**SQL FUNDAMENTALS**

### Order By Clause

SELECT statement returns records in an unspecified order. In case you want to retrieve data in alphabetical or numeric order, we use ORDER BY keyword.

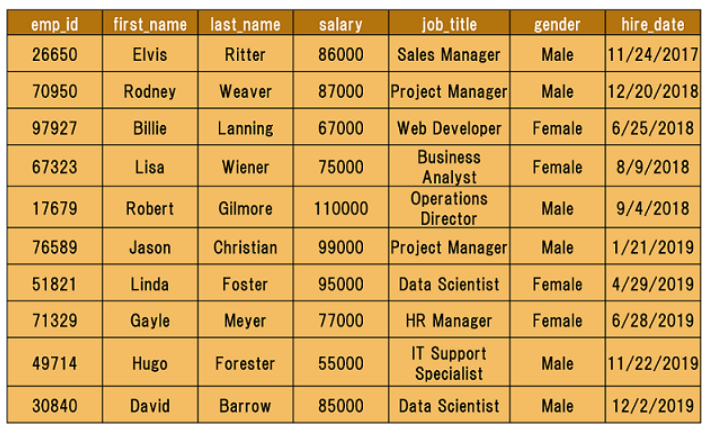
The ORDER BY keyword sorts the result-set in descending or ascending order.

By default ORDER BY keyword sorts the records in ascending order. Use the keyword DESC to sort the records in descending order. You can also use ASC to sort the data in ascending order. You have to use either of them.

Here is the syntax of ORDER BY:

SELECT column\_name(s) FROM table\_name ORDER BY column\_name(s) ASC|DESC;

Herein **"|"** symbol means "use either ASC or DESC". If you don't use any of them, the default value is ASC (ascending order).  
Let's see ORDER BY in an action.



Here is our employees table. I want to sort the *first\_name*column in alphabetical order (A-Z). This is the appropriate query:  
  
query :

SELECT \* FROM employees ORDER BY first\_name ASC;

After executing the query, we get the result table below. Our table is now sorted by the first names in ascending order. Not only the *first\_name* column is sorted, but also other columns are affected by the sort accordingly.

output :

emp\_id first\_name last\_name salary job\_title gender

    hire\_date

---------- ---------- ---------- ---------- ------------- ----------

    ----------

97927 Billie Lanning 67000 Web Developer Female 6/25

    /2018

30840 David Barrow 85000 Data Scientis Male 12/2

    /2019

26650 Elvis Ritter 86000 Sales Manager Male 11/24

    /2017

71329 Gayle Meyer 77000 HR Manager Female 6/28

    /2019

49714 Hugo Forester 55000 IT Support Sp Male 11/22

    /2019

76589 Jason Christian 99000 Project Manag Male 1/21

    /2019

51821 Linda Foster 95000 Data Scientis Female 4/29

    /2019

67323 Lisa Wiener 75000 Business Anal Female 8/9/2018

17679 Robert Gilmore 110000 Operations Di Male 9/4/2018

70950 Rodney Weaver 87000 Project Manag Male 12/20

    /2018

You could write the query which returns the same result table as below. Since ASC is the default order value in case you don't specify any ascending or descending order, both queries will yield the same result.

SELECT \* FROM employees ORDER BY first\_name;

Alright. Now it's time to put your theory into practice. Try to write as many queries as you want in the coding playground.

### Sorting in Descending Order

In the previous part, we sorted our table by the first names of the employees in ascending order. What if we asked you to sort it in descending order (Z-A)? You would probably say that "I use the **DESC** keyword." You're right. Let's write the query.

query :

SELECT \* FROM employees ORDER BY first\_name DESC;

Below is the query result. Other columns are also affected.  
  
output :

emp\_id first\_name last\_name salary job\_title gender

    hire\_date

---------- ---------- ---------- ---------- --------------- ----------

    ----------

70950 Rodney Weaver 87000 Project Manager Male 12/20

    /2018

17679 Robert Gilmore 110000 Operations Dire Male 9/4

    /2018

67323 Lisa Wiener 75000 Business Analys Female 8/9

    /2018

51821 Linda Foster 95000 Data Scientist Female 4/29

    /2019

76589 Jason Christian 99000 Project Manager Male 1/21

    /2019

49714 Hugo Forester 55000 IT Support Spec Male 11/22

    /2019

71329 Gayle Meyer 77000 HR Manager Female 6/28

    /2019

26650 Elvis Ritter 86000 Sales Manager Male 11/24

    /2017

30840 David Barrow 85000 Data Scientist Male 12/2

    /2019

97927 Billie Lanning 67000 Web Developer Female 6/25

    /2018

**💡 Tip:**When you sort the data, the original table's order is not affected. Remember from the previous lessons that a query returns a result table. Thus, we sort the result table, not the original one.

Until now, we've sorted the column containing textual data. It's time to sort numerical data in our table. You may wonder whose salary is the highest. Let's write the query. This time we don't want to retrieve all columns instead we want first name, last name and salary.  
  
query :

SELECT first\_name, last\_name, salary FROM employees ORDER BY salary DESC;

output :

first\_name last\_name salary

---------- ---------- ----------

Robert Gilmore 110000

Jason Christian 99000

Linda Foster 95000

Rodney Weaver 87000

Elvis Ritter 86000

David Barrow 85000

Gayle Meyer 77000

Lisa Wiener 75000

Billie Lanning 67000

Hugo Forester 55000

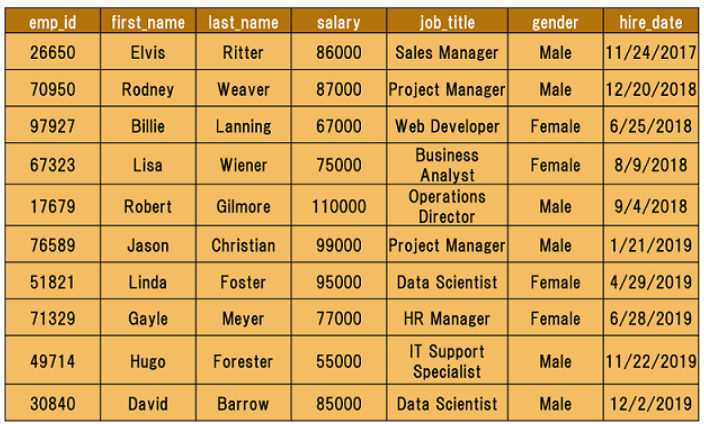
Robert Gilmore, Operations Director, has the highest salary. We sorted the result table from largest to smallest.

### Sorting By Multiple Columns

We are now able to sort by one column using the **ORDER BY**keyword. In some cases, we may need to sort our data by two columns or more. To do this, separate the columns by a comma. Here is the **syntax:**

SELECT column\_name(s) FROM table\_name ORDER BY column1 ASC|DESC, column2 ASC|DESC,

    columnN ASC|DESC;



Above is our original table. Let's sort it by gender in descending order.

query :

SELECT \* FROM employees ORDER BY gender DESC;

Here is the result-set:  
  
output :

emp\_id first\_name last\_name salary job\_title gender

    hire\_date

---------- ---------- ---------- ---------- ------------------- ----------

    ----------

17679 Robert Gilmore 110000 Operations Director Male 9

    /4/2018

26650 Elvis Ritter 86000 Sales Manager Male 11

    /24/2017

30840 David Barrow 85000 Data Scientist Male 12

    /2/2019

49714 Hugo Forester 55000 IT Support Speciali Male 11

    /22/2019

70950 Rodney Weaver 87000 Project Manager Male 12

    /20/2018

76589 Jason Christian 99000 Project Manager Male 1

    /21/2019

51821 Linda Foster 95000 Data Scientist Female 4

    /29/2019

67323 Lisa Wiener 75000 Business Analyst Female 8

    /9/2018

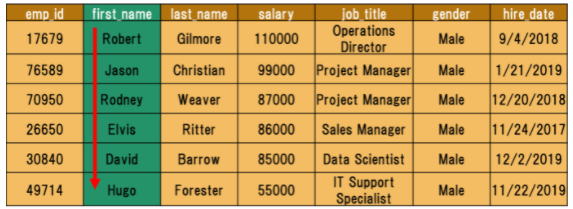
71329 Gayle Meyer 77000 HR Manager Female 6

    /28/2019

97927 Billie Lanning 67000 Web Developer Female 6

    /25/2018

In the above output, you can see that the result table is sorted in descending order according to the gender of the employees. Suppose that we also want to sort it by first names.



In this case, we have to append the first\_name column to our existing query. Use a comma to add the new column.  
  
query :

SELECT \* FROM employees ORDER BY gender DESC, first\_name ASC;

Here is the output:  
  
output :

emp\_id first\_name last\_name salary job\_title gender

    hire\_date

---------- ---------- ---------- ---------- -------------- ----------

    ----------

30840 David Barrow 85000 Data Scientist Male 12/2

    /2019

26650 Elvis Ritter 86000 Sales Manager Male 11/24

    /2017

49714 Hugo Forester 55000 IT Support Spe Male 11/22

    /2019

76589 Jason Christian 99000 Project Manage Male 1/21

    /2019

17679 Robert Gilmore 110000 Operations Dir Male 9/4

    /2018

70950 Rodney Weaver 87000 Project Manage Male 12/20

    /2018

97927 Billie Lanning 67000 Web Developer Female 6/25

    /2018

71329 Gayle Meyer 77000 HR Manager Female 6/28

    /2019

51821 Linda Foster 95000 Data Scientist Female 4/29

    /2019

67323 Lisa Wiener 75000 Business Analy Female 8/9

    /2018

In the above example, we first sorted the data by gender, then we sorted by first names.

### ORDER BY Clause with WHERE Clause

In this part, we will use ORDER BY with the WHERE clause.

This is the syntax:

SELECT column\_name(s) FROM table\_name WHERE condition ORDER BY column\_name(s)s

    ASC|DESC;

ORDER BY clause is placed after the WHERE clause.

Technically, any SQL statement can be written on a single line. However, it will become difficult to read when you start to write long queries. The solution in such cases is to organize the code, not just horizontally, but also vertically. This is called **beautifying.**Let's rewrite the syntax above.

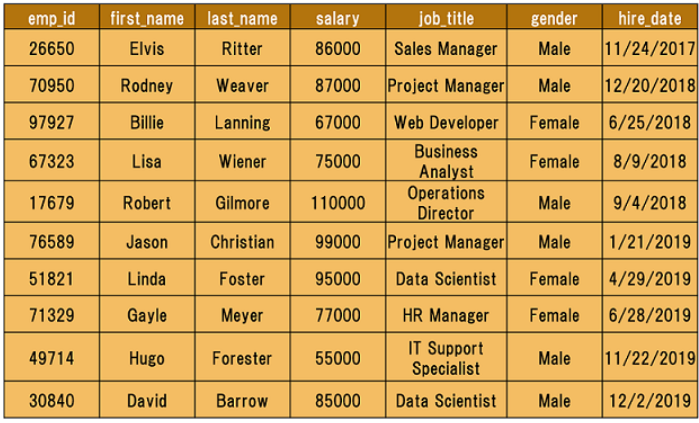
SELECT column\_name(s)

FROM table\_name

WHERE condition

ORDER BY column\_name(s)s ASC|DESC;

As you see that we put each clause on a separate line. Now let's continue with an example.



Assume that we try to find the employees whose salary is higher than $80,000. Next, we will sort it by first\_name in descending order.

Here is the query:

query :

SELECT \*

FROM employees

WHERE salary > 80000

ORDER BY first\_name DESC;

This is our result table:

output :

emp\_id first\_name last\_name salary job\_title gender

    hire\_date

---------- ---------- ---------- ---------- --------------- ----------

    ----------

70950 Rodney Weaver 87000 Project Manager Male 2018-12

    -20

17679 Robert Gilmore 110000 Operations Dire Male 2018-09

    -04

51821 Linda Foster 95000 Data Scientist Female 2019-04

    -29

76589 Jason Christian 99000 Project Manager Male 2019-01

    -21

26650 Elvis Ritter 86000 Sales Manager Male 2017-11

    -24

30840 David Barrow 85000 Data Scientist Male 2019-12

    -02

We first returned the employees whose salary is higher than $80,000. Next, we sorted this by the first names in descending order. Alright, we now know how to sort our table. It's time to put them into practice.