SQL Server: Transact-SQL Basic Data Retrieval

Module 5: Using Functions

Joe Sack

Joe@SQLskills.com

pluralsight
hardcore developer training

Module Introduction

- **This module covers a variety of functions which can be used within your data retrieval queries to meet application and reporting result set requirements**
- **You'll learn what the more commonly used functions are and *how* to use them, including coverage of:**
  - Aggregate functions
  - Mathematical functions
  - Ranking functions
  - Conversion functions
  - Date and Time functions
  - Logical functions
  - NULL handling
  - String functions
  - Analytic functions

Aggregate Functions

- **AVG**
  - Average of values in a group
- **CHECKSUM_AGG**
  - Checksum of values in a group
- **COUNT and COUNT_BIG**
  - Number of items in a group
    - COUNT returns int and COUNT_BIG returns bigint
  - If adding the DISTINCT keyword, COUNT and COUNT_BIG return the number of unique non-null values
  - COUNT(*) or COUNT_BIG(*) specifies all rows – including null values
- **MIN and MAX**
  - Minimum and maximum values in an expression
  - Both ignore null values

Aggregate Functions (2)

- **SUM**
  - Sum of all values
  - With the DISTINCT keyword – returns SUM of unique values
  - Null values are ignored
- **STDEV and STDEVP**
  - STDEV returns the standard deviation of all values in the expression
    - Assumes partial sampling of the whole population
  - STDEVP returns the standard deviation for the entire population of all values in the expression
- **VAR and VARP**
  - VAR is the statistical variance of all values in the specified expression
    - Assumes a partial sampling
  - VARP is the same as VAR, but assumes a entire population of all values

Mathematical Functions

- **More than 20 mathematical functions included natively in SQL Server 2005 – SQL Server 2012**
- **Common mathematical functions include:**
  - CEILING – smallest integer greater than or equal to the numeric expression
  - FLOOR – largest integer less than or equal to the numeric expression
  - PI – returns the constant value of PI
  - POWER – raises the numeric expression to a specified power
  - SQRT – square root of the numeric expression
  - ROUND – rounds number to the specified number of digits
  - RAND – generates a pseudo-random float value from 0 through 1
    - Has an optional “seed” integer value

OVER Clause

- **Applicable to ranking, aggregate and analytic functions**
- **OVER clause defines a "window" within a specific query result set**
  - Think of a window like a user-defined row set within the total result set
- **Allows you to apply aggregations to rows without a GROUP BY**
- **Window functions can then compute values for individual rows within the window**
  - Several examples of OVER in this module

Ranking Functions

- **ROW_NUMBER**
  - Sequential number of a row within a result set partition
- **RANK**
  - Returns rank of a row within the partition of the result set
    - Rank is calculated as one plus the number of ranks before the row
  - Tied rows based on logical order get the same rank
- **DENSE_RANK**
  - Same as RANK but with no gaps in ranking values
- **NTILE**
  - Map rows into equally sized row groups
    - You specify the number of groups
    - When the rows are not evenly divisible by groups, the group sizes will differ

Conversion Functions

- **PARSE (new in SQL Server 2012)**
  - Convert from a string data type to a date/time or number data type
- **TRY_PARSE (new in SQL Server 2012)**
  - Same as PARSE, but if the convert fails, returns NULL
- **TRY_CONVERT (new in SQL Server 2012)**
  - Converts from one data type to another and returns NULL if unsuccessful
- **CAST / CONVERT**
  - Both functions convert an expression from one data type to another
  - For CAST you designate the expression to be converted, data type and optional data type length
  - For CONVERT, you additionally can designate a style argument to determine output formats

Validating Data Types

- **ISDATE**
  - Validates if an input expression is a valid date or time
- **ISNUMERIC**
  - Validates if an input expression is a valid numeric data value

System Time Functions

- **SYSDATETIME**
  - Date and time of the SQL Server instance server
  - Returns datetime2(7) data type
- **SYSDATETIMEOFFSET**
  - Date and time of the SQL Server instance server
  - Includes time zone offset
  - Returns datetimeoffset(7) data type
- **SYSUTCDATETIME**
  - Date and time of the SQL Server instance server returned as Coordinated Universal Time (UTC)
  - Returns datetime2(7) data type
- **Lower precision functions are available – returning datetime**
  - CURRENT_TIMESTAMP, GETDATE, GETUTCDATE

Returning Date and Time Parts

- **DAY**
  - Returns integer representing day part of a provided date
- **MONTH**
  - Returns integer representing month part of a provided date
- **YEAR**
  - Returns integer representing year part of a provided date
- **DATEPART**
  - Returns integer value representing datepart of a date
  - Datepart examples include year (yy, yyyy), quarter (qq, q), month (mm, m), day (dd, d), hour (hh), minute (mi, n), second (ss, s), millisecond (ms)
- **DATENAME**
  - Returns string representing datepart of a date

Constructing Date and Time Values

- **DATEFROMPARTS**
  - Returns **date** data type value for a specified year, month and day
- **DATETIMEFROMPARTS**
  - Returns **datetime** value based on a specified year, month, day, hour, minute, seconds and milliseconds
- **DATETIME2FROMPARTS**
  - Returns **datetime2** value based on a specified year, month, day, hour, minute, seconds, fractions and precision

Constructing Date and Time Values (2)

- **DATETIMEOFFSETFROMPARTS**
  - Returns **datetimeoffset** value based on a specified year, month, day, hour, minute, seconds, fractions, hour_offset, minute_offset and precision
- **SMALLDATETIMEFROMPARTS**
  - Returns **smalldatetime** value based on a specified year, month, day, hour and minute
- **TIMEFROMPARTS**
  - Returns **time** value for specified hour, minute, seconds, fractions and precision

Calculating Time Differences

- **DATEDIFF returns the date/time difference between two input dates**
  - Return data type is integer
- **Takes a datepart argument followed by the start and end date**
  - Same datepart values accepted as would be provided for DATEPART

Modifying Dates

- **DATEADD**
  - Returns a new datetime value based on a datepart interval
- **EOMONTH (new in SQL Server 2012)**
  - Returns the last day of the month for a specified date
  - Optional integer that specifies months to add to the start date
- **SWITCHOFFSET**
  - Changes input datetimeoffset value to a new time zone offset
- **TODATETIMEOFFSET**
  - Converts a datetime2 value to a datetimeoffset value

Logical Functions

- **CHOOSE (new in SQL Server 2012)**
  - Choose an item from a list of values
  - First parameter is the 1-based index and the consecutive arguments are the list of values of any data type
- **IIF (new in SQL Server 2012)**
  - Return one of two values based on a Boolean expression
  - First parameter is the Boolean expression
  - Second parameter is the “true” value
  - Third parameter is the “false” value

Logical Functions (2)

- **Simple CASE expression**
  - Compare an expression to a set of expressions in order to determine the result
- **Searched CASE expression**
  - Evaluate a set of Boolean expressions in order to determine the result

Working with NULL

- **COALESCE**
  - Returns the first non-null expression from a list of arguments
- **ISNULL**
  - Replaces NULL with another value
  - Value is the same data type as the input expression
- **Understanding CONCAT_NULL_YIELDS_NULL**
  - When ON, concatenating a null value and a non-null string yields a NULL result

String Functions

- **ASCII**
  - Returns the ASCII integer code for the leftmost character of the expression
- **CHAR**
  - Converts an ASCII integer code to the char(1) data type equivalent
- **NCHAR**
  - Returns the Unicode character based on the input integer expression
- **UNICODE**
  - Returns the integer value representing the first character of the Unicode expression based on the Unicode standard

String Functions (2)

- **FORMAT (new in SQL Server 2012)**
  - Formats an input expression into an nvarchar value based on a format argument and optional culture argument
- **LEFT**
  - Outputs the left part of an input argument character string based on the positive integer specifying the number of characters to be returned
- **RIGHT**
  - Outputs the right part of an input argument character string based on the positive integer specifying the number of characters to be returned

String Functions (3)

- **LEN**
  - Outputs an int or bigint value representing the number of characters in the input string expression
  - Number of characters excludes trailing blanks
- **DATALENGTH**
  - Outputs number of bytes representing the number of characters in the input string expression
  - Number of characters excludes trailing blanks

String Functions (4)

- **LOWER**
  - Outputs character expression to lowercase
- **UPPER**
  - Outputs character expression to uppercase
- **LTRIM**
  - Removes leading blanks from the character expression
- **RTRIM**
  - Removes trailing blanks from the character expression

String Functions (5)

- **CHARINDEX and PATINDEX**
  - Both return the start position of a pattern within a character expression
  - PATINDEX allows for wildcard characters and CHARINDEX does not
- **REPLACE**
  - Replaces occurrences of a string pattern within a string expression with a replacement string
- **STUFF**
  - Inserts character data into a character expression
  - The point of insertion is determined by the start argument
  - The number of characters to delete is designated by the length argument
- **SUBSTRING**
  - Returns part of a character expression
  - Position argument determines the starting point and length argument determines the number of characters to return

String Functions (6)

- **REPLICATE**
  - Repeats a string expression a specified number of times
- **REVERSE**
  - Reverses the order of characters within a string expression
- **SPACE**
  - Returns a string of repeated spaces based on an integer expression
- **STR**
  - Converts numeric data into non-Unicode character data
- **CONCAT (new in SQL Server 2012)**
  - Returns a character string that is the result of two or more strings
- **QUOTENAME**
  - Delimits an input value, returning a Unicode string that produces a valid SQL Server identifier

Analytic Functions

- **LAG**
  - Compare values in current row with values in the previous row
- **LEAD**
  - Compares values in current row with values in a following row
- **FIRST_VALUE**
  - Return the first value in a result set
  - Result set may or may not be logically partitioned
- **LAST_VALUE**
  - Returns the last value in a result set
  - Result set may or may not be logically partitioned

Analytic Functions (2)

- **CUME_DIST**
  - Returns a value greater than 0 and less than 1 representing the number of rows less than or equal to the current row value divided by the number of rows in the partition/result set
- **PERCENT_RANK**
  - Returns a value greater than 0 and less than 1 representing the relative rank of a row within the partition/result set
- **PERCENTILE_CONT**
  - Calculates a percentile based on a continuous distribution of values
  - Result is interpolated, so the returned value may not actually be in the result set
- **PERCENTILE_DISC**
  - Similar to PERCENTILE_CONT, but the result will choose a number from the result set
    - Smallest CUME_DIST value greater than or equal to the percentile value

Course Summary

- **We've covered:**
  - Setting up your own Transact-SQL learning environment
  - Writing a basic SELECT statement
  - Writing a query that accesses multiple data sources
  - Using functions to meet application and business requirements
- **To solidify what you've learned, keep *applying* your knowledge in real life scenarios**
- **Thanks for watching!**