

Nisan Shrestha - 18 - DB Assignment no 2.

Day 2

A.

Create queries:

-- 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)  
);
```

-- Grades table

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

-- Courses table

```
CREATE TABLE Courses (  
  course_id INT PRIMARY KEY,  
  course_name VARCHAR(50)  
);
```

-- 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)  
);
```

Insert queries:

-- Insert into Grades table

```
INSERT INTO Grades (grade_id,  
  grade_name) VALUES  
(1, 'A'),  
(2, 'B'),  
(3, 'C');
```

-- Insert into Courses table

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

-- Insert into Students table

```
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);
```

-- Insert into Enrollments table

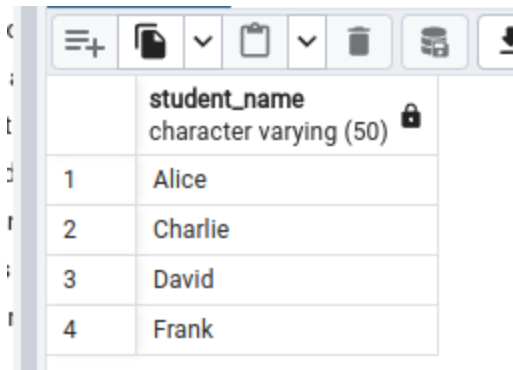
```
INSERT INTO Enrollments (enrollment_id,  
  student_id, course_id, enrollment_date)  
VALUES  
(1, 1, 101, '2023-09-01'),  
(2, 1, 102, '2023-09-01'),  
(3, 2, 102, '2023-09-01'),  
(4, 3, 101, '2023-09-01'),  
(5, 3, 103, '2023-09-01'),  
(6, 4, 101, '2023-09-01'),  
(7, 4, 102, '2023-09-01'), 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

Questions:

1. Find all students enrolled in the Math course.

```
SELECT  
  student_name  
FROM  
  Students  
WHERE  
  student_id IN (  
    SELECT  
      student_id  
    FROM  
      Enrollments  
    WHERE  
      course_id = (  
        SELECT  
          course_id  
        FROM  
          Courses  
        WHERE  
          course_name = 'Math'  
      )  
  )  
);
```

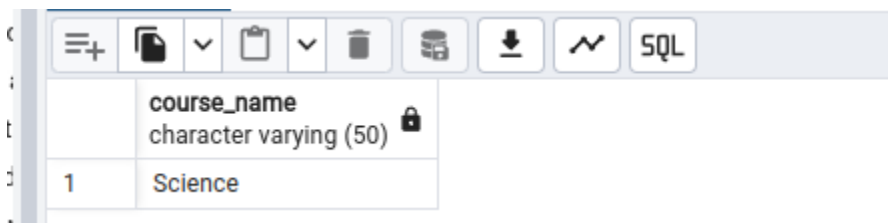
Nisan Shrestha - 18 - DB Assignment no 2.



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

2. List all courses taken by students named Bob.

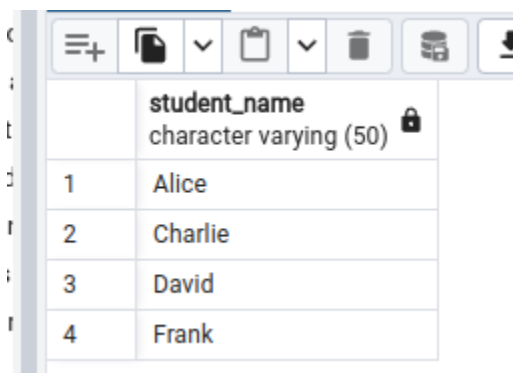
```
SELECT
  course_name
FROM
  courses
WHERE
  course_id IN (
    SELECT
      course_id
    FROM
      enrollments
    WHERE
      student_id = (
        SELECT
          student_id
        FROM
          students
        WHERE
          student_name = 'Bob'
      )
  );
```



courses	
course_name	
character varying (50)	
1	Science

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

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

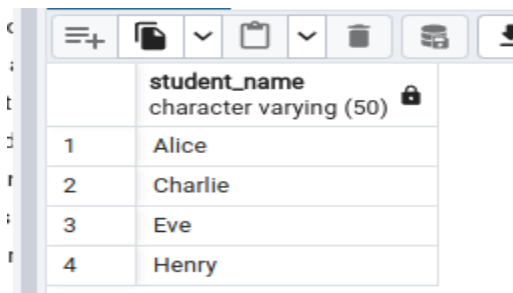


A screenshot of a database query result. The table has a single column labeled 'student_name' with a data type of 'character varying (50)' and a lock icon. The table contains four rows of data:

	student_name
1	Alice
2	Charlie
3	David
4	Frank

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

```
SELECT
    student_name
FROM
    students
WHERE
    student_grade_id = 1;
```

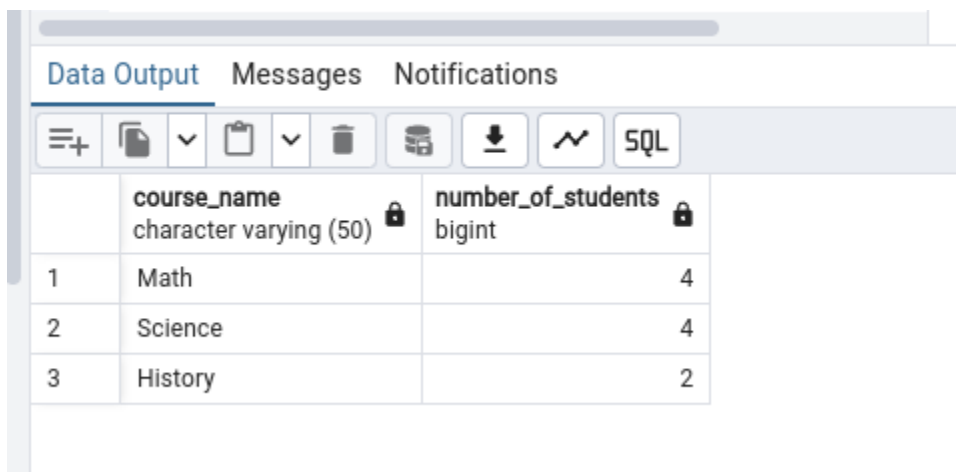


A screenshot of a database query result. The table has a single column labeled 'student_name' with a data type of 'character varying (50)' and a lock icon. The table contains four rows of data:

	student_name
1	Alice
2	Charlie
3	Eve
4	Henry

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

```
SELECT
    course_name,
    (
        SELECT
            COUNT(*)
        FROM
            Enrollments E
        WHERE
            E.course_id = C.course_id
    ) AS number_of_students
FROM
    Courses C;
```



The screenshot shows a database management interface with three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, displaying a table with two columns: 'course_name' (character varying (50)) and 'number_of_students' (bigint). The table contains three rows of data: Math (4 students), Science (4 students), and History (2 students). The interface includes a toolbar with icons for expand, save, undo, redo, delete, refresh, download, and a 'SQL' button.

	course_name character varying (50)	number_of_students bigint
1	Math	4
2	Science	4
3	History	2

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

```
SELECT
    course_name,
    (
        SELECT
            COUNT(*)
        FROM
            Enrollments E
        WHERE
            E.course_id = C.course_id
    ) AS number_of_students
FROM
    Courses C
ORDER BY
    number_of_students desc
LIMIT
    1;
```

Data Output Messages Notifications		
	course_name character varying (50) 🔒	number_of_students bigint 🔒
1	Math	4

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

```
SELECT
    student_name
FROM
    Students
WHERE
    student_id IN (
        SELECT
            student_id
        FROM
            Enrollments
        GROUP BY
            student_id
        HAVING
            COUNT(course_id) = (
                SELECT
                    COUNT(course_id)
                FROM
                    Courses
            )
    );
```

Data Output			Messages			Notifications		
<div><div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div></div>								
			student_name character varying (50)					

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

```
SELECT
    student_name
FROM
    Students
WHERE
    student_id NOT IN (
        SELECT
            student_id
        FROM
```

Nisan Shrestha - 18 - DB Assignment no 2.

```
Enrollments
GROUP BY
    student_id
HAVING
    COUNT(course_id) > 0
);
```

Data Output		Messages	Notificat
	student_name character varying (50) 🔒		
1	Henry		
2	Ivy		
3	Jack		

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

```
SELECT
    AVG(student_age) as Avg_Age
FROM
    Students
WHERE
    student_id IN (
        SELECT
            student_id
        FROM
            Enrollments
        WHERE
            course_id = (
                SELECT
                    course_id
                FROM
                    courses
                WHERE
                    course_name = 'Science'
            )
    )
);
```

Data Output			Messages	Notificat
	avg_age			
	numeric			
1	16.5000000000000000			

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

```
SELECT
    student_name,
    (
        Select
            grade_name
        from
            grades G
        where
            S.student_grade_id = G.grade_id
    ) as Grade
from
    Students S
where
    student_id in (
        select
            student_id
        from
            Enrollments E
        where
            E.course_id = (
                Select
                    course_id
                from
                    courses
                where
                    course_name = 'History'
            )
    )
)
```

Data Output			Messages	Notifications
	student_name character varying (50)	grade character varying (10)		
1	Charlie	A		
2	Grace	B		

2. Please design and create the necessary tables (Books, Authors, Publishers, Customers, Orders, Book_Authors, Order_Items) for an online bookstore database. Ensure each table includes appropriate columns, primary keys, and foreign keys where necessary. Consider the relationships between these tables and how they should be defined.

...

Created different DB

```
create table
    publishers (publisher_id int primary key, publisher_name varchar(100),
country varchar(50));

create table
    books (book_id int primary key, title Varchar(100), genre varchar(50),
publisher_id int, publication_year date, foreign key (publisher_id)
references publishers (publisher_id));

create table
    customers (customer_id int primary key, customer_name varchar(50) not
null, email varchar(150) unique, address varchar(50));

create table
    authors (author_id int primary key, author_name varchar(50) not null,
birth_date date, nationality varchar(50));

create table
    orders (order_id int primary key, order_date date default current_date,
customer_id int, total_amount int default 1, foreign key (customer_id)
references customers (customer_id));

create table
    book_authors (book_id int, author_id int, primary key (book_id,
author_id), foreign key (book_id) references books (book_id), foreign key
```



```
(author_id) references authors (author_id));
```

```
create table
```

```
order_items (order_id int, book_id int, primary key (order_id,  
book_id), foreign key (book_id) references books (book_id), foreign key  
(order_id) references orders (order_id));
```

