





Office

Products

All Microsoft

Apps

Resources

Cart   
Search 

Install

Templates

Account

**Support**

Buy Office 365

Training

Admin

[Access](#) / [Queries](#) / [SQL syntax](#) / Access SQL: basic concepts, vocabulary, and syntax

# Access SQL: basic concepts, vocabulary, and syntax

*Access for Office 365, Access 2019, Access 2016, Access 2013, Access 2010,*

When you want to retrieve data from a database, you ask for the data by using Structured Query Language, or SQL. SQL is a computer language that closely resembles English, but that database programs understand. Every query that you run uses SQL behind the scenes.

Understanding how SQL works can help you create better queries, and can make it easier for you to understand how to fix a query that is not returning the results that you want.

This is one of a set of articles about Access SQL. This article describes the basic use of SQL to select data, and uses examples to illustrate SQL syntax.

## In this article

[What is SQL?](#)

[Basic SQL clauses: SELECT, FROM, and WHERE](#)

[Sorting the results: ORDER BY](#)

Was this information helpful?

Yes

No

×

## Combining query results: UNION

# What is SQL?

SQL is a computer language for working with sets of facts and the relationships between them. Relational database programs, such as Microsoft Office Access, use SQL to work with data. Unlike many computer languages, SQL is not difficult to read and understand, even for a novice. Like many computer languages, SQL is an international standard that is recognized by standards bodies such as [ISO](#) and [ANSI](#).

You use SQL to describe sets of data that can help you answer questions. When you use SQL, you must use the correct syntax. Syntax is the set of rules by which the elements of a language are correctly combined. SQL syntax is based on English syntax, and uses many of the same elements as Visual Basic for Applications (VBA) syntax.

For example, a simple SQL statement that retrieves a list of last names for contacts whose first name is Mary might resemble this:

```
SELECT Last_Name
FROM Contacts
WHERE First_Name = 'Mary';
```

**Note:** SQL is not only used for manipulating data, but also for creating and altering the design of database objects, such as tables. The part of SQL that is used for creating and altering database objects is called data-definition language (DDL). This topic does not cover DDL. For more information, see the article [Create or modify tables or indexes by using a data-definition query](#).

## SELECT statements

To describe a set of data by using SQL, you write a SELECT statement. A SELECT statement contains a complete description of a set of data that you want to obtain from a database. This includes the following:

- What tables contain the data.
- How data from different sources is related.

Was this information helpful?

Yes

No



- Criteria that data must match to be included.
- Whether and how to sort the results.

## SQL clauses

Like a sentence, a SQL statement has clauses. Each clause performs a function for the SQL statement. Some clauses are required in a SELECT statement. The following table lists the most common SQL clauses.

SQL clause	What it does	Required
SELECT	Lists the fields that contain data of interest.	Yes
FROM	Lists the tables that contain the fields listed in the SELECT clause.	Yes
WHERE	Specifies field criteria that must be met by each record to be included in the results.	No
ORDER BY	Specifies how to sort the results.	No
GROUP BY	In a SQL statement that contains aggregate functions, lists fields that are not summarized in the SELECT clause.	Only if there are such fields
HAVING	In a SQL statement that contains aggregate functions, specifies conditions that apply to fields that are summarized in the SELECT statement.	No

## SQL terms

Was this information helpful?

Yes

No



Each SQL clause is composed of terms — comparable to parts of speech. The following table lists types of SQL terms.

SQL term	Comparable part of speech	Definition	Example
identifier	noun	A name that you use to identify a database object, such as the name of a field.	Customers. [Phone Number]
operator	verb or adverb	A keyword that represents an action or modifies an action.	AS
constant	noun	A value that does not change, such as a number or NULL.	42
expression	adjective	A combination of identifiers, operators, constants, and functions that evaluates to a single value.	> = Products. [Unit Price]

[Top of Page](#)

## Basic SQL clauses: SELECT, FROM, and WHERE

A SQL statement takes the general form:

```
SELECT field_1
FROM table_1
WHERE criterion_1
;
```

### Notes:

Access ignores line breaks in a SQL statement. However, consider using a line for each

Was this information helpful?

Yes

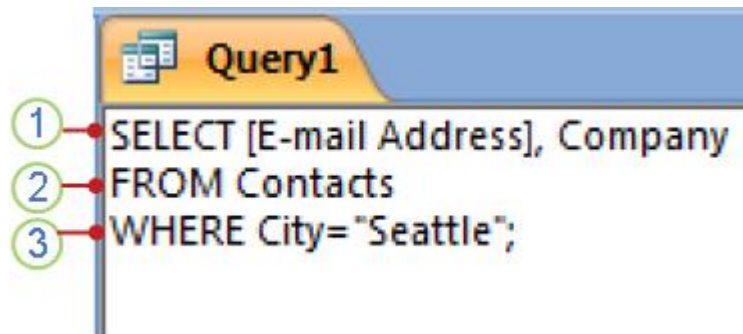
No

×

- Every SELECT statement ends with a semi-colon (;). The semi-colon can appear at the end of the last clause or on a line by itself at the end of the SQL statement.

## An example in Access

The following illustrates what a SQL statement for a simple select query might look like in Access:



1. SELECT clause
2. FROM clause
3. WHERE clause

This example SQL statement reads "Select the data that is stored in the fields named E-mail Address and Company from the table named Contacts, specifically those records in which the value of the field City is Seattle."

Let's look at the example, one clause at a time, to see how SQL syntax works.

## The SELECT clause

```
SELECT [E-mail Address], Company
```

This is the SELECT clause. It consists of an operator (SELECT) followed by two identifiers ([E-mail Address] and Company).

If an identifier contains spaces or special characters (such as "E-mail Address"), it must be enclosed in square brackets.

A SELECT clause does not have to say which tables contain the fields, and it cannot specify any

Was this information helpful?

Yes

No

×

The SELECT clause always appears in front of the FROM clause in a SELECT statement.

## The FROM clause

```
FROM Contacts
```

This is the FROM clause. It consists of an operator (FROM) followed by an identifier (Contacts).

A FROM clause does not list the fields to be selected.

## The WHERE clause

```
WHERE City="Seattle"
```

This is the WHERE clause. It consists of an operator (WHERE) followed by an expression (City="Seattle").

**Note:** Unlike the SELECT and FROM clauses, the WHERE clause is not a required element of a SELECT statement.

You can accomplish many of the actions that SQL enables you to do by using SELECT, FROM, and WHERE clauses. More information about how you use these clauses is presented in these additional articles:

- [Access SQL: SELECT clause](#)
- [Access SQL: FROM clause](#)
- [Access SQL: WHERE clause](#)

[Top of Page](#)

## Sorting the results: ORDER BY

Like Microsoft Excel, Access lets you sort query results in a datasheet. You can also specify in the query how you want to sort the results when the query is run by using an ORDER BY

Was this information helpful?

Yes

No

×

An ORDER BY clause contains a list of the fields that you want to use for sorting, in the same order that you want to apply the sort operations.

For example, suppose that you want your results sorted first by the value of the field Company in descending order, and — if there are records with the same value for Company — sorted next by the values in the field E-mail Address in ascending order. Your ORDER BY clause would resemble the following:

```
ORDER BY Company DESC, [E-mail Address]
```

**Note:** By default, Access sorts values in ascending order (A-Z, smallest to largest). Use the DESC keyword to sort values in descending order instead.

For more information about the ORDER BY clause, see the topic [ORDER BY Clause](#).

[Top of Page](#)

## Working with summarized data: GROUP BY and HAVING

Sometimes you want to work with summarized data, such as the total sales in a month, or the most expensive items in an inventory. To do this, you apply an aggregate function to a field in your SELECT clause. For example, if you want your query to show the count of e-mail addresses listed for each company, your SELECT clause might resemble the following:

```
SELECT COUNT([E-mail Address]), Company
```

The aggregate functions that you can use depend on the type of data that is in the field or expression that you want to use. For more information about the available aggregate functions, see the article [SQL Aggregate Functions](#).

## Specifying fields that are not used in an aggregate function: The GROUP BY clause

When you use aggregate functions, you usually must also create a GROUP BY clause. A GROUP BY clause lists all the fields to which you do not apply an aggregate function. If you

Was this information helpful?

Yes

No



A GROUP BY clause immediately follows the WHERE clause, or the FROM clause if there is no WHERE clause. A GROUP BY clause lists the fields as they appear in the SELECT clause.

For example, continuing the previous example, if your SELECT clause applies an aggregate function to [E-mail Address] but not to Company, your GROUP BY clause would resemble the following:

```
GROUP BY Company
```

For more information about the GROUP BY clause, see the topic [GROUP BY Clause](#).

## Limiting aggregate values by using group criteria: the HAVING clause

If you want to use criteria to limit your results, but the field that you want to apply criteria to is used in an aggregate function, you cannot use a WHERE clause. Instead, you use a HAVING clause. A HAVING clause works like a WHERE clause, but is used for aggregated data.

For example, suppose that you use the AVG function (which calculates an average value) with the first field in your SELECT clause:

```
SELECT COUNT([E-mail Address]), Company
```

If you want the query to restrict the results based on the value of that COUNT function, you cannot use a criteria for that field in the WHERE clause. Instead, you put the criteria in a HAVING clause. For example, if you only want the query to return rows if there are more than one e-mail addresses associated with the company, the HAVING clause might resemble the following:

```
HAVING COUNT([E-mail Address])>1
```

**Note:** A query can have a WHERE clause and a HAVING clause — criteria for fields that are not used in an aggregate function go in the WHERE clause, and criteria for fields that are used with aggregate functions go in the HAVING clause.

For more information about the HAVING clause, see the topic [HAVING Clause](#).

[Top of Page](#)

Was this information helpful?

Yes

No

×



# Combining query results: UNION

When you want to review all the data that is returned by several similar select queries together, as a combined set, you use the UNION operator.

The UNION operator lets you combine two SELECT statements into one. The SELECT statements that you combine must have the same number of output fields, in the same order, and with the same or compatible data types. When you run the query, data from each set of corresponding fields is combined into one output field, so that the query output has the same number of fields as each of the select statements.

**Note:** For the purposes of a union query, the Number and Text data types are compatible.

When you use the UNION operator, you can also specify whether the query results should include duplicate rows, if any exist, by using the ALL key word.

The basic SQL syntax for a union query that combines two SELECT statements is as follows:

```
SELECT field_1
FROM table_1
UNION [ALL]
SELECT field_a
FROM table_a
;
```

For example, suppose that you have a table named Products and another table named Services. Both tables have fields that contain the name of the product or service, the price, warranty or guarantee availability, and whether you offer the product or service exclusively. Although the Products table stores warranty information, and the Services table stores guarantee information, the basic information is the same (whether a particular product or service includes a promise of quality). You can use a union query, such as the following, to combine the four fields from the two tables:

```
SELECT name, price, warranty_available, exclusive_offer
FROM Products
UNION ALL
SELECT name, price, guarantee_available, exclusive_offer
FROM Services
;
```

For more information about how to combine SELECT statements by using the UNION operator, see [Combine the results of several select queries by using a union query](#).

Was this information helpful?

Yes

No





Expand your Office skills

[EXPLORE TRAINING >](#)

Get new features first

[JOIN OFFICE INSIDERS >](#)

What's new	Microsoft Store	Education	Enterprise	Developer	Company
Surface Pro 6	Account profile	Microsoft in education	Azure	Microsoft Visual Studio	Careers
Surface Laptop 2	Download Center	Office for students	AppSource	Windows Dev Center	About Microsoft
Surface Go	Microsoft Store support	Office 365 for schools	Automotive	Developer Network	Company news
Xbox One X	Returns	Deals for students & parents	Government	TechNet	Privacy at Microsoft
Xbox One S	Order tracking	Microsoft Azure in education	Healthcare	Microsoft developer program	Investors
VR & mixed reality	Store locations		Manufacturing	Channel 9	Diversity and inclusion
Windows 10 apps	Buy online, pick up in store		Financial services	Office Dev Center	Accessibility
Office apps			Retail	Microsoft Garage	Security



English (United States)

[Contact Us](#)

[Privacy & Cookies](#)

[Terms of use & sale](#)

[Trademarks](#)

[Office accessibility](#)

[Legal](#)

© Microsoft 2019

Was this information helpful?

Yes

No

