# **SQL Self Join**

**SQL Self Join:**

1. The SQL Self Join is used to join a table to itself as if the table were two tables. To carry this out, alias of the tables should be used at least once.
2. Self Join is a type of inner join, which is performed in cases where the comparison between two columns of a same table is required; probably to establish a relationship between them. In other words, a table is joined with itself when it contains both Foreign Key and Primary Key in it.
3. The syntax:  
   Here, the WHERE clause could be any given expression based on your requirement.

|  |
| --- |
| SELECT column\_name(s)  FROM table1 a, table1 b  WHERE a.common\_field = b.common\_field; |

1. The table used will be this one:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **NAME** | **AGE** | **ADDRESS** | **SALARY** |
| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |
| 2 | Khilan | 25 | Delhi | 1500.00 |
| 3 | Kaushik | 23 | Kota | 2000.00 |
| 4 | Chaitali | 25 | Mumbai | 6500.00 |
| 5 | Hardik | 27 | Bhopal | 8500.00 |
| 6 | Komal | 22 | Hyderabad | 4500.00 |
| 7 | Muffy | 24 | Indore | 10000.00 |

1. Now, let us join this table using the following Self Join query. Our aim is to establish a relationship among the said Customers on the basis of their earnings. We are doing this with the help of the WHERE clause.

|  |
| --- |
| SELECT a.ID, b.NAME as EARNS\_HIGHER, a.NAME  as EARNS\_LESS, a.SALARY as LOWER\_SALARY  FROM CUSTOMERS a, CUSTOMERS b  WHERE a.SALARY < b.SALARY; |

Output:

The resultant table displayed will list out all the customers that earn lesser than other customers −

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **EARNS\_HIGHER** | **EARNS\_LESS** | **LOWER\_SALARY** |
| 2 | Ramesh | Khilan | 1500.00 |
| 2 | Kaushik | Khilan | 1500.00 |
| 6 | Chaitali | Komal | 4500.00 |
| 3 | Chaitali | Kaushik | 2000.00 |
| 2 | Chaitali | Khilan | 1500.00 |
| 1 | Chaitali | Ramesh | 2000.00 |
| 6 | Hardik | Komal | 4500.00 |
| 4 | Hardik | Chaitali | 6500.00 |
| 3 | Hardik | Kaushik | 2000.00 |
| 2 | Hardik | Khilan | 1500.00 |
| 1 | Hardik | Ramesh | 2000.00 |
| 3 | Komal | Kaushik | 2000.00 |
| 2 | Komal | Khilan | 1500.00 |
| 1 | Komal | Ramesh | 2000.00 |
| 6 | Muffy | Komal | 4500.00 |
| 5 | Muffy | Hardik | 8500.00 |
| 4 | Muffy | Chaitali | 6500.00 |
| 3 | Muffy | Kaushik | 2000.00 |
| 2 | Muffy | Khilan | 1500.00 |
| 1 | Muffy | Ramesh | 2000.00 |

**Self Join with ORDER BY Clause:**

1. After joining a table with itself using self join, the records in the combined table can also be sorted in an order, using the ORDER BY clause.
2. The syntax:

|  |
| --- |
| SELECT column\_name(s)  FROM table1 a, table1 b  WHERE a.common\_field = b.common\_field  ORDER BY column\_name; |

1. Example:  
   Let us join the CUSTOMERS table with itself using self join on a WHERE clause; then, arrange the records in an ascending order using the ORDER BY clause with respect to a specified column, as shown in the following query.

|  |
| --- |
| SELECT a.ID, b.NAME as EARNS\_HIGHER, a.NAME  as EARNS\_LESS, a.SALARY as LOWER\_SALARY  FROM CUSTOMERS a, CUSTOMERS b  WHERE a.SALARY < b.SALARY  ORDER BY a.SALARY; |

Output:  
The resultant table is displayed as follows −

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **EARNS\_HIGHER** | **EARNS\_LESS** | **LOWER\_SALARY** |
| 2 | Ramesh | Khilan | 1500.00 |
| 2 | Kaushik | Khilan | 1500.00 |
| 2 | Chaitali | Khilan | 1500.00 |
| 2 | Hardik | Khilan | 1500.00 |
| 2 | Komal | Khilan | 1500.00 |
| 2 | Muffy | Khilan | 1500.00 |
| 3 | Chaitali | Kaushik | 2000.00 |
| 1 | Chaitali | Ramesh | 2000.00 |
| 3 | Hardik | Kaushik | 2000.00 |
| 1 | Hardik | Ramesh | 2000.00 |
| 3 | Komal | Kaushik | 2000.00 |
| 1 | Komal | Ramesh | 2000.00 |
| 3 | Muffy | Kaushik | 2000.00 |
| 1 | Muffy | Ramesh | 2000.00 |
| 6 | Chaitali | Komal | 4500.00 |
| 6 | Hardik | Komal | 4500.00 |
| 6 | Muffy | Komal | 4500.00 |
| 4 | Hardik | Chaitali | 6500.00 |
| 4 | Muffy | Chaitali | 6500.00 |
| 5 | Muffy | Hardik | 8500.00 |

Not just the salary column, the records can be sorted based on the alphabetical order of names, numerical order of Customer IDs etc.