SQL Datatypes

Datatypes and varchars

Character Strings : char, varchar, text

- **char**(n) is a fixed-length string where n is # of bytes and must be between 1 and 8,000.
  - if you enter a string too short, the server will add trailing spaces. Too many, you'll get an error
  - char is best used when the column data entries are a consistent length
- **varchar(n)** is a variable-length string.
  - You should always specify length, or a default of 1 character will be imposed.
  - n is # of bytes and must be between 1 and 8,000.
  - instead of n you can specify 'max' which will allow a column size of up to 2GB
  - total size of varchar is n + 2 bytes to store length info
- **text** is very similar to varchar, in that it is a Non-Unicode large Variable Length character data type, which can store maximum of 2GB. It is an older data type and may be discontinued in the future.
- nchar, nvarchar, and ntext all support Unicode characters, and so are best for multi-language support

Date and Time data types

SQL has many different ways of representing the date and time

different data types

- date - smallest in size. 3 bytes. default format: YYYY-MM-DD
- datetime - most widely used. 8 bytes.
  - YYYY-MM-DD hh: mm: ss may include fractional seconds (23:59:59.996)
- datetime2 - similar to datetime but with more precision. 6 to 8 bytes.
  - Up to 7 decimal places for seconds, user can define any fractional length <= 7
- datetimeoffset - 10 bytes - includes timezone awareness
- smalldatetime - 4 bytes. like datetime but without fractional seconds
- time(7) - hh:mm:ss[.nnnnnnn]
- The TIMESTAMP() function is not supported in Microsoft Azure.
  - Use GetDate() or GeTUTCDate() for current time
- Why not use varchar for date? Because you lose functionality, like comparing dates.

different styles

2004-05-23 14:25:10.487
2005-05-23 14:35:19.923
2006-08-10 14:35:37.823

May 23, 2005
Aug 10, 2006
Feb 28, 2012
Oct 23, 2020

02/28/12
10/23/20
09/23/19

10 Aug 2006 14:35:10:000
28 Feb 2012 03:35:10:000
23 Oct 2020 19:35:10:000

convert and cast: These functions convert an expression of one data type to another.

```
-- CAST Syntax:
CAST ( expression AS data_type [ ( length ) ] )

-- CONVERT Syntax:
CONVERT ( data_type [ ( length ) ] , expression [ , style ] )
```

Date and Time data types in SQL

Data type	Format	Range	Accuracy	Storage size (bytes)	User-defined fractional second precision	Time zone offset
time	hh:mm:ss[.nnnnnnn]	00:00:00.0000000 through 23:59:59.9999999	100 nanoseconds	3 to 5	Yes	No
date	YYYY-MM-DD	0001-01-01 through 9999-12-31	1 day	3	No	No
smalldatetime	YYYY-MM-DD hh:mm:ss	1900-01-01 through 2079-06-06	1 minute	4	No	No
datetime	YYYY-MM-DD hh:mm:ss[.nnn]	1753-01-01 through 9999-12-31	0.00333 second	8	No	No
datetime2	YYYY-MM-DD hh:mm:ss[.nnnnnnn]	0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999	100 nanoseconds	6 to 8	Yes	No
datetimeoffset	YYYY-MM-DD hh:mm:ss[.nnnnnnn] [+|-]hh:mm	0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999 (in UTC)	100 nanoseconds	8 to 10	Yes	Yes

Monetary data types: money & smallmoney

Money is an exact numeric type in SQL, like integer or bit.

money is 8 bytes, with a max/min value around +/- 922 trillion

smallmoney is 4 bytes with a max/min value around +/- 214,000

Casting money type to varchar:

select convert(varchar, column, style#) FROM table;

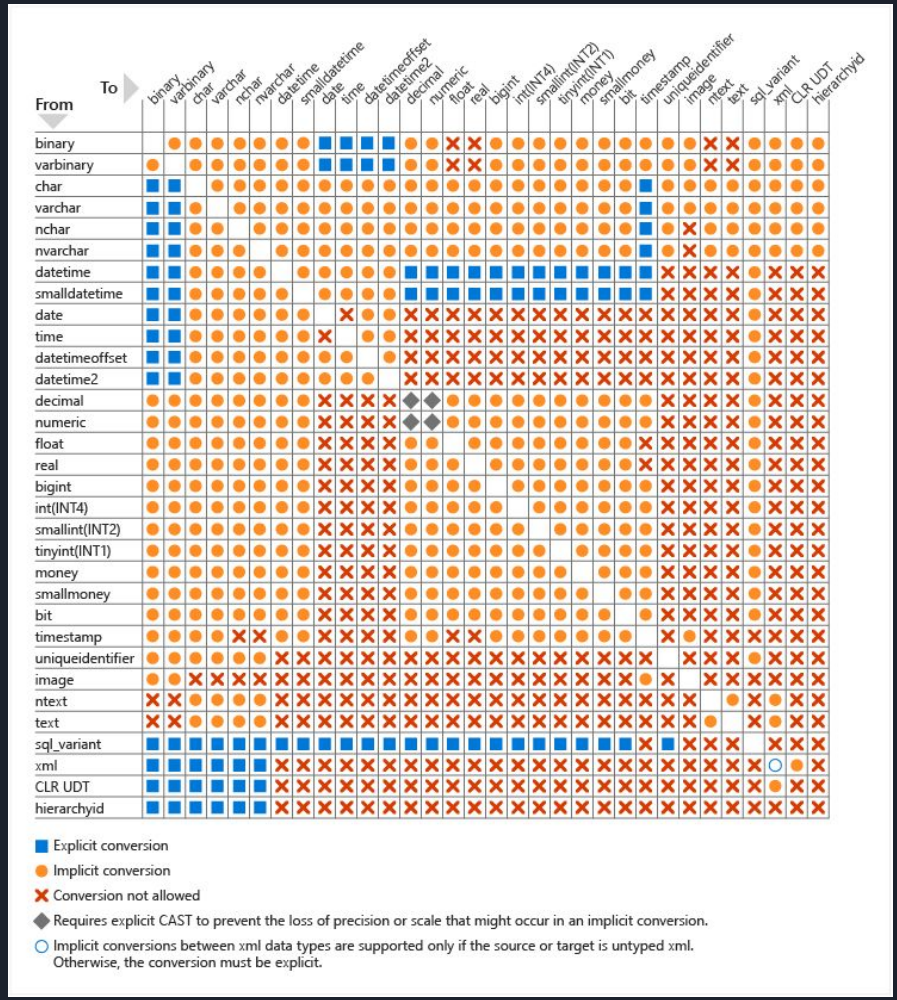
Style affects the visual representation of monetary values in SQL.

Commas for thousands:
$7500.00 vs $7,500.00
2 or 4 decimal places:
$55.1234 vs $55.12

Data type Conversions in SQL

Cast and Convert are used for explicit conversion

Implicit conversion is hidden from the user



Bit

- Smallest numeric data type.
- It can only store 1, 0, and NULL.
- The strings TRUE and FALSE are converted to one and zero respectively for optimization.
- Converting to bit turns any nonzero value into one.
- Another optimization is that if there are eight or fewer bit columns in the table, they can all be stored as one byte. nine to 16 will be stored as two bytes, 17 to 24 as three, and so on.
- Why? This can be useful for saving space if the data can be stored as binary values.
- How? CREATE TABLE Employees(ID BIT);
- That yields a table with an ID column that holds bits.

Exact Numerics: Int, Bigint, Smallint, and Tinyint

- Bigint: -2^63 (-9,223,372,036,854,775,808) to 2^63-1 (9,223,372,036,854,775,807). Eight bytes.
- Int: -2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647). Four bytes.
- Smallint: -2^15 (-32,768) to 2^15-1 (32,767). Four bytes.
- Tinyint: Zero to 255. 1 byte.
- These only store whole numbers within the specified range.
- When converting any int to a character, it will become (*) if it exceeds 1 digit in length.
- Attempting to use integer constants that exceed 2,147,483,647 will result in a decimal datatype, not bigint.
- SELECT 2147483647 / 2 AS Result1, 2147483649 / 2 AS Result2 ;
- Will yield:
- Result1: 1073741823, Result2: 1073741824.500000.

More About Ints

- Why? Use to store whole numbers. Pick the smallest one that can contain the data.
- How: CREATE TABLE dbo.MyTable
- (
- MyBigIntColumn BIGINT
- ,MyIntColumn INT
- ,MySmallIntColumn SMALLINT
- ,MyTinyIntColumn TINYINT
-);
- The query above yields a table of the various int types.

Exact Numerics: Decimal and Numeric

- Decimal, numeric are interchangeable. They can reach up to 17 bytes in size.
- When creating a column of these types, they must supplied with a precision and scale. Precision is the total number of digits to store. Scale must be less than or equal to precision and determines how many out of the total number of digits will be to the right of the decimal point.
- Precision of one to 9 requires five bytes, 10 to 19 requires 9, 20 to 28 takes 13, and 29 to 38 takes 17.
- Variations of of scale are considered different data types even if they have the same precision. So decimal(5,5) is different from decimal(5,0).
- Constants with decimal points are converted into numerics with the minimum necessary precision and scale.
- Converting to float/real can cause loss of precision. Converting any int to decimal can cause overflow.
- Why? Use to store decimal numbers.
- How: CREATE TABLE dbo.MyTable
- (MyDecimalColumn DECIMAL(5,2)
- ,MyNumericColumn NUMERIC(10,5));

Binary Strings: Binary and Varbinary

- Very similar to char and varchar, but elements are compared byte by byte rather than character by character. The two strings, “HELLO” and “hello” would be considered unequal if stored in binary or varbinary form, but would be considered equal if stored in char or varchar form.
- Requires a parameter n(1 to 8000) which determines number of characters that can be stored.
- Use binary if entries will be of a consistent size. Use varbinary if entries will vary in size.
- Use varbinary(max) if the column data entries exceed 8,000 bytes.
- Why: Use if byte comparison are preferred over character comparisons.
- How: CREATE TABLE dbo.MyTable
- (MyBinaryColumn BINARY(52)
- ,MyVarbinarylColumn VARBINARY(max));

Ntext, Text, and Image

- ATTENTION: Do NOT use as these will be removed in a future version of SQL Server according to the Microsoft docs. Use nvarchar(max), varchar(max), and varbinary(max) instead.
- Ntext: Variable-length Unicode data with a max string length of 2^30 - 1 (1,073,741,823) bytes. Storage size, in bytes, is twice the string length that is entered.
- Text: Variable-length non-Unicode data in the code page of the server and with a maximum string length of 2^31-1 (2,147,483,647).
- Image: Variable-length binary data from 0 through 2,147,483,647 bytes.

Unicode character strings: nchar and nvarchar, and how they differ from char and varchar

- These store characters as byte-pairs, so a minimum of two bytes.
- They take a parameter n(0 to 4000) which determines the capacity.
- N does not correlate precisely with number of characters. This is because some characters can occupy more than one byte pair. Most of the time this won't be a concern however.
- Nchar is fixed while nvarchar is variable.
- Why? These support utf-16(two bytes minimum per character) while char and varchar support utf-8 and each character takes a minimum of 1 byte. Choose depending on which characters need to be supported. For most cases, non-n variants are preferred to save space and since most of characters that will be encountered are still supported.
- How: CREATE TABLE dbo.MyTable
- (MyNcharColumn NCHAR(2000)
- ,MyNvarcharColumn NVARCHAR(4000));

Data Type Precedence

- When using operators with operands of different data types, the lower precedence type is upcast to the higher precedence type.
- An error is returned if the conversion is not supported.
- If the data types are the same, the return is of that data type.
- The precedence list can be found in Microsoft SQL docs, at https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-type-precedence-transact-sql?view=sql-server-ver15/