(T10)比較 Cast、Convert、Parse，介紹 GetDateTime
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc
==========
(T10)比較 Cast、Convert、Parse，介紹 GetDateTime
==========

==========

0. Summary

1.
Conversion Syntax :
--CONVERT(dataType , value, [style])
--TRY_CONVERT (dataType , value, [style])
--CAST(value AS dataType)
--TRY_CAST(value AS dataType)
--PARSE(stringValue AS date/time.../number)
--TRY_PARSE(stringValue AS date/time.../number)
Reference:
https://docs.microsoft.com/en-us/sql/t-sql/functions/cast-and-convert-transact-sql
https://docs.microsoft.com/en-us/sql/t-sql/functions/parse-transact-sql
1.1.
With "TRY", it returns a value cast/parse/convert to the specified data type.
If fail, then it returns NULL.
Without "TRY", then return ERROR.
1.2.
PARSE relies on the presence of the .NET Framework Common Language Runtime (CLR).
Remoting a function that requires the CLR would cause an error on the remote server.
1.3.
In Sumary,
PARSE() need .NET Framework Common Language Runtime (CLR).
Use PARSE only for converting from string to date/time and number types.
For general type conversions, continue to use CAST or CONVERT.
Convert() is specific to SQL Server.
Cast() is based on ANSI standard can work with other database.
Only if you need Style, then use Convert().
Otherwise better use Cast() for most of time.

2.
System Date and Time Functions
Reference:
https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql
2.1.
GETDATE()
Returns current "datetime" value
E.g.
2017-09-08 17:03:34.270
2.2.

CURRENT_TIMESTAMP
Reference:
https://docs.microsoft.com/en-us/sql/t-sql/functions/current-timestamp-transact-sql
Returns current "datetime" value.
ANSI SQL equivalent to GETDATE()
E.g.
2017-09-08 17:03:34.270
2.3.
SYSDATETIME()
Returns current "datetime2(7)" value
E.g.
2017-09-08 17:03:34.2715896
2.4.
SYSDATETIMEOFFSET()
Returns current "datetimeoffset(7)" value which includes time zone.
E.g.
2017-09-08 17:03:34.2715896 +10:00
2.5.
GETUTCDATE()
Returns current UTC "datetime" value
E.g.
2017-09-08 07:03:34.270
2.6.
SYSUTCDATETIME() AS [SYSUTCDATETIME()]
Returns current "datetime2(7)" value
E.g.
2017-09-08 07:03:34.2715896
2.7.
2017-09-08 18:54:32.033 is DateTime value.
2017-09-08 18:54:32.0352403 is DateTime2 value.
2017-09-08 17:03:34.2715896 +10:00 is datetimeoffset(7) value

==========

1. Cast_Convert

```
--=====================================================================
--T010_01_Cast_Convert
--=====================================================================
--=====================================================================
--T010_01_01
--DECLARE @Date1 DATETIME = CAST('2017-09-08 18:54:32.033' AS DATETIME);
DECLARE @Date1 DATETIME = '2017-09-08 18:54:32.033';
PRINT @Date1;
--Sep  8 2017  6:54PM
DECLARE @DateStr1 NVARCHAR(50) = CONVERT(NVARCHAR, @Date1);
PRINT @DateStr1;
--Sep  8 2017  6:54PM
DECLARE @DateStr2 NVARCHAR(50) = CONVERT(NVARCHAR, @Date1, 103);
PRINT @DateStr2;
--08/09/2017
```

```
GO -- Run the prvious command and begins new batch
--================================================================
--T010_01_02
--CONVERT Style
DECLARE @tempdate2 DATETIME ,
  @tempdate2_1 NVARCHAR(50);
SET @tempdate2 = CONVERT(DATETIME, '2017-09-08 18:54:32.033');
PRINT @tempdate2;
--Output : Sep  8 2017  6:54PM
SET @tempdate2_1 = CAST(@tempdate2 AS NVARCHAR);
PRINT 'CAST(@tempdate2 AS NVARCHAR)   :   ' + @tempdate2_1;
--Output : CAST(@tempdate2 AS NVARCHAR)   :   Sep  8 2017  6:54PM
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 101);
PRINT 'CONVERT(NVARCHAR, @tempdate2, 101)   :   ' + @tempdate2_1;
--Output : CONVERT(NVARCHAR, @tempdate2, 101)   :   09/08/2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2);
PRINT @tempdate2_1;
--Output : Sep  8 2017  6:54PM
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 0);
PRINT @tempdate2_1;
--Output : Sep  8 2017  6:54PM
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 100);
PRINT @tempdate2_1;
--Output : Sep  8 2017  6:54PM
--Default for datetime and smalldatetime
--0 or 100  = mon dd yyyy hh:miAM (or PM)
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 1);
PRINT @tempdate2_1;
--Output : 09/08/17
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 101);
PRINT @tempdate2_1;
--Output : 09/08/2017
--US, 1 = mm/dd/yy, 101 = mm/dd/yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 102);
PRINT @tempdate2_1;
--Output : 2017.09.08
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 2);
PRINT @tempdate2_1;
--Output : 17.09.08
--ANSI, 2 = yy.mm.dd, 102 = yyyy.mm.dd
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 103);
PRINT @tempdate2_1;
--Output : 08/09/2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 3);
PRINT @tempdate2_1;
--Output : 08/09/17
--British/French/Australia, 3 = dd/mm/yy, 103 = dd/mm/yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 104);
PRINT @tempdate2_1;
--Output : 08.09.2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 4);
PRINT @tempdate2_1;
--Output : 08.09.17
```

```
--German, 4 = dd.mm.yy, 104 = dd.mm.yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 105);
PRINT @tempdate2_1;
--Output : 08-09-2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 5);
PRINT @tempdate2_1;
--Output : 08-09-17
--Italian, 5 = dd-mm-yy, 105 = dd-mm-yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 106);
PRINT @tempdate2_1;
--Output : 08 Sep 2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 6);
PRINT @tempdate2_1;
--Output : 08 Sep 17
--6 = dd mon yy, 106 = dd mon yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 107);
PRINT @tempdate2_1;
--Output : Sep 08, 2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 7);
PRINT @tempdate2_1;
--Output : Sep 08, 17
--7 = Mon dd, yy, 107 = Mon dd, yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 108);
PRINT @tempdate2_1;
--Output : 18:54:32
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 8);
PRINT @tempdate2_1;
--Output : 18:54:32
--8 or 108 = hh:mi:ss
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 109);
PRINT @tempdate2_1;
--Output : Sep  8 2017  6:54:32:033PM
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 9);
PRINT @tempdate2_1;
--Sep  8 2017  6:54:32:033PM
--Default + milliseconds, 9 or 109 = mon dd yyyy hh:mi:ss:mmmAM (or PM)
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 110);
PRINT @tempdate2_1;
--Output : 09-08-2017
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 10);
PRINT @tempdate2_1;
--Output : 09-08-17
--10 = mm-dd-yy, 110 = mm-dd-yyyy
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 111);
PRINT @tempdate2_1;
--Output : 2017/09/08
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 11);
PRINT @tempdate2_1;
--Output : 17/09/08
--JAPAN, 11 = yy/mm/dd, 111 = yyyy/mm/dd
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 112);
PRINT @tempdate2_1;
--Output : 20170908
```

```
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 12);
PRINT @tempdate2_1;
--Output : 170908
--ISO, 12 = yymmdd, 112 = yyyymmdd
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 113);
PRINT @tempdate2_1;
--Output : 08 Sep 2017 18:54:32:033
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 13);
PRINT @tempdate2_1;
--Output : 08 Sep 2017 18:54:32:033
--Europe default + milliseconds, 13 or 113 = dd mon yyyy hh:mi:ss:mmm(24h)
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 114);
PRINT @tempdate2_1;
--Output : 18:54:32:033
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 14);
PRINT @tempdate2_1;
--Output : 18:54:32:033
--14 or 114 = hh:mi:ss:mmm(24h)
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 120);
PRINT @tempdate2_1;
--Output : 2017-09-08 18:54:32
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 20);
PRINT @tempdate2_1;
--Output : 2017-09-08 18:54:32
--20 or 120 = yyyy-mm-dd hh:mi:ss(24h)
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 121);
PRINT @tempdate2_1;
--Output : 2017-09-08 18:54:32.033
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 21);
PRINT @tempdate2_1;
--Output : 2017-09-08 18:54:32.033
--21 or 121 =yyyy-mm-dd hh:mi:ss.mmm(24h)
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 126);
PRINT @tempdate2_1;
--Output : 2017-09-08T18:54:32.033
--ISO8601, 126 = yyyy-mm-ddThh:mi:ss.mmm (no spaces)
--Note: When the value for milliseconds (mmm) is 0,
--the millisecond value is not displayed.
--E.g.
--the value '2012-11-07T18:26:20.000 is
--displayed as '2012-11-07T18:26:20
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 127);
PRINT @tempdate2_1;
--Output : 2017-09-08T18:54:32.033
--ISO8601 with time zone Z.
--127 = yyyy-mm-ddThh:mi:ss.mmmZ (no spaces)
--Note: When the value for milliseconds (mmm) is 0,
--the milliseconds value is not displayed.
--E.g.
--the value '2012-11-07T18:26:20.000 is
--displayed as '2012-11-07T18:26:20'.
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 130);
PRINT @tempdate2_1;
```

--Output : 17 6:54:32:033 1438 ذو الحجةP
--130 = dd mon yyyy hh:mi:ss:mmmAM
--In this style, mon represents a multi-token Hijri unicode
--representation of the full month's name.
--This value does not render correctly
--on a default US installation of SSMS.
SET @tempdate2_1 = CONVERT(NVARCHAR, @tempdate2, 131);
PRINT @tempdate2_1;
--Output : 17/12/1438 6:54:32:033PM
--131 = dd/mm/yyyy hh:mi:ss:mmmAM
GO -- Run the prvious command and begins new batch

/*
1.
2017-09-08 18:54:32.033 is DateTime value.
2017-09-08 18:54:32.0352403 is DateTime2 value.
2017-09-08 17:03:34.2715896 +10:00 is datetimeoffset(7) value
2.
CAST (expression AS data_type [(length)])
CONVERT (data_type [(length)] , expression [, style])
3.
Date and Time Styles
Reference
https://docs.microsoft.com/en-us/sql/t-sql/functions/cast-and-convert-transact-sql
3.1.
Default for datetime and smalldatetime
0 or 100 = mon dd yyyy hh:miAM (or PM)

3.2.
US
1 = mm/dd/yy
101 = mm/dd/yyyy

3.3.
ANSI
2 = yy.mm.dd
102 = yyyy.mm.dd

3.4.
British/French/Australia
3 = dd/mm/yy
103 = dd/mm/yyyy

3.5.
German
4 = dd.mm.yy
104 = dd.mm.yyyy

3.6.
Italian
5 = dd-mm-yy
105 = dd-mm-yyyy

3.7.
6 = dd mon yy
106 = dd mon yyyy

3.8.
7 = Mon dd, yy
107 = Mon dd, yyyy

3.9.
8 or 108 = hh:mi:ss

3.10.
Default + milliseconds
9 or 109 = mon dd yyyy hh:mi:ss:mmmAM (or PM)

3.11.
USA
10 = mm-dd-yy
110 = mm-dd-yyyy

3.12.
JAPAN
11 = yy/mm/dd
111 = yyyy/mm/dd

3.13.
ISO
12 = yymmdd
112 = yyyymmdd

3.14.
Europe default + milliseconds
13 or 113 = dd mon yyyy hh:mi:ss:mmm(24h)

3.15.
14 or 114 = hh:mi:ss:mmm(24h)

3.16.
ODBC canonical
20 or 120 = yyyy-mm-dd hh:mi:ss(24h)

3.17
ODBC canonical (with milliseconds) default for time, date, datetime2, and datetimeoffset
21 or 121 =yyyy-mm-dd hh:mi:ss.mmm(24h)

3.18.
ISO8601
126 = yyyy-mm-ddThh:mi:ss.mmm (no spaces)
Note: When the value for milliseconds (mmm) is 0,
the millisecond value is not displayed.
E.g.
the value '2012-11-07T18:26:20.000 is
displayed as '2012-11-07T18:26:20'.

3.19.
ISO8601 with time zone Z.
127 = yyyy-mm-ddThh:mi:ss.mmmZ (no spaces)
Note: When the value for milliseconds (mmm) is 0,
the milliseconds value is not displayed.
E.g.
the value '2012-11-07T18:26:20.000 is
displayed as '2012-11-07T18:26:20'.

3.20.
Hijri
130 = dd mon yyyy hh:mi:ss:mmmAM
In this style, mon represents a multi-token Hijri unicode
representation of the full month's name.
This value does not render correctly
on a default US installation of SSMS.

3.20.
Hijri
131 = dd/mm/yyyy hh:mi:ss:mmmAM
*/

--==
--T010_01_03
PRINT CAST(GETDATE() AS DATETIME);
--Output : Sep 10 2017 9:25PM
PRINT CONVERT(DATETIME, GETDATE());
--Output : Sep 10 2017 9:25PM
GO -- Run the prvious command and begins new batch
/*
1.
System Date and Time Functions
Reference:
https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql
1.1.
GETDATE()
Returns current "datetime" value
E.g.
2017-09-08 17:03:34.270
1.2.
CURRENT_TIMESTAMP
Reference:
https://docs.microsoft.com/en-us/sql/t-sql/functions/current-timestamp-transact-sql
Returns current "datetime" value.
ANSI SQL equivalent to GETDATE()
E.g.
2017-09-08 17:03:34.270
1.3.
SYSDATETIME()
Returns current "datetime2(7)" value

E.g.
2017-09-08 17:03:34.2715896
1.4.
SYSDATETIMEOFFSET()
Returns current "datetimeoffset(7)" value which includes time zone.
E.g.
2017-09-08 17:03:34.2715896 +10:00
1.5.
GETUTCDATE()
Returns current UTC "datetime" value
E.g.
2017-09-08 07:03:34.270
1.6.
SYSUTCDATETIME() AS [SYSUTCDATETIME()]
Returns current "datetime2(7)" value
E.g.
2017-09-08 07:03:34.2715896
2.
2017-09-08 18:54:32.033 is DateTime value.
2017-09-08 18:54:32.0352403 is DateTime2 value.
2017-09-08 17:03:34.2715896 +10:00 is datetimeoffset(7) value
*/

```
--=====================================================================
--T010_01_04
--Create Sample Data
IF ( EXISTS ( SELECT   *
         FROM     INFORMATION_SCHEMA.TABLES
         WHERE    TABLE_NAME = 'Person2' ) )
  BEGIN
     TRUNCATE TABLE dbo.Person2;
     DROP TABLE Person2;
  END;
GO -- Run the previous command and begins new batch
CREATE TABLE Person2
(
  Id INT IDENTITY(1, 1)
       PRIMARY KEY ,
  [NAME] NVARCHAR(100) NULL ,
  Email NVARCHAR(500) NULL ,
  RegisteredDateTime DATETIME NULL,
);
INSERT  Person2
VALUES ( N'Name6', N'6@6.com', CAST(N'2016-09-08T18:54:32.033' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name7', N'7@7.com', CAST(N'2016-01-27T21:30:28.473' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name8', N'8@8.com', CAST(N'2016-09-08T12:35:29.050' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name9', N'9@9.com', CAST(N'2016-01-27T13:19:34.267' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name10', N'10@10.com',
       CAST(N'2016-09-08T12:22:37.597' AS DATETIME) );
INSERT  Person2
```

```
VALUES ( N'Name11', N'11@11.com',
      CAST(N'2016-01-27T12:22:37.597' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name12', N'12@12.com',
      CAST(N'2011-11-01T07:51:48.177' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name13', N'13@13.com',
      CAST(N'2012-09-03T22:01:04.580' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name14', N'14@14.com',
      CAST(N'2016-01-27T01:28:02.657' AS DATETIME) );
INSERT  Person2
VALUES ( N'Name15', N'15@15.com',
      CAST(N'2016-09-08T00:28:44.183' AS DATETIME) );
SELECT *
FROM    Person2;



--=====================================================================
--T010_01_05
--GROUP BY Date
SELECT CAST(RegisteredDateTime AS DATE) AS RegisteredDate ,
    COUNT(Id) AS NumberOfPersonReister
FROM    Person2
GROUP BY CAST(RegisteredDateTime AS DATE);
GO -- Run the prvious command and begins new batch
/*
Group the Register Date
Counnt how many people who register on that date.
*/
--=====================================================================
--T010_01_06
--Clean up
IF ( EXISTS ( SELECT   *
         FROM    INFORMATION_SCHEMA.TABLES
         WHERE    TABLE_NAME = 'Person2' ) )
 BEGIN
    TRUNCATE TABLE dbo.Person2;
    DROP TABLE Person2;
 END;
GO -- Run the previous command and begins new batch


=======================================================================
```

2. Parse_Convert

```
--=====================================================================
--T010_02_Parse_Convert
--=====================================================================
```

```
/*
1.
Conversion Syntax :
--CONVERT( dataType , value, [style] )
--TRY_CONVERT ( dataType , value, [style] )
--CAST( value AS dataType )
--TRY_CAST( value AS dataType )
--PARSE( stringValue AS date/time.../number )
--TRY_PARSE( stringValue AS  date/time.../number )
Reference:
https://docs.microsoft.com/en-us/sql/t-sql/functions/cast-and-convert-transact-sql
https://docs.microsoft.com/en-us/sql/t-sql/functions/parse-transact-sql
1.1.
With "TRY", it returns a value cast/parse/convert to the specified data type.
If fail, then it returns NULL.
Without "TRY", then return ERROR.
1.2.
PARSE relies on the presence of the .NET Framework Common Language Runtime (CLR).
Remoting a function that requires the CLR would cause an error on the remote server.
1.3.
In Sumary,
PARSE() need .NET Framework Common Language Runtime (CLR).
Use PARSE only for converting from string to date/time and number types.
For general type conversions, continue to use CAST or CONVERT.
Convert() is specific to SQL Server.
Cast() is based on ANSI standard can work with other database.
Only if you need Style, then use Convert().
Otherwise better use Cast() for most of time.
*/
--=====================================================================
--T010_02_01
--Conversion success
DECLARE @Int1 INT = TRY_CONVERT(INT, '99');
PRINT @Int1;
--99
DECLARE @Int2 INT = TRY_PARSE( '99' AS INT
);
PRINT @Int2;
--99
DECLARE @Int3 INT = TRY_CAST('99' AS INT);
PRINT @Int3;
--99
DECLARE @Int4 INT = CONVERT(INT, '99');
PRINT @Int4;
--99
DECLARE @Int5 INT = PARSE( '99' AS INT
);
PRINT @Int5;
--99
DECLARE @Int6 INT = CAST('99' AS INT);
PRINT @Int6;
--99
GO -- Run the previous command and begins new batch
```

```
--=====================================================================
--T010_02_02
--Conversion fail
DECLARE @Int1 INT = TRY_CONVERT(INT, 'ABC');
SELECT  @Int1;
--NULL
DECLARE @Int2 INT = TRY_PARSE( 'ABC' AS INT
);
SELECT  @Int2;
--NULL
DECLARE @Int3 INT = TRY_CAST('ABC' AS INT);
SELECT  @Int3;
--NULL
DECLARE @Int4 INT = CONVERT(INT, 'ABC');
SELECT  @Int4;
/*
ERROR
--Msg 245, Level 16, State 1, Line 599
--Conversion failed when converting the varchar value 'ABC' to data type int.
*/
DECLARE @Int5 INT = PARSE( 'ABC' AS INT
);
SELECT  @Int5;
/*
ERROR
--Msg 9819, Level 16, State 1, Line 606
--Error converting string value 'ABC' into data type int using culture ''.
*/
DECLARE @Int6 INT = CAST('ABC' AS INT);
SELECT  @Int6;
/*
ERROR
--Msg 245, Level 16, State 1, Line 613
--Conversion failed when converting the varchar value 'ABC' to data type int.
*/
GO -- Run the previous command and begins new batch



--=====================================================================
--T010_02_03
--Conversion XML
DECLARE @Xml1 XML = TRY_CONVERT(XML, 10);
SELECT  @Xml1;
GO -- Run the previous command and begins new batch
/*
Error
--Msg 529, Level 16, State 2, Line 629
--Explicit conversion from data type int to xml is not allowed.
*/
DECLARE @Xml2 XML = TRY_CONVERT(XML, '<root><child/></root>');
SELECT  @Xml2;
GO -- Run the previous command and begins new batch
```

```
--<root><child /></root>
DECLARE @Xml3 XML = TRY_PARSE( '<root><child/></root>' AS XML
);
SELECT  @Xml3;
GO -- Run the previous command and begins new batch
/*
Error
--Msg 10761, Level 15, State 2, Line 648
--Invalid data type xml in function TRY_PARSE.
--Msg 137, Level 15, State 2, Line 649
--Must declare the scalar variable "@Xml3".
*/
DECLARE @Xml4 XML = TRY_CAST('<root><child/></root>' AS XML);
SELECT  @Xml4;
GO -- Run the previous command and begins new batch
--<root><child /></root>
--====================================================================
--T010_02_04
--IFF, CASE WHEN ...
--====================================================================
--T010_02_04_01
--IFF, CASE WHEN ... TRY_PARSE
DECLARE @Str1 NVARCHAR(10) = CASE WHEN TRY_PARSE( 'ABC' AS INT) IS NULL
                        THEN 'Failed'
                        ELSE 'Successful'
                    END;
PRINT @Str1;
GO -- Run the previous command and begins new batch
--Failed
DECLARE @Str2 NVARCHAR(10) = CASE WHEN TRY_PARSE( '30' AS INT) IS NULL
                        THEN 'Failed'
                        ELSE 'Successful'
                    END;
PRINT @Str2;
GO -- Run the previous command and begins new batch
--Successful
DECLARE @Str3 NVARCHAR(10) = IIF(TRY_PARSE( 'ABC' AS INT) IS NULL, 'Failed', 'Successful');
PRINT @Str3;
GO -- Run the previous command and begins new batch
--Failed
DECLARE @Str4 NVARCHAR(10) = IIF(TRY_PARSE( '30' AS INT) IS NULL, 'Failed', 'Successful');
PRINT @Str4;
--Successful
GO -- Run the previous command and begins new batch
--====================================================================
--T010_02_04_02
--IFF, CASE WHEN ... TRY_CONVERT
DECLARE @Str1 NVARCHAR(10) = CASE WHEN TRY_CONVERT(INT, 'ABC') IS NULL
                        THEN 'Failed'
                        ELSE 'Successful'
                    END;
PRINT @Str1;
GO -- Run the previous command and begins new batch
```

```
--Failed
DECLARE @Str2 NVARCHAR(10) = CASE WHEN TRY_CONVERT(INT, '30') IS NULL
                        THEN 'Failed'
                        ELSE 'Successful'
                    END;
PRINT @Str2;
GO -- Run the previous command and begins new batch
--Successful
DECLARE @Str3 NVARCHAR(10) = IIF(TRY_CONVERT(INT, 'ABC') IS NULL, 'Failed', 'Successful');
PRINT @Str3;
GO -- Run the previous command and begins new batch
--Failed
DECLARE @Str4 NVARCHAR(10) = IIF(TRY_CONVERT(INT, '30') IS NULL, 'Failed', 'Successful');
PRINT @Str4;
--Successful
GO -- Run the previous command and begins new batch


--=================================================================
--T010_02_04_03
--IFF, CASE WHEN ... TRY_CAST
DECLARE @Str1 NVARCHAR(10) = CASE WHEN TRY_CAST('ABC' AS INT) IS NULL
                        THEN 'Failed'
                        ELSE 'Successful'
                    END;
PRINT @Str1;
GO -- Run the previous command and begins new batch
--Failed
DECLARE @Str2 NVARCHAR(10) = CASE WHEN TRY_CAST('30' AS INT) IS NULL
                        THEN 'Failed'
                        ELSE 'Successful'
                    END;
PRINT @Str2;
GO -- Run the previous command and begins new batch
--Successful
DECLARE @Str3 NVARCHAR(10) = IIF(TRY_CAST('ABC' AS INT) IS NULL, 'Failed', 'Successful');
PRINT @Str3;
GO -- Run the previous command and begins new batch
--Failed
DECLARE @Str4 NVARCHAR(10) = IIF(TRY_CAST('30' AS INT) IS NULL, 'Failed', 'Successful');
PRINT @Str4;
--Successful
GO -- Run the previous command and begins new batch
```