upGrad

PLACEMENT REFRESHER PROGRAM

Session 4 - SQL 2

Working with Multiple Tables and Sub-Queries

By
Ritesh Kumar Pandey

Agenda

- Joins
- Nested Queries

Joins - Introduction

upGrad

Why to use Joins ?

ID	Name	Roll No.	Department
1	Ritesh	143	Information Technology
2	Satyam	28	Computer
3	Rahul	88	Computer

ID	Marks
1	98
2	68
3	77

Joins - Introduction

upGrad

- Joins in SQL are used to combine rows from two or more tables based on a related column between those tables.
- ‘JOIN’ keyword is used in SQL queries for joining two or more tables.
- Predominantly used when data is to be extracted from tables having one-to-many or many-to-many relationships between them.

Joins - Types

upGrad

Inner Join

- Natural Join

Cross Join

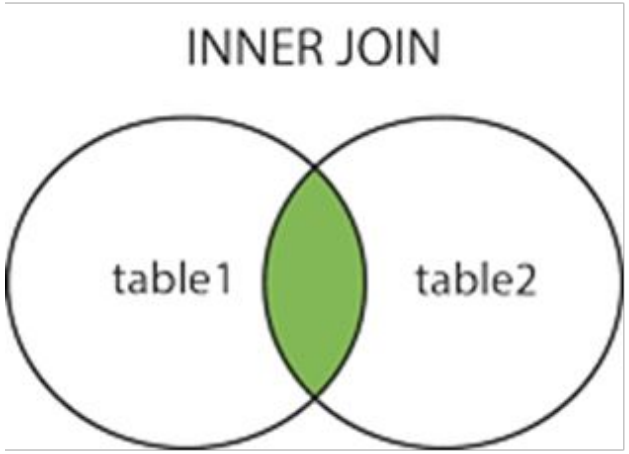
Outer Join

- Left Outer Join
- Right Outer Join
- Full Outer Join

Joins - Types - INNER JOIN

upGrad

```
SELECT column-name-list
from table-name1
INNER JOIN
table-name2
WHERE table-name1.column-name = table-
  name2.column-name;
```



ID	NAME
1	ABHI
2	ADAM
3	ALEX
4	ANU

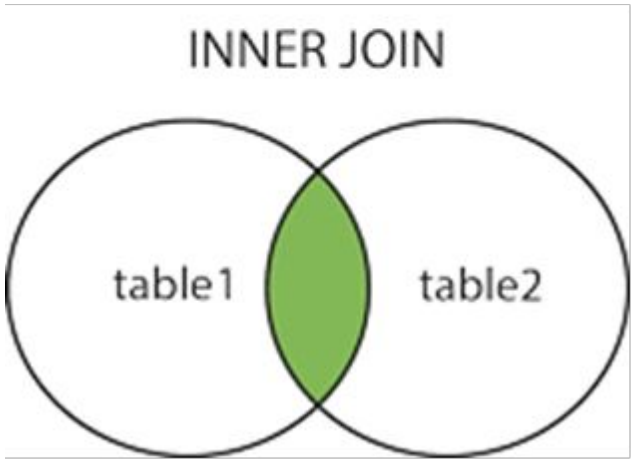
ID	ADDRESS
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

ID	NAME	ID	Address
1	ABHI	1	DELHI
2	ADAM	2	MUMBAI
3	ALEX	3	CHENNAI

Joins - Types - NATURAL JOIN

upGrad

```
SELECT *
from table-name1
NATURAL JOIN
table-name2;
```



ID	NAME
1	ABHI
2	ADAM
3	ALEX
4	ANU

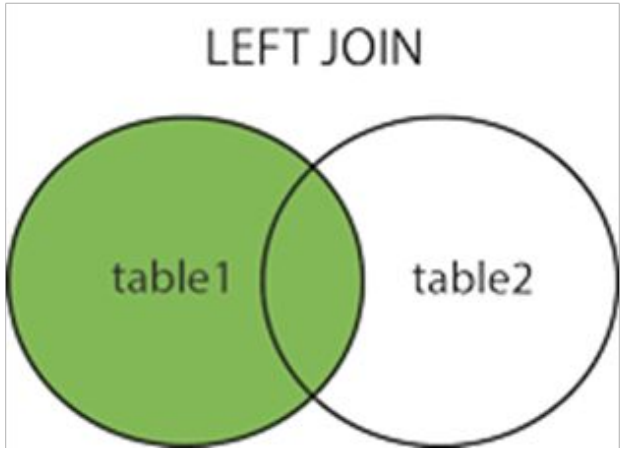
ID	ADDRESS
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

ID	NAME	Address
1	ABHI	DELHI
2	ADAM	MUMBAI
3	ALEX	CHENNAI

Joins - Types - LEFT OUTER JOIN

upGrad

```
SELECT column-name-list
from table-name1
LEFT OUTER JOIN
table-name2
on table-name1.column-name = table-name2.column-name;
```



ID	NAME
1	ABHI
2	ADAM
3	ALEX
4	ANU

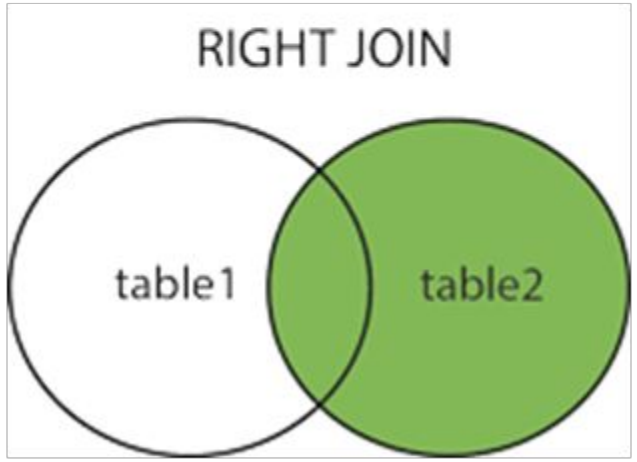
ID	ADDRESS
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

ID	NAME	ID	ADDRESS
1	ABHI	1	DELHI
2	ADAM	2	MUMBAI
3	ALEX	3	CHENNAI
4	ANU	NULL	NULL

Joins - Types - RIGHT OUTER JOIN

upGrad

```
select column-name-list
from table-name1
RIGHT OUTER JOIN
table-name2
on table-name1.column-name = table-name2.column-name;
```



ID	NAME
1	ABHI
2	ADAM
3	ALEX
4	ANU

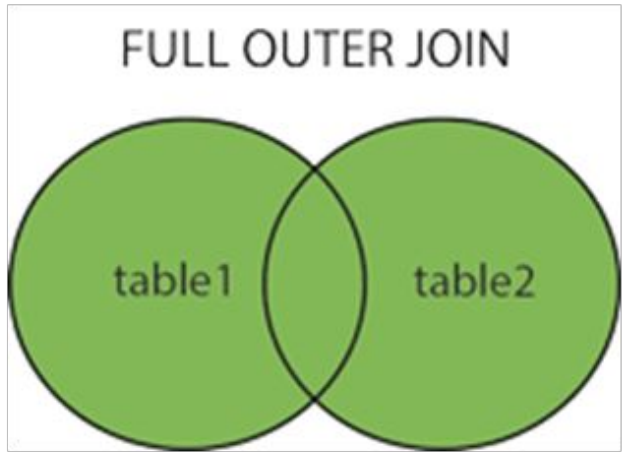
ID	ADDRESS
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

ID	NAME	ID	ADDRESS
1	ABHI	1	DELHI
2	ADAM	2	MUMBAI
3	ALEX	3	CHENNAI
NULL	NULL	7	NOIDA
NULL	NULL	8	PANIPAT

Joins - Types - FULL OUTER JOIN

upGrad

```
select column-name-list
from table-name1
FULL OUTER JOIN
table-name2
on table-name1.column-name = table-name2.column-name;
```



ID	NAME
1	ABHI
2	ADAM
3	ALEX
4	ANU

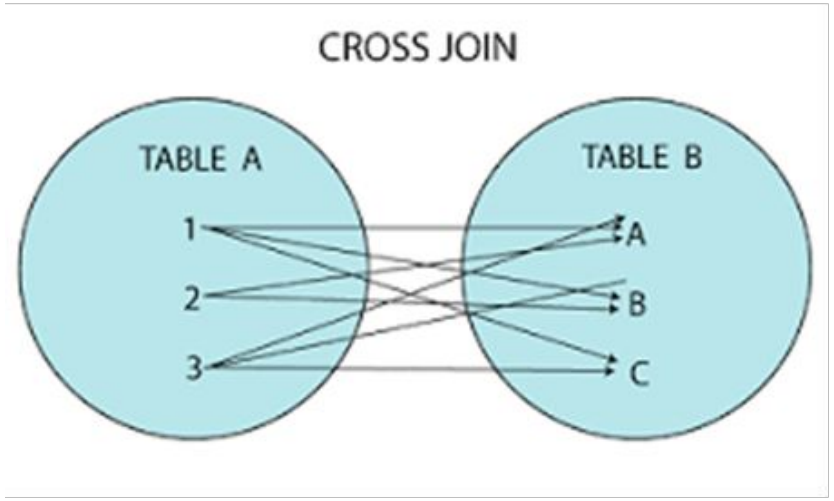
ID	ADDRESS
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

ID	NAME	ID	ADDRESS
1	ABHI	1	DELHI
2	ADAM	2	MUMBAI
3	ALEX	3	CHENNAI
4	ANU	NULL	NULL
NULL	NULL	7	NOIDA
NULL	NULL	8	PANIPAT

Joins - Types - CROSS JOIN

upGrad

```
Select * from CountryStatus
Cross Join PRODUCTS
```



ID	Name
1	Abhi
2	Alex

Sr. No.	Address
4	Delhi
5	Mumbai

ID	Name	Sr. No.	Address
1	Abhi	4	Delhi
1	Abhi	5	Mumbai
2	Alex	4	Delhi
2	Alex	5	Mumbai

Joins - Examples

upGrad

Inner Join

```
select * from student_info inner join student_marks on
student_info.student_id = student_marks.student_id;
```

Natural Join

```
select * from student_info natural join student_marks;
```

Joins - Examples

upGrad

Left Outer Join

```
select * from student_info left outer join student_marks on
student_info.student_id = student_marks.student_id;
```

Right Outer Join

```
select * from student_info right outer join student_marks on
student_info.student_id = student_marks.student_id;
```

Cross Join

```
select * from student_info cross join student_marks;
```

Joins - Question

upGrad

Write a SQL query to find the average marks for each department in the "student_marks" table and display the department name.

Joins - Question

upGrad

Write a SQL query to find the average marks for each department in the "student_marks" table and display the department name.

```
SELECT si.department, AVG(sm.marks) AS average_marks
FROM student_info si
JOIN student_marks sm ON si.student_id = sm.student_id
GROUP BY si.department;
```

Nested Queries

upGrad

- Nested queries are a way to perform more complex queries by embedding one query within another.
- A nested query is a query that appears inside another query, and it helps retrieve data from multiple tables or apply conditions based on the results of another query.
- The result of inner query is used in execution of outer query.

Nested Queries - Questions

upGrad

Write a SQL query to retrieve the names of students who scored the highest marks in the "student_marks" table.

Nested Queries - Questions

upGrad

Write a SQL query to retrieve the names of students who scored the highest marks in the "student_marks" table.

```
SELECT student_name
FROM student_info
WHERE student_id = (SELECT student_id FROM student_marks
WHERE marks = (SELECT MAX(marks) FROM student_marks));
```

Nested Queries - Questions

upGrad

Write a SQL query to list the names of students who scored marks higher than the average marks in their respective departments.

Nested Queries - Questions

upGrad

Write a SQL query to list the names of students who scored marks higher than the average marks in their respective departments.

```
SELECT si.student_name
FROM student_info si
JOIN student_marks sm ON si.student_id = sm.student_id
WHERE sm.marks > (SELECT AVG(sm2.marks) FROM student_marks
sm2 WHERE sm2.student_id = si.student_id);
```

Nested Queries - Questions

upGrad

Write a SQL query to find the department(s) with the highest average marks and display the department name(s).

Nested Queries - Questions

upGrad

Write a SQL query to find the department(s) with the highest average marks and display the department name(s).

```
SELECT si.department
FROM student_info si
JOIN student_marks sm ON si.student_id = sm.student_id
GROUP BY si.department
HAVING AVG(sm.marks) = (SELECT MAX(avg_marks) FROM
(SELECT si2.department, AVG(sm2.marks) AS avg_marks FROM
student_info si2 JOIN student_marks sm2 ON si2.student_id =
sm2.student_id GROUP BY si2.department) AS department_avg);
```

Nested Queries - Questions

upGrad

Write a SQL query to retrieve the names of students who scored marks higher than the lowest marks in the "student_marks" table.

Nested Queries - Questions

upGrad

Write a SQL query to retrieve the names of students who scored marks higher than the lowest marks in the "student_marks" table.

```
SELECT student_name
FROM student_info
WHERE student_id IN (SELECT student_id FROM student_marks
WHERE marks > (SELECT MIN(marks) FROM student_marks));
```

THANK YOU