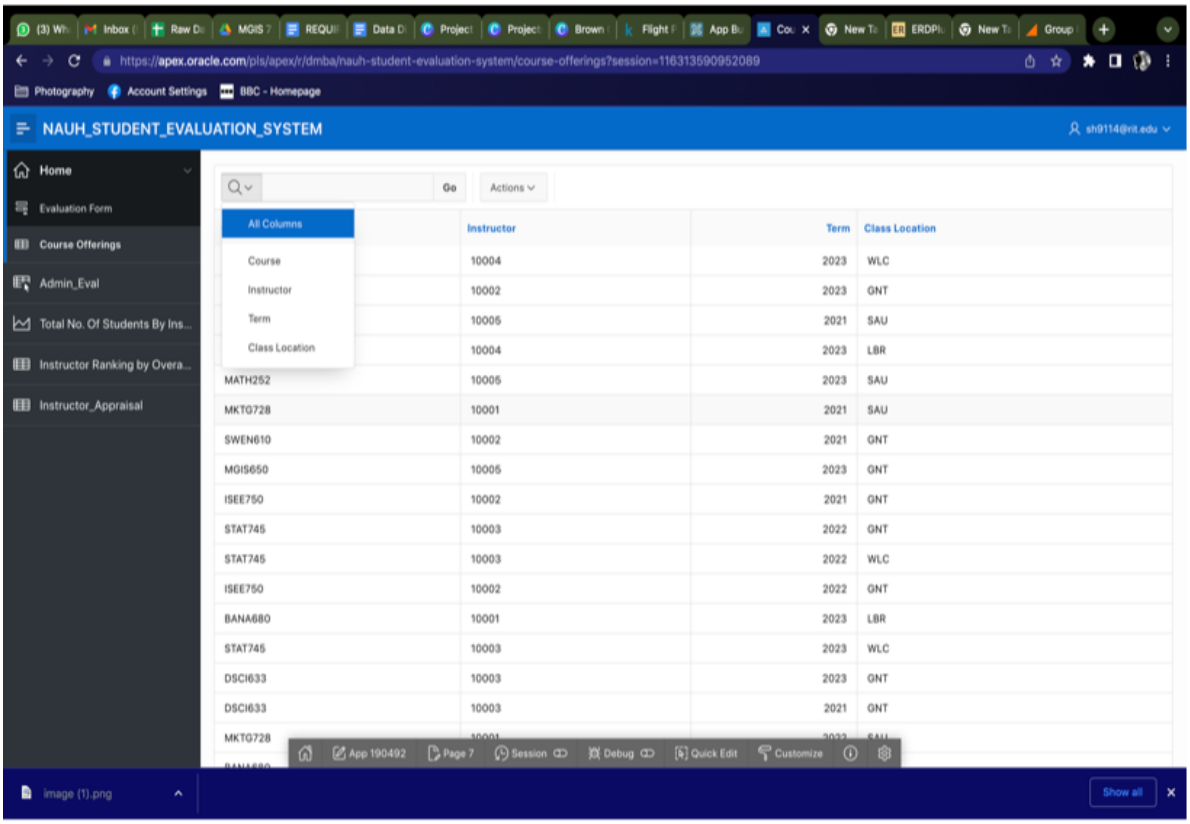


Figure 2. Course Offerings View

Admin Evaluation From

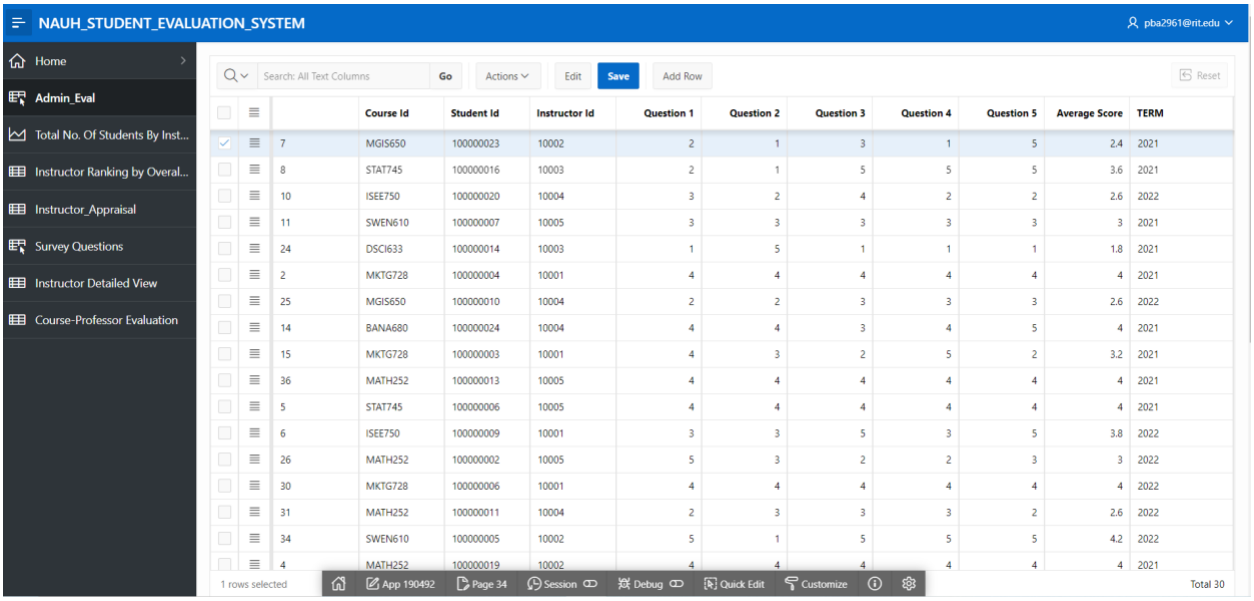
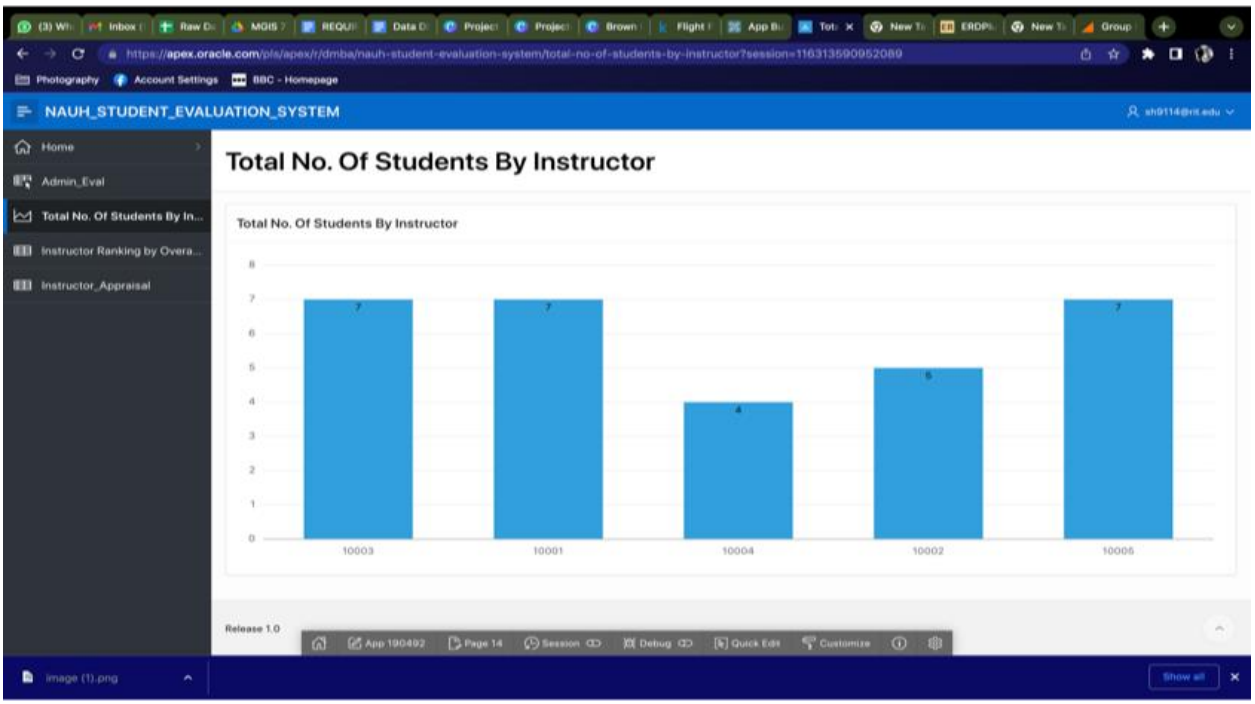


Figure 3. Admin Evaluation Form

Description

The admin evaluation form above is based on the Evaluation_NAUH table from our database. The Admin_Evaluation Report will be populated once the students fill out the evaluation form. This table is the bridge table which includes the evaluation id, course id, instructor id, the score for each question, and the average score. The average score is a derived variable of all the five questions. The report also has the year the evaluation

was completed. In addition to this later on we have to add another column that is



“average score” and so we had use the following SQL query:

ALTER TABLE Evaluation_NAUH add Average_Score number(10,2);

UPDATE Evaluation_NAUH e

SET e.Average_Score = (SELECT
(QUESTION_1+QUESTION_2+QUESTION_3+QUESTION_4+QUESTION_5)/5

FROM Evaluation_NAUH

WHERE EVALUATION_ID = t.EVALUATION_ID);

Total Number of Students by Instructor

Description: The figure above shows the bar chart application for the total students taught by each instructor the application was created using the following query:

Query:

```
SELECT instructor_id, COUNT( student_id)

FROM Evaluation_NAUH

GROUP BY instructor_id;
```

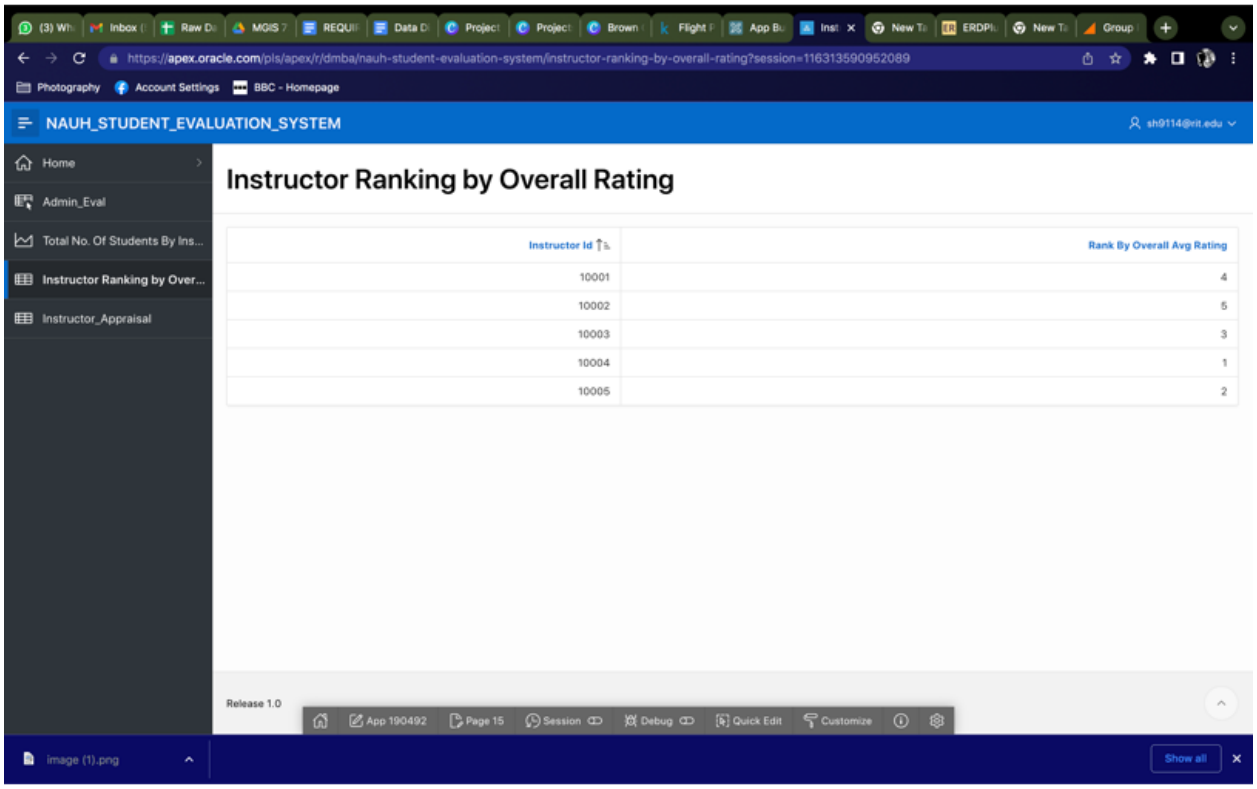
Instructor Ranking by Overall Rating

This view shows the ranking of the overall rating of each instructor for all 5 questions. The application was made using the following query:

Query:

```
SELECT instructor_id, RANK() OVER (ORDER BY AVG(question_1) +
AVG(question_2) + AVG(question_3) + AVG(question_4) + AVG(question_5) )as "Rank
by Overall AVG Rating"

FROM Evaluation_NAUH
```

GROUP BY instructor_id;



Instructor Appraisal

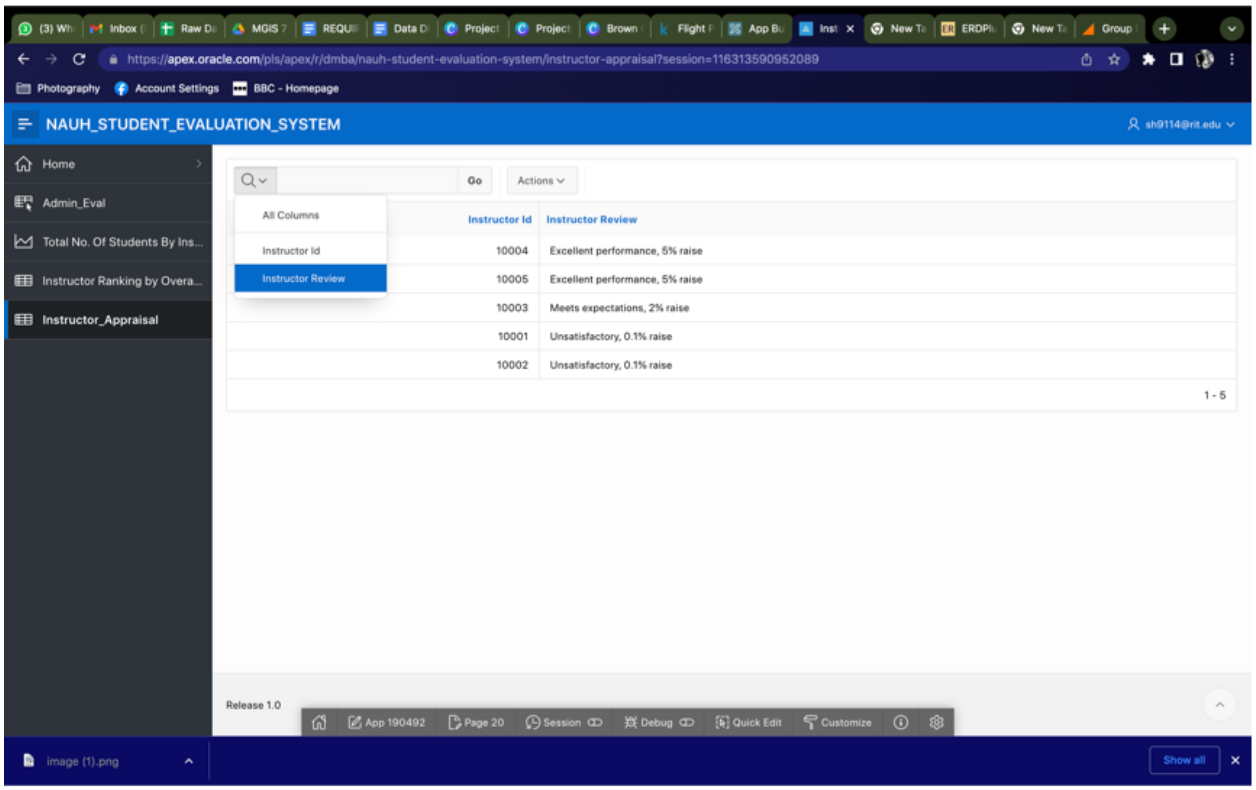
This view shows the appraisal that the instructors will be given based on their ranking in terms of the overall evaluation score average they received.

Query:

SELECT instructor_id,

CASE

```
WHEN RANK() OVER (ORDER BY AVG(question_1) + AVG(question_2) +
AVG(question_3) + AVG(question_4) + AVG(question_5)) <= 2 THEN 'Excellent
performance, 5% raise'

WHEN RANK() OVER (ORDER BY AVG(question_1) + AVG(question_2) +
AVG(question_3) + AVG(question_4) + AVG(question_5)) = 3 THEN  'Meets
expectations, 2% raise'

ELSE 'Unsatisfactory, 0.1% raise'

END

AS Instructor_Review

FROM Evaluation_NAUH

GROUP BY instructor_id;
```

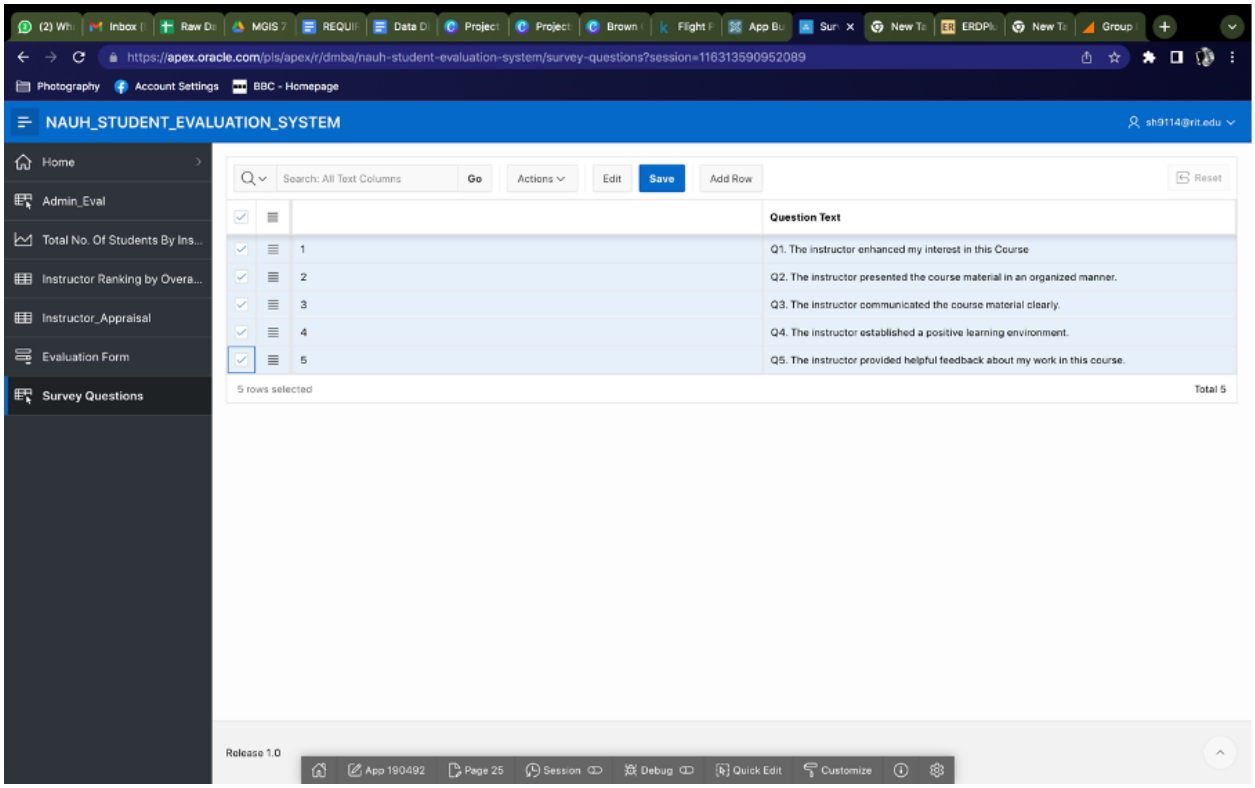


Survey Questions

The administrators need to have the ability to alter the five questions in the evaluation form of the application. The Question_NAUH table stores the question num and the question text. To dynamically populate the question in the form, we used the following query in the UI Editor:

Query:

```
SELECT question_text

FROM Question_NAUH

WHERE question_id = 1
```



Instructor Detail View

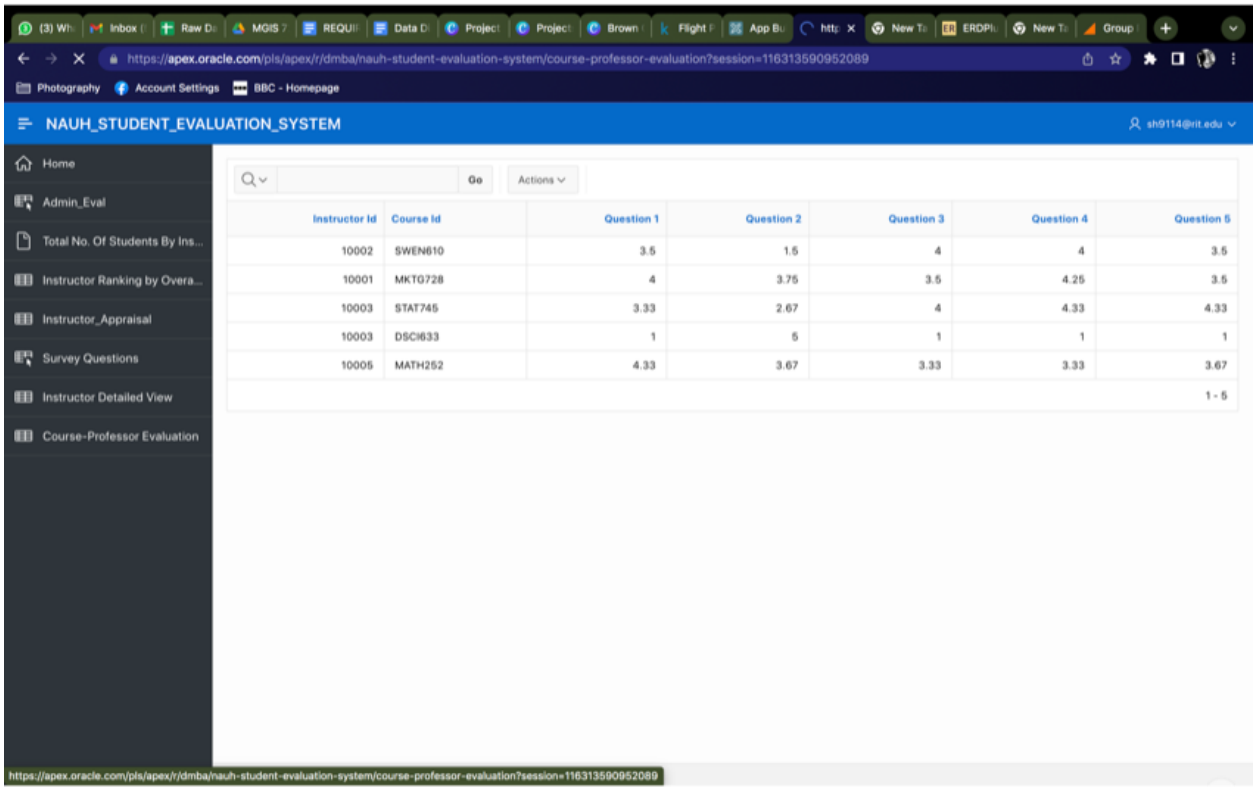
The application below is to show the instructor how they are rated across all the questions through the average score for each question.

Query:

SELECT INSTRUCTOR_ID,

ROUND(AVG(question_1),2), ROUND(AVG(question_2),2),
ROUND(AVG(question_3),2), ROUND(AVG(question_4),2),
ROUND(AVG(question_5),2)

FROM Evaluation_NAUH GROUP BY INSTRUCTOR_ID;

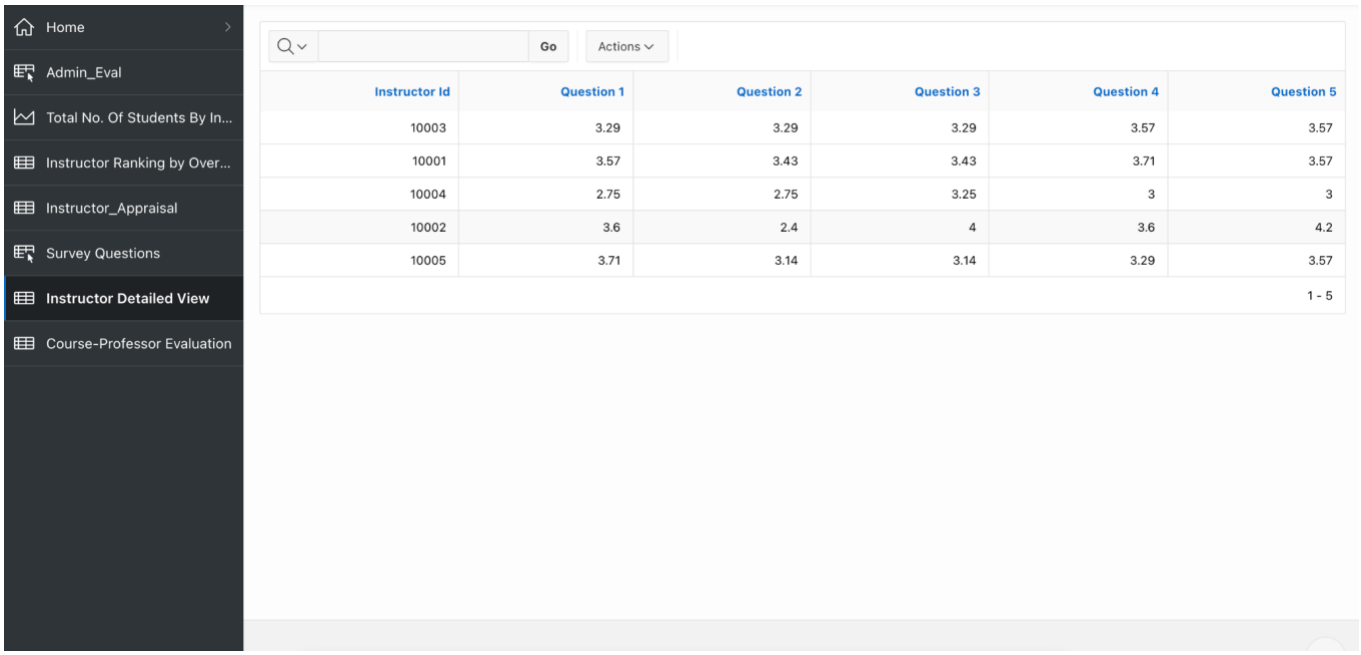


Course Professor Evaluation

This page will allow an administrator to compare ratings of one instructor to another instructor who have both taught the same course. If there are more than two instructors who have taught the same course, then the average scores across all five questions may also be compared.

Query:

```
SELECT Course_Instructor_NAUH.instructor_id,Course_Instructor_NAUH.course_id,
ROUND(AVG(question_1),2), ROUND(AVG(question_2),2),
ROUND(AVG(question_3),2), ROUND(AVG(question_4),2),
ROUND(AVG(question_5),2)

FROM Evaluation_NAUH

JOIN Course_Instructor_NAUH ON Course_Instructor_NAUH.instructor_id =
Evaluation_NAUH.instructor_id

WHERE Course_Instructor_NAUH.course_id = Evaluation_NAUH.course_id

GROUP BY Course_Instructor_NAUH.instructor_id,Course_Instructor_NAUH.course_id
```


Home
Admin_Eval
Total No. Of Students By In...
Instructor Ranking by Over...
Instructor_Appraisal
Survey Questions
Instructor Detailed View
Course-Professor Evaluation
Go
Actions
Instructor Id
Question 1
Question 2
Question 3
Question 4
Question 5
10003 3.29 3.29 3.29 3.57 3.57
10001 3.57 3.43 3.43 3.71 3.57
10004 2.75 2.75 3.25 3 3
10002 3.6 2.4 4 3.6 4.2
10005 3.71 3.14 3.14 3.29 3.57
1 - 5