You can control the sort order of SQL query results using the ORDER BY clause. When sorting on a numeric column, the resulting order typically makes intuitive sense, but when sorting on a string column, you might be surprised by the resulting order. This is especially true when the strings include numbers, or a mix of numbers and letters or other characters within a value.

Unfortunately, there isn't a simple explanation to tell you how SQL will sort your results, because it depends on what *collation* you are using.

> A DBMS uses a *collating sequence,* or *collation,* to determine the order in which characters are sorted. The collation defines the order of precedence for every character in your character set. Your character set depends on the language that you're using—European languages (a Latin character set), Hebrew (the Hebrew alphabet), or Chinese (ideographs), for example. The collation also determines case sensitivity (is 'A' < 'a'?), accent sensitivity (is 'A' < 'À' ?), width sensitivity (for multibyte or Unicode characters), and other factors such as linguistic practices. The SQL standard doesn't define particular collations and character sets, so each DBMS uses its own sorting strategy and default collation... Search your DBMS documentation for *collation* or *sort order*. (1)

Collations have different options associated with them, and many can be customized depending on the system you are using. For English, *case sensitivity* is a major one to consider—should "A" and "a" be considered the same character for the purposes of ordering? Others include *accent sensitivity* (for example, should "a" and "á" be considered the same), *Kana sensitivity* (which distinguishes between the two types of Japanese characters), and *script order* (for example, which should be ordered first: Hebrew, Greek, or Cyrillic). See "Customization" (2) and "Collation" (3) for more examples of these and other options.

When using Unicode—an industry standard that assigns a number to each character or symbol— SQL will most likely follow the Unicode ordering to distinguish the order of two characters, while taking customizations into account. Non-Unicode data may have a different order:

> When you use a SQL collation you might see different results for comparisons of the same characters, depending on the underlying data type. For example, if you are using the SQL collation "SQL_Latin1_General_CP1_CI_AS", the non-Unicode string 'a-c' is less than the string 'ab' because the hyphen ("-") is sorted as a separate character that comes before "b". However, if you convert these strings to Unicode and you perform the same comparison, the Unicode string N'a-c' is considered to be greater than N'ab' because the Unicode sorting rules use a "word sort" that ignores the hyphen. (4)