# **New Data Type Conversion Functions in SQL Server 2012**

**Problem**

SQL Server 2012 has a lot of new features to offer. A new set of features includes new **Conversion Functions** such as **PARSE**, **TRY\_PARSE** and **TRY\_CONVERT** which support data type casting and converting.  In this tip we will take a look at these new conversion functions.

**Solution**

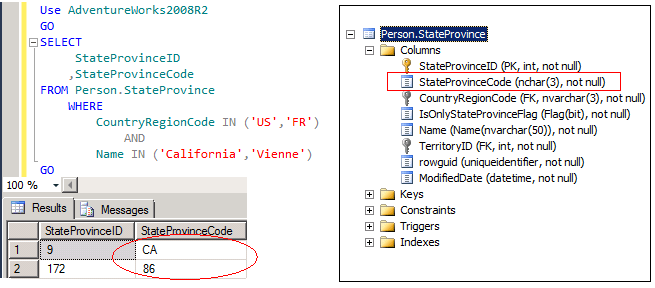
In this tip we will discuss how to utilize the below newly introduced Conversion Functions in SQL Server 2012.

**1. PARSE Conversion Function  
2. TRY\_PARSE Conversion Function  
3. TRY\_CONVERT Conversion Function**

**PARSE Conversion Function**

The PARSE function which is available in SQL Server 2012 can be used to convert any string value to a **Numeric** or to a **Date and Time**format. However, SQL Server 2012 will return an Error if it cannot convert a passed value to Numeric or to a Date and Time format. It is important to note the PARSE function relies on the presence of the .NET Framework Common Language Runtime (CLR).

To demonstrate this new conversion function, I will be using the **StateProvinceCode** column in the **Person.StateProvince** table of the **AdventureWorks2008R2** database which uses NCHAR(3) as the data type. In the snippet below you can see the StateProvinceCode column of Person.StateProvince table stores character data, but we have both a character value "CA" and an numeric value "86" stored in this column.



Let's execute the below TSQL code which will use the PARSE function to convert the StateProvinceCode column value to INT. Note that I am limiting the result set to a record that I know can be converted from character to integer otherwise this will error out.

Use AdventureWorks2008R2

GO

SELECT

StateProvinceID

,StateProvinceCode

,PARSE(StateProvinceCode AS INT) AS [Using PARSE Function]

FROM Person.StateProvince

WHERE

CountryRegionCode IN ('FR')

AND

Name IN ('Vienne')

GO

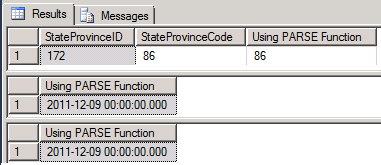
/\* Using PARSE Function to Convert String to Date Time \*/

SELECT PARSE('12/09/2011' AS datetime) AS [Using PARSE Function]

GO

SELECT PARSE('Friday, 09 December 2011' AS datetime USING 'en-US') AS [Using PARSE Function]

GO



**TRY\_PARSE Conversion Function**

The **TRY\_PARSE**function which is available in SQL Server 2012 can be used to convert any string value to a **Numeric** or to a **Date and Time** format. However, if SQL Server 2012 cannot convert a passed value to Numeric or to a Date and Time format, then it will return a **NULL** value rather than failing the entire TSQL code.

Use AdventureWorks2008R2

GO

SELECT

StateProvinceID

,StateProvinceCode

,TRY\_PARSE(StateProvinceCode AS INT) AS [Using TRY\_PARSE Function]

FROM Person.StateProvince

WHERE

CountryRegionCode IN ('US','FR')

AND

Name IN ('California','Vienne')

GO

/\* Using PARSE Function to Convert String to Date Time \*/

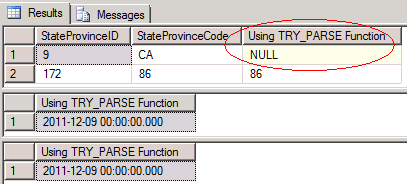
SELECT TRY\_PARSE('12/09/2011' AS datetime) AS [Using TRY\_PARSE Function]

GO

SELECT TRY\_PARSE('Friday, 09 December 2011' AS datetime USING 'en-US')

AS [Using TRY\_PARSE Function]

GO



In the above snippet you can see that since SQL Server couldn't convert **StateProvinceCode** column value **CA** to an integer it returned a **NULL** value instead of failing the entire TSQL code.

**TRY\_CONVERT Conversion Function**

The TRY\_CONVERT function which is available in SQL Server 2012 takes the values passed to it and tries to convert it to a specified Data Type. If the conversion is successful then it will return the value as the specified data type. Otherwise it will return a NULL value. However if you request a conversion that is explicitly not permitted, then the TRY\_CONVERT fails with an error.

Let's execute the below TSQL code wherein we are trying to convert NCHAR data type to INT using the CONVERT function which is available in the previous versions of SQL Server. You will notice that the conversion fails as StateProvinceCode column has both STRING and INT values.

Use AdventureWorks2008R2

GO

SELECT

StateProvinceID

,StateProvinceCode

,CONVERT(INT,StateProvinceCode) AS [Using CONVERT Function]

FROM Person.StateProvince

WHERE

CountryRegionCode IN ('US','FR')

AND

NAME IN ('California','Vienne')

GO

Error Message

Msg 245, Level 16, State 1, Line 1

Conversion failed when converting the nvarchar value 'CA ' to data type int.

Now lets us execute the below TSQL code using the TRY\_CONVERT function.

Use AdventureWorks2008R2

GO

SELECT

StateProvinceID

,StateProvinceCode

,TRY\_CONVERT(INT,StateProvinceCode) AS [Using TRY\_CONVERT Function]

FROM Person.StateProvince

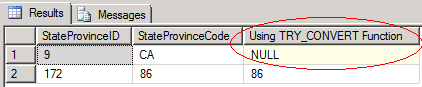
WHERE

CountryRegionCode IN ('US','FR')

AND

Name IN ('California','Vienne')

GO



In the above snippet you can see that when we used the TRY\_CONVERT function the query executed successfully and it returned a NULL value for StateProvinceCode value 'CA' since SQL Server cannot convert a String value to an Integer.

# **LOG (Transact-SQL)**

**SQL Server 2012**

Returns the natural logarithm of the specified float expression.

Topic link icon[Transact-SQL Syntax Conventions](https://technet.microsoft.com/en-us/library/ms177563(v=sql.110).aspx)

## Syntax

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_3ac1ea1f-7203-479e-855e-297cdc57d29d');" \o "Copy to clipboard.)

LOG ( float\_expression [, base ] )

## Arguments

float\_expression

Is an [expression](https://technet.microsoft.com/en-us/library/ms190286(v=sql.110).aspx) of type float or of a type that can be implicitly converted to float.

base

Optional integer argument that sets the base for the logarithm.

## Return Types

float

## Remarks

By default, LOG() returns the natural logarithm. You can change the base of the logarithm to another value using the optional base parameter.

The natural logarithm is the logarithm to the base **e**, where **e** is an irrational constant approximately equal to 2.718281828.

The natural logarithm of the exponential of a number is the number itself: LOG( EXP( n ) ) = n. And the exponential of the natural logarithm of a number is the number itself: EXP( LOG( n ) ) = n.

## Examples

### A. Calculating the logarithm for a number.

The following example calculates the LOG for the specified float expression.

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_62a0bb3b-8ed3-4a11-a131-33bf4558212d');)

DECLARE @var float;

SET @var = 10;

SELECT 'The LOG of the variable is: ' + CONVERT(varchar, LOG(@var));

GO

Here is the result set.

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_98d21ae9-2926-427e-b61a-0c7d2b3e34c7');)

-------------------------------------

The LOG of the variable is: 2.30259

(1 row(s) affected)

### B. Calculating the logarithm of the exponent of a number.

The following example calculates the LOG for the exponent of a number.

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_3a46f6d3-e2bc-49d2-96ca-bc528ff6ca68');)

SELECT LOG (EXP (10));

Here is the result set.

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_7f8a366f-fd90-46b2-b89e-dff579cee5bc');)

----------------------------------

10

(1 row(s) affected)