Querying Data by Using Joins

Name: Aathirainathan P

Date: 05-11-2024

---

1. **Inner Join:**

```sql
-- INNER JOIN Example: List all students with their course names
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s INNER JOIN Enrollments e ON s.student_id = e.student_id
INNER JOIN Courses c ON e.course_id = c.course_id;

-- LEFT JOIN Example: Show all students and the courses they are enrolled in (if any)
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
LEFT JOIN Enrollments e ON s.student_id = e.student_id
LEFT JOIN Courses c ON e.course_id = c.course_id;

-- RIGHT JOIN Example: List all courses and their enrolled students (if any)
```

90 %

Results | Messages

	StudentName	CourseName
1	Aarushi	Chemistry
2	Karthik	Computer Science
3	Priya	Biology
4	Anirudh	History
5	Divya	Geography
6	Vignesh	Political Science
7	Lakshmi	Economics
8	Jane	Advanced Database Management
9	Lakshmi	Chemistry

2. Left Join:

```
-- LEFT JOIN Example: Show all students and the courses they are enrolled in (if any)
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
LEFT JOIN Enrollments e ON s.student_id = e.student_id
LEFT JOIN Courses c ON e.course_id = c.course_id;

-- RIGHT JOIN Example: List all courses and their enrolled students (if any)
SELECT c.course_name AS CourseName, s.first_name AS StudentName
FROM Courses c
RIGHT JOIN Enrollments e ON c.course_id = e.course_id
RIGHT JOIN Students s ON e.student_id = s.student_id;

-- FULL OUTER JOIN Example: List all students and all courses, showing nulls where there is no match
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
FULL OUTER JOIN Enrollments e ON s.student_id = e.student_id
```

Results | Messages

	StudentName	CourseName
1	Aadhithya	NULL
2	Aarushi	Chemistry
3	Karthik	Computer Science
4	Priya	Biology
5	Anirudh	History
6	Divya	Geography
7	Vignesh	Political Science
8	Lakshmi	Economics
9	Lakshmi	Chemistry
10	Shyam	NULL
11	Sowmya	NULL
12	John	NULL
13	Jane	Advanced Database Management

3. Right Join:

```
-- RIGHT JOIN Example: List all courses and their enrolled students (if any)
SELECT c.course_name AS CourseName, s.first_name AS StudentName
FROM Courses c
RIGHT JOIN Enrollments e ON c.course_id = e.course_id
RIGHT JOIN Students s ON e.student_id = s.student_id;

-- FULL OUTER JOIN Example: List all students and all courses, showing nulls where there is no match
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
FULL OUTER JOIN Enrollments e ON s.student_id = e.student_id
```

90 %

Results | Messages

	CourseName	StudentName
1	NULL	Aadhithya
2	Chemistry	Aarushi
3	Computer Science	Karthik
4	Biology	Priya
5	History	Anirudh
6	Geography	Divya
7	Political Science	Vignesh
8	Economics	Lakshmi
9	Chemistry	Lakshmi
10	NULL	Shyam
11	NULL	Sowmya
12	NULL	John
13	Advanced Database Management	Jane

4. Full Outer Join:

```sql
-- FULL OUTER JOIN Example: List all students and all courses, showing nulls where there is no match
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
FULL OUTER JOIN Enrollments e ON s.student_id = e.student_id
FULL OUTER JOIN Courses c ON e.course_id = c.course_id;

-- CROSS JOIN Example: Get a cartesian product of students and courses
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
CROSS JOIN Courses c;

-- Using GROUP BY with Aggregation: Number of students enrolled in each course
SELECT c.course_name, COUNT(e.student_id) AS EnrolledStudents
FROM Courses c
LEFT JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_name
```

90 %

Results | Messages

	StudentName	CourseName
1	NULL	Mathematics
2	NULL	Physics
3	Aarushi	Chemistry
4	Lakshmi	Chemistry
5	Karthik	Computer Science
6	Priya	Biology
7	Anirudh	History
8	Divya	Geography
9	Vignesh	Political Science
10	Lakshmi	Economics
11	Jane	Advanced Database Management
12	Aadhithya	NULL
13	Shyam	NULL
14	Sowmya	NULL
15	John	NULL

5. Cross Join:

```sql
-- CROSS JOIN Example: Get a cartesian product of students and courses
SELECT s.first_name AS StudentName, c.course_name AS CourseName
FROM Students s
CROSS JOIN Courses c;

-- Using GROUP BY with Aggregation: Number of students enrolled in each course
SELECT c.course_name, COUNT(e.student_id) AS EnrolledStudents
FROM Courses c
LEFT JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_name
```

90 %

Results | Messages

	StudentName	CourseName
1	Aadhithya	Mathematics
2	Aarushi	Mathematics
3	Karthik	Mathematics
4	Priya	Mathematics
5	Anirudh	Mathematics
6	Divya	Mathematics
7	Vignesh	Mathematics
8	Lakshmi	Mathematics
9	Shyam	Mathematics
10	Sowmya	Mathematics
11	John	Mathematics
12	Jane	Mathematics
13	Aadhithya	Physics
14	Aarushi	Physics
15	Karthik	Physics
16	Priya	Physics

6. Joins with Aggregation:

```sql
-- Aggregation: Total payments by each student
SELECT s.first_name AS StudentName, SUM(p.amount) AS TotalPayment
FROM Students s
INNER JOIN Payments p ON s.student_id = p.student_id
GROUP BY s.first_name
ORDER BY TotalPayment DESC;

-- Using HAVING to filter groups: Get courses with more than 1 student enrolled
SELECT c.course_name, COUNT(e.student_id) AS EnrolledStudents
FROM Courses c
LEFT JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_name
HAVING COUNT(e.student_id) > 1
ORDER BY EnrolledStudents DESC;

-- Using INNER JOIN with Aggregation: Total payments for each course (aggregating by course)
```

90 %

Results | Messages

	StudentName	TotalPayment
1	Jane	4000.00
2	Priya	700.00
3	Shyam	700.00
4	Vignesh	650.00
5	Lakshmi	600.00
6	Aarushi	600.00
7	Sowmya	550.00
8	Karthik	550.00
9	Aadhithya	500.00
10	Divya	500.00
11	Anirudh	450.00