

# Learn SQL Joins



**DIGIBYTES**

**LEARN  
CS / IP WITH  
PYTHON**

**FOR CLASSES  
XI & XII**

**ONLINE CLASSES**  
COMPLETE COURSE IN  
3 MONTHS  
INDIVIDUAL / BATCH

Surya Nagar, Ghaziabad  
<http://digibytes.co.in>  
Call : 9312352488

## SQL JOIN

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

## Different Types of SQL JOINS

Here are the different types of the JOINS in SQL:

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table

# POINTS TO REMEMBER BEFORE LEARNING SQL JOINS

- For this we will create 2 tables (student & class)
- Each table will have some entries
- For making concept of joins clear,
  - **in student table we will have a student entry which will not exists in class table**
  - **In class table we will have a entry which does not exists in student table**
- In SQL statement
  - The name of the table which appears first is known as left table
  - The name of the table which appears next to first table is known as right table

```
mysql> describe class;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| stu_id     | varchar(5)    | YES  | UNI | NULL    |      |
| stu_class  | varchar(10)   | YES  |     | NULL    |      |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from class;
+-----+-----+
| stu_id | stu_class |
+-----+-----+
| s001   | XI        |
| s002   | X         |
| s003   | IX        |
| s004   | IX        |
| s005   | V         |
| s006   | XI        |
| s007   | X         |
| s008   | IX        |
| s009   | IX        |
| s010   | V         |
| sRRR   | XI        |
+-----+-----+
11 rows in set (0.00 sec)
```

Note: stu\_id sRRR exists only in class table.

This stu\_id sRRR does not exist in student table

(Purposefully to explain you the concept of joins

```
mysql> describe student;
```

```
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| stu_id     | varchar(5)    | YES  | UNI | NULL    |      |
| stu_name   | varchar(50)   | YES  |     | NULL    |      |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.01 sec)
```

```
mysql> select * from student;
```

```
+-----+-----+
| stu_id | stu_name |
+-----+-----+
| s001   | gagan   |
| s002   | seema   |
| s003   | anuj    |
| s004   | nitin   |
| s005   | amit    |
| sLLL   | left singh |
+-----+-----+
6 rows in set (0.00 sec)
```

Note: stu\_id sLLL exists only in student table.

This stu\_id sLLL does not exist in class table

(Purposefully to explain you the concept of joins

## Easiest to understand is INNER JOIN

### INNER JOIN

The INNER JOIN keyword selects records that have matching values in both tables.

#### INNER JOIN

The INNER JOIN keyword selects records that have matching values in both tables.

```
SELECT student.stu_id, class.stu_class FROM student INNER JOIN class ON student.stu_id=class.stu_id;
```

```
mysql> SELECT student.stu_id, class.stu_class FROM student INNER JOIN class ON student.stu_id=class.stu_id;
```

```
+-----+-----+
| stu_id | stu_class |
+-----+-----+
| s001   | XI        |
| s002   | X         |
| s003   | IX        |
| s004   | IX        |
| s005   | V         |
+-----+-----+
5 rows in set (0.00 sec)
```

Note : Neither 'sLLL' not appear in result nor 'sRRR' because sLLL was not present in student table

And sRRR was not present in class Table

Note : 'sLLL' dose not appear in result because sLLL was not present in student table;

## LEFT JOIN

The LEFT JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

LEFT JOIN

The LEFT JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

```
mysql> SELECT student.stu_id, class.stu_class FROM student LEFT JOIN class ON student.stu_id=class.stu_id;
```

stu_id	stu_class
s001	XI
s002	X
s003	IX
s004	IX
s005	V
sLLL	NULL

6 rows in set (0.00 sec)

'sLLL' appear in result but sLLL was not present in class table , because

The LEFT JOIN keyword returns all records from the left table (student), and the matched records from the right table (class). The result is NULL from the right side, if there is no match.

Note : 'sLLL' appear in result but sLLL was not present in student table , because

The LEFT JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

## RIGHT JOIN

The RIGHT JOIN keyword returns all records from the right table (class), and the matched records from the left table (student). The result is NULL from the left side, when there is no match.

```
RIGHT JOIN

The RIGHT JOIN keyword returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, when there is no match.

mysql> SELECT student.stu_id, class.stu_class FROM student RIGHT JOIN class ON student.stu_id=class.stu_id;
+-----+-----+
| stu_id | stu_class |
+-----+-----+
| s001   | XI        |
| s002   | X         |
| s003   | IX       |
| s004   | IX       |
| s005   | V        |
| NULL   | XI       |
| NULL   | X        |
| NULL   | IX       |
| NULL   | IX       |
| NULL   | V        |
| NULL   | XI       |
+-----+-----+
11 rows in set (0.00 sec)

Note : Non matching stu_id not displayed, but all the class are displayed from class table
```

Here in Right Join , all the 11 records from table class are shown , because table class was my right table in the command.

stu_id	stu_class
s006	XI
s007	X
s008	IX
s009	IX
s010	V
sRRR	XI

But all other students id such as didn't match with the student id in student table hence NULL was placed

## UNION

The UNION operator is used to combine the result-set of two or more SELECT statements.

1. Each SELECT statement within UNION must have the same number of columns
2. The columns must also have similar data types
3. The columns in each SELECT statement must also be in the same order

```
UNION
The UNION operator is used to combine the result-set of two or more SELECT statements.
Each SELECT statement within UNION must have the same number of columns
The columns must also have similar data types
The columns in each SELECT statement must also be in the same order

mysql> select * from student union
-> select * from class;
+-----+-----+
| stu_id | stu_name |
+-----+-----+
| s001   | gagan   |
| s002   | seema   |
| s003   | anuj    |
| s004   | nitin   |
| s005   | amit    |
| sLLL   | left singh |
+-----+-----+
| s001   | XI      |
| s002   | X       |
| s003   | IX      |
| s004   | IX      |
| s005   | V       |
| s006   | XI      |
| s007   | X       |
| s008   | IX      |
| s009   | IX      |
| s010   | V       |
| sRRR   | XI      |
+-----+-----+
17 rows in set (0.00 sec)
```

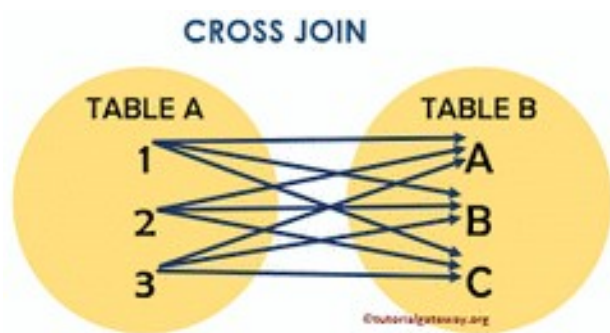
Output of the first command  
Select \* from student

Output of the second command  
Select \* from class

Both results combined

## Cross Join / Cartesian Product

The SQL **CROSS JOIN** produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table if no **WHERE** clause is used along with **CROSS JOIN**. This kind of result is called as **Cartesian Product**.



**Why Use Cross Join,**

**When to use Cross Join ?**

A **cross join** is used when you wish to create a combination of every row from two tables. All row combinations are included in the result; this is commonly called **cross product join**. A common **use** for a **cross join** is to create obtain all combinations of items,

The SQL CROSS JOIN produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table if no WHERE clause is used along with CROSS JOIN. This kind of result is called as Cartesian Product.

```
mysql> SELECT student.stu_id, class.stu_class FROM student CROSS JOIN class ;
```

```

+-----+-----+
| stu_id | stu_class |
+-----+-----+
| s111   | XI        |
| s005   | XI        |
| s004   | XI        |
| s003   | XI        |
| s002   | XI        |
| s001   | XI        |
| s111   | X         |
| s005   | X         |
| s004   | X         |
| s003   | X         |
| s002   | X         |
| s001   | X         |
| s111   | IX       |
| s005   | IX       |
| s004   | IX       |
| s003   | IX       |
| s002   | IX       |
| s001   | IX       |
| s111   | XI       |
| s005   | X        |
| s004   | X        |
| s003   | X        |
| s002   | X        |
| s001   | X        |
| s111   | IX       |
| s005   | IX       |
| s004   | IX       |
| s003   | IX       |
| s002   | IX       |
| s001   | IX       |
| s111   | IX       |
| s005   | IX       |
| s004   | IX       |
| s003   | IX       |
| s002   | IX       |
| s001   | IX       |
| s111   | V        |
| s005   | V        |
| s004   | V        |
| s003   | V        |
| s002   | V        |
| s001   | V        |
| s111   | XI       |
| s005   | XI       |
| s004   | XI       |
| s003   | XI       |
| s002   | XI       |
| s001   | XI       |
| s111   | X        |
| s005   | X        |
| s004   | X        |
| s003   | X        |
| s002   | X        |
| s001   | X        |
| s111   | IX       |
| s005   | IX       |
| s004   | IX       |
| s003   | IX       |
| s002   | IX       |
| s001   | IX       |
| s111   | IX       |
| s005   | IX       |
| s004   | IX       |
| s003   | IX       |
| s002   | IX       |
| s001   | IX       |
| s111   | V        |
| s005   | V        |
| s004   | V        |
| s003   | V        |
| s002   | V        |
| s001   | V        |
| s111   | XI       |
| s005   | XI       |
| s004   | XI       |
| s003   | XI       |
| s002   | XI       |
| s001   | XI       |
+-----+-----+

```

66 rows in set (0.00 sec)

In this case the cross join has returned us all the possible combinations :

But remember we use cross joins in real life such as for making all the possible combinations

in real life Eg

### Three cars

Alto  
Swift  
Ciaz

### 3 colours

Grey  
Red  
cherry

Will produce  $3*3=9$  combinations

car	colour
-----	--------

Alto	Grey
Swift	Grey
Ciaz	Grey
Alto	Red
Swift	Red
Ciaz	Red
Alto	Cherry
Swift	Cherry
Ciaz	Cherry