

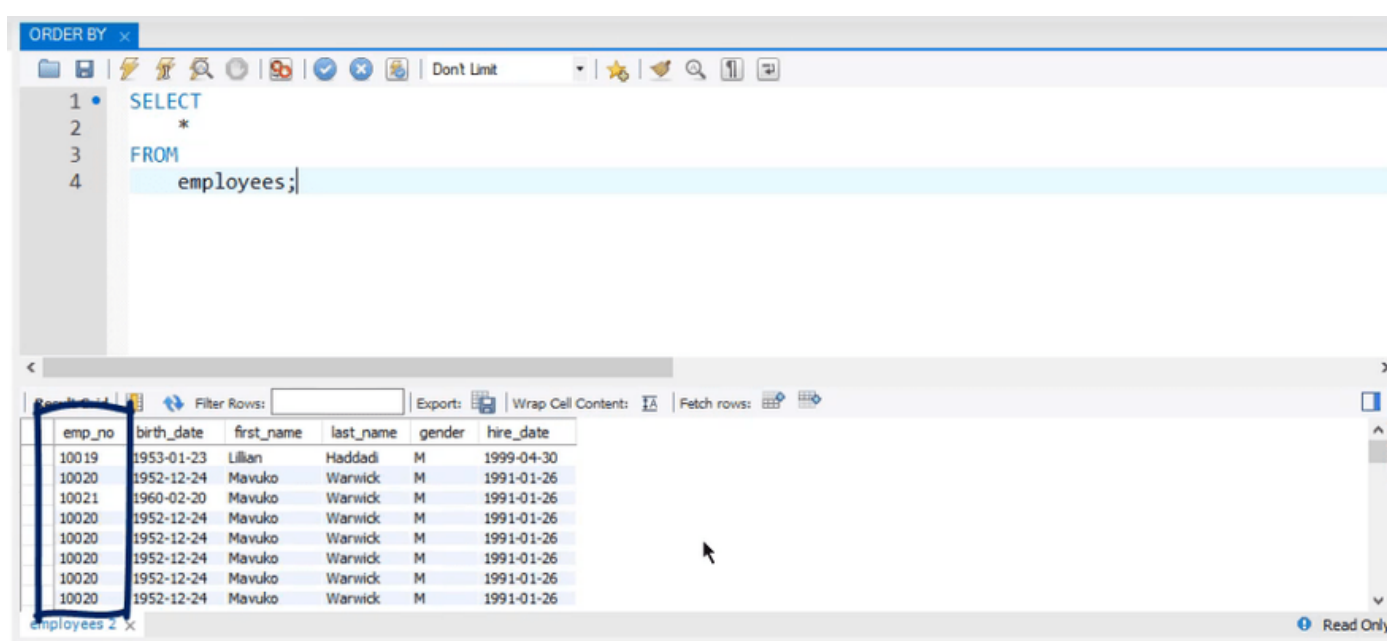
HOW TO USE

ORDER BY Clause in SQL



If you want to refine your output when coding, the SQL ORDER BY clause is an awesome way to go about it.

Let's see what happens when we retrieve all records from the “employees” table. As shown in the picture below, the list we get is automatically ordered based on employee numbers.



The screenshot shows a SQL IDE window titled "ORDER BY". The query editor contains the following SQL code:

```
1 SELECT
2 *
3 FROM
4 employees;
```

Below the query editor, the results are displayed in a table. The table has columns: emp_no, birth_date, first_name, last_name, gender, and hire_date. The results are ordered by employee number (emp_no).

emp_no	birth_date	first_name	last_name	gender	hire_date
10019	1953-01-23	Lillian	Haddadi	M	1999-04-30
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26
10021	1960-02-20	Mavuko	Warwick	M	1991-01-26
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26
10020	1952-12-24	Mavuko	Warwick	M	1991-01-26

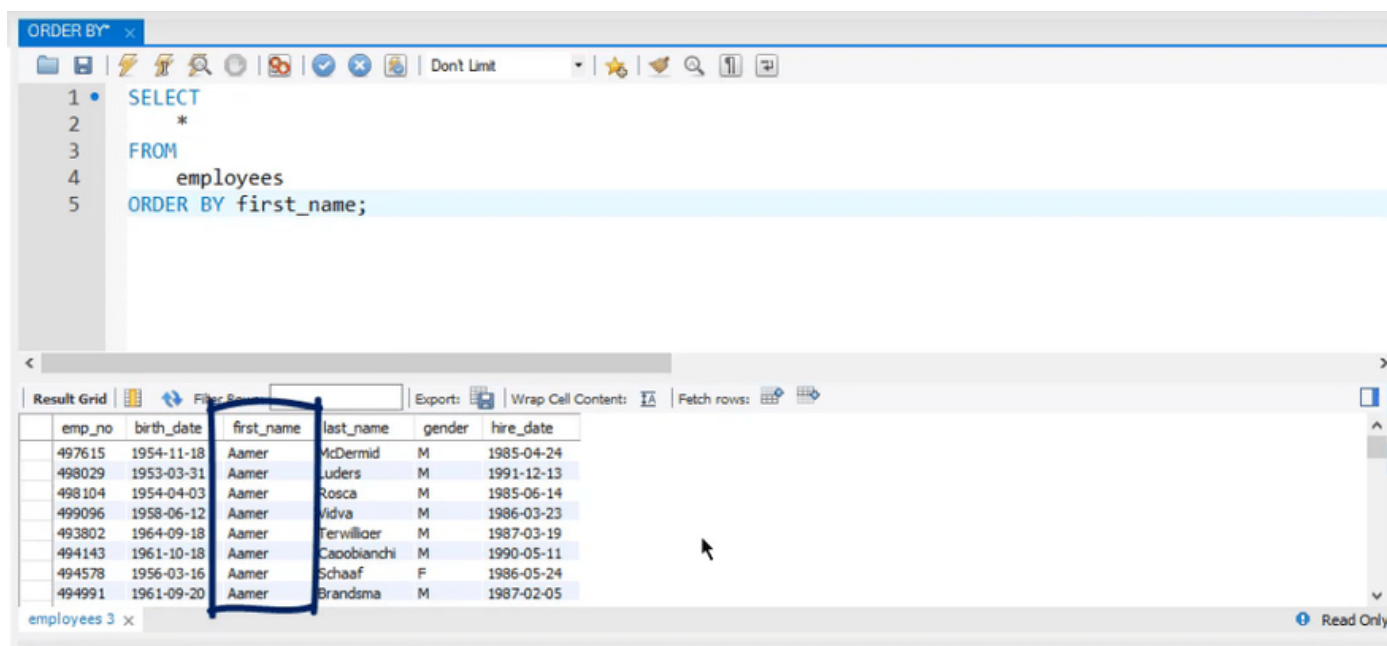
Assume your boss has just asked you to order the people by first name, instead of by employee number. So, how can you do that?



When to Use the SQL ORDER BY clause

Adding “ORDER BY first name” at the end of this query will provide the desired outcome.

```
SELECT
  *
FROM
  employees
ORDER BY first_name;
```



The screenshot shows a SQL IDE window titled "ORDER BY". The query editor contains the following SQL code:

```
1 SELECT
2   *
3 FROM
4   employees
5 ORDER BY first_name;
```

Below the query editor is a "Result Grid" showing the results of the query. The grid has columns: emp_no, birth_date, first_name, last_name, gender, and hire_date. The results are sorted alphabetically by first_name. A blue box highlights the first_name column in the result grid.

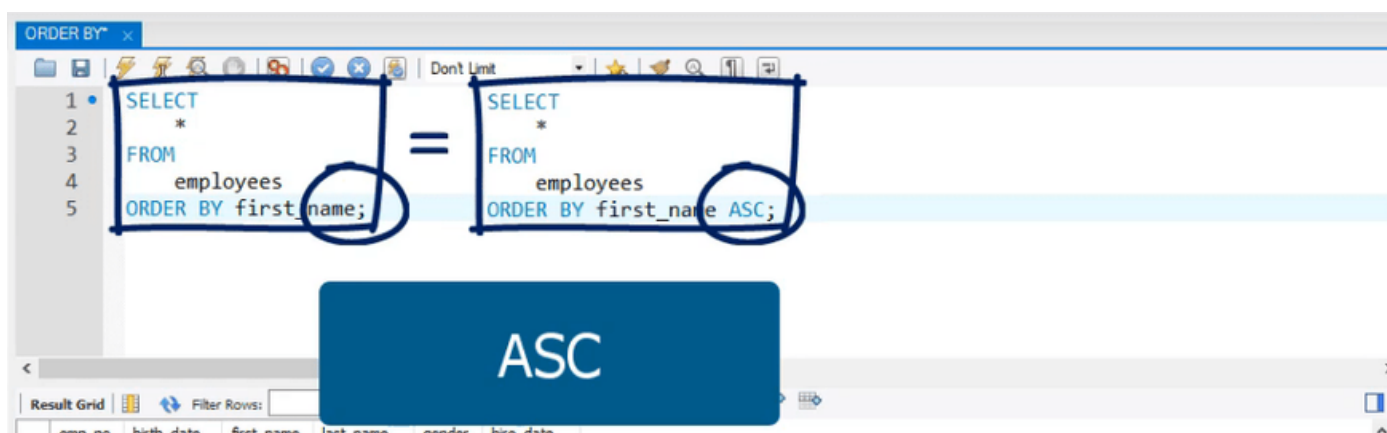
emp_no	birth_date	first_name	last_name	gender	hire_date
497615	1954-11-18	Aamer	McDermid	M	1985-04-24
498029	1953-03-31	Aamer	Luders	M	1991-12-13
498104	1954-04-03	Aamer	Rosca	M	1985-06-14
499096	1958-06-12	Aamer	Vidva	M	1986-03-23
493802	1964-09-18	Aamer	Terwilliger	M	1987-03-19
494143	1961-10-18	Aamer	Caoobianchi	M	1990-05-11
494578	1956-03-16	Aamer	Schaaf	F	1986-05-24
494991	1961-09-20	Aamer	Brandsma	M	1987-02-05

As you can see in the picture above, the entire list was reorganized in alphabetical order, according to the field containing employee names.



Ascending Order

The first one is ASC, abbreviated from “ascending”, requiring the output to be sorted by the values in the designated field in ascending order. If this keyword is not attached to the statement at the end, SQL will implicitly understand you want things ordered precisely in ascending order anyway.



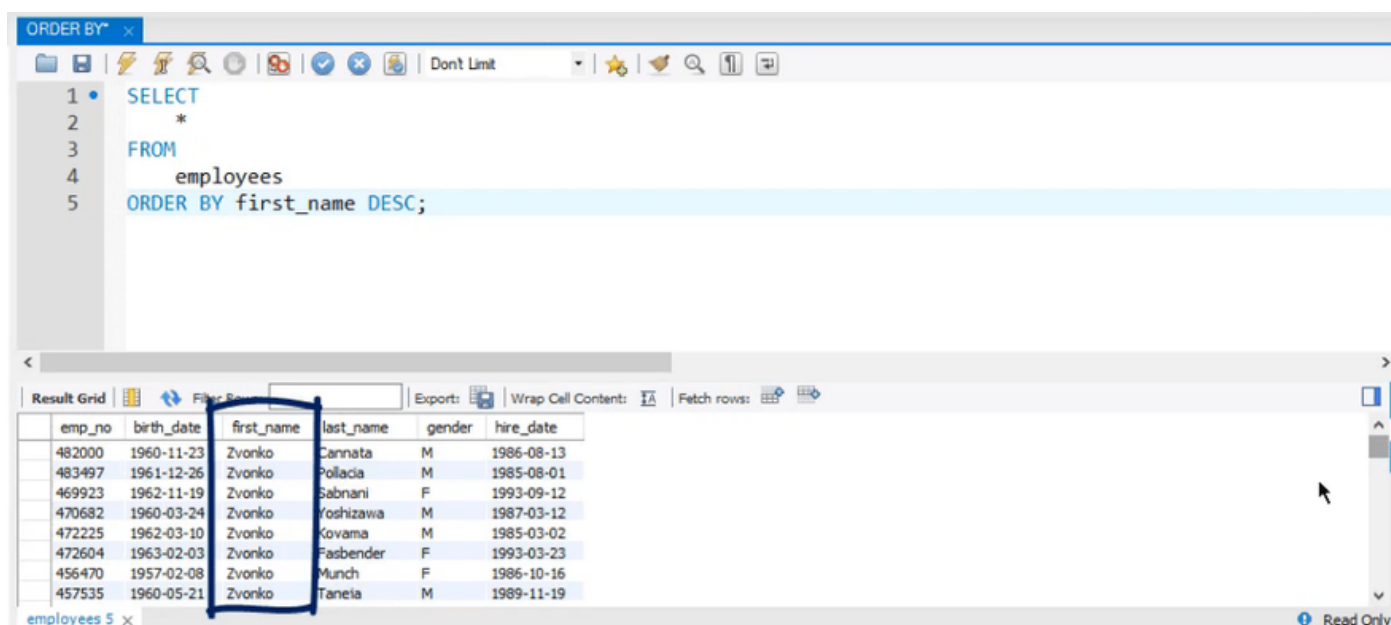
Therefore, if we add ASC at the end and re-run the query, we will obtain the same output.

Descending Order

The alternative is to use DESC, which is abbreviated from “descending”. Hence, if you would like your results plotted in reverse order, DESC is the keyword to add at the end of the ORDER BY clause.

Let’s check if this places the names starting with Z first.





ORDER BY

```

1 • SELECT
2   *
3 FROM
4   employees
5 ORDER BY first_name DESC;

```

Result Grid

emp_no	birth_date	first_name	last_name	gender	hire_date
482000	1960-11-23	Zvonko	Cannata	M	1986-08-13
483497	1961-12-26	Zvonko	Pollacia	M	1985-08-01
469923	1962-11-19	Zvonko	Sabnani	F	1993-09-12
470682	1960-03-24	Zvonko	Yoshizawa	M	1987-03-12
472225	1962-03-10	Zvonko	Kovama	M	1985-03-02
472604	1963-02-03	Zvonko	Fasbender	F	1993-03-23
456470	1957-02-08	Zvonko	Munch	F	1986-10-16
457535	1960-05-21	Zvonko	Taneia	M	1989-11-19

employees 5 x

Read Only

Working with Numbers

Now, the SQL ORDER BY clause does not only work for columns containing string values. It can handle numbers as well! Let's sort the list by employee number in descending order. Our query will look like this:

```

SELECT
  *
FROM
  employees
ORDER BY emp_no DESC;

```

This way, we start with a larger employee number that decreases as we scroll down!

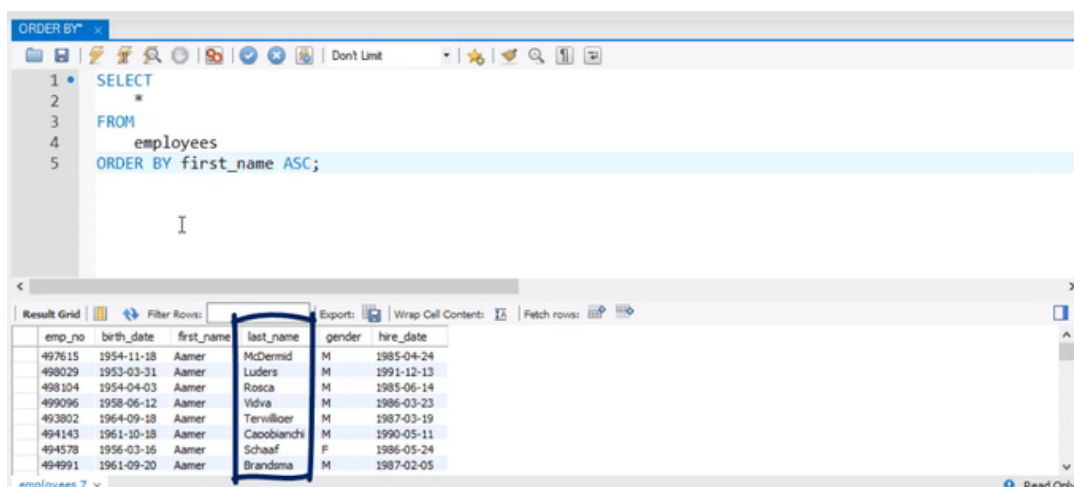
[linkedin.com/in/ileonjose](https://www.linkedin.com/in/ileonjose)



Ordering by More than One Field

Let's see another interesting feature of this clause. You can order your results by more than one field. For instance, we can order employees by first and last name.

To make the comparison, let's re-run the query where we sorted our output by first name in ascending order.



The screenshot shows a database query editor with a SQL query in the top pane and a result grid in the bottom pane. The query is:

```
1 SELECT
2 *
3 FROM
4 employees
5 ORDER BY first_name ASC;
```

The result grid displays the following data:

emp_no	birth_date	first_name	last_name	gender	hire_date
497615	1954-11-18	Aamer	McDermid	M	1985-04-24
498029	1953-03-31	Aamer	Luders	M	1991-12-13
498104	1954-04-03	Aamer	Rosca	M	1985-06-14
499096	1958-06-12	Aamer	Vidva	M	1986-03-23
493802	1964-09-18	Aamer	Terwilliger	M	1987-03-19
494143	1961-10-18	Aamer	Cacobiandhi	M	1990-05-11
494578	1956-03-16	Aamer	Schaaf	F	1986-05-24
494991	1961-09-20	Aamer	Brandma	M	1987-02-05

Now, look at the order of last names.

We can sort employees sharing the same first name according to their last name. To do this, we must simply type:

```
SELECT
  *
FROM
  employees
ORDER BY first_name, last_name ASC;
```

By writing this “, last_name”, we designated the second column of interest.

[linkedin.com/in/ileonjose](https://www.linkedin.com/in/ileonjose)



ORDER BY

```

1 SELECT
2 *
3 FROM
4 employees
5 ORDER BY first_name, last_name ASC;

```

Result Grid

emp_no	birth_date	first_name	last_name	gender	hire_date
69256	1962-04-14	Aamer	Anoer	M	1998-03-16
486584	1952-08-12	Aamer	Armand	M	1990-09-15
237165	1962-02-23	Aamer	Azevedo	F	1991-06-28
413688	1955-06-26	Aamer	Azuma	M	1989-12-10
281363	1956-05-18	Aamer	Bask	F	1994-03-10
242368	1959-07-26	Aamer	Baaleh	F	1989-08-06
206549	1960-02-24	Aamer	Baar	M	1995-06-12
259089	1963-06-08	Aamer	Baba	M	1995-02-02

Whate Are the Benefits of Ordering Data?

To sum up, the SQL ORDER BY clause is not mandatory. Rather, it is a helpful tool.

Whenever you need to sort your data in ascending or descending order, you can use it.

Otherwise, there's yet another extremely powerful tool that will allow you to retrieve more meaningful data by avoiding repetition - The GROUP BY clause.

I hope you got the clue about the next post!

[linkedin.com/in/ileonjose](https://www.linkedin.com/in/ileonjose)



HELPFUL? REPOST

[linkedin.com/in/ileonjose](https://www.linkedin.com/in/ileonjose)