(T8)討論DateTimeFunction，實作RandomDateTime  
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc  
=======================================================================  
(T8)討論DateTimeFunction，實作RandomDateTime  
=======================================================================  
0. What to learn

1. System Date and Time Functions

2. Date and Time data types

3. ISDATE(expression)

4. DatePart\_DateName\_Day\_Month\_Year

5. DATEADD(datepart,number,date)

6. DATEDIFF(datepart,startdate,enddate)

7. fnDurationByDate

8. Get Random DateTime

9. Get Random DateTime stored procedure

10. Clean up  
=======================================================================  
  
0. What to learn

What to learn

1.

GETDATE

CURRENT\_TIMESTAMP

SYSDATETIME

SYSDATETIMEOFFSET

GETUTCDATE

SYSUTCDATETIME

2.

ISDATE

DAY

MONTH

YEAR

3.

DATEPART

DATENAME

DATEADD

DATEDIFF

1. System Date and Time Functions

--================================================================

--T008\_01\_System Date and Time Functions

--================================================================

--System Date and Time Functions

SELECT  GETDATE() AS [ GETDATE()];

--2017-09-08 17:03:34.270

SELECT  CURRENT\_TIMESTAMP AS [CURRENT\_TIMESTAMP];

--2017-09-08 17:03:34.270

SELECT  SYSDATETIME() AS [SYSDATETIME()];

--2017-09-08 17:03:34.2715896

SELECT  SYSDATETIMEOFFSET() AS [SYSDATETIMEOFFSET()];

--2017-09-08 17:03:34.2715896 +10:00

SELECT  GETUTCDATE() AS [GETUTCDATE()];

--2017-09-08 07:03:34.270

SELECT  SYSUTCDATETIME() AS [SYSUTCDATETIME()];

--2017-09-08 07:03:34.2715896

/\*

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.

As a programmer, All you need to know is that

GMT time and UTC time are basically the same in this earth.

In addition, London UK time zone is UTC time 0.

Brisbane Queensland Australia Time zone is UTC+10.

Washington, D.C. time zone is UTC-5

That means the time in Brisbane is 10 hours faster than London UK.

The time in Washington, D.C. is 5 hours slower than London UK.

Programmer normally stored UTC time value in database,

then display the local time to local user by adding time offset to UTC time value from database.

E.g.

In database, the row created UTC time is 2017-09-08 07:03:34.270

When user is in Brisbane Queensland Australia, add 10 hours to UTC time.

That will be 2017-09-08 17:03:34.270

4.

Regarding Seconds

<http://whatis.techtarget.com/definition/nanosecond-ns-or-nsec>

4.1.

1 millisecond(ms) = 0.001 = 10E-3 second

A millisecond(ms or msec) is one thousandth of a second.

4.2.

1 microsecond(us) = 10E-6 second

A microsecond(us or Greek letter mu plus s) is one millionth (10E-6) of a second.

4.3.

1 nanosecond(ns) = 10E-9 second

A nanosecond(ns or nsec) is one billionth(10E-9) of a second.

4.4.

1 picosecond = 10E-12 second

A picosecond is one trillionth(10E-12) of a second, or one millionth of a microsecond.

4.5.

1 femtosecond = 10E-15 second

A femtosecond is one millionth of a nanosecond or (10E-15) of a second

and is a measurement sometimes used in laser technology.

4.6.

1 attosecond = 10E-18 second

An attosecond is one quintillionth (10E-18) of a second

and is a term used in photon research.

\*/

==================================================

2. Date and Time data types

--================================================================

--T008\_02\_Date and Time data types

--================================================================

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'DateTimeTypes' ) )

    BEGIN

        TRUNCATE TABLE DateTimeTypes;

        DROP TABLE DateTimeTypes;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE DateTimeTypes

(

  [TIME] TIME(7) NULL ,  --18:54:32.0333333

  [DATE] DATE NULL ,   --2017-09-08

  [SMALLDATETIME] SMALLDATETIME NULL ,  --2017-09-08 18:55:00

  [DATETIME] DATETIME NULL ,  --2017-09-08 18:54:32.033

  [DATETIME2] DATETIME2(7) NULL ,  --22017-09-08 18:54:32.0352403

  [DATETIMEOFFSET] DATETIMEOFFSET(7) NULL  --2017-09-08 18:54:32.0352403 +10:00

 );

INSERT  INTO [DateTimeTypes]

VALUES  ( GETDATE(), GETDATE(), GETDATE(), GETDATE(), SYSDATETIME(),

          SYSDATETIMEOFFSET() );

SELECT  \*

FROM    [DateTimeTypes];

/\*

1.

Date and Time data types

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql>

1.1.

[TIME](7)

Format:  hh:mm:ss[.nnnnnnn]

[TIME](7)  means 7 n, [TIME](6) means 6 n

Range:  00:00:00.0000000   through   23:59:59.9999999

Accuracy: 100 nanosecond(ns).  1 nanosecond(ns) = 10E-9 second

E.g.  18:54:32.0333333

1.2.

[DATE]

Format:  YYYY-MM-DD

Range:  0001-01-01 through 9999-12-31

Accuracy: 1 day

E.g.  2017-09-08

1.3.

[SMALLDATETIME]

Format:  YYYY-MM-DD hh:mm:ss

Range:  1900-01-01 through 2079-06-06

Accuracy: 1 minute

E.g.  2017-09-08 18:55:00

1.4.

[DATETIME]

Format:  YYYY-MM-DD hh:mm:ss[.nnn]

Range:  1753-01-01 through 9999-12-31

Accuracy: 0.00333 second

E.g.

2017-09-08 18:54:32.033

1.5.

[DATETIME2](7)

Format:  YYYY-MM-DD hh:mm:ss[.nnnnnnn]

[TIME](7)  means 7 n, [TIME](6) means 6 n

Range:  0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999

Accuracy: 100 nanosecond(ns).  1 nanosecond(ns) = 10E-9 second

E.g.

2017-09-08 18:54:32.0352403

1.6.

[DATETIMEOFFSET](7)

Format:  YYYY-MM-DD hh:mm:ss[.nnnnnnn] [+|-]hh:mm

[TIME](7)  means 7 n, [TIME](6) means 6 n

Range:  0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999 (in UTC)

Accuracy: 100 nanosecond(ns).  1 nanosecond(ns) = 10E-9 second

E.g.

2017-09-08 18:54:32.0352403 +10:00

\*/

==================================================

3. ISDATE(expression)

--================================================================

--T008\_03\_01

----The default setting

PRINT ISDATE('I am not DateTime');

-- returns 0

PRINT ISDATE(GETDATE());

-- returns 1

PRINT ISDATE('2017-09-08 18:54:32.033');

-- returns 1

PRINT ISDATE('22017-09-08 18:54:32.0352403');

-- returns 0, beucase ISDATE() does not work for "DateTime2" data type value

--================================================================

--T008\_03\_02

--SET LANGUAGE us\_english;  and  SET DATEFORMAT MDY;

--The U.S. English default is mdy

SET LANGUAGE us\_english;

SET DATEFORMAT MDY;

PRINT ISDATE('04/15/2008');

--Returns 1.

PRINT ISDATE('04-15-2008');

--Returns 1.

PRINT ISDATE('04.15.2008');

--Returns 1.

PRINT ISDATE('04/2008/15');

--Returns 1.  --> Non sense

SET DATEFORMAT MDY;

PRINT ISDATE('15/04/2008');

--Returns 0.

PRINT ISDATE('15/2008/04');

--Returns 0.

PRINT ISDATE('2008/15/04');

--Returns 0.

PRINT ISDATE('2008/04/15');

--Returns 1.    --> Non sense

--================================================================

--T008\_03\_03

--SET DATEFORMAT DMY;

--For Australia

SET DATEFORMAT DMY;

PRINT ISDATE('15/04/2008');

--Returns 1

PRINT ISDATE('15/2008/04');

--Returns 1. --> Non sense.

PRINT ISDATE('2008/15/04');

--Returns 1. --> Non sense.

PRINT ISDATE('2008/04/15');

--Returns 0.

PRINT ISDATE('04/15/2008');

--Returns 0.

--================================================================

--T008\_03\_04

--SET DATEFORMAT DYM;

--Not common

SET DATEFORMAT DYM;

PRINT ISDATE('15/2008/04');

--Returns 1.

PRINT ISDATE('15/04/2008');

--Returns 1.  --> Non sense.

PRINT ISDATE('04/15/2008');

--Returns 0.

PRINT ISDATE('2008/04/15');

--Returns 0.

--================================================================

--T008\_03\_05

--SET DATEFORMAT YDM;

--Not common

SET DATEFORMAT YDM;

PRINT ISDATE('2008/15/04');

--Returns 1.

PRINT ISDATE('15/2008/04');

--Returns 1.  --> Non sense.

PRINT ISDATE('15/04/2008');

--Returns 1.  --> Non sense.

PRINT ISDATE('04/15/2008');

--Returns 0.

PRINT ISDATE('2008/04/15');

--Returns 0.

--================================================================

--T008\_03\_06

--SET DATEFORMAT YMD;

--Very common in most country.

SET DATEFORMAT YMD;

PRINT ISDATE('2008/04/15');

--Returns 1.

PRINT ISDATE('2008/15/04');

--Returns 0.

PRINT ISDATE('15/2008/04');

--Returns 0.

PRINT ISDATE('15/04/2008');

--Returns 0.

PRINT ISDATE('04/15/2008');

--Returns 1.  --> Non sense.

--================================================================

--T008\_03\_07

--SET LANGUAGE English;

SET LANGUAGE English;

PRINT ISDATE('15/04/2008');

--Returns 0.   --> Not for Australia Date format  DMY

PRINT ISDATE('2008/04/15');

--Returns 1.  --> For USA

PRINT ISDATE('2008/15/04');

--Returns 0.

PRINT ISDATE('15/2008/04');

--Returns 0.

PRINT ISDATE('04/15/2008');

--Returns 1.  --> For USA

--================================================================

--T008\_03\_08

--SET LANGUAGE Hungarian;

--The result is same as   SET LANGUAGE English;

SET LANGUAGE Hungarian;

PRINT ISDATE('15/04/2008');

--Returns 0.   --> Not for Australia Date format  DMY

PRINT ISDATE('2008/04/15');

--Returns 1.  --> For USA

PRINT ISDATE('2008/15/04');

--Returns 0.

PRINT ISDATE('15/2008/04');

--Returns 0.

PRINT ISDATE('04/15/2008');

--Returns 1.  --> For USA

--================================================================

--T008\_03\_09

--SET LANGUAGE Swedish;

--The result is same as   SET LANGUAGE English;

SET LANGUAGE Swedish;

PRINT ISDATE('15/04/2008');

--Returns 0.   --> Not for Australia Date format  DMY

PRINT ISDATE('2008/04/15');

--Returns 1.  --> For USA

PRINT ISDATE('2008/15/04');

--Returns 0.

PRINT ISDATE('15/2008/04');

--Returns 0.

PRINT ISDATE('04/15/2008');

--Returns 1.  --> For USA

--================================================================

--T008\_03\_10

--SET LANGUAGE Italian;

SET LANGUAGE Italian;

PRINT ISDATE('15/04/2008');

--Returns 1.  --> For Australia

PRINT ISDATE('2008/04/15');

--Returns 0.  --> Not For USA

PRINT ISDATE('2008/15/04');

--Returns 1.

PRINT ISDATE('15/2008/04');

--Returns 1.

PRINT ISDATE('04/15/2008');

--Returns 0.  --> Not For USA

--================================================================

--T008\_03\_11

PRINT ISDATE(3000);

--Returns 1.   --> Non Sense

PRINT ISDATE(2000);

--Returns 1.   --> Non Sense

PRINT ISDATE(100);

--Returns 0.

--================================================================

--T008\_03\_12

-- Set back to default setting.

SET LANGUAGE us\_english;

SET DATEFORMAT MDY;

/\*

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

2.

Date and Time data types

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql>

2.1.

[DATETIME]

Format:  YYYY-MM-DD hh:mm:ss[.nnn]

Range:  1753-01-01 through 9999-12-31

Accuracy: 0.00333 second

E.g.

2017-09-08 18:54:32.033

2.2.

[DATETIME2](7)

Format:  YYYY-MM-DD hh:mm:ss[.nnnnnnn]

[TIME](7)  means 7 n, [TIME](6) means 6 n

Range:  0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999

Accuracy: 100 nanosecond(ns).  1 nanosecond(ns) = 10E-9 second

E.g.

22017-09-08 18:54:32.0352403

3.

ISDATE(expression)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/isdate-transact-sql>

Returns 1 if the expression is a valid date, time, or datetime value; otherwise, 0.

For datetime2 values, IsDate returns ZERO.

4.

In samarry of ISDATE(expression)

ISDATE(expression)  sometimes work, and sometimes does not work.

I personally don't recommend to use it to check if the date is valid.

I personally suggest create a function to restrict to the format YYYY/MM/DD.

It is always easer to convert the format in C# application,

and harder to convert the format in SQL.

\*/

==================================================

4. DatePart\_DateName\_Day\_Month\_Year

--================================================================

--T008\_04\_DatePart\_DateName\_Day\_Month\_Year

--================================================================

--================================================================

--T008\_04\_01

--Day(date)

PRINT DAY(GETDATE());

-- Returns the day number of the current month.

PRINT DAY('01/31/2017');

-- Returns 31

PRINT DAY('31/01/2017');

-- ERROR: Conversion failed when converting date and/or time from character string.

PRINT DAY('2017/01/31');

-- Returns 31

/\*

Day(date)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/day-transact-sql>

Returns an integer representing the day (day of the month) of the specified date.

\*/

--================================================================

--T008\_04\_02

--MONTH(date)

PRINT MONTH(GETDATE());

-- Returns the day number of the current month.

PRINT MONTH('01/31/2017');

-- Returns 1

PRINT MONTH('31/01/2017');

-- ERROR: Conversion failed when converting date and/or time from character string.

PRINT MONTH('2017/01/31');

-- Returns 1

/\*

MONTH(date)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/month-transact-sql>

Returns an integer that represents the month of the specified date.

\*/

--================================================================

--T008\_04\_03

--YEAR(date)

PRINT YEAR(GETDATE());

-- Returns the day number of the current month.

PRINT YEAR('01/31/2017');

-- Returns 2017

PRINT YEAR('31/01/2017');

-- ERROR: Conversion failed when converting date and/or time from character string.

PRINT YEAR('2017/01/31');

-- Returns 2017

/\*

YEAR(date)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/year-transact-sql>

Returns an integer that represents the year of the specified date.

\*/

--================================================================

--T008\_04\_04

--DATENAME(datepart , date)

PRINT DATENAME(YEAR, '2017-09-08 18:54:32.0352403');

-- Returns 2017

PRINT DATENAME(QUARTER, '2017-09-08 18:54:32.0352403');

-- Returns 3, January to March is QUARTER 1, April to June is QUARTER 2

PRINT DATENAME(MONTH, '2017-09-08 18:54:32.0352403');

-- Returns September

PRINT DATENAME(DAYOFYEAR, '2017-09-08 18:54:32.0352403');

-- Returns 251

PRINT DATENAME(DAY, '2017-09-08 18:54:32.0352403');

-- Returns 8

PRINT DATENAME(WEEK, '2017-09-08 18:54:32.0352403');

-- Returns 36

PRINT DATENAME(WEEKDAY, '2017-09-08 18:54:32.0352403');

-- Returns Friday, DATEPART will return 6

PRINT DATENAME(HOUR, '2017-09-08 18:54:32.0352403');

-- Returns 18

PRINT DATENAME(MINUTE, '2017-09-08 18:54:32.0352403');

-- Returns 54

PRINT DATENAME(SECOND, '2017-09-08 18:54:32.0352403');

-- Returns 32

PRINT DATENAME(MILLISECOND, '2017-09-08 18:54:32.0352403');

-- Returns 35

PRINT DATENAME(MICROSECOND, '2017-09-08 18:54:32.0352403');

-- Returns 35240

PRINT DATENAME(NANOSECOND, '2017-09-08 18:54:32.0352403');

-- Returns 35240300

PRINT DATENAME(TZoffset, '2017-09-08 17:03:34.2715896 +10:00');

-- Returns +10:00

/\*

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.

DATENAME(datepart , date)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/datename-transact-sql>

Returns a character string that represents the specified datepart of the specified date

3.

DATEPART ( datepart , date )

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/datepart-transact-sql>

Returns an integer that represents the specified datepart of the specified date.

\*/

--================================================================

--T008\_04\_05

--DATEPART(datepart , date)

PRINT DATEPART(YEAR, '2017-09-08 18:54:32.0352403');

-- Returns 2017

PRINT DATEPART(QUARTER, '2017-09-08 18:54:32.0352403');

-- Returns 3, January to March is QUARTER 1, April to June is QUARTER 2

PRINT DATEPART(MONTH, '2017-09-08 18:54:32.0352403');

-- Returns 9

PRINT DATEPART(DAYOFYEAR, '2017-09-08 18:54:32.0352403');

-- Returns 251

PRINT DATEPART(DAY, '2017-09-08 18:54:32.0352403');

-- Returns 8

PRINT DATEPART(WEEK, '2017-09-08 18:54:32.0352403');

-- Returns 36

PRINT DATEPART(WEEKDAY, '2017-09-08 18:54:32.0352403');

-- Returns 6, DATENAME will return Friday, Sunday is 1, Saturday is 7.

PRINT DATEPART(HOUR, '2017-09-08 18:54:32.0352403');

-- Returns 18

PRINT DATEPART(MINUTE, '2017-09-08 18:54:32.0352403');

-- Returns 54

PRINT DATEPART(SECOND, '2017-09-08 18:54:32.0352403');

-- Returns 32

PRINT DATEPART(MILLISECOND, '2017-09-08 18:54:32.0352403');

-- Returns 35

PRINT DATEPART(MICROSECOND, '2017-09-08 18:54:32.0352403');

-- Returns 35240

PRINT DATEPART(NANOSECOND, '2017-09-08 18:54:32.0352403');

-- Returns 35240300

PRINT DATEPART(TZoffset, '2017-09-08 17:03:34.2715896 +10:00');

-- Returns 600

/\*

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.

DATENAME(datepart , date)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/datename-transact-sql>

Returns a character string that represents the specified datepart of the specified date

3.

DATEPART ( datepart , date )

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/datepart-transact-sql>

Returns an integer that represents the specified datepart of the specified date.

\*/

==================================================

5. DATEADD(datepart,number,date)

--================================================================

--T008\_05\_DATEADD(datepart,number,date)

--================================================================

--================================================================

--T008\_05\_01

--PRINT

PRINT DATEADD(DAY, 20, '2017-08-30 19:45:31.793');

-- Returns Sep 19 2017  7:45PM (Add 20 days)

PRINT DATEADD(DAY, -20, '2017-08-30 19:45:31.793');

-- Returns Aug 10 2017  7:45PM   (Minus 20 days)

PRINT DATEADD(MONTH, 1, '2017-08-30 19:45:31.793');

--Returns Sep 30 2017  7:45PM

PRINT DATEADD(MONTH, -1, '2017-08-30 19:45:31.793');

--Returns Jul 30 2017  7:45PM

PRINT DATEADD(YEAR, 2, '2017-08-30 19:45:31.793');

--Returns Aug 30 2019  7:45PM

PRINT DATEADD(YEAR, -2, '2017-08-30 19:45:31.793');

--Returns Aug 30 2015  7:45PM

--================================================================

--T008\_05\_02

--SELECT

SELECT  DATEADD(DAY, 20, '2017-08-30 19:45:31.793');

-- Returns 2017-09-19 19:45:31.793      (Add 20 days)

SELECT  DATEADD(DAY, -20, '2017-08-30 19:45:31.793');

-- Returns 2017-08-10 19:45:31.793      (Minus 20 days)

SELECT  DATEADD(MONTH, 1, '2017-08-30 19:45:31.793');

--Returns 2017-09-30 19:45:31.793

SELECT  DATEADD(MONTH, -1, '2017-08-30 19:45:31.793');

--Returns 2017-07-30 19:45:31.793

SELECT  DATEADD(YEAR, 2, '2017-08-30 19:45:31.793');

--Returns 2019-08-30 19:45:31.793

SELECT  DATEADD(YEAR, -2, '2017-08-30 19:45:31.793');

--Returns 2015-08-30 19:45:31.793

/\*

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.

DATEADD(datepart,number,date)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/dateadd-transact-sql>

Returns a specified date with the specified number interval (signed integer)

added to a specified datepart of that date.

\*/

==================================================

6. DATEDIFF(datepart,startdate,enddate)

--================================================================

--T008\_06\_DATEDIFF(datepart,startdate,enddate)

--================================================================

SELECT  DATEDIFF(MONTH, '2017/01/31', '2017/05/31');

 -- returns 4

SELECT  DATEDIFF(DAY, '2017/01/31', '2017/05/31');

GO

 -- returns 120

/\*

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.

DATEDIFF(datepart,startdate,enddate)

Reference:

<https://docs.microsoft.com/en-us/sql/t-sql/functions/datediff-transact-sql>

Returns the count (signed integer) of the specified datepart boundaries crossed

between the specified startdate and enddate.

\*/

==================================================

7. fnDurationByDate

--================================================================

--T008\_07\_01

--fnDurationByDate

/\*

/// <summary>

/// Input a date, then return the string value of duration between that date to today.

/// E.g. 33 Years 5 Months 14 Days

/// </summary>

/// <param name="Date">The input date</param>

/// <returns>The string value of duration between that date to today </returns>

\*/

--If function exists then DROP it

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'FUNCTION'

                        AND LEFT(ROUTINE\_NAME, 2) NOT IN ( '@@' )

                        AND SPECIFIC\_NAME = 'fnDurationByDate' ) )

    BEGIN

        DROP FUNCTION fnDurationByDate;

    END;

GO -- Run the previous command and begins new batch

CREATE FUNCTION fnDurationByDate ( @Date DATETIME )

RETURNS NVARCHAR(50)

AS

    BEGIN

        DECLARE @tempdate DATETIME ,

            @years INT ,

            @months INT ,

            @days INT;

        SELECT  @tempdate = @Date;

             -- Caculate Years

        SELECT  @years = DATEDIFF(YEAR, @tempdate, GETDATE())

                - CASE WHEN ( MONTH(@Date) > MONTH(GETDATE()) )

                            OR ( MONTH(@Date) = MONTH(GETDATE())

                                 AND DAY(@Date) > DAY(GETDATE())

                               ) THEN 1

                       ELSE 0

                  END;

        SELECT  @tempdate = DATEADD(YEAR, @years, @tempdate);

             -- Caculate Months

        SELECT  @months = DATEDIFF(MONTH, @tempdate, GETDATE())

                - CASE WHEN DAY(@Date) > DAY(GETDATE()) THEN 1

                       ELSE 0

                  END;

        SELECT  @tempdate = DATEADD(MONTH, @months, @tempdate);

             -- Caculate Days

        SELECT  @days = DATEDIFF(DAY, @tempdate, GETDATE());

        DECLARE @Duration NVARCHAR(50);

        SET @Duration = CAST(@years AS NVARCHAR(4)) + ' Years '

            + CAST(@months AS NVARCHAR(2)) + ' Months '

            + CAST(@days AS NVARCHAR(2)) + ' Days';

        RETURN @Duration;

    END;

GO -- Run the prvious command and begins new batch

--================================================================

--T008\_07\_02

--fnDurationByDate2

/\*

/// <summary>

/// Input a date, then return the string value of duration between that date to today.

/// E.g. 33 Years 5 Months 14 Days

/// </summary>

/// <param name="Date">The input date</param>

/// <returns>The string value of duration between that date to today </returns>

\*/

--If function exists then DROP it

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'FUNCTION'

                        AND LEFT(ROUTINE\_NAME, 2) NOT IN ( '@@' )

                        AND SPECIFIC\_NAME = 'fnDurationByDate2' ) )

    BEGIN

        DROP FUNCTION fnDurationByDate2;

    END;

GO -- Run the previous command and begins new batch

CREATE FUNCTION fnDurationByDate2 ( @Date DATETIME )

RETURNS NVARCHAR(50)

AS

    BEGIN

        DECLARE @tempdate DATETIME ,

            @years INT ,

            @months INT ,

            @days INT;

        SET @tempdate = @Date;

             -- Caculate Years

        IF ( MONTH(@Date) > MONTH(GETDATE()) )

            OR ( MONTH(@Date) = MONTH(GETDATE())

                 AND DAY(@Date) > DAY(GETDATE())

               )

            BEGIN

                SET @years = DATEDIFF(YEAR, @tempdate, GETDATE()) - 1;

            END;

        ELSE

            BEGIN

                SET @years = DATEDIFF(YEAR, @tempdate, GETDATE());

            END;

             -- Caculate Months

        SET @tempdate = DATEADD(YEAR, @years, @tempdate);

        IF DAY(@Date) > DAY(GETDATE())

            BEGIN

                SET @months = DATEDIFF(MONTH, @tempdate, GETDATE()) - 1;

            END;

        ELSE

            BEGIN

                SET @months = DATEDIFF(MONTH, @tempdate, GETDATE());

            END;

             -- Caculate Days

        SET @tempdate = DATEADD(MONTH, @months, @tempdate);

        SET @days = DATEDIFF(DAY, @tempdate, GETDATE());

        DECLARE @Duration NVARCHAR(50);

        SET @Duration = CAST(@years AS NVARCHAR(4)) + ' Years '

            + CAST(@months AS NVARCHAR(2)) + ' Months '

            + CAST(@days AS NVARCHAR(2)) + ' Days';

        RETURN @Duration;

    END;

GO -- Run the prvious command and begins new batch

PRINT [dbo].fnDurationByDate('1984/11/26');

PRINT [dbo].fnDurationByDate2('1984-11-26');

--32 Years 9 Months 14 Days

PRINT [dbo].fnDurationByDate('1984/09/10');

PRINT [dbo].fnDurationByDate2('1984-09-10');

--32 Years 11 Months 30 Days

PRINT [dbo].fnDurationByDate('1984/09/09');

PRINT [dbo].fnDurationByDate2('1984-09-09');

--33 Years 0 Months 0 Days

PRINT [dbo].fnDurationByDate('1984/09/08');

PRINT [dbo].fnDurationByDate2('1984-09-08');

--33 Years 0 Months 1 Days

DECLARE @tempdate2 DATETIME;

SET @tempdate2 = CAST('1984/11/26' AS DATETIME);

PRINT @tempdate2;

--Nov 26 1984 12:00AM

SET @tempdate2 = DATEADD(YEAR, 32, @tempdate2);

PRINT @tempdate2;

--Nov 26 2016 12:00AM

SET @tempdate2 = DATEADD(MONTH, 9, @tempdate2);

PRINT @tempdate2;

--Aug 26 2017 12:00AM

SET @tempdate2 = DATEADD(DAY, 14, @tempdate2);

PRINT @tempdate2;

GO -- Run the previous command and begins new batch

--Sep  9 2017 12:00AM

/\*

I assume today is 2017/09/09  (YYYY/MM/DD)

I assume inputDate is 1984/11/26  (YYYY/MM/DD)

The difference shoud be '32 Years 9 Months 14 Days'

\*/

/\*

1.

---- Caculate Years

--IF ( MONTH(@Date) > MONTH(GETDATE()) )

--    OR ( MONTH(@Date) = MONTH(GETDATE())

--            AND DAY(@Date) > DAY(GETDATE())

--        )

--    BEGIN

--        SET @years = DATEDIFF(YEAR, @tempdate, GETDATE()) - 1;

--    END;

--ELSE

--    BEGIN

--        SET @years = DATEDIFF(YEAR, @tempdate, GETDATE());

--    END;

--SET @tempdate = DATEADD(YEAR, @years, @tempdate);

1.1.

I assume today is 2017/09/09  (YYYY/MM/DD)

I assume inputDate is 1984/11/26  (YYYY/MM/DD)

Shoud return '32 Years 9 Months 14 Days'

but 2017-1984=33, thus, It should minus 1, 33-1=32

1.2.

I assume today is 2017/09/09  (YYYY/MM/DD)

I assume inputDate is 1984/09/10  (YYYY/MM/DD)

Shoud return '32 Years 11 Months 30 Days'

but 2017-1984=33, thus, It should minus 1, 33-1=32

1.3.

I assume today is 2017/09/09  (YYYY/MM/DD)

I assume inputDate is 1984/09/09  (YYYY/MM/DD)

Shoud return '33 Years 0 Months 0 Dayss'

2017-1984=33

1.4.

I assume today is 2017/09/09  (YYYY/MM/DD)

I assume inputDate is 1984/09/08  (YYYY/MM/DD)

Should return 33 Years 0 Months 1 Days

2017-1984=33

1.5.

In Summary, when caculating the "Years"

--IF ( MONTH(@Date) > MONTH(GETDATE()) )

--    OR ( MONTH(@Date) = MONTH(GETDATE())

--            AND DAY(@Date) > DAY(GETDATE())

--        )

If the Month and Day of inputDate is later than the Month and Day of currentDate

Then the years is  DATEDIFF(YEAR, @tempdate, GETDATE()) - 1

If the Month and Day of inputDate is earlier than the Month and Day of currentDate

Then the years is  DATEDIFF(YEAR, @tempdate, GETDATE())

2.

---- Caculate Months

--SET @tempdate = DATEADD(YEAR, @years, @tempdate);

--IF DAY(@Date) > DAY(GETDATE())

--    BEGIN

--        SET @months = DATEDIFF(MONTH, @tempdate, GETDATE()) - 1;

--    END;

--ELSE

--    BEGIN

--        SET @months = DATEDIFF(MONTH, @tempdate, GETDATE());

--    END;

2.1.

--SET @tempdate = DATEADD(YEAR, @years, @tempdate);

After we get the years, then we add the years to TempDate which was originally currentDate.

Then the different between @tempdate and currentDate should be less than 1 year.

The @tempdate is less than 1 year means between 0 Months 0 days to 11 months and 30 Days.

2.2.

In Summary, when caculating the "Months"

--IF DAY(@Date) > DAY(GETDATE())

If the Day of inputDate is later than the Day of currentDate

Then the Month is  DATEDIFF(MONTH, @tempdate, GETDATE()) - 1

If the Day of inputDate is earlier than the Day of currentDate

Then the Month is  DATEDIFF(MONTH, @tempdate, GETDATE())

3.

---- Caculate Days

--SET @tempdate = DATEADD(MONTH, @months, @tempdate);

--SET @days = DATEDIFF(DAY, @tempdate, GETDATE());

3.1.

--SET @tempdate = DATEADD(YEAR, @years, @tempdate);

...

--SET @tempdate = DATEADD(MONTH, @months, @tempdate);

After we get the Months and Years, then we add the Months and Years to TempDate which was originally currentDate.

Then the different between @tempdate and currentDate should be less than the Days.

3.1.1.

--DECLARE @tempdate2 DATETIME;

--SET @tempdate2 = CAST('1984/11/26' AS DATETIME);

--PRINT @tempdate2

----Nov 26 1984 12:00AM

--SET @tempdate2 = DATEADD(YEAR, 32, @tempdate2);

--PRINT @tempdate2

----Nov 26 2016 12:00AM

--SET @tempdate2 = DATEADD(MONTH, 9, @tempdate2);

--PRINT @tempdate2

----Aug 26 2017 12:00AM

--SET @tempdate2 = DATEADD(DAY, 14, @tempdate2);

--PRINT @tempdate2

----Sep  9 2017 12:00AM

I assume today is 2017/09/09  (YYYY/MM/DD)

I assume inputDate is 1984/11/26  (YYYY/MM/DD)

Shoud return '32 Years 9 Months 14 Days'

but 2017-1984=33, thus, It should minus 1, 33-1=32

Nov to Sep is 10 months different, but, it should minus 1, 10-1=9

Then 1984/11/26 add 32 yaers and 10 Month

Thus, we add 32 years and 9 month into the inputDate.

The the difference between inputDate and CurrentDate will be less than 30 adys.

Thus,

--SET @days = DATEDIFF(DAY, @tempdate, GETDATE());

To caculate the date, we do not need the if statmet to minus 1 any more.

Go straight to get the DATEDIFF to get Days.

\*/

==================================================

8. Get Random DateTime

--================================================================

--T008\_08\_Get Random DateTime

--================================================================

--Get Random DateTime

--Reference: <http://crodrigues.com/sql-server-generate-random-datetime-within-a-range/>

DECLARE @RandomDateTime DATETIME;

DECLARE @DateFrom DATETIME = '2012-01-01';

DECLARE @DateTo DATETIME = '2017-06-30';

DECLARE @DaysRandom INT= 0;

DECLARE @MillisRandom INT= 0;

--get random number of days

 SELECT  @DaysRandom = DATEDIFF(DAY, @DateFrom, @DateTo);

SELECT  @DaysRandom = ROUND(( ( @DaysRandom - 1 ) \* RAND() ), 0);

--get random millis

SELECT  @MillisRandom = ROUND(( ( 99999999 ) \* RAND() ), 0);

SELECT  @RandomDateTime = DATEADD(DAY, @DaysRandom, @DateFrom);

SELECT  @RandomDateTime = DATEADD(MILLISECOND, @MillisRandom, @RandomDateTime);

SELECT  @RandomDateTime;

GO -- Run the previous command and begins new batch

==================================================

9. Get Random DateTime stored procedure

--================================================================

--T008\_09\_Get Random DateTime stored procedure

--================================================================

/\*

1.

Get Random DateTime

Reference: <http://crodrigues.com/sql-server-generate-random-datetime-within-a-range/>

2.

You can not use RAND() in function,

but you can use RAND() in Store procedure and return view.

\*/

--Drop Store Procedure if it exist

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spGetRandomDate' ) )

    BEGIN

        DROP PROCEDURE spGetRandomDate;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spGetRandomDate

(

  @DateFrom DATETIME ,

  @DateTo DATETIME ,

  @RandomDateTime DATETIME OUTPUT

     --@parameterB INT OUT

) --WITH ENCRYPTION

AS

    BEGIN

             --DECLARE @RandomDateTime DATETIME;

             --DECLARE @DateFrom DATETIME = '2012-01-01';

             --DECLARE @DateTo DATETIME = '2017-06-30';

        DECLARE @DaysRandom INT= 0;

        DECLARE @MillisRandom INT= 0;

             --get random number of days

        SELECT  @DaysRandom = DATEDIFF(DAY, @DateFrom, @DateTo);

        SELECT  @DaysRandom = ROUND(( ( @DaysRandom - 1 ) \* RAND() ), 0);

             --get random millis

        SELECT  @MillisRandom = ROUND(( ( 99999999 ) \* RAND() ), 0);

        SELECT  @RandomDateTime = DATEADD(DAY, @DaysRandom, @DateFrom);

        SELECT  @RandomDateTime = DATEADD(MILLISECOND, @MillisRandom,

                                          @RandomDateTime);

        SELECT  @RandomDateTime;

    END;

GO -- Run the previous command and begins new batch

DECLARE @RandomDateTime DATETIME;

DECLARE @DateFrom DATETIME = '2012-01-01';

DECLARE @DateTo DATETIME = '2017-06-30';

--EXECUTE @RandomDateTime = spGetRandomDate '2012-01-01','2017-06-30',@RandomDateTime

EXECUTE @RandomDateTime = spGetRandomDate @DateFrom, @DateTo,@RandomDateTime

PRINT @RandomDateTime;

GO -- Run the previous command and begins new batch

==================================================

10. Clean up

--================================================================

--T008\_10\_Clean up

--================================================================

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'DateTimeTypes' ) )

    BEGIN

        TRUNCATE TABLE DateTimeTypes;

        DROP TABLE DateTimeTypes;

    END;

GO -- Run the previous command and begins new batch

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

--Drop Store Procedure if it exist

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spGetRandomDate' ) )

    BEGIN

        DROP PROCEDURE spGetRandomDate;

    END;

GO -- Run the previous command and begins new batch

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

--If function exists then DROP it

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'FUNCTION'

                        AND LEFT(ROUTINE\_NAME, 2) NOT IN ( '@@' )

                        AND SPECIFIC\_NAME = 'fnDurationByDate2' ) )

    BEGIN

        DROP FUNCTION fnDurationByDate2;

    END;

GO -- Run the previous command and begins new batch

--If function exists then DROP it

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'FUNCTION'

                        AND LEFT(ROUTINE\_NAME, 2) NOT IN ( '@@' )

                        AND SPECIFIC\_NAME = 'fnDurationByDate' ) )

    BEGIN

        DROP FUNCTION fnDurationByDate;

    END;

GO -- Run the previous command and begins new batch