15,657,589 members Sign in

CODE PROJECT For those who code

articles Q&A forums features

Sign in to codeproject.com with Google

Raj Dalvi
rajdalvi529@gmail.com

2020BTECS00027 DALVI RAJ DATTATRAY
raj.dalvi@walchandsangli.ac.in

1 more account

Create and Populate Date Dimension for Data Warehouse

Mubin M. Shaikh
4 Sep 2013 CPOL

Rate me: 4.88/5 (77 votes)

This article shows how to create and populate date dimension for data warehouse.

Download script - 5.3 KB

Introduction

This article will especially help those people who work in Data warehouse and Business Intelligence. Whenever as a starting point, they need to set New Data warehouse, during this time they need to create and fill their Date Dimension with various values of Date, Date Keys, Day Type, Day Name Of Week, Month, Month Name, Quarter, etc.

Date dimension plays an important role in your data warehouse designing, it provides the ability to study behavior and trend of your data over a period of time.

You can study your data by grouping them using various fields of date dimension.

For example:

If I want to analyze my data of total sales by each month of the year, or show total sales by each quarter of the year, or show me on which days total sales takes place more in the entire year or month.

After implementing the complete solution in data warehouse, the relationship of date dimension gives you all this facility to do slice and dice of your data.

So as an initial step, you need to design your date dimension, time dimension and populate them with range of values.

For designing of time dimension, you can refer to my other tip posted on CodeProject, “Design and Populate Time Dimension with 24 Hour plus Values”.

This date dimension will have values of date stored as per various date formats used across the world, like “dd-MM-yyyy” is used in Europe, UK, India, etc. while “MM-dd-yyyy” format is used in US.

Even some countries refer to Monday as 1st day of the week like UK, and some of them refer to Sunday as the 1st day of the week like US, so I have placed both types of values in this date dimension so that it can be utilized as per the need.

National holiday list is also different in both the countries US and UK, script is there to update date dimension with these values, it will help to study data on a particular national holiday.

I expect some intermediate knowledge of T-SQL from the reader to understand and to use the script given below, even I have placed steps to follow to achieve a particular task , T-SQL script also contains in line comments to explain the purpose of various steps.

This article contains two scripts:

Script 1

This script will create date dimension table for you and populate it with all standard values. Please refer to figure 1. You need to follow 4 easy steps given under section of the Script 1.

	DateKey	FullDateUK	FullDateUSA	DayOfMonth	DayName	DayOfWeekUK	DayOfWeekUSA	DayOfQuarter	WeekOfMonth	WeekOfQuarter	MonthName	MONTH	Quarter	YEAR	MonthYear	IsHolidayUK	HolidayUK	IsHolidayUSA	HolidayUSA
1	20130101	01/01/2013	01/01/2013	1	Tuesday	2	3	1	1	1	January	1	1	2013	Jan-2013	1	New Year's Day	1	New Year's Day
2	20130102	02/01/2013	01/02/2013	2	Wednesday	3	4	1	1	1	January	1	1	2013	Jan-2013	0	NULL	0	NULL
3	20130103	03/01/2013	01/03/2013	3	Thursday	4	5	1	1	1	January	1	1	2013	Jan-2013	0	NULL	0	NULL
4	20130104	04/01/2013	01/04/2013	4	Friday	5	6	1	1	1	January	1	1	2013	Jan-2013	0	NULL	0	NULL
5	20130105	05/01/2013	01/05/2013	5	Saturday	6	7	1	1	1	January	1	1	2013	Jan-2013	0	NULL	0	NULL
6	20130106	06/01/2013	01/06/2013	6	Sunday	7	1	1	2	1	January	1	1	2013	Jan-2013	0	NULL	0	NULL
7	20130107	07/01/2013	01/07/2013	7	Monday	1	2	1	2	1	January	1	1	2013	Jan-2013	0	NULL	0	NULL
8	20130108	08/01/2013	01/08/2013	8	Tuesday	2	3	2	2	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
9	20130109	09/01/2013	01/09/2013	9	Wednesday	3	4	2	2	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
10	20130110	10/01/2013	01/10/2013	10	Thursday	4	5	2	2	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
11	20130111	11/01/2013	01/11/2013	11	Friday	5	6	2	2	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
12	20130112	12/01/2013	01/12/2013	12	Saturday	6	7	2	2	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
13	20130113	13/01/2013	01/13/2013	13	Sunday	7	1	2	3	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
14	20130114	14/01/2013	01/14/2013	14	Monday	1	2	2	3	2	January	1	1	2013	Jan-2013	0	NULL	0	NULL
15	20130115	15/01/2013	01/15/2013	15	Tuesday	2	3	3	3	3	January	1	1	2013	Jan-2013	0	NULL	0	NULL

Figure 1

Script 2

This script can be used to extend your date dimension with Fiscal Calendar fields like Fiscal Year, Fiscal month, fiscal Quarter, etc and populate these fields with appropriate values, Fiscal Calendar fields can be used to study data as per Financial Year Defined. Please refer to figure 2 for further details. Follow the steps given under section of Script 2 to accomplish this task.

	DateKey	FullDateUK	FullDateUSA	DayOfMonth	DayName	MMYYYY	MonthName	MONTH	YEAR	MonthYear	FirstDayOfMonth	LastDayOfMonth	FiscalDayOfYear	FiscalWeekOfYear	FiscalMMYYYY	FiscalMonth	FiscalQuarter	FiscalYear	FiscalYearName
1	20130101	01/01/2013	01/01/2013	1	Tuesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	360	52	122012	12	4	2012	FY 2012
2	20130102	02/01/2013	01/02/2013	2	Wednesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	361	52	122012	12	4	2012	FY 2012
3	20130103	03/01/2013	01/03/2013	3	Thursday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	362	52	122012	12	4	2012	FY 2012
4	20130104	04/01/2013	01/04/2013	4	Friday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	363	52	122012	12	4	2012	FY 2012
5	20130105	05/01/2013	01/05/2013	5	Saturday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	364	52	122012	12	4	2012	FY 2012
6	20130106	06/01/2013	01/06/2013	6	Sunday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	1	1	012013	1	1	2013	FY 2013
7	20130107	07/01/2013	01/07/2013	7	Monday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	2	1	012013	1	1	2013	FY 2013
8	20130108	08/01/2013	01/08/2013	8	Tuesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	3	1	012013	1	1	2013	FY 2013
9	20130109	09/01/2013	01/09/2013	9	Wednesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	4	1	012013	1	1	2013	FY 2013
10	20130110	10/01/2013	01/10/2013	10	Thursday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	5	1	012013	1	1	2013	FY 2013
11	20130111	11/01/2013	01/11/2013	11	Friday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	6	1	012013	1	1	2013	FY 2013
12	20130112	12/01/2013	01/12/2013	12	Saturday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	7	1	012013	1	1	2013	FY 2013
13	20130113	13/01/2013	01/13/2013	13	Sunday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	8	2	012013	1	1	2013	FY 2013
14	20130114	14/01/2013	01/14/2013	14	Monday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	9	2	012013	1	1	2013	FY 2013
15	20130115	15/01/2013	01/15/2013	15	Tuesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	10	2	012013	1	1	2013	FY 2013

Figure 2

SCRIPT 1

Follow the 4 easy steps in sequence to create and populate date dimension with values.

Using the Code

Follow the given steps to create date dimension:

1. Open SQL Server Management Studio
2. Connect Database Engine
3. Open New query editor
4. Copy paste scripts given below in various steps in new query editor window one by one
5. To run the given SQL script, press F5

Step 1

Please refer to the inline comments given with T-SQL Script for further explanation of each field in table and which type of values it will hold.

Create Table for Date Dimension

SQL Shrink ▲

```
BEGIN TRY
    DROP TABLE [dbo].[DimDate]
END TRY

BEGIN CATCH
    /*No Action*/
END CATCH

/*****************************************************************************/

CREATE TABLE    [dbo].[DimDate]
    (   [DateKey] INT primary key,
        [Date] DATETIME,
        [FullDateUK] CHAR(10), -- Date in dd-MM-yyyy format
        [FullDateUSA] CHAR(10),-- Date in MM-dd-yyyy format
        [DayOfMonth] VARCHAR(2), -- Field will hold day number of Month
        [DaySuffix] VARCHAR(4), -- Apply suffix as 1st, 2nd ,3rd etc
        [DayName] VARCHAR(9), -- Contains name of the day, Sunday, Monday
        [DayOfWeekUSA] CHAR(1),-- First Day Sunday=1 and Saturday=7
        [DayOfWeekUK] CHAR(1),-- First Day Monday=1 and Sunday=7
        [DayOfWeekInMonth] VARCHAR(2), --1st Monday or 2nd Monday in Month
        [DayOfWeekInYear] VARCHAR(2),
        [DayOfQuarter] VARCHAR(3),
        [DayOfYear] VARCHAR(3),
        [WeekOfMonth] VARCHAR(1),-- Week Number of Month
        [WeekOfQuarter] VARCHAR(2), --Week Number of the Quarter
        [WeekOfYear] VARCHAR(2),--Week Number of the Year
        [Month] VARCHAR(2), --Number of the Month 1 to 12
```

```
        [MonthName] VARCHAR(9),--January, February etc
        [MonthOfQuarter] VARCHAR(2),-- Month Number belongs to Quarter
        [Quarter] CHAR(1),
        [QuarterName] VARCHAR(9),--First,Second..
        [Year] CHAR(4),-- Year value of Date stored in Row
        [YearName] CHAR(7), --CY 2012,CY 2013
        [MonthYear] CHAR(10), --Jan-2013,Feb-2013
        [MMYYYY] CHAR(6),
        [FirstDayOfMonth] DATE,
        [LastDayOfMonth] DATE,
        [FirstDayOfQuarter] DATE,
        [LastDayOfQuarter] DATE,
        [FirstDayOfYear] DATE,
        [LastDayOfYear] DATE,
        [IsHolidayUSA] BIT,-- Flag 1=National Holiday, 0-No National Holiday
        [IsWeekday] BIT,-- 0=Week End ,1=Week Day
        [HolidayUSA] VARCHAR(50),--Name of Holiday in US
        [IsHolidayUK] BIT Null,-- Flag 1=National Holiday, 0-No National Holiday
        [HolidayUK] VARCHAR(50) Null --Name of Holiday in UK
    )
GO
```

Brief introduction to functions used in script to populate Date Dimension

	Function	Detail (e.g. for 16-Aug-2013)
1	`Select DATEPART(MM, Getdate()) as MonthNumber`	Return Integer Number=8 of Month from Current Date
2	`Select DATEPART(YY , Getdate()) as YearValue`	Return Value of the Year=2013 from Current Date
3	`Select DATEPART(QQ , Getdate()) as QuarterValue`	Return Value of the Quarter=3 for Current Date
4	`Select DATEPART(DW, Getdate()) as DayOfWeekValue`	Return integer Value of day=6 (Friday) in Week for Current Date as per US standard
5	`Select CONVERT (char(8),Getdate(),112)`	Return Key=20130816 Value for current Date
6	`Select CONVERT (char(10),Getdate(),103)`	Return date =16/08/2013 in “dd-MM-yyyy” format, UK, Europe
7	`Select CONVERT (char(10),Getdate(),101)`	Return date=08/16/2013 in “MM-dd-yyyy” format, US
8	`Select DATEPART(DD , Getdate()) as DayOfMonthValue`	Return integer Day=16 Value for Current Date
9	`select DATENAME(DW, Getdate()) AS DayName`	Return Name=Friday of the Day for Current Date.
10	`select DATEPART(WW, Getdate()) AS WeekOfYear`	Returns Value of Week in Year=33

Step 2

Populate Date dimension with values

You can specify start date and end date value of date range which you want to populate in your date dimension.

Please refer to the inline comments given with T-SQL script for further explanation of steps.

SQL Shrink ▲

```
/*****************************************************************************
*****/
--Specify Start Date and End date here
--Value of Start Date Must be Less than Your End Date

DECLARE @StartDate DATETIME = '01/01/2013' --Starting value of Date Range
DECLARE @EndDate DATETIME = '01/01/2015' --End Value of Date Range

--Temporary Variables To Hold the Values During Processing of Each Date of Year
DECLARE
    @DayOfWeekInMonth INT,
    @DayOfWeekInYear INT,
    @DayOfQuarter INT,
    @WeekOfMonth INT,
    @CurrentYear INT,
    @CurrentMonth INT,
    @CurrentQuarter INT

/*Table Data type to store the day of week count for the month and year*/
DECLARE @DayOfWeek TABLE (DOW INT, MonthCount INT, QuarterCount INT, YearCount INT)

INSERT INTO @DayOfWeek VALUES (1, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (2, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (3, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (4, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (5, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (6, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (7, 0, 0, 0)

--Extract and assign various parts of Values from Current Date to Variable

DECLARE @CurrentDate AS DATETIME = @StartDate
SET @CurrentMonth = DATEPART(MM, @CurrentDate)
SET @CurrentYear = DATEPART(YY, @CurrentDate)
SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)

/*****************************************************************************
*****/
--Proceed only if Start Date(Current date ) is less than End date you specified above

WHILE @CurrentDate < @EndDate
BEGIN

/*Begin day of week logic*/

         /*Check for Change in Month of the Current date if Month changed then
          Change variable value*/
    IF @CurrentMonth != DATEPART(MM, @CurrentDate)
```

```
    BEGIN
        UPDATE @DayOfWeek
        SET MonthCount = 0
        SET @CurrentMonth = DATEPART(MM, @CurrentDate)
    END

        /* Check for Change in Quarter of the Current date if Quarter changed then
change
         Variable value*/

    IF @CurrentQuarter != DATEPART(QQ, @CurrentDate)
    BEGIN
        UPDATE @DayOfWeek
        SET QuarterCount = 0
        SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)
    END

        /* Check for Change in Year of the Current date if Year changed then change
         Variable value*/


    IF @CurrentYear != DATEPART(YY, @CurrentDate)
    BEGIN
        UPDATE @DayOfWeek
        SET YearCount = 0
        SET @CurrentYear = DATEPART(YY, @CurrentDate)
    END

        -- Set values in table data type created above from variables

    UPDATE @DayOfWeek
    SET
        MonthCount = MonthCount + 1,
        QuarterCount = QuarterCount + 1,
        YearCount = YearCount + 1
    WHERE DOW = DATEPART(DW, @CurrentDate)

    SELECT
        @DayOfWeekInMonth = MonthCount,
        @DayOfQuarter = QuarterCount,
        @DayOfWeekInYear = YearCount
    FROM @DayOfWeek
    WHERE DOW = DATEPART(DW, @CurrentDate)

/*End day of week logic*/


/* Populate Your Dimension Table with values*/

    INSERT INTO [dbo].[DimDate]
    SELECT

        CONVERT (char(8),@CurrentDate,112) as DateKey,
        @CurrentDate AS Date,
        CONVERT (char(10),@CurrentDate,103) as FullDateUK,
        CONVERT (char(10),@CurrentDate,101) as FullDateUSA,
        DATEPART(DD, @CurrentDate) AS DayOfMonth,
        --Apply Suffix values like 1st, 2nd 3rd etc..
        CASE
            WHEN DATEPART(DD,@CurrentDate) IN (11,12,13) _
            THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
```

```
    WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 1 _
    THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'st'
    WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 2 _
    THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'nd'
    WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 3 _
    THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'rd'
    ELSE CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
    END AS DaySuffix,

DATENAME(DW, @CurrentDate) AS DayName,
DATEPART(DW, @CurrentDate) AS DayOfWeekUSA,

-- check for day of week as Per US and change it as per UK format
CASE DATEPART(DW, @CurrentDate)
    WHEN 1 THEN 7
    WHEN 2 THEN 1
    WHEN 3 THEN 2
    WHEN 4 THEN 3
    WHEN 5 THEN 4
    WHEN 6 THEN 5
    WHEN 7 THEN 6
    END
    AS DayOfWeekUK,

@DayOfWeekInMonth AS DayOfWeekInMonth,
@DayOfWeekInYear AS DayOfWeekInYear,
@DayOfQuarter AS DayOfQuarter,
DATEPART(DY, @CurrentDate) AS DayOfYear,
DATEPART(WW, @CurrentDate) + 1 - DATEPART(WW, CONVERT(VARCHAR, _
DATEPART(MM, @CurrentDate)) + '/1/' + CONVERT(VARCHAR, _
DATEPART(YY, @CurrentDate))) AS WeekOfMonth,
(DATEDIFF(DD, DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0), _
@CurrentDate) / 7) + 1 AS WeekOfQuarter,
DATEPART(WW, @CurrentDate) AS WeekOfYear,
DATEPART(MM, @CurrentDate) AS Month,
DATENAME(MM, @CurrentDate) AS MonthName,
CASE
    WHEN DATEPART(MM, @CurrentDate) IN (1, 4, 7, 10) THEN 1
    WHEN DATEPART(MM, @CurrentDate) IN (2, 5, 8, 11) THEN 2
    WHEN DATEPART(MM, @CurrentDate) IN (3, 6, 9, 12) THEN 3
    END AS MonthOfQuarter,
DATEPART(QQ, @CurrentDate) AS Quarter,
CASE DATEPART(QQ, @CurrentDate)
    WHEN 1 THEN 'First'
    WHEN 2 THEN 'Second'
    WHEN 3 THEN 'Third'
    WHEN 4 THEN 'Fourth'
    END AS QuarterName,
DATEPART(YEAR, @CurrentDate) AS Year,
'CY ' + CONVERT(VARCHAR, DATEPART(YEAR, @CurrentDate)) AS YearName,
LEFT(DATENAME(MM, @CurrentDate), 3) + '-' + CONVERT(VARCHAR, _
DATEPART(YY, @CurrentDate)) AS MonthYear,
RIGHT('0' + CONVERT(VARCHAR, DATEPART(MM, @CurrentDate)),2) + _
CONVERT(VARCHAR, DATEPART(YY, @CurrentDate)) AS MMYYYY,
CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD, _
@CurrentDate) - 1), @CurrentDate))) AS FirstDayOfMonth,
CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD, _
(DATEADD(MM, 1, @CurrentDate)))), DATEADD(MM, 1, _
@CurrentDate)))) AS LastDayOfMonth,
DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0) AS FirstDayOfQuarter,
DATEADD(QQ, DATEDIFF(QQ, -1, @CurrentDate), -1) AS LastDayOfQuarter,
```

```
        CONVERT(DATETIME, '01/01/' + CONVERT(VARCHAR, DATEPART(YY, _
        @CurrentDate))) AS FirstDayOfYear,
        CONVERT(DATETIME, '12/31/' + CONVERT(VARCHAR, DATEPART(YY, _
        @CurrentDate))) AS LastDayOfYear,
        NULL AS IsHolidayUSA,
        CASE DATEPART(DW, @CurrentDate)
            WHEN 1 THEN 0
            WHEN 2 THEN 1
            WHEN 3 THEN 1
            WHEN 4 THEN 1
            WHEN 5 THEN 1
            WHEN 6 THEN 1
            WHEN 7 THEN 0
            END AS IsWeekday,
        NULL AS HolidayUSA, Null, Null

    SET @CurrentDate = DATEADD(DD, 1, @CurrentDate)
END

/*********************************************************************************
*****/

Step 3.
Update Values of Holiday as per UK Government Declaration for National Holiday.

/*Update HOLIDAY fields of UK as per Govt. Declaration of National Holiday*/

-- Good Friday  April 18
    UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Good Friday'
    WHERE [Month] = 4 AND [DayOfMonth]  = 18

-- Easter Monday  April 21
    UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Easter Monday'
    WHERE [Month] = 4 AND [DayOfMonth]  = 21

-- Early May Bank Holiday   May 5
   UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Early May Bank Holiday'
    WHERE [Month] = 5 AND [DayOfMonth]  = 5

-- Spring Bank Holiday  May 26
    UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Spring Bank Holiday'
    WHERE [Month] = 5 AND [DayOfMonth]  = 26

-- Summer Bank Holiday  August 25
    UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Summer Bank Holiday'
    WHERE [Month] = 8 AND [DayOfMonth]  = 25

-- Boxing Day  December 26
    UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Boxing Day'
    WHERE [Month] = 12 AND [DayOfMonth]  = 26

--CHRISTMAS
    UPDATE [dbo].[DimDate]
        SET HolidayUK = 'Christmas Day'
    WHERE [Month] = 12 AND [DayOfMonth]  = 25
```

```
--New Years Day
    UPDATE [dbo].[DimDate]
        SET HolidayUK  = 'New Year''s Day'
    WHERE [Month] = 1 AND [DayOfMonth] = 1

--Update flag for UK Holidays 1= Holiday, 0=No Holiday

    UPDATE [dbo].[DimDate]
        SET IsHolidayUK  = CASE WHEN HolidayUK   IS NULL _
        THEN 0 WHEN HolidayUK   IS NOT NULL THEN 1 END


Step 4.
Update Values of Holiday as per USA Govt. Declaration for National Holiday.

/*Update HOLIDAY Field of USA In dimension*/

    /*THANKSGIVING - Fourth THURSDAY in November*/
    UPDATE [dbo].[DimDate]
        SET HolidayUSA = 'Thanksgiving Day'
    WHERE
        [Month] = 11
        AND [DayOfWeekUSA] = 'Thursday'
        AND DayOfWeekInMonth = 4

    /*CHRISTMAS*/
    UPDATE [dbo].[DimDate]
        SET HolidayUSA = 'Christmas Day'

    WHERE [Month] = 12 AND [DayOfMonth]  = 25

    /*4th of July*/
    UPDATE [dbo].[DimDate]
        SET HolidayUSA = 'Independance Day'
    WHERE [Month] = 7 AND [DayOfMonth] = 4

    /*New Years Day*/
    UPDATE [dbo].[DimDate]
        SET HolidayUSA = 'New Year''s Day'
    WHERE [Month] = 1 AND [DayOfMonth] = 1

    /*Memorial Day - Last Monday in May*/
    UPDATE [dbo].[DimDate]
        SET HolidayUSA = 'Memorial Day'
    FROM [dbo].[DimDate]
    WHERE DateKey IN
        (
        SELECT
            MAX(DateKey)
        FROM [dbo].[DimDate]
        WHERE
            [MonthName] = 'May'
            AND [DayOfWeekUSA]  = 'Monday'
        GROUP BY
            [Year],
            [Month]
        )

    /*Labor Day - First Monday in September*/
    UPDATE [dbo].[DimDate]
```

```
    SET HolidayUSA = 'Labor Day'
FROM [dbo].[DimDate]
WHERE DateKey IN
    (
    SELECT
        MIN(DateKey)
    FROM [dbo].[DimDate]
    WHERE
        [MonthName] = 'September'
        AND [DayOfWeekUSA] = 'Monday'
    GROUP BY
        [Year],
        [Month]
    )

/*Valentine's Day*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'Valentine''s Day'
WHERE
    [Month] = 2
    AND [DayOfMonth] = 14

/*Saint Patrick's Day*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'Saint Patrick''s Day'
WHERE
    [Month] = 3
    AND [DayOfMonth] = 17

/*Martin Luthor King Day - Third Monday in January starting in 1983*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'Martin Luthor King Jr Day'
WHERE
    [Month] = 1
    AND [DayOfWeekUSA]  = 'Monday'
    AND [Year] >= 1983
    AND DayOfWeekInMonth = 3

/*President's Day - Third Monday in February*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'President''s Day'
WHERE
    [Month] = 2
    AND [DayOfWeekUSA] = 'Monday'
    AND DayOfWeekInMonth = 3

/*Mother's Day - Second Sunday of May*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'Mother''s Day'
WHERE
    [Month] = 5
    AND [DayOfWeekUSA] = 'Sunday'
    AND DayOfWeekInMonth = 2

/*Father's Day - Third Sunday of June*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'Father''s Day'
WHERE
    [Month] = 6
    AND [DayOfWeekUSA] = 'Sunday'
    AND DayOfWeekInMonth = 3
```

```sql
/*Halloween 10/31*/
UPDATE [dbo].[DimDate]
    SET HolidayUSA = 'Halloween'
WHERE
    [Month] = 10
    AND [DayOfMonth] = 31

/*Election Day - The first Tuesday after the first Monday in November*/
BEGIN
DECLARE @Holidays TABLE (ID INT IDENTITY(1,1), _
DateID int, Week TINYINT, YEAR CHAR(4), DAY CHAR(2))

    INSERT INTO @Holidays(DateID, [Year],[Day])
    SELECT
        DateKey,
        [Year],
        [DayOfMonth]
    FROM [dbo].[DimDate]
    WHERE
        [Month] = 11
        AND [DayOfWeekUSA] = 'Monday'
    ORDER BY
        YEAR,
        DayOfMonth

    DECLARE @CNTR INT, @POS INT, @STARTYEAR INT, @ENDYEAR INT, @MINDAY INT

    SELECT
        @CURRENTYEAR = MIN([Year])
        , @STARTYEAR = MIN([Year])
        , @ENDYEAR = MAX([Year])
    FROM @Holidays

    WHILE @CURRENTYEAR <= @ENDYEAR
    BEGIN
        SELECT @CNTR = COUNT([Year])
        FROM @Holidays
        WHERE [Year] = @CURRENTYEAR

        SET @POS = 1

        WHILE @POS <= @CNTR
        BEGIN
            SELECT @MINDAY = MIN(DAY)
            FROM @Holidays
            WHERE
                [Year] = @CURRENTYEAR
                AND [Week] IS NULL

            UPDATE @Holidays
                SET [Week] = @POS
            WHERE
                [Year] = @CURRENTYEAR
                AND [Day] = @MINDAY

            SELECT @POS = @POS + 1
        END

        SELECT @CURRENTYEAR = @CURRENTYEAR + 1
    END
```

```
        UPDATE [dbo].[DimDate]
            SET HolidayUSA  = 'Election Day'
        FROM [dbo].[DimDate] DT
            JOIN @Holidays HL ON (HL.DateID + 1) = DT.DateKey
        WHERE
            [Week] = 1
    END
    --set flag for USA holidays in Dimension
    UPDATE [dbo].[DimDate]
SET IsHolidayUSA = CASE WHEN HolidayUSA  IS NULL THEN 0 WHEN HolidayUSA  IS NOT NULL
THEN 1 END
/*****************************************************************************
**/

SELECT * FROM [dbo].[DimDate]
```

SCRIPT 2

Extension of date dimension with fiscal calendar fields like Fiscal Year, Fiscal Month, and Fiscal Quarter, etc.

Using the Code

Follow the given steps to add new fields related to Fiscal calendar in date dimension and populate them with values.

1. Open SQL Server Management Studio
2. Connect Database Engine
3. Open New query editor
4. Copy paste scripts given below in sequence of one by one
5. To run the given SQL Script, press F5

Step 1

Add new Fields in Date dimension related to Fiscal Calendar

SQL

```
/*Add Fiscal Calendar columns into table DimDate*/

ALTER TABLE [dbo].[DimDate] ADD
    [FiscalDayOfYear] VARCHAR(3),
    [FiscalWeekOfYear] VARCHAR(3),
    [FiscalMonth] VARCHAR(2),
    [FiscalQuarter] CHAR(1),
    [FiscalQuarterName] VARCHAR(9),
    [FiscalYear] CHAR(4),
    [FiscalYearName] CHAR(7),
    [FiscalMonthYear] CHAR(10),
    [FiscalMMYYYY] CHAR(6),
    [FiscalFirstDayOfMonth] DATE,
    [FiscalLastDayOfMonth] DATE,
    [FiscalFirstDayOfQuarter] DATE,
```

```
    [FiscalLastDayOfQuarter] DATE,
    [FiscalFirstDayOfYear] DATE,
    [FiscalLastDayOfYear] DATE
    GO
```

Step 2

Populate Fiscal Calendar fields in Dim date table

SQL Shrink ▲

```
/*************************************************************************
The following section needs to be populated for defining the fiscal calendar
*************************************************************************/

DECLARE
    @dtFiscalYearStart SMALLDATETIME = 'January 01, 1995',
    @FiscalYear INT = 1995,
    @LastYear INT = 2025,
    @FirstLeapYearInPeriod INT = 1996

/*************************************************************************************
**/

DECLARE
    @iTemp INT,
    @LeapWeek INT,
    @CurrentDate DATETIME,
    @FiscalDayOfYear INT,
    @FiscalWeekOfYear INT,
    @FiscalMonth INT,
    @FiscalQuarter INT,
    @FiscalQuarterName VARCHAR(10),
    @FiscalYearName VARCHAR(7),
    @LeapYear INT,
    @FiscalFirstDayOfYear DATE,
    @FiscalFirstDayOfQuarter DATE,
    @FiscalFirstDayOfMonth DATE,
    @FiscalLastDayOfYear DATE,
    @FiscalLastDayOfQuarter DATE,
    @FiscalLastDayOfMonth DATE

/*Holds the years that have 455 in last quarter*/

DECLARE @LeapTable TABLE (leapyear INT)

/*TABLE to contain the fiscal year calendar*/

DECLARE @tb TABLE(
    PeriodDate DATETIME,
    [FiscalDayOfYear] VARCHAR(3),
    [FiscalWeekOfYear] VARCHAR(3),
    [FiscalMonth] VARCHAR(2),
    [FiscalQuarter] VARCHAR(1),
    [FiscalQuarterName] VARCHAR(9),
    [FiscalYear] VARCHAR(4),
    [FiscalYearName] VARCHAR(7),
    [FiscalMonthYear] VARCHAR(10),
    [FiscalMMYYYY] VARCHAR(6),
```

```
    [FiscalFirstDayOfMonth] DATE,
    [FiscalLastDayOfMonth] DATE,
    [FiscalFirstDayOfQuarter] DATE,
    [FiscalLastDayOfQuarter] DATE,
    [FiscalFirstDayOfYear] DATE,
    [FiscalLastDayOfYear] DATE)

/*Populate the table with all leap years*/

SET @LeapYear = @FirstLeapYearInPeriod
WHILE (@LeapYear < @LastYear)
    BEGIN
        INSERT INTO @leapTable VALUES (@LeapYear)
        SET @LeapYear = @LeapYear + 5
    END

/*Initiate parameters before loop*/

SET @CurrentDate = @dtFiscalYearStart
SET @FiscalDayOfYear = 1
SET @FiscalWeekOfYear = 1
SET @FiscalMonth = 1
SET @FiscalQuarter = 1
SET @FiscalWeekOfYear = 1

IF (EXISTS (SELECT * FROM @LeapTable WHERE @FiscalYear = leapyear))
    BEGIN
        SET @LeapWeek = 1
    END
    ELSE
    BEGIN
        SET @LeapWeek = 0
    END

/*************************************************************************************
****/

/* Loop on days in interval*/

WHILE (DATEPART(yy,@CurrentDate) <= @LastYear)
BEGIN

/*SET fiscal Month*/
    SELECT @FiscalMonth = CASE
        /*Use this section for a 4-5-4 calendar.
        Every leap year the result will be a 4-5-5*/
        WHEN @FiscalWeekOfYear BETWEEN 1 AND 4 THEN 1 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 5 AND 9 THEN 2 /*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 10 AND 13 THEN 3 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 14 AND 17 THEN 4 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 18 AND 22 THEN 5 /*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 23 AND 26 THEN 6 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 27 AND 30 THEN 7 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 31 AND 35 THEN 8 /*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 36 AND 39 THEN 9 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 40 AND 43 THEN 10 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 44 AND (48+@LeapWeek) THEN 11
/*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN (49+@LeapWeek) AND (52+@LeapWeek) THEN 12
/*4 weeks (5 weeks on leap year)*/
```

```
/*Use this section for a 4-4-5 calendar.
Every leap year the result will be a 4-5-5*/
        /*
        WHEN @FiscalWeekOfYear BETWEEN 1 AND 4 THEN 1 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 5 AND 8 THEN 2 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 9 AND 13 THEN 3 /*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 14 AND 17 THEN 4 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 18 AND 21 THEN 5 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 22 AND 26 THEN 6 /*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 27 AND 30 THEN 7 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 31 AND 34 THEN 8 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 35 AND 39 THEN 9 /*5 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 40 AND 43 THEN 10 /*4 weeks*/
        WHEN @FiscalWeekOfYear BETWEEN 44 AND _
        (47+@leapWeek) THEN 11 /*4 weeks (5 weeks on leap year)*/
WHEN @FiscalWeekOfYear BETWEEN (48+@leapWeek) AND (52+@leapWeek) THEN 12 /*5 weeks*/
        */
    END

    /*SET Fiscal Quarter*/
    SELECT @FiscalQuarter = CASE
        WHEN @FiscalMonth BETWEEN 1 AND 3 THEN 1
        WHEN @FiscalMonth BETWEEN 4 AND 6 THEN 2
        WHEN @FiscalMonth BETWEEN 7 AND 9 THEN 3
        WHEN @FiscalMonth BETWEEN 10 AND 12 THEN 4
    END

    SELECT @FiscalQuarterName = CASE
        WHEN @FiscalMonth BETWEEN 1 AND 3 THEN 'First'
        WHEN @FiscalMonth BETWEEN 4 AND 6 THEN 'Second'
        WHEN @FiscalMonth BETWEEN 7 AND 9 THEN 'Third'
        WHEN @FiscalMonth BETWEEN 10 AND 12 THEN 'Fourth'
    END

    /*Set Fiscal Year Name*/
    SELECT @FiscalYearName = 'FY ' + CONVERT(VARCHAR, @FiscalYear)

    INSERT INTO @tb (PeriodDate, FiscalDayOfYear, FiscalWeekOfYear, _
    fiscalMonth, FiscalQuarter, FiscalQuarterName, FiscalYear, FiscalYearName) VALUES
    (@CurrentDate, @FiscalDayOfYear, @FiscalWeekOfYear, @FiscalMonth, _
    @FiscalQuarter, @FiscalQuarterName, @FiscalYear, @FiscalYearName)

    /*SET next day*/
    SET @CurrentDate = DATEADD(dd, 1, @CurrentDate)
    SET @FiscalDayOfYear = @FiscalDayOfYear + 1
    SET @FiscalWeekOfYear = ((@FiscalDayOfYear-1) / 7) + 1


    IF (@FiscalWeekOfYear > (52+@LeapWeek))
    BEGIN
        /*Reset a new year*/
        SET @FiscalDayOfYear = 1
        SET @FiscalWeekOfYear = 1
        SET @FiscalYear = @FiscalYear + 1
        IF ( EXISTS (SELECT * FROM @leapTable WHERE @FiscalYear = leapyear))
        BEGIN
            SET @LeapWeek = 1
        END
        ELSE
        BEGIN
            SET @LeapWeek = 0
```

```
        END
    END
END

/*****************************************************************************
*****/

/*Set first and last days of the fiscal months*/
UPDATE @tb
SET
    FiscalFirstDayOfMonth = minmax.StartDate,
    FiscalLastDayOfMonth = minmax.EndDate
FROM
@tb t,
    (
    SELECT FiscalMonth, FiscalQuarter, FiscalYear, _
    MIN(PeriodDate) AS StartDate, MAX(PeriodDate) AS EndDate
    FROM @tb
    GROUP BY FiscalMonth, FiscalQuarter, FiscalYear
    ) minmax
WHERE
    t.FiscalMonth = minmax.FiscalMonth AND
    t.FiscalQuarter = minmax.FiscalQuarter AND
    t.FiscalYear = minmax.FiscalYear

/*Set first and last days of the fiscal quarters*/

UPDATE @tb
SET
    FiscalFirstDayOfQuarter = minmax.StartDate,
    FiscalLastDayOfQuarter = minmax.EndDate
FROM
@tb t,
    (
    SELECT FiscalQuarter, FiscalYear, min(PeriodDate) _
    as StartDate, max(PeriodDate) as EndDate
    FROM @tb
    GROUP BY FiscalQuarter, FiscalYear
    ) minmax
WHERE
    t.FiscalQuarter = minmax.FiscalQuarter AND
    t.FiscalYear = minmax.FiscalYear

/*Set first and last days of the fiscal years*/

UPDATE @tb
SET
    FiscalFirstDayOfYear = minmax.StartDate,
    FiscalLastDayOfYear = minmax.EndDate
FROM
@tb t,
    (
    SELECT FiscalYear, min(PeriodDate) as StartDate, max(PeriodDate) as EndDate
    FROM @tb
    GROUP BY FiscalYear
    ) minmax
WHERE
    t.FiscalYear = minmax.FiscalYear

/*Set FiscalYearMonth*/
UPDATE @tb
```

```
SET
    FiscalMonthYear =
        CASE FiscalMonth
        WHEN 1 THEN 'Jan'
        WHEN 2 THEN 'Feb'
        WHEN 3 THEN 'Mar'
        WHEN 4 THEN 'Apr'
        WHEN 5 THEN 'May'
        WHEN 6 THEN 'Jun'
        WHEN 7 THEN 'Jul'
        WHEN 8 THEN 'Aug'
        WHEN 9 THEN 'Sep'
        WHEN 10 THEN 'Oct'
        WHEN 11 THEN 'Nov'
        WHEN 12 THEN 'Dec'
        END + '-' + CONVERT(VARCHAR, FiscalYear)

/*Set FiscalMMYYYY*/
UPDATE @tb
SET
    FiscalMMYYYY = RIGHT('0' + CONVERT(VARCHAR, FiscalMonth),2) + CONVERT(VARCHAR,
FiscalYear)

/**************************************************************************************
*****/

UPDATE [dbo].[DimDate]
    SET
    FiscalDayOfYear = a.FiscalDayOfYear
    , FiscalWeekOfYear = a.FiscalWeekOfYear
    , FiscalMonth = a.FiscalMonth
    , FiscalQuarter = a.FiscalQuarter
    , FiscalQuarterName = a.FiscalQuarterName
    , FiscalYear = a.FiscalYear
    , FiscalYearName = a.FiscalYearName
    , FiscalMonthYear = a.FiscalMonthYear
    , FiscalMMYYYY = a.FiscalMMYYYY
    , FiscalFirstDayOfMonth = a.FiscalFirstDayOfMonth
    , FiscalLastDayOfMonth = a.FiscalLastDayOfMonth
    , FiscalFirstDayOfQuarter = a.FiscalFirstDayOfQuarter
    , FiscalLastDayOfQuarter = a.FiscalLastDayOfQuarter
    , FiscalFirstDayOfYear = a.FiscalFirstDayOfYear
    , FiscalLastDayOfYear = a.FiscalLastDayOfYear
FROM @tb a
    INNER JOIN [dbo].[DimDate] b ON a.PeriodDate = b.[Date]

/**************************************************************************************
*****/

SELECT * FROM [dbo].[DimDate]
```

Enjoy T-SQLization.

License

This article, along with any associated source code and files, is licensed under The Code Project Open License (CPOL)

Written By
Mubin M. Shaikh
Architect Cybage Software Pvt. Ltd.
India

Microsoft® Certified Professional (Microsoft Certification ID: 8918672).

Microsoft Certified Technology Specialist with more than 16+ years of expertise to architect and implement effective solutions for Data Analytics, Reporting and Data Visualization solutioning need on Azure Cloud or On-Premise

Technology :
Azure (Data Lake, Data Factory, Synapse Analytics, Databricks, SQL),
Microsoft BI (SSIS, SSAS, SSRS, SQL-Server), C#.Net, Pentaho,
Data Warehousing, Dimension modelling, Snowflake DW, SQL DW, MySQL
Data Visualization using (Tableau, Power BI, QlikView, Pentaho),
Domain : Sales, Retail, CRM, Public Transport, Media & Entertainment, Insurance
Data Integration and Analytics Experience with MS. Dynamic CRM, Salesforce CRM, Dataverse, SAP- FI, Dynamics AX etc.

Linked In Profile:
Click Here to View Linked In Profile

Change will not come if we keep waiting for some other person !!, or keep waiting for some other time !!, We are the one we are waiting for, We are the change that we are looking for.

Comments and Discussions

You must Sign In to use this message board.

Search Comments

First Prev Next

My vote of 5
Member 14973764 24-Oct-20 4:38

Memorial Day Missing?
Member 14902153 **29-Jul-20 5:21**

Error On US Holidays
Member 14902153 29-Jul-20 5:41

Errors
Member 14864861 **19-Jun-20 3:58**

Fiscal year problem
Member 14191456 **21-Mar-19 9:55**

Outstanding script although there are issues with fiscal portion
Steven Lundahl **27-Feb-18 7:59**

Fiscal year error
Euniq **16-Jan-18 4:15**

fical year start day = 1/may
Member 13620177 **12-Jan-18 4:30**

Mistake in the 4-5-4 calendar generation
Jrn86 **25-Aug-17 0:09**

I am getting error on fascul script (last script)
EngrKI **3-Jul-17 22:20**

Good Code article
Member 13274327 **22-Jun-17 10:13**

US Holiday List 2010-2020
Member 13106423 **4-Apr-17 15:13**

Fiscal Year
Fraser Stewart (TowelsRus) **4-Apr-17 1:12**

South Africa date dimension
Keobakile Mothobi **21-Mar-17 21:31**

DateDimension Script
Member 12833075 **6-Nov-16 9:05**

Re: DateDimension Script
SimotheBlue 8-Mar-17 11:26

Re: DateDimension Script
chirag_parikh01 30-Nov-17 22:41

My vote of 5
Member 12661873 **30-Jul-16 14:33**

Data-warehousing

mariasimmons **10-Jul-16 3:33**

Fiscal Section Totally wrong

Member 12476682 **21-Apr-16 10:15**

My vote of 5

0x29A **29-Mar-16 9:33**

Just a little note

jacktric **21-Dec-15 22:13**

Nice One

Saintgr8 **24-Sep-15 10:10**

My vote of 5

Member 11957129 **3-Sep-15 6:37**

My vote of 5

Member 11907560 **13-Aug-15 8:23**

Permalink
Advertise
Privacy
Cookies
Terms of Use

Layout: fixed | fluid

Article Copyright 2013 by Mubin M. Shaikh
Everything else Copyright © CodeProject, 1999-2023

Web03 2.8:2023-05-13:1