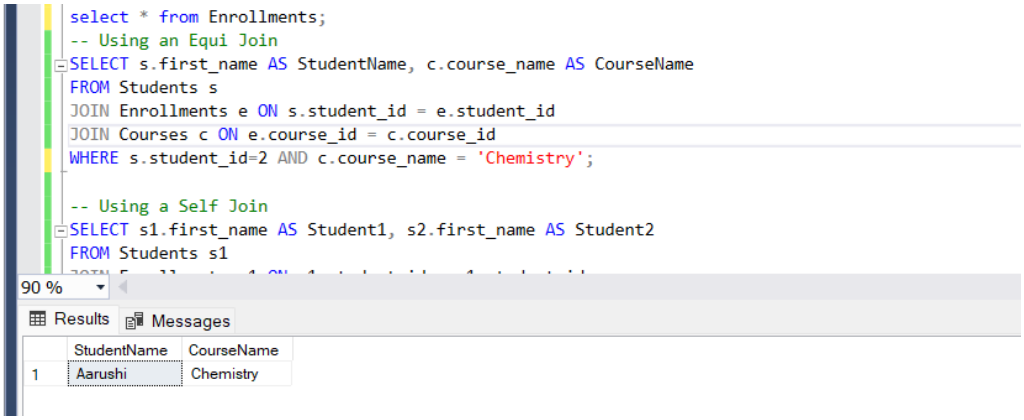
Querying Data by Using Joins and Subqueries

Name: Aathirainathan P

Date: 05-11-2024

---

1. **Using an Equi Join:**



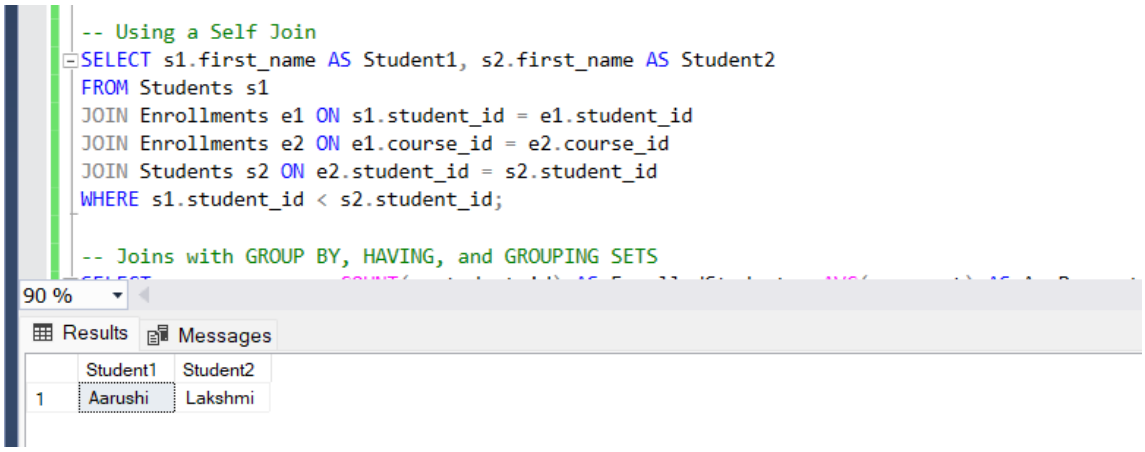
```
select * from Enrollments;
-- Using an Equi Join
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id
WHERE s.student_id=2 AND c.course_name = 'Chemistry';

-- Using a Self Join
SELECT s1.first_name AS Student1, s2.first_name AS Student2
FROM Students s1
```

90 %

Results | Messages

	StudentName	CourseName
1	Aarushi	Chemistry

2. **Using a Self Join:**



```
-- Using a Self Join
SELECT s1.first_name AS Student1, s2.first_name AS Student2
FROM Students s1
JOIN Enrollments e1 ON s1.student_id = e1.student_id
JOIN Enrollments e2 ON e1.course_id = e2.course_id
JOIN Students s2 ON e2.student_id = s2.student_id
WHERE s1.student_id < s2.student_id;

-- Joins with GROUP BY, HAVING, and GROUPING SETS
```

90 %

Results | Messages

	Student1	Student2
1	Aarushi	Lakshmi

3. Joins with groupby,having by,grouping sets:

```sql
-- Joins with GROUP BY, HAVING, and GROUPING SETS
SELECT c.course_name, COUNT(e.student_id) AS EnrolledStudents, AVG(p.amount) AS AvgPayment
FROM Courses c
JOIN Enrollments e ON c.course_id = e.course_id
JOIN Payments p ON e.student_id = p.student_id
GROUP BY c.course_name
HAVING COUNT(e.student_id) > 2
ORDER BY AVG(p.amount) DESC;

-- Querying Data by Using Subqueries - Get students who paid more than the average payment
SELECT s.first_name, s.last_name
FROM Students s
WHERE s.student_id IN (
```

%

Results | Messages

course_name	EnrolledStudents	AvgPayment
Advanced Database Management	8	500.000000

4. Querying Data by Using Subqueries:

```sql
-- Querying Data by Using Subqueries - Get students who paid more than the average payment
SELECT s.first_name, s.last_name
FROM Students s
WHERE s.student_id IN (
    SELECT p.student_id
    FROM Payments p
    GROUP BY p.student_id
    HAVING SUM(p.amount) > (SELECT AVG(amount) FROM Payments)
);
```

90 %

Results | Messages

	first_name	last_name
1	Aarushi	Narayan
2	Karthik	Venugopal
3	Priya	Balasubramaniam
4	Vignesh	Iyer
5	Lakshmi	Sankaran
6	Shyam	Kumar
7	Sowmya	Narayanan
8	Jane	Johnson

5. Using the EXISTS,ANY,ALL Keywords:

```sql
-- Using the EXISTS keyword - Check if a student is enrolled in any course
SELECT s.first_name, s.last_name
FROM Students s
WHERE EXISTS (
    SELECT 1
    FROM Enrollments e
    WHERE e.student_id = s.student_id
);

-- Using the ANY keyword - Find students who paid more than any other student
SELECT s.first_name, s.last_name
FROM Students s
WHERE (SELECT SUM(p.amount) FROM Payments p WHERE p.student_id = s.student_id) > ANY (
    SELECT SUM(p.amount) FROM Payments p GROUP BY p.student_id
);

-- Using the ALL keyword - Get students who have paid more than all other students
SELECT s.first_name, s.last_name
FROM Students s
WHERE (SELECT SUM(p.amount) FROM Payments p WHERE p.student_id = s.student_id) > ALL (
    SELECT SUM(p.amount) FROM Payments p GROUP BY p.student_id
);
```

90 %

Results | Messages

	first_name	last_name
1	Aadhithya	Srinivasan
2	Aarushi	Narayan
3	Karthik	Venugopal
4	Priya	Balasubramaniam
5	Divya	Rajendran
6	Vignesh	Iyer
7	Lakshmi	Sankaran
8	Shyam	Kumar
9	Sowmya	Narayanan
10	Jane	Johnson

6. Using Nested Subqueries:

```sql
-- Using Nested Subqueries - Find courses with no students enrolled
SELECT c.course_name
FROM Courses c
WHERE c.course_id NOT IN (
    SELECT e.course_id
    FROM Enrollments e
    WHERE e.course_id = c.course_id
);
```

90 %

Results | Messages

	course_name
1	Mathematics
2	Physics

7. Using Correlated Subqueries:

```sql
-- Using Correlated Subqueries - Find students who are enrolled in the same course as 'Aadhithya'
SELECT s.first_name, s.last_name
FROM Students s
WHERE EXISTS (
    SELECT 1
    FROM Enrollments e1
    WHERE e1.student_id = s.student_id
    AND e1.course_id IN (
        SELECT e2.course_id
        FROM Enrollments e2
        WHERE e2.student_id = (SELECT student_id FROM Students WHERE first_name = 'Aarushi')
    )
);

-- Using UNION - Get all unique student names and course names
```

Results | Messages

first_name	last_name
Aarushi	Narayan
Lakshmi	Sankaran

8. Using UNION,INTERSECT,EXCEPT,MERGE:

```sql
-- Using EXCEPT - Get students who are enrolled in 'Physics' but not in 'Chemistry'
SELECT s.first_name
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id
WHERE c.course_name = 'Physics'
EXCEPT
SELECT s.first_name
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id
WHERE c.course_name = 'Chemistry';

-- Using MERGE
MERGE INTO Payments p
USING Students s ON p.student_id = s.student_id
WHEN MATCHED AND p.amount > 500 THEN
    UPDATE SET p.payment_date = '2024-09-10';
```

Messages

```
(7 rows affected)

Completion time: 2024-11-05T14:14:23.1509762+05:30
```