Module 6

Working with SQL Server 2014 Data Types.

Zyead Ahmed.
Aspiring To Learning Data Engineer.

Module Overview

- Introducing SQL Server 2014 Data Types
- Working with Character Data
- Working with Date and Time Data

Lesson 1: Introducing SQL Server 2014 Data Types

- SQL Server Data Types
- Numeric Data Types
- Binary String Data Types
- Other Data Types
- Data Type Precedence
- When Are Data Types Converted?
- Demonstration: SQL Server Data Types

SQL Server Data Types

- SQL Server associate columns, expressions, variables, and parameters with data types
- Data types determine what kind of data can be held:
  - Integers, characters, dates, money, binary strings, etc.
- SQL Server supplies built-in data types
- Developers can also define custom types
  - Aliases in T-SQL
  - User-defined types in .NET code

SQL Server Data Type Categories	
Exact numeric	Unicode characters
Approximate numeric	Binary strings
Date and time	Other
Character strings	

Numeric Data Types

- Exact Numeric

Data type	Range	Storage (bytes)
tinyint	0 to 255	1
smallint	-32,768 to 32,768	2
int	2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)	4
Bigint	-2^63 - 2^63-1 (+/- 9 quintillion)	8
bit	1, 0 or NULL	1
decimal/numeric	- 10^38 +1 through 10^38 – 1 when maximum precision is used	5-17
money	-922,337,203,685,477.5808 to 922,337,203,685,477.5807	8
smallmoney	- 214,748.3648 to 214,748.3647	4

Binary String Data Types

- Binary Strings

Data Type	Range	Storage (bytes)
binary(n)	1-8000 bytes	n bytes
varbinary(n)	1-8000 bytes	n bytes + 2
varbinary(MAX)	1-2.1 billion (approx) bytes	actual length + 2

Other Data Types

Data Type	Range	Storage (bytes)	Remarks
rowversion	Auto-generated	8	Successor type to timestamp.
uniqueidentifier	Auto-generated	16	Globally unique identifier (GUID).
xml	0-2 GB	0-2 GB	Stores XML in native hierarchical structure.
cursor	N/A	N/A	Not a storage data type.
hierarchyid	N/A	Depends on content	Represents position in a hierarchy.
sql_variant	0-8000 bytes	Depends on content	Can store data of various data types.
table	N/A	N/A	Not a storage data type, used for query and programmatic operations.

Data Type Precedence

- Data type precedence determines which data type will be chosen when expressions of different types are combined
- Data type with the lower precedence is converted to the data type with the higher precedence
- Important for understanding implicit conversions
  - Conversion to type of lower precedence must be made explicitly (with CAST function)
- Example (low to high):
  - CHAR -> VARCHAR -> NVARCHAR -> TINYINT -> INT -> DECIMAL -> TIME -> DATE -> DATETIME2 -> XML

When Are Data Types Converted?

- Data type conversion scenarios
  - When data is moved, compared, or combined with other data
  - During variable assignment
- Implicit conversion
  - When comparing data of one type to another
  - Transparent to user

```
WHERE <column of smallint type> = <value of int type>
```

- Explicit conversion
  - Uses CAST or CONVERT functions

```
CAST(unitprice AS int)
```

- Not all conversions allowed by SQL Server

Demonstration: SQL Server Data Types

In this demonstration, you will see how to:

- Convert data types

Lesson 2: Working with Character Data

- Character Data Types
- Collation
- String Concatenation
- Character String Functions
- The LIKE Predicate
- Demonstration: Working with Character Data

Character Data Types

- SQL Server supports two kinds of character data types:
  - Regular: CHAR, VARCHAR
    - One byte stored per character
      - Only 256 possible characters – limits language support
  - Unicode: NCHAR, NVARCHAR
    - Two bytes stored per character
      - 65k characters represented – multiple language support
    - Precede characters with N' (National)
- TEXT, NTEXT deprecated
    - Use VARCHAR(MAX), NVARCHAR(MAX) instead

Collation

- Collation is a collection of character properties
  - Supported language, sort order
  - Case sensitivity, accent sensitivity
- In querying, collation awareness important for comparison
  - Is database case-sensitive?
    - If so, N'Funk' <> N'funk'
- Add COLLATE option to WHERE clause to control collation comparison

```
SELECT empid, lastname
FROM HR.employees
WHERE lastname COLLATE Latin1_General_CS_AS =
N'Funk';
```

String Concatenation

- SQL Server uses the + (plus) sign to concatenate characters:
  - Concatenating a value with a NULL returns a NULL

```
SELECT      empid, lastname, firstname,
            firstname + N' ' + lastname AS fullname
FROM HR.Employees;
```

- SQL Server 2012 introduced the CONCAT function
  - Converts NULL to empty string before concatenation

```
SELECT custid, city, region, country,
  CONCAT(city, ', ' + region, ', ' + country) AS location
FROM Sales.Customers
```

Character String Functions

- Common functions that modify character

Function	Syntax	Remarks
SUBSTRING()	SUBSTRING (expression , start , length)	Returns part of an expression.
LEFT(), RIGHT()	LEFT (expression , integer_value) RIGHT (expression , integer_value)	LEFT() returns left part of string up to integer_value. RIGHT() returns right part of string.
LEN(), DATALENGTH()	LEN (string_expression) DATALENGTH (expression)	LEN() returns the number of characters of the specified string expression, excluding trailing blanks. DATALENGTH() returns the number of bytes used.
CHARINDEX()	CHARINDEX (expressionToFind, expressionToSearch)	Searches an expression for another expression and returns its starting position if found. Optional start position.
REPLACE()	REPLACE (string_expression , string_pattern , string_replacement)	Replaces all occurrences of a specified string value with another string value.
UPPER(), LOWER()	UPPER (character_expression) LOWER (character_expression)	UPPER() returns a character expression with lowercase character data converted to uppercase. LOWER() converts uppercase to lowercase.

The LIKE Predicate

- The LIKE predicate used to check a character string against a pattern
- Patterns expressed with symbols
  - % (Percent) represents a string of any length
  - _ (Underscore) represents a single character
  - [<List of characters>] represents a single character within the supplied list
  - [<Character> - <character>] represents a single character within the specified range
  - [^<Character list or range>] represents a single character not in the specified list or range
  - ESCAPE Character allows you to search for a character that is also a wildcard character (%, _, [,] for example)

```
SELECT categoryid, categoryname, description
FROM Production.Categories
WHERE description LIKE 'Sweet%'
```

Demonstration: Working with Character Data

In this demonstration, you will see how to:

- Manipulate character data

Lesson 3: Working with Date and Time Data

- Date and Time Data Types
- Date and Time Data Types: Literals
- Working with Date and Time Separately
- Querying Date and Time Values
- Date and Time Functions
- Demonstration: Working with Date and Time Data

Date and Time Data Types

- Older versions of SQL Server supported only DATETIME and SMALLDATETIME
- DATE, TIME, DATETIME2, and DATETIMEOFFSET introduced in SQL Server 2008
- SQL Server 2012 added further functionality for working with date and time data types

Data Type	Storage (bytes)	Date Range	Accuracy	Recommended Entry Format
DATETIME	8	January 1, 1753 to December 31, 9999	3-1/3 milliseconds	'YYMMDD hh:mm:ss:nnn'
SMALLDATETIME	4	January 1, 1900 to June 6, 2079	1 minute	'YYMMDD hh:mm:ss:nnn'
DATETIME2	6 to 8	January 1, 0001 to December 31, 9999	100 nanoseconds	'YYMMDD hh:mm:ss.nnnnnn'
DATE	3	January 1, 0001 to December 31, 9999	1 day	'YYYY-MM-DD'
TIME	3 to 5		100 nanoseconds	'hh:mm:ss:nnnnnnn'
DATETIMEOFFSET	8 to 10	January 1, 0001 to December 31, 9999	100 nanoseconds	'YY-MM-DD hh:mm:ss:nnnnnnn [+|-]hh:mm'

Date and Time Data Types: Literals

- SQL Server doesn't offer an option for entering a date or time value explicitly
  - Dates and times are entered as character literals and converted explicitly or implicitly
    - For example, CHAR converted to DATETIME due to precedence
  - Formats are language-dependent, can cause confusion
- Best practices:
  - Use character strings to express date and time values
  - Use language-neutral formats

```
SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE orderdate = '20070825';
```

Working with Date and Time Separately

- DATETIME, SMALLDATETIME, DATETIME2, and DATETIMEOFFSET include both date and time data
- If only date is specified, time set to midnight (all zeroes)
- If only time is specified, date set to base date (January 1, 1900)

```
DECLARE @DateOnly DATETIME = '20120212';
SELECT @DateOnly;
```

```
RESULT
-----------------------
2012-02-12 00:00:00.000
```

Querying Date and Time Values

- Date values converted from character literals often omit time
  - Queries written with equality operator for date will match midnight

```
SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE orderdate = '20070825';
```

- If time values are stored, queries need to account for time past midnight on a date
  - Use range filters instead of equality

```
SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE orderdate >= '20070825'
AND    orderdate <  '20070826';
```

Date and Time Functions

- Functions that return current date and time

Function	Return Type	Remarks
GETDATE()	datetime	Current date and time. No time zone offset.
GETUTCDATE()	datetime	Current date and time in UTC.
CURRENT_TIMESTAMP()	datetime	Current date and time. No time zone offset. ANSI standard.
SYSDATETIME()	datetime2	Current date and time. No time zone offset.
STSUTCDATETIME()	datetime2	Current date and time in UTC.
SYSDATETIMEOFFSET()	datetimeoffset	Current date and time. Includes time zone offset.

```
SELECT CURRENT_TIMESTAMP();
SELECT SYSUTCDATETIME();
```

Demonstration: Working with Date and Time Data

In this demonstration, you will see how to:

- Query date and time values

Lab: Working with SQL Server 2014 Data Types

- Exercise 1: Writing Queries That Return Date and Time Data
- Exercise 2: Writing Queries That Use Date and Time Functions
- Exercise 3: Writing Queries That Return Character Data
- Exercise 4: Writing Queries That Use Character Functions

Logon Information
Virtual machine: **20461C-MIA-SQL**
User name: **ADVENTUREWORKS\Student**
Password: **Pa$$w0rd**

Estimated Time: 60 Minutes

Lab Scenario

- You are a business analyst for Adventure Works who will be writing reports using corporate databases stored in SQL Server. You have been provided with a set of business requirements for data and will write T-SQL queries to retrieve the specified data from the databases. You will need to retrieve and convert character and temporal data into various formats.

Module Review and Takeaways

- Review Question(s)
- Common Issues and Troubleshooting Tips