# **An overview of SQL Server data types**

## Introduction

In SQL Server, a data type defines the type of data in a table column or variable. It is a mandatory and essential step in designing a table. A table with inappropriate data types can lead to various issues such as improper query optimization, performance issues, data truncation.

To understand SQL Server data types, let’s look at the following page to create a new Google account (for reference purpose only):

It requires the following inputs:

* First name and Last name: Suppose we require only alphabets in these fields
* Username: We can use letter, numbers and periods in the user name
* Password: We can use alphabets, numbers, and special characters

We can consider a few more examples:

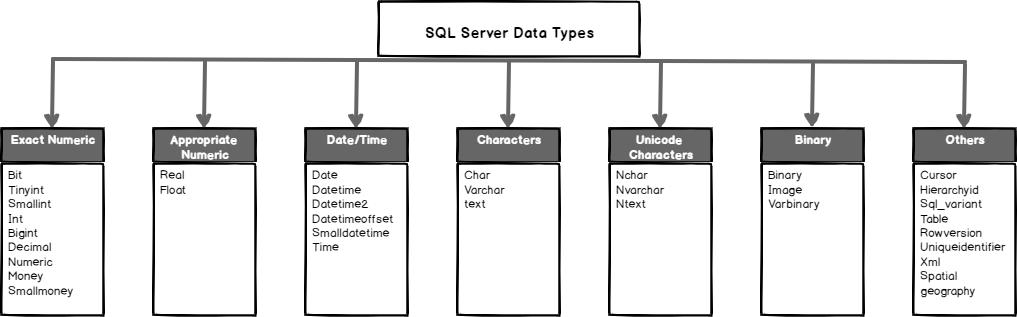
* Contact Number: It should be a numeric field
* Pin Code: We might expect both alphabets and numeric fields in this column

We store all these columns data in a table of SQL Server. We can use various data types in SQL Server and use them as per the requirement. Let’s explore the SQL Server data types.

## Categories of SQL Server data types

SQL Server supports the following data type’s categories:

* **Exact numeric**: bit, tinyint, smallint, int, bigint, decimal, numeric, money and smallmoney
* **Approximate numeric:** Read and float
* **Date and time:** date, DateTime, datetime2, datetimeoffset, smalldatetime, time
* **Character strings**:char, varchar, text
* **Unicode character strings**: Nchar, Nvarchar, Ntext
* **Binary strings:** Binary, image and varbinary
* **Other data types:** Cursor, hierarchyid, sql\_variant, table, rowversion, uniqueidentifier, XML, Spatial and geography

[](https://www.sqlshack.com/wp-content/uploads/2019/12/categories-of-sql-server-data-types02.png)

**Note:** SQL Server will deprecate ntext, text and image data type in future releases. We should avoid the use of these data types. You can use **Nvarchar(max),** **varchar(max)**, and **varbinary(max).**

### **Exact numeric SQL Server data type**

We use exact numeric data types for integer, decimal, and money. Each data type has its own lower, upper limit and memory requirements. We should use the smallest data type to save memory requirements as well. For example, we can use the bit data type for storing true (1) or false (0) values.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Lower Range** | **Upper Range** | **Storage** | **Remarks** |
| **Bit** | 0 | 1 | 1 byte | We can also store NULL values in this. |
| **tinyint** | 0 | 255 | 1 byte | We can store whole numbers up to 255 in this data type. |
| **Smallint** | -2^15 (-32,768) | 2^15-1 (32,767) | 2 bytes | We can store whole numbers between a lower and higher range. |
| **Int** | −2^31 (−2,147, 483,648) | 2^31−1 (−2,147, 483,647 | 4 bytes | It also stores the whole number similar to a smallint but its lower and upper limits changes as defined. |
| **Bigint** | −2^63 (−9,223,372, 036,854,775,808) | 2^63−1 (−9,223,372, 036,854,775,807) | 8 bytes | We should use bigint data type if we cannot accommodate data in the integer data type. |
| **Decimal** | −10^38+1 | 10^381−1 | It depends upon precision.  1 – 9 -> 5 bytes  10-19->9 bytes  20-28->13 bytes  29-28->17 bytes | We use decimal data type for  scale and fixed precision numbers. |
| **Numeric** | −10^38+1 | 10^381−1 | It depends upon precision.  1 – 9 -> 5 bytes  10-19->9 bytes  20-28->13 bytes  29-28->17 bytes | Decimal and numeric are synonyms. We can use them interchangeably. |
| **Smallmoney** | -214,478.3648 | +214,478.3647 | 4 bytes | We can use this data type for monetary or currency values. |
| **Money** | −922,337, 203, 685,477.5808 | +922,337, 203, 685,477.5807 | 8 bytes |  |

### **Approximate numeric SQL Server data type**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Lower Range** | **Upper Range** | **Storage** | **Remarks** |
| **Real** | −3.40E+38 | 3.40E+38 | 4 bytes | We can use float924) as ISO synonym for **real**. |
| **Float(n)** | −1.79E+308 | 1.79E+308 | Its storage depends upon value (n)  N(1-24) ->4 bytes  N(25-53)->8 bytes | It is an Approximate-number data types.  The default value of N is 53. |

### **Date and Time SQL Server Data types**

We stored date and time data along with time offset using these data types.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Lower Range** | **Upper Range** | **Storage** | **Remarks** |
| **Date** | 0001-01-01 | 9999-12-31 | 3 bytes | 1. It stores only dates in SQL Server.  2. Its default value is 1900-01-01.  3. It provides default format  YYYY-MM-DD. |
| **Datetime** | 1753-01-01 | 9999-12-31 | 8 bytes | 1. We can define a date along with time with fractional seconds.  2. The default value for this data type is 1900-01-01 00:00:00.  3.It provides accuracy in increments of .000, .003, or .007 seconds.  4. We should avoid using this data type. We can use Datetime2 instead. |
| **Datetime2** | 0001-01-01 00:00:00 | 9999-12-31 23:59:59.9999999 | 6-8 bytes  1. Precision<3 -> 6 bytes  2.Precision 3 or 4-> 7 bytes | 1. the default format for this is YYYY-MM-DD hh:mm: ss[.fractional seconds].  2. It provides precision from 0 to 7 digits, with an accuracy of 100ns.  2. The default precision for datetime2 is 7 digits. |
| **Datetimeoffset** | 0001-01-01 00:00:00 | 9999-12-31 23:59:59.9999999 | 10 bytes | 1. It is similar to a datetime2 data type but includes time zone offset as well.  2. Timezone offset is -14:00 through +14:00. |
| **smalldatetime** | 1900-01-01 00:00:00 | 2079-06-06 23:59:59 | 4 bytes | 1. It defines a date with the time of the day.  2. Its default value is 1900-01-01 00:00:00.  3. It provides an accuracy of one minute. |
| **Time** | 00:00:00.0000000 | 23:59:59.9999999 | 5 bytes | 1. We can use it for storing only time data.  2. Its default format is hh:mm:ss[.nnnnnnn].  3. It provides an accuracy of 100 nanoseconds. |

### **Character Strings SQL Server Data types**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Lower Range** | **Upper Range** | **Storage** | **Remarks** |
| **Char(n)** | 0 characters | 8000 characters | N bytes | 1. It provides a fixed-width character data type. |
| **Varchar(n)** | 0 characters | 8000 characters | n bytes + 2 bytes | 1.It is a variable length character data type.  2.N defines the string size. |
| **Varchar (max)** | 0 characters | 2^31 chars | n bytes + 2 bytes ~ 2 GB | We should avoid using this data type unless required due to its huge storage requirement. |
| **Text** | 0 chars | 2,147,483,647 chars | n bytes + 4 bytes | 1. It is a variable-length character data type.  2. We should avoid using this data type as it might get deprecated in future versions of SQL Server. |

### **Unicode character string SQL Server data types**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Lower Range** | **Upper Range** | **Storage** | **Remarks** |
| **Nchar** | 0 characters | 4000 characters | 2 times n bytes | It is a Unicode string of fixed width. |
| **Nvarchar** | 0 chars | 4000 Chars | 2 times n bytes | Nvarchar is a Unicode string of variable width. |
| **Ntext** | 0 chars | 1,073,741,823 char | 2 times the string length | 1. It is a Variable-length Unicode data  2. We should avoid using this data type as it will be deprecated in future SQL releases. |

### **Binary SQL Server data types**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Lower Range** | **Upper Range** | **Storage** | **Remarks** |
| **Binary** | 0 bytes | 8000 bytes | N bytes | This data type is a fixed-width binary string. |
| **varbinary** | 0 bytes | 8000 bytes | Its storage is the actual length of string + 2 bytes. |  |
| **Image** | 0 bytes | 2,147,483,647 bytes |  | Avoid using this data type, as it will be deprecated in future SQL Server releases. |

### **Other data types**

There are few data types as well that can be used as per the requirement:

* [Cursor](https://docs.microsoft.com/en-us/sql/t-sql/data-types/cursor-transact-sql?view=sql-server-ver15): It is useful for variables or stored procedure OUTPUT parameter referencing to a cursor
* [Rowversion](https://docs.microsoft.com/en-us/sql/t-sql/data-types/rowversion-transact-sql?view=sql-server-ver15): It returns automatically generated, unique binary numbers within a database
* [Hierarchyid](https://docs.microsoft.com/en-us/sql/t-sql/data-types/hierarchyid-data-type-method-reference?view=sql-server-ver15)**: it is a system data type with variable length. We use it to represent a**position in a hierarchy
* [Uniqueidentifier](https://docs.microsoft.com/en-us/sql/t-sql/data-types/uniqueidentifier-transact-sql?view=sql-server-ver15): It provides 16 bytes GUID
* [XML](https://docs.microsoft.com/en-us/sql/t-sql/xml/xml-transact-sql?view=sql-server-ver15): It is a special data type for storing the XML data in SQL Server tables
* [Spatial Geometry type](https://docs.microsoft.com/en-us/sql/t-sql/spatial-geometry/spatial-types-geometry-transact-sql?view=sql-server-ver15): We can use this for representing data in a flat (Euclidean) coordinate system
* [Spatial Geography Types:](https://docs.microsoft.com/en-us/sql/t-sql/spatial-geography/spatial-types-geography?view=sql-server-ver15) We can use Spatial Geography type for storing ellipsoidal (round-earth) data, such as GPS latitude and longitude coordinates. It represents data in a round-earth coordinate system
* [Table](https://docs.microsoft.com/en-us/sql/t-sql/data-types/table-transact-sql?view=sql-server-ver15): It is a special data type useful for storing result set temporarily in a table-valued function. We can use data from this for processing later. It can be used in functions, stored procedures, and batches

SQL Data Type is an attribute that specifies the type of data of any object. Each column, variable and expression has a related data type in SQL. You can use these data types while creating your tables. You can choose a data type for a table column based on your requirement.

SQL Server offers six categories of data types for your use which are listed below −

Exact Numeric Data Types

|  |  |  |
| --- | --- | --- |
| **DATA TYPE** | **FROM** | **TO** |
| Bigint | -9,223,372,036,854,775,808 | 9,223,372,036,854,775,807 |
| Int | -2,147,483,648 | 2,147,483,647 |
| Smallint | -32,768 | 32,767 |
| Tinyint | 0 | 255 |
| Bit | 0 | 1 |
| Decimal | -10^38 +1 | 10^38 -1 |
| Numeric | -10^38 +1 | 10^38 -1 |
| Money | -922,337,203,685,477.5808 | +922,337,203,685,477.5807 |
| Smallmoney | -214,748.3648 | +214,748.3647 |

Approximate Numeric Data Types

|  |  |  |
| --- | --- | --- |
| **DATA TYPE** | **FROM** | **TO** |
| float | -1.79E + 308 | 1.79E + 308 |
| real | -3.40E + 38 | 3.40E + 38 |

Date and Time Data Types

|  |  |  |
| --- | --- | --- |
| **DATA TYPE** | **FROM** | **TO** |
| datetime | Jan 1, 1753 | Dec 31, 9999 |
| smalldatetime | Jan 1, 1900 | Jun 6, 2079 |
| Date | Stores a date like June 30, 1991 | |
| Time | Stores a time of day like 12:30 P.M. | |

**Note** − Here, datetime has 3.33 milliseconds accuracy where as smalldatetime has 1 minute accuracy.

Character Strings Data Types

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **char**  Maximum length of 8,000 characters.( Fixed length non-Unicode characters) |
| 2 | **varchar**  Maximum of 8,000 characters.(Variable-length non-Unicode data). |
| 3 | **varchar(max)**  Maximum length of 2E + 31 characters, Variable-length non-Unicode data (SQL Server 2005 only). |
| 4 | **text**  Variable-length non-Unicode data with a maximum length of 2,147,483,647 characters. |

Unicode Character Strings Data Types

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **nchar**  Maximum length of 4,000 characters.( Fixed length Unicode) |
| 2 | **nvarchar**  Maximum length of 4,000 characters.(Variable length Unicode) |
| 3 | **nvarchar(max)**  Maximum length of 2E + 31 characters (SQL Server 2005 only).( Variable length Unicode) |
| 4 | **ntext**  Maximum length of 1,073,741,823 characters. ( Variable length Unicode ) |

Binary Data Types

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **binary**  Maximum length of 8,000 bytes(Fixed-length binary data ) |
| 2 | **varbinary**  Maximum length of 8,000 bytes.(Variable length binary data) |
| 3 | **varbinary(max)**  Maximum length of 2E + 31 bytes (SQL Server 2005 only). ( Variable length Binary data) |
| 4 | **image**  Maximum length of 2,147,483,647 bytes. ( Variable length Binary Data) |

Misc Data Types

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **sql\_variant**  Stores values of various SQL Server-supported data types, except text, ntext, and timestamp. |
| 2 | **timestamp**  Stores a database-wide unique number that gets updated every time a row gets updated |
| 3 | **uniqueidentifier**  Stores a globally unique identifier (GUID) |
| 4 | **xml**  Stores XML data. You can store xml instances in a column or a variable (SQL Server 2005 only). |
| 5 | **cursor**  Reference to a cursor object |
| 6 | **table**  Stores a result set for later processing |