Assignment Report

Database Assignment 2

(Without JOIN)

Prepared By:

Devraj Neupane

Roll No: 7

Group: D

Create grades table

```
CREATE TABLE Grades (grade_id INT PRIMARY KEY, grade_name
VARCHAR(10));
```

Screenshot:

```
Data Output   Messages   Notifications

CREATE TABLE

Query returned successfully in 59 msec.
```

Create students table

```
CREATE TABLE Students (
  student_id INT PRIMARY KEY,
  student_name VARCHAR(50),
  student_age INT,
  student_grade_id INT,
  FOREIGN KEY (student_grade_id) REFERENCES Grades (grade_id)
);
```

Screenshot:

```
Data Output   Messages   Notifications

CREATE TABLE

Query returned successfully in 59 msec.
```

Create courses table

```
CREATE TABLE Courses (
  course_id INT PRIMARY KEY,
```

```
  course_name VARCHAR(50)
);
```

Screenshot:

```
Data Output   Messages   Notifications

CREATE TABLE

Query returned successfully in 59 msec.
```

Enrollments table

```
CREATE TABLE Enrollments (
  enrollment_id INT PRIMARY KEY,
  student_id INT,
  course_id INT,
  enrollment_date DATE,
  FOREIGN KEY (student_id) REFERENCES Students (student_id),
  FOREIGN KEY (course_id) REFERENCES Courses (course_id)
);
```

Screenshot

```
Data Output   Messages   Notifications

CREATE TABLE

Query returned successfully in 45 msec.
```

Insert queries:

Insert into Grades table

```
INSERT INTO
  Grades (grade_id, grade_name)
```

```sql
VALUES
  (1, 'A'),
  (2, 'B'),
  (3, 'C');
```

Screenshot:

Data Output Messages Notifications

	grade_id [PK] integer	grade_name character varying (10)
1	1	A
2	2	B
3	3	C

Insert into Courses table

```sql
INSERT INTO
  Courses (course_id, course_name)
VALUES
  (101, 'Math'),
  (102, 'Science'),
  (103, 'History');
```

Screenshot:

Data Output Messages Notifications

	grade_id [PK] integer	grade_name character varying (10)
1	1	A
2	2	B
3	3	C

Insert into Students table

```sql
INSERT INTO
  Students (
    student_id,
    student_name,
    student_age,
    student_grade_id
  )
VALUES
  (1, 'Alice', 17, 1),
  (2, 'Bob', 16, 2),
  (3, 'Charlie', 18, 1),
  (4, 'David', 16, 2),
  (5, 'Eve', 17, 1),
  (6, 'Frank', 18, 3),
  (7, 'Grace', 17, 2),
  (8, 'Henry', 16, 1),
  (9, 'Ivy', 18, 2),
  (10, 'Jack', 17, 3);
```

Screenshot:

Data Output Messages Notifications

	student_id [PK] integer	student_name character varying (50)	student_age integer	student_grade_id integer
1	1	Alice	17	1
2	2	Bob	16	2
3	3	Charlie	18	1
4	4	David	16	2
5	5	Eve	17	1
6	6	Frank	18	3
7	7	Grace	17	2
8	8	Henry	16	1
9	9	Ivy	18	2
10	10	Jack	17	3

Insert into Enrollments table

```sql
INSERT INTO
  Enrollments (
    enrollment_id,
    student_id,
    course_id,
    enrollment_date
  )
VALUES
  (1, 1, 101, '2023-09-01'),
  (2, 1, 102, '2023-09-01'),
  (3, 2, 102, '2023-09-01'),
  (4, 3, 101, '2023-09-01'),
  (5, 3, 103, '2023-09-01'),
  (6, 4, 101, '2023-09-01'),
  (7, 4, 102, '2023-09-01'),
  (8, 5, 102, '2023-09-01'),
  (9, 6, 101, '2023-09-01'),
  (10, 7, 103, '2023-09-01');
```

Screenshots:

Data Output Messages Notifications

	enrollment_id [PK] integer	student_id integer	course_id integer	enrollment_date date
1	1	1	101	2023-09-01
2	2	1	102	2023-09-01
3	3	2	102	2023-09-01
4	4	3	101	2023-09-01
5	5	3	103	2023-09-01
6	6	4	101	2023-09-01
7	7	4	102	2023-09-01
8	8	5	102	2023-09-01
9	9	6	101	2023-09-01
10	10	7	103	2023-09-01

Questions:

1. Find all students enrolled in the Math course.

```sql
SELECT
  s.*
FROM
  students s
WHERE
  s.student_id IN (
    SELECT
      e.student_id
    FROM
      enrollments e
    WHERE
      e.course_id IN (
        SELECT
          c.course_id
        FROM
          courses c
        WHERE
          c.course_name = 'Math'
      )
  )
GROUP BY
  s.student_id;
```

Screenshots:

Data Output Messages Notifications

	student_id [PK] integer	student_name character varying (50)	student_age integer	student_grade_id integer
1	1	Alice	17	1
2	3	Charlie	18	1
3	4	David	16	2
4	6	Frank	18	3

2. List all courses taken by students named Bob.

```
SELECT
  c.*
FROM
  courses c
WHERE
  c.course_id IN (
    SELECT
      e.course_id
    FROM
      enrollments e
    WHERE
      e.student_id IN (
        SELECT
          s.student_id
        FROM
          students s
        WHERE
          s.student_name = 'Bob'
      )
  );
```

Screenshots:

Data Output Messages Notifications

	course_id [PK] integer	course_name character varying (50)
1	102	Science

3. Find the names of students who are enrolled in more than one course.

```sql
SELECT
  s.student_name
FROM
  students s
WHERE
  s.student_id IN (
    SELECT
      e.student_id
    FROM
      enrollments e
    GROUP BY
      e.student_id
    HAVING
      COUNT(e.course_id) > 1
  );
```

Screenshots:

Data Output Messages Notifications

	student_name character varying (50)
1	Alice
2	Charlie
3	David

4. List all students who are in Grade A (grade_id = 1).

```
SELECT
  s.*
FROM
  students s
WHERE
  s.student_grade_id IN (
    SELECT
      g.grade_id
    FROM
      grades g
    WHERE
      g.grade_id = 1
  );
```

Screenshot:

Data Output Messages Notifications

	student_id [PK] integer	student_name character varying (50)	student_age integer	student_grade_id integer
1	1	Alice	17	1
2	3	Charlie	18	1
3	5	Eve	17	1
4	8	Henry	16	1

5. Find the number of students enrolled in each course.

```
SELECT
  course_id,
  (
    SELECT
      c.course_name
    FROM
```

```
      courses c
    WHERE
      c.course_id = e.course_id
  ) AS course_name,
  COUNT(student_id) AS students_enrolled
FROM
  enrollments e
GROUP BY
  course_id
ORDER BY
  students_enrolled DESC;
```

Screenshot:

Data Output Messages Notifications

	course_id integer	course_name character varying (50)	students_enrolled bigint
1	101	Math	4
2	102	Science	4
3	103	History	2

6. Retrieve the course with the highest number of enrollments.

```
SELECT
  course_name,
  (
    SELECT
      Count(*)
    FROM
      enrollments e
    WHERE
      c.course_id = e.course_id
  )
FROM
  Courses c
```

```sql
WHERE
  c.course_id IN (
    SELECT
      course_id
    FROM
      enrollments e
    GROUP BY
      course_id
    HAVING
      COUNT(e.student_id) = (
        SELECT
          COUNT(student_id)
        FROM
          enrollments
        GROUP BY
          course_id
        ORDER BY
          COUNT(student_id) DESC
        LIMIT
          1
      )
  );
```

Screenshot:

Data Output Messages Notifications

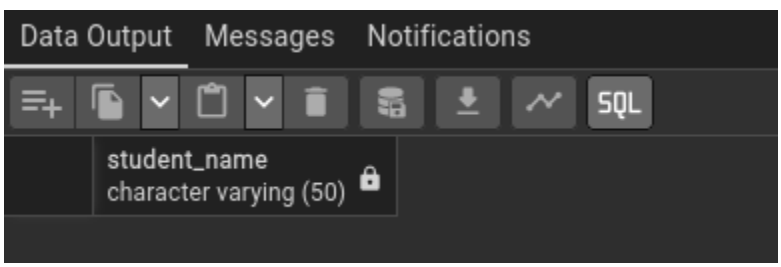
	course_name character varying (50)	count bigint
1	Math	4
2	Science	4

7. List students who are enrolled in all available courses.

```sql
SELECT
  student_name
FROM
```

```
  students s
WHERE
  s.student_id IN (
    SELECT
      e.student_id
    FROM
      enrollments e
    GROUP BY
      e.student_id
    HAVING
      COUNT(e.course_id) = (
        SELECT
          COUNT(c.course_id)
        FROM
          courses c
      )
  );
```

Screenshot:



8. Find students who are not enrolled in any courses.

```
SELECT
  s.*
FROM
  students s
WHERE
  s.student_id NOT IN (
    SELECT
      e.student_id
```

```
    FROM
      enrollments e
  );
```

Screenshot:

Data Output Messages Notifications

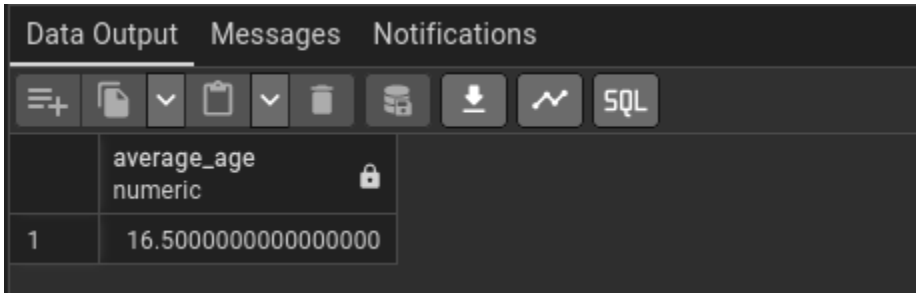
	student_id [PK] integer	student_name character varying (50)	student_age integer	student_grade_id integer
1	8	Henry	16	1
2	9	Ivy	18	2
3	10	Jack	17	3

9. Retrieve the average age of students enrolled in the Science course.

```
SELECT
  AVG(student_age) AS average_age
FROM
  students
WHERE
  student_id IN (
    SELECT
      e.student_id
    FROM
      enrollments e
    WHERE
      e.course_id IN (
        SELECT
          c.course_id
        FROM
          courses c
        WHERE
          c.course_name = 'Science'
      )
```

```
);
```

Screenshot:



10. Find the grade of students enrolled in the History course.

```
SELECT
  student_name,
  (
    SELECT
      g.grade_name
    FROM
      grades g
    WHERE
      s.student_grade_id = g.grade_id
  )
FROM
  students s
WHERE
  student_id IN (
    SELECT
      student_id
    FROM
      enrollments
    WHERE
      course_id = (
        SELECT
          course_id
        FROM
          courses
```

```
        WHERE
          course_name = 'History'
      )
  );
```

Screenshot:

Data Output Messages Notifications

	student_name character varying (50)	grade_name character varying (10)
1	Charlie	A
2	Grace	B