Nabin Thapa - 17 - I Database Assignment 3

```sql
CREATE SCHEMA IF NOT EXISTS assignment3;
-- Create Students table
CREATE TABLE
 IF NOT EXISTS assignment3.Students (
   student_id INT PRIMARY KEY,
   student_name VARCHAR(100),
   student_major VARCHAR(100)
 );
-- Create Courses table
CREATE TABLE
 IF NOT EXISTS assignment3.Courses (
   course_id INT PRIMARY KEY,
   course_name VARCHAR(100),
   course_description VARCHAR(255)
 );
-- Create Enrollments table
CREATE TABLE
 IF NOT EXISTS assignment3.Enrollments (
   enrollment_id INT PRIMARY KEY,
   student_id INT,
   course_id INT,
   enrollment_date DATE,
   FOREIGN KEY (student_id) REFERENCES assignment3.Students (student_id),
   FOREIGN KEY (course_id) REFERENCES assignment3.Courses (course_id)
 );
-- Insert data into Students table
INSERT INTO
 assignment3.Students (student_id, student_name, student_major)
VALUES
 (1, 'Alice', 'Computer Science'),
 (2, 'Bob', 'Biology'),
 (3, 'Charlie', 'History'),
 (4, 'Diana', 'Mathematics');
-- Insert data into Courses table
INSERT INTO
 assignment3.Courses (course_id, course_name, course_description)
VALUES
 (
   101,
   'Introduction to CS',
   'Basics of Computer Science'
 ),
 (102, 'Biology Basics', 'Fundamentals of Biology'),
```

Nabin Thapa - 17 - I Database Assignment 3

```
 (
   103,
   'World History',
   'Historical events and cultures'
 ),
 (104, 'Calculus I', 'Introduction to Calculus'),
 (105, 'Data Structures', 'Advanced topics in CS');
-- Insert data into Enrollments table
INSERT INTO
 assignment3.Enrollments (
   enrollment_id,
   student_id,
   course_id,
   enrollment_date
 )
VALUES
 (1, 1, 101, '2023-01-15'),
 (2, 2, 102, '2023-01-20'),
 (3, 3, 103, '2023-02-01'),
 (4, 1, 105, '2023-02-05'),
 (5, 4, 104, '2023-02-10'),
 (6, 2, 101, '2023-02-12'),
 (7, 3, 105, '2023-02-15'),
 (8, 4, 101, '2023-02-20'),
 (9, 1, 104, '2023-03-01'),
 (10, 2, 104, '2023-03-05');
```

```
--  Retrieve the list of students and their enrolled courses.
SELECT student_name
FROM assignment3.students s
JOIN assignment3.enrollments e
ON e.student_id = s.student_id
GROUP BY s.student_name;
```

	student_name
1	Alice
2	Bob
3	Charlie
4	Diana

Nabin Thapa - 17 - I Database Assignment 3

```sql
-- List all students and their enrolled courses,
-- including those who haven't enrolled in any course.
SELECT student_name, c.course_name
FROM assignment3.students s
LEFT JOIN assignment3.enrollments e
ON s.student_id = e.student_id
LEFT JOIN assignment3.courses c
ON e.course_id = c.course_id
ORDER BY s.student_name;
```

	student_name	course_name
1	Alice	Data Structures
2	Alice	Introduction to CS
3	Alice	Calculus I
4	Bhuban Bam	[NULL]
5	Bob	Calculus I
6	Bob	Biology Basics
7	Bob	Introduction to CS
8	Charlie	Data Structures
9	Charlie	World History
10	Diana	Introduction to CS
11	Diana	Calculus I

```sql
-- Display all courses and the students enrolled in each course,
-- including courses with no enrolled students.
SELECT c.course_name, student_name
FROM assignment3.students s
RIGHT JOIN assignment3.enrollments e
ON s.student_id = e.student_id
RIGHT JOIN assignment3.courses c
ON e.course_id = c.course_id
ORDER BY c.course_name;
```

	course_name	student_name
1	Biology Basics	Bob
2	Calculus I	Bob
3	Calculus I	Alice
4	Calculus I	Diana
5	Data Structures	Charlie
6	Data Structures	Alice
7	Introduction to CS	Bob
8	Introduction to CS	Alice
9	Introduction to CS	Diana
10	World History	Charlie

Nabin Thapa - 17 - I Database Assignment 3

```sql
-- Find pairs of students who are enrolled in at least one common course.
SELECT DISTINCT s.student_name AS student_1 , s2.student_name AS student_2
FROM assignment3.enrollments e
JOIN assignment3.enrollments e2 ON e.course_id = e2.course_id
JOIN assignment3.students s
ON e.student_id = s.student_id
JOIN assignment3.students s2
ON e2.student_id = s2.student_id
WHERE e.student_id < e2.student_id
```

	student_1	student_2
1	Alice	Diana
2	Bob	Diana
3	Alice	Bob
4	Alice	Charlie

```sql
-- Retrieve students who are enrolled in 'Introduction to CS' but not in
'Data Structures'.
SELECT s.student_name
FROM assignment3.students s
JOIN assignment3.enrollments e
ON s.student_id = e.student_id
JOIN assignment3.courses c
ON e.course_id = c.course_id
WHERE c.course_name = 'Introduction to CS'
AND s.student_id NOT IN (
   SELECT e2.student_id
   FROM assignment3.enrollments e2
   JOIN assignment3.courses c2
   ON e2.course_id = c2.course_id
   WHERE c2.course_name = 'Data Structures'
);
```

	student_name
1	Bob
2	Diana

Nabin Thapa - 17 - I Database Assignment 3

```sql
-- List all students along with a row number based on their enrollment date
in ascending order.
-- Using ROW_NUMBER()
SELECT s.student_name,
c.course_name,
ROW_NUMBER () OVER(
      -- PARTITION BY e.course_id
      ORDER BY e.enrollment_date
) AS row_number_count
FROM assignment3.enrollments e
JOIN assignment3.students s
ON s.student_id = e.student_id
JOIN assignment3.courses c
ON e.course_id = c.course_id
```

	student_name	course_name	row_number_count
1	Alice	Introduction to CS	1
2	Bob	Biology Basics	2
3	Charlie	World History	3
4	Alice	Data Structures	4
5	Diana	Calculus I	5
6	Bob	Introduction to CS	6
7	Charlie	Data Structures	7
8	Diana	Introduction to CS	8
9	Alice	Calculus I	9
10	Bob	Calculus I	10

```sql
-- Rank students based on the number of courses they are enrolled in,
-- handling ties by assigning the same rank. Using RANK()
SELECT s.student_name,
COUNT(e.course_id) AS course_count,
RANK () OVER(
      ORDER BY COUNT(e.course_id) DESC
)
FROM assignment3.enrollments e
JOIN assignment3.students s
ON s.student_id = e.student_id
GROUP BY s.student_i
```

	student_name	course_count	rank
1	Bob	3	1
2	Alice	3	1
3	Charlie	2	3
4	Diana	2	3

Nabin Thapa - 17 - I Database Assignment 3

```sql
-- Rank students based on the number of courses they are enrolled in,
-- handling ties by assigning the same rank. Using DENSE_RANK();
SELECT c.course_name,
COUNT(e.student_id) AS student_count,
DENSE_RANK () OVER(
      ORDER BY COUNT(e.student_id) DESC
)
FROM assignment3.enrollments e
JOIN assignment3.courses c
ON e.course_id = c.course_id
GROUP BY c.course_name
```

	course_name	student_count	dense_rank
1	Calculus I	3	1
2	Introduction to CS	3	1
3	Data Structures	2	2
4	World History	1	3
5	Biology Basics	1	3