PROG8080
Data Types and CREATE TABLE in SQL Server 2014

Glenn Paulley

Fall 2015

PART 1: SQL DATA TYPES

CONESTOGA
Connect Life and Learning

Data Types

- Each column in a table has a data type
- The data type determines the type and amount of data you can store in a column
  - Different types have different semantics and support different sets of operations

Character Types

- CHAR(*n*)
- VARCHAR(*n*)
- TEXT
- NCHAR(n)
- NVARCHAR(n)
- nTEXT

- Fixed-length, up to 8K characters
- Variable length, up to 8K characters
- Variable length, up to 2GB characters
- Fixed length Unicode data, up to 4000 characters
- Variable length Unicode data, up to 2GB
- Variable length Unicode data, up to 2^30 – 1 characters

Binary (bit string) Types

- BINARY(*n*)
- VARBINARY(*n*)
- IMAGE
- BIT

- Fixed-length, up to 8K bytes
- Variable length, up to 8K bytes
- Variable length, up to 2GB bytes
- Supported values are 0,1, or NULL

Exact Numeric Types

- INT
- BIGINT
- SMALLINT
- TINYINT

- Whole numbers +/- 2^31
  - 32-bit signed values
- Whole numbers +/- 2^63
  - 64-bit signed values
- Whole numbers +/- 32K
  - 16-bit signed values
- Positive whole numbers 0-255

DECIMAL and NUMERIC types

- DECIMAL and NUMERIC are synonyms
- DECIMAL is an exact numeric type that supports numeric values with fractional component
- Declaration specifies *precision* (between 1 and 38) and an optional *scale*
  - DECIMAL(7,2)
    - 7 significant digits (precision), 2 of these digits after the decimal point (scale)
  - DECIMAL(10)
    - 10 significant digits, no fractional component
- Range: +/- 10^38
- Storage: between 5 and 17 bytes
- Default: if a value is declared DECIMAL, assumed to be DECIMAL(18)

Approximate Numeric Types

- Approximate numeric types store floating-point values; specification uses scientific notation (mantissa and exponent)
  - Not all numbers can be represented exactly (mantissa is in base-2)
  - Should never be used for primary keys
- REAL
  - Storage: 4 bytes
  - Range: - 3.40E + 38 to -1.18E - 38, 0 and 1.18E - 38 to 3.40E + 38
- FLOAT(n)
  - Value of n is between 1 and 53; number of bits used to store the mantissa of the number
  - Storage: 4 bytes if n <= 24; 8 bytes if n > 24
- DOUBLE PRECISION
  - Synonym for FLOAT(53)

Date/Time types in SQL Server 2014

- DATE
- TIME
- DATETIME
- SMALLDATETIME
- DATETIME2
- DATETIMEOFFSET

- Note: No TIMESTAMP type in this list (from the SQL Standard)
  - The TIMESTAMP type means something else in SQL Server!

DATE type

- Range: 1 January 0001 AD through 31 December 9999
  - Cannot be used for dates BC
- Date only, no time portion
- Default string format: ‘YYYY-MM-DD’
- Storage size: 3 bytes
- Default value: 1 January 1900

TIME type

- Range: 00:00:00.0000000 through 23:59:59.9999999
  - A TIME type can be declared with millisecond precision
  - Default is TIME(7), 100ns precision
- Time only, no date portion
- Default string format: ‘hh:mm:ss[.nnnnnnn]’
- Storage size: between 3 and 5 bytes, depending on precision
- Default value: 00:00:00

DATETIME type

- Range: January 1, 1753, through December 31, 9999
  - Time portion is between 00:00:00 and 23:59:59:997
  - Approximately 1/300 second precision with the time portion
- Character length: 19 minimum to 23 characters maximum
- Default string format: 'YYYY-MM-DD hh:mm:ss[.nnnnnnn]'
- Storage size: 8 bytes
- Default value: 1900-01-01 00:00:00.000
- SET DATEFORMAT option controls interpretation of DATETIME values when specified as string literals

SMALLDATETIME type

- Range: January 1, 1900, through June 6, 2079
  - Time portion is between 00:00:00 and 23:59:59
  - Approximately 1 minute precision with the time portion
- Character length: 19 characters maximum
- Default string format: 'YYYY-MM-DD hh:mm:ss'
- Storage size: 4 bytes
- Default value: 1900-01-01 00:00:00

DATETIME2 type

- SQL Server 2014's implementation of the TIMESTAMP type from the ISO SQL Standard
  - Now recommended for all applications instead of DATETIME
- Range: January 1, 0001, through December 31, 9999
  - Cannot be used for dates BC
  - Time portion is between 00:00:00 and 23:59:59:9999999
  - DATETIME2 supports specification of the time precision, up to 100ns
  - DATETIME2(7) is the default (100ns precision)
- Character length: 19 minimum to 27 characters maximum
- Default string format: 'YYYY-MM-DD hh:mm:ss[.fractional seconds]'
- Storage size: between 6 and 8 bytes depending on time precision
- Default value: 1900-01-01 00:00:00.0000000

DATETIMEOFFSET type

- SQL Server 2014's implementation of the TIMESTAMP WITH TIME ZONE type from the ISO SQL Standard
  - Adds a time zone component to a DATETIME2 type
- Range: January 1, 0001, through December 31, 9999
  - Cannot be used for dates BC
  - Time portion is between 00:00:00 and 23:59:59:9999999
  - DATETIMEOFFSET supports specification of the time precision, up to 100ns
  - DATETIME2(7) is the default (100ns precision)
  - Time zones: -14h through +14h
- Character length: 26 minimum to 34 characters maximum
- Default string format: 'YYYY-MM-DD hh:mm:ss.nnnnnnn {+|-}hh:mm
- Storage size: 10 bytes
- Default value: 1900-01-01 00:00:00.0000000 +00.00

Other (vendor-specific) Types

- MONEY
  - Money amounts to the trillions with 4 decimal places accuracy
- ROWVERSION (formerly known as TIMESTAMP)
  - A database-wide unique row number
  - TIMESTAMP is still supported as a row version type

CONESTOGA
Connect Life and Learning

User Defined Data Types

- It is possible to define user-defined types (UDTs)
  - UDTs are existing types with specific properties
- UDTs are an advanced technique used to ensure consistency across tables

PART 2: CREATE TABLE

CONESTOGA
Connect Life and Learning

CREATE TABLE Syntax

- Basic CREATE TABLE syntax and format:

```
CREATE TABLE tableName
( column1 TYPE [qualifiers],
  column2 TYPE,
  column3 TYPE [qualifiers],
   . . .
  [, table-level constraints]
)
```

- It is customary, but not essential, to put primary key columns at the beginning of a table

CREATE TABLE Example 1

```
CREATE TABLE product
(
  product_id VARCHAR(11),
  description VARCHAR(75),
  vendor_id VARCHAR(4),
  vendor_part_number VARCHAR(20),
  price NUMERIC(7,2),
  reorder_threshold INT,
  product_category_code CHAR(1)
)
```

NULL and NOT NULL

- Each column definition can include:
  - NULL to indicate that NULL values are permitted
    - This is the default and is therefore seldom seen
  - NOT NULL to indicate that NULL values are prohibited

```
...
vendor_id VARCHAR(4) NOT NULL,
...
```

CREATE TABLE

- All database systems have various optional clauses on CREATE TABLE that make these statements non-portable across various database management systems
- CREATE TABLE can contain vendor-specific syntax for:
  - Column and/or table-level encryption
  - Column compression
  - FOREIGN KEY constraints and various options
  - Table partitioning specification
  - Indexing options
  - Use of system-generated values for primary keys
    - SQL Server 2014 supports both IDENTITY values and SEQUENCE types

CONESTOGA
Connect Life and Learning

PART 3: CREATING A TABLE FROM A SELECT STATEMENT

CONESTOGA
Connect Life and Learning

CREATE TABLE

- In SQL Server, CREATE TABLE can only be used to create a table from scratch
  - Populating a table afterwards can be done using an INSERT INTO <table> FROM SELECT …
- This differs from ORACLE, for example, where one can specify

  CREATE TABLE myCourses AS

  SELECT * FROM Course
- To do this in SQL Server, one uses a SELECT statement with an INTO clause

CONESTOGA
Connect Life and Learning

SELECT INTO

- Syntax:

```
SELECT <select list>
    INTO <table>
FROM …
WHERE …
GROUP BY …
HAVING …
```

- The columns in <table> have the same names and data types as the attributes in the query's SELECT list
- Rows are populated from the result of the SELECT statement

SELECT INTO - limitations

- Cannot be used to create a partitioned table
- Indexes, constraints, and triggers defined in the source table are not transferred to the new table, nor can they be specified in the SELECT...INTO statement.
  - If these objects are required, you can create them after executing the SELECT...INTO statement.
- Specifying an ORDER BY clause does not guarantee the rows are inserted in the specified order.
- When a computed column is included in the select list, the corresponding column in the new table is not a computed column. The values in the new column are the values that were computed at the time SELECT...INTO was executed.

PART 4: DROP TABLE AND IF EXISTS

CONESTOGA
Connect Life and Learning

DROP TABLE

- To completely remove a table from a database, use DROP TABLE
  - DROP TABLE table

IF EXISTS

- Use code like this in a script to drop a table if it already exists:

```
IF EXISTS ( SELECT NAME FROM sysobjects
                    WHERE name = 'mytable' )
    DROP TABLE mytable;
```

or conversely

```
IF OBJECT_ID('dbo.your_table', 'U') IS NOT NULL
  DROP TABLE dbo.your_table;
```

CONESTOGA
Connect Life and Learning