**Collation and Unicode Support**

**SQL Server 2012**

[Other Versions](javascript:;)

Description: http://i.technet.microsoft.com/Areas/Epx/Content/Images/ImageSprite.png

* [SQL Server 2008 R2](http://technet.microsoft.com/en-us/library/ms143726(d=printer,v=sql.105).aspx)
* [SQL Server 2008](http://technet.microsoft.com/en-us/library/ms143726(d=printer,v=sql.100).aspx)
* [SQL Server 2005](http://technet.microsoft.com/en-us/library/ms143726(d=printer,v=sql.90).aspx)

Collations in SQL Server provide sorting rules, case, and accent sensitivity properties for your data. Collations that are used with character data types such as char and varchar dictate the code page and corresponding characters that can be represented for that data type. Whether you are installing a new instance of SQL Server, restoring a database backup, or connecting server to client databases, it is important that you understand the locale requirements, sorting order, and case and accent sensitivity of the data that you will be working with.

When you select a collation for your server, database, column, or expression, you are assigning certain characteristics to your data that will affect the results of many operations in the database. For example, when you construct a query by using ORDER BY, the sort order of your result set might depend on the collation that is applied to the database or dictated in a COLLATE clause at the expression level of the query.

To best use collation support in SQL Server, you must understand the terms that are defined in this topic, and how they relate to the characteristics of your data.

[In This Topic](javascript:void(0))

[Collation Terms](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Terms)

[Unicode Support](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Unicode_Defn)

[Supplementary Characters](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Supplementary_Characters)

[GB18030 Support](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#GB18030)

[Complex Script Support](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Complex_script)

[Related Tasks](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Related_Tasks)

[Related Content](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Related_Content)

[Collation Terms](javascript:void(0))

* [Collation](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Collation_Defn)
* [Locale](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Locale_Defn)
* [Code page](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Code_Page_Defn)
* [Sort order](http://technet.microsoft.com/en-us/library/ms143726(d=printer).aspx#Sort_Order_Defn)

**Collation**

A collation specifies the bit patterns that represent each character in a data set. Collations also determine the rules that sort and compare data. SQL Server supports storing objects that have different collations in a single database. For non-Unicode columns, the collation setting specifies the code page for the data and which characters can be represented. Data that is moved between non-Unicode columns must be converted from the source code page to the destination code page.

Transact-SQL statement results can vary when the statement is run in the context of different databases that have different collation settings. If it is possible, use a standardized collation for your organization. This way, you do not have to explicitly specify the collation in every character or Unicode expression. If you must work with objects that have different collation and code page settings, code your queries to consider the rules of collation precedence. For more information, see [Collation Precedence (Transact-SQL)](http://technet.microsoft.com/en-us/library/ms179886.aspx).

The options associated with a collation are case sensitivity, accent sensitivity, Kana-sensitivity, width sensitivity. These options are specified by appending them to the collation name. For example, this collation Japanese\_Bushu\_Kakusu\_100\_CS\_AS\_KS\_WS is case-sensitive, accent-sensitive, Kana-sensitive, and width-sensitive. The following table describes the behavior associated with these options.

|  |  |
| --- | --- |
| **Option** | **Description** |
| Case-sensitive (\_CS) | Distinguishes between uppercase and lowercase letters. If selected, lowercase letters sort ahead of their uppercase versions. If this option is not selected, the collation will be case-insensitive. That is, SQL Server considers the uppercase and lowercase versions of letters to be identical for sorting purposes. You can explicitly select case insensitivity by specifying \_CI. |
| Accent-sensitive (\_AS) | Distinguishes between accented and unaccented characters. For example, 'a' is not equal to 'ấ'. If this option is not selected, the collation will be accent-insensitive. That is, SQL Server considers the accented and unaccented versions of letters to be identical for sorting purposes. You can explicitly select accent insensitivity by specifying \_AI. |
| Kana-sensitive (\_KS) | Distinguishes between the two types of Japanese kana characters: Hiragana and Katakana.If this option is not selected, the collation is Kana-insensitive. That is, SQL Server considers Hiragana and Katakana characters to be equal for sorting purposes. Omitting this option is the only method of specifying Kana-insensitivity. |
| Width-sensitive (\_WS) | Distinguishes between a single-byte character and the same character when represented as a double-byte character. If this option is not selected, SQL Server considers the single-byte and double-byte representation of the same character to be identical for sorting purposes. Omitting this option is the only method of specifying width-insensitivity. |

SQL Server supports the following collation sets:

Windows collations

Windows collations define rules for storing character data that are based on an associated Windows system locale. For a Windows collation, comparison of non-Unicode data is implemented by using the same algorithm as Unicode data. The base Windows collation rules specify which alphabet or language is used when dictionary sorting is applied, and the code page that is used to store non-Unicode character data. Both Unicode and non-Unicode sorting are compatible with string comparisons in a particular version of Windows. This provides consistency across data types within SQL Server, and it also lets developers sort strings in their applications by using the same rules that are used by SQL Server. For more information, see [Windows Collation Name (Transact-SQL)](http://technet.microsoft.com/en-us/library/ms188046.aspx).

Binary collations

Binary collations sort data based on the sequence of coded values that are defined by the locale and data type. They are case sensitive. A binary collation in SQL Server defines the locale and the ANSI code page that will be used. This enforces a binary sort order. Because they are relatively simple, binary collations help improve application performance. For non-Unicode data types, data comparisons are based on the code points that are defined in the ANSI code page. For Unicode data types, data comparisons are based on the Unicode code points. For binary collations on Unicode data types, the locale is not considered in data sorts. For example, Latin\_1\_General\_BIN and Japanese\_BIN yield identical sorting results when they are used on Unicode data.

Binary collations in earlier versions of SQL Server compared the first character as WCHAR, followed by a byte-by-byte comparison. For backward compatibility reasons, existing binary collation semantics will not be changed.

Binary collations in this release of SQL Server include a set of pure code-point comparison collations. You can migrate to the binary collations to take advantage of true code-point comparisons. You should use binary collations to develop new applications. The BIN2 suffix identifies collation names that implement the new code-point collation semantics. There is also a comparison flag that corresponds to BIN2 for the new binary sort. Advantages include simpler application development and clearer semantics.

SQL Server collations

SQL Server collations (SQL\_\*) provide sort order compatibility with earlier versions of SQL Server. The dictionary sorting rules for non-Unicode data are incompatible with any sorting routine that is provided by Windows operating systems. However, sorting Unicode data is compatible with a particular version of Windows sorting rules. Because SQL Server collations use different comparison rules for non-Unicode and Unicode data, you will see different results for comparisons of the same data, depending on the underlying data type. For more information, see [SQL Server Collation Name (Transact-SQL)](http://technet.microsoft.com/en-us/library/ms180175.aspx).

|  |
| --- |
| **Note** |
| When you upgrade an English-language instance of SQL Server, SQL Server collations (SQL\_\*) can be specified for compatibility with existing instances of SQL Server. Because the default collation for an instance of SQL Server is defined during setup, make sure that you specify collation settings carefully when the following are true: |

* Your application code depends on the behavior of previous SQL Server collations.
* You must store character data that reflects multiple languages.

Setting collations are supported at the following levels of an instance of SQL Server:

Server-level collations

The default server collation is set during SQL Server setup, and also becomes the default collation of the system databases and all user databases. Note that Unicode-only collations cannot be selected during SQL Server setup because they are not supported as server-level collations.

After a collation has been assigned to the server, you cannot change the collation except by exporting all database objects and data, rebuilding the master database, and importing all database objects and data. Instead of changing the default collation of an instance of SQL Server, you can specify the desired collation at the time that you create a new database or database column.

Database-level collations

When a database is created or modified, you can use the COLLATE clause of the CREATE DATABASE or ALTER DATABASE statement to specify the default database collation. If no collation is specified, the database is assigned the server collation.

You cannot change the collation of system databases except by changing the collation for the server.

The database collation is used for all metadata in the database, and is the default for all string columns, temporary objects, variable names, and any other strings used in the database. When you change the collation of a user database, know that there is the potential for collation conflicts when queries in the database access temporary tables. Temporary tables are always stored in the tempdb system database, which will use the collation for the instance. Queries that compare character data between the user database and tempdb may fail if the collations cause a conflict in evaluating the character data. You can resolve this by specifying the COLLATE clause in the query. For more information, see [COLLATE (Transact-SQL)](http://technet.microsoft.com/en-us/library/ms184391.aspx).

Column-level collations

When you create or alter a table, you can specify collations for each character-string column by using the COLLATE clause. If no collation is specified, the column is assigned the default collation of the database.

Expression-level collations

Expression-level collations are set when a statement is run, and they affect the way a result set is returned. This enables ORDER BY sort results to be locale-specific. Use a COLLATE clause such as the following to implement expression-level collations:

SELECT name FROM customer ORDER BY name COLLATE Latin1\_General\_CS\_AI;

**Locale**

A locale is a set of information that is associated with a location or a culture. This can include the name and identifier of the spoken language, the script that is used to write the language, and cultural conventions. Collations can be associated with one or more locales. For more information, see [Locale IDs Assigned by Microsoft](http://msdn.microsoft.com/en-us/goglobal/bb964664.aspx).

**Code Page**

A code page is an ordered set of characters of a given script in which a numeric index, or code point value, is associated with each character. A Windows code page is typically referred to as a character set or charset. Code pages are used to provide support for the character sets and keyboard layouts that are used by different Windows system locales.

**Sort Order**

Sort order specifies how data values are sorted. This affects the results of data comparison. Data is sorted by using collations, and it can be optimized by using indexes.

[Unicode Support](javascript:void(0))

Unicode is a standard for mapping code points to characters. Because it is designed to cover all the characters of all the languages of the world, there is no need for different code pages to handle different sets of characters. If you store character data that reflects multiple languages, always use Unicode data types (nchar, nvarchar, and ntext) instead of the non-Unicode data types (char, varchar, and text).

Significant limitations are associated with non-Unicode data types. This is because a non-Unicode computer will be limited to use of a single code page. You might experience performance gain by using Unicode because fewer code-page conversions are required. Unicode collations must be selected individually at the database, column or expression level because they are not supported at the server level.

The code pages that a client uses are determined by the operating system settings. To set client code pages on the Windows operating system, use Regional Settings in Control Panel.

When you move data from a server to a client, your server collation might not be recognized by older client drivers. This can occur when you move data from a Unicode server to a non-Unicode client. Your best option might be to upgrade the client operating system so that the underlying system collations are updated. If the client has database client software installed, you might consider applying a service update to the database client software.

You can also try to use a different collation for the data on the server. Choose a collation that will map to a code page on the client.

To use the UTF-16 collations available in SQL Server 2012, you can select one of the supplementary characters \_SC collations to improve searching and sorting of some Unicode characters.

To evaluate issues that are related to using Unicode or non-Unicode data types, test your scenario to measure performance differences in your environment. It is a good practice to standardize the collation that is used on systems across your organization, and deploy Unicode servers and clients wherever possible.

In many situations, SQL Server will interact with other servers or clients, and your organization might use multiple data access standards between applications and server instances. SQL Server clients are one of two main types:

* Unicode clients that use OLE DB and Open Database Connectivity (ODBC) version 3.7 or a later version.
* Non-Unicode clients that use DB-Library and ODBC version 3.6 or an earlier version.

The following table provides information about using multilingual data with various combinations of Unicode and non-Unicode servers.

|  |  |  |
| --- | --- | --- |
| **Server** | **Client** | **Benefits or Limitations** |
| Unicode | Unicode | Because Unicode data will be used throughout the system, this scenario provides the best performance and protection from corruption of retrieved data. This is the situation with ActiveX Data Objects (ADO), OLE DB, and ODBC version 3.7 or a later version. |
| Unicode | Non-Unicode | In this scenario, especially with connections between a server that is running a newer operating system and a client that is running an older version of SQL Server, or on an older operating system, there can be limitations or errors when you move data to a client computer. Unicode data on the server will try to map to a corresponding code page on the non-Unicode client to convert the data. |
| Non-Unicode | Unicode | This is not an ideal configuration for using multilingual data. You cannot write Unicode data to the non-Unicode server. Problems are likely to occur when data is sent to servers that are outside the server's code page. |
| Non-Unicode | Non-Unicode | This is a very limiting scenario for multilingual data. You can use only a single code page. |

[Supplementary Characters](javascript:void(0))

SQL Server provides data types such as nchar and nvarchar to store Unicode data. These data types encode text in a format called UTF-16. The Unicode Consortium allocates each character a unique codepoint, which is a value in the range 0x0000 to 0x10FFFF. The most frequently used characters have codepoint values that will fit into a 16-bit word in memory and on disk, but characters with codepoint values larger than 0xFFFF require two consecutive 16-bit words. These characters are called supplementary characters, and the two consecutive 16-bit words are called surrogate pairs.

If you use supplementary characters:

* Supplementary characters can be used in ordering and comparison operations in collation versions 90 or greater.
* All \_100 level collations support linguistic sorting with supplementary characters.
* Supplementary characters are not supported for use in metadata, such as in names of database objects.
* Introduced in SQL Server 2012, a new family of supplementary character (SC) collations can be used with the data types nchar, nvarchar and sql\_variant. For example: Latin1\_General\_100\_CI\_AS\_SC, or if using a Japanese collation, Japanese\_Bushu\_Kakusu\_100\_CI\_AS\_SC.

The SC flag can be applied to:

* + Version 90 Windows collations
  + Version 100 Windows collations

The SC flag cannot be applied to:

* + Version 80 non-versioned Windows collations
  + The BIN or BIN2 binary collations
  + The SQL\* collations

The following table compares the behavior of some string functions and string operators when they use supplementary characters with and without a SC collation.

|  |  |  |
| --- | --- | --- |
| **String Function or Operator** | **With an SC Collation** | **Without an SC Collation** |
| [CHARINDEX](http://technet.microsoft.com/en-us/library/ms186323.aspx)  [LEN](http://technet.microsoft.com/en-us/library/ms190329.aspx)  [PATINDEX](http://technet.microsoft.com/en-us/library/ms188395.aspx) | The UTF-16 surrogate pair is counted as a single codepoint. | The UTF-16 surrogate pair is counted as two codepoints. |
| [LEFT](http://technet.microsoft.com/en-us/library/ms177601.aspx)  [REPLACE](http://technet.microsoft.com/en-us/library/ms186862.aspx)  [REVERSE](http://technet.microsoft.com/en-us/library/ms180040.aspx)  [RIGHT](http://technet.microsoft.com/en-us/library/ms177532.aspx)  [SUBSTRING](http://technet.microsoft.com/en-us/library/ms187748.aspx)  [STUFF](http://technet.microsoft.com/en-us/library/ms188043.aspx) | These functions treat each surrogate pair as a single codepoint and work as expected. | These functions may split any surrogate pairs and lead to unexpected results. |
| [NCHAR](http://technet.microsoft.com/en-us/library/ms182673.aspx) | Returns the character corresponding to the specified Unicode codepoint value in the range 0 to 0x10FFFF. If the value specified lies in the range 0 through 0xFFFF, one character is returned. For higher values, the corresponding surrogate is returned. | A value higher than 0xFFFF returns NULL instead of the corresponding surrogate. |
| [UNICODE](http://technet.microsoft.com/en-us/library/ms180059.aspx) | Returns a UTF-16 codepoint in the range 0 through 0x10FFFF. | Returns a UCS-2 codepoint in the range 0 through 0xFFFF. |
| [Match One Character Wildcard](http://technet.microsoft.com/en-us/library/ms174424.aspx)  [Wildcard - Character(s) Not to Match](http://technet.microsoft.com/en-us/library/ms188342.aspx) | Supplementary characters are supported for all wildcard operations. | Supplementary characters are not supported for these wildcard operations. Other wildcard operators are supported. |

[GB18030 Support](javascript:void(0))

GB18030 is a separate standard used in the People's Republic of China for encoding Chinese characters. In GB18030, characters can be 1, 2, or 4 bytes in length. SQL Server provides support for GB18030-encoded characters by recognizing them when they enter the server from a client-side application and converting and storing them natively as Unicode characters. After they are stored in the server, they are treated as Unicode characters in any subsequent operations. You can use any Chinese collation, preferably the latest 100 version. All \_100 level collations support linguistic sorting with GB18030 characters. If the data includes supplementary characters (surrogate pairs), you can use the SC collations available in SQL Server 2012 to improve searching and sorting.

[Complex Script Support](javascript:void(0))

SQL Server can support inputting, storing, changing, and displaying complex scripts. Complex scripts include the following:

* Scripts that include the combination of both right-to-left and left-to-right text, such as a combination of Arabic and English text.
* Scripts whose characters change shape depending on their position, or when combined with other characters, such as Arabic, Indic, and Thai characters.
* Languages such as Thai that require internal dictionaries to recognize words because there are no breaks between them.

Database applications that interact with SQL Server must use controls that support complex scripts. Standard Windows form controls that are created in managed code are complex script-enabled.

[Related Tasks](javascript:void(0))

|  |  |
| --- | --- |
| **Task** | **Topic** |
| Describes how to set or change the collation of the instance of SQL Server. | [Set or Change the Server Collation](http://technet.microsoft.com/en-us/library/ms179254.aspx) |
| Describes how to set or change the collation of a user database. | [Set or Change the Database Collation](http://technet.microsoft.com/en-us/library/ms175835.aspx) |
| Describes how to set or change the collation of a column in the database. | [Set or Change the Column Collation](http://technet.microsoft.com/en-us/library/ms190920.aspx) |
| Describes how to return collation information at the server, database, or column level. | [View Collation Information](http://technet.microsoft.com/en-us/library/hh230914.aspx) |
| Describes how to write Transact-SQL statements that will make them more portable from one language to another, or support multiple languages more easily. | [Write International Transact-SQL Statements](http://technet.microsoft.com/en-us/library/ms191307.aspx) |
| Describes how to change the language of error messages and preferences for how date, time, and currency data are used and displayed. | [Set a Session Language](http://technet.microsoft.com/en-us/library/ms142797.aspx) |