Tech Tips: Calculation Manager Custom-Defined Date Functions

Celvin Kattookaran, Huron Consulting Group

All the functions described here are available from Calc Manager 11.1.2.3.502 onwards.

All the examples given below are done on a modified Sample Basic application and not from a real-world application. However, the logic can be ported to any real-world application, and I believe these functions will be of great help in Workforce and Capex modules where you're working with Start and End date.

All these date functions expect the date to be in Planning format, which is YYYYMMDD. So get your dates straight ☺

Essbase (epoch time) DATE use @DATEPART/@DATEROLL et al.
Planning DATE use the new functions

Let's start in the order they appear in Custom-Defined Function Manager:

@CalcMgrAddDate

Syntax

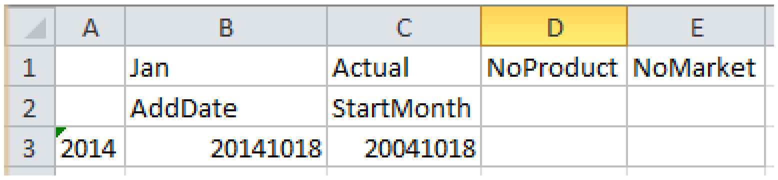
@CalcMgrAddDate(date,yearsToAdd,monthsToAdd,daysToAdd)

Parameter	**Description**
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
yearsToAdd	Signed Integer, specify number of years to add. Using a negative value will go back to the number of years specified.
monthsToAdd	Signed Integer, specify number of months to add. Using a negative value will go back to the number of months specified.
daysToAdd	Signed Integer, specify number of days to add. Using a negative value will go back to the number of days specified.

Example

1. In this example, 10 years are added to the StartMonth

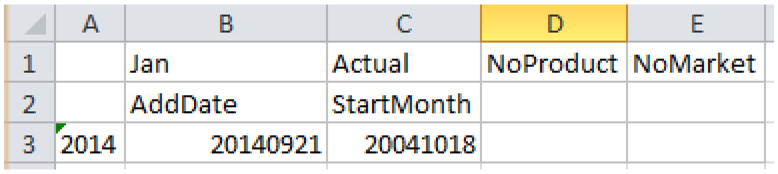
```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "AddDate"(
     @CalcMgrAddDate ("StartMonth",10,0,0);
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		AddDate	StartMonth		
3	2014	20141018	20041018		

2. In this example 10 years are added, go back a month and add 3 days

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "AddDate"(
     @CalcMgrAddDate ("StartMonth",10,-1,3);
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		AddDate	StartMonth		
3	2014	20140921	20041018		

@CalcMgrAddDays

Syntax

@CalcMgrAddDays(date,daysToAdd)

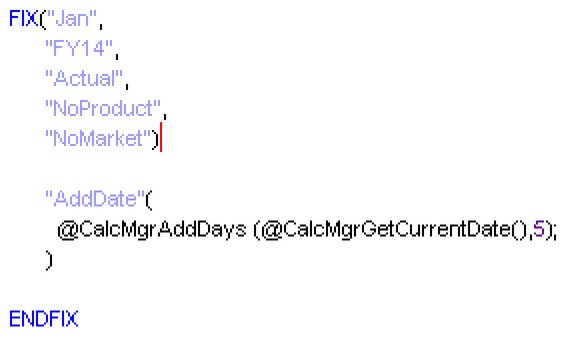
Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
daysToAdd	Signed Integer, specify number of days to add. Using a negative value will go back to the number of days specified.

Notes

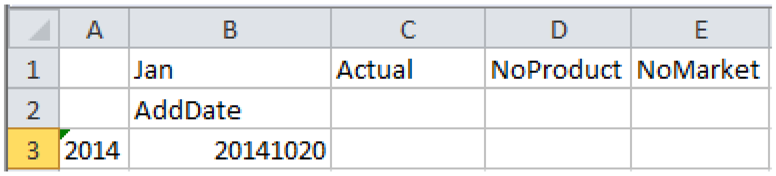
This function works similar to @CalcMgrAddDate function; the only difference is you are just mentioning days.

Example

In this example, five days are added to today’s date.



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "AddDate"(
     @CalcMgrAddDays (@CalcMgrGetCurrentDate(),5);
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		AddDate			
3	2014	20141020			

@CalcMgrAddMonths

Syntax

@CalcMgrAddMonths(date,monthsToAdd)

Parameter	**Description**
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
monthsToAdd	Signed Integer, specify number of months to add. Using a negative value will go back to the number of months specified.

Notes

This function works similar to @CalcMgrAddDate function, the only difference is you are just mentioning months.

@CalcMgrAddWeeks

Syntax

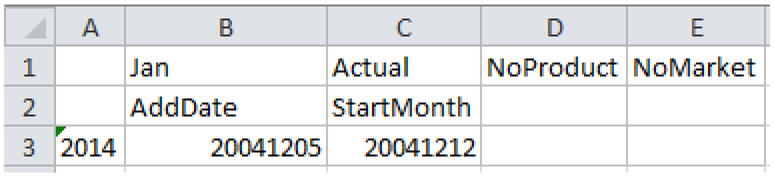
@CalcMgrAddWeeks(date,weeksToAdd)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
weeksToAdd	Signed Integer, specify number of weeks to add. Using a negative value will go back to the number of weeks specified.

Notes
This function works similar to @CalcMgrAddDate function; only difference is you are just mentioning weeks.

Example
In this example, we are going back a week.

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "AddDate"(
     @CalcMgrAddWeeks("StartMonth",-1);
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		AddDate	StartMonth		
3	2014	20041205	20041212		

@CalcMgrAddYears

Syntax
@CalcMgrAddYears(date,yearsToAdd)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
yearsToAdd	Signed Integer, specify number of years to add. Using a negative value will go back to the number of years specified.

Notes

This function works similar to @CalcMgrAddDate function; the only difference is you are just mentioning years.

@CalcMgrDateToString

Syntax

@CalcMgrDateToString(date,format)

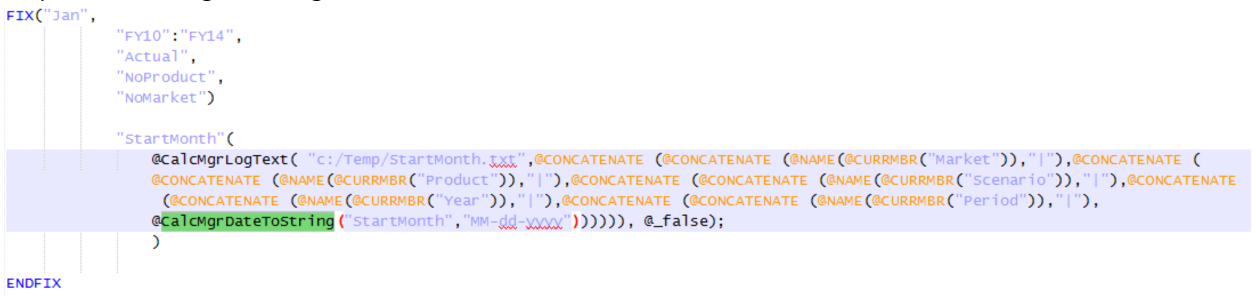
Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
format	SimpleDateFormat used in java, have a look at http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html for a full list of format that can be used.

Notes

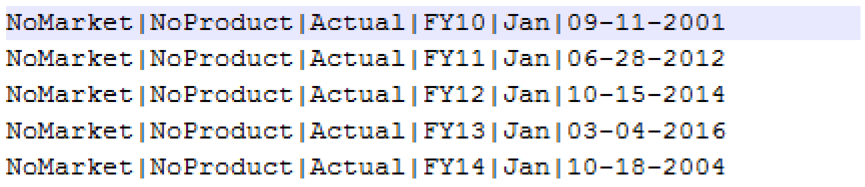
Considering the fact that Essbase deals with numbers, most of the SimplDateFormat cannot be used. (Remember the number we have in hand is yyyymmdd.)

Example

In this example, we are exporting data from a Planning application to a Reporting application, which uses Essbase date measures. DataExport gives you numbers as yyyymmdd where CalcMgrDateToString can help in formatting Planning date to Essbase date.



```
FIX("Jan",
            "FY10":"FY14",
            "Actual",
            "NoProduct",
            "NoMarket")

            "StartMonth"(
                @CalcMgrLogText( "c:/Temp/StartMonth.txt",@CONCATENATE (@CONCATENATE (@NAME(@CURRMBR("Market")),"|"),@CONCATENATE (
                @CONCATENATE (@NAME(@CURRMBR("Product")),"|"),@CONCATENATE (@CONCATENATE (@NAME(@CURRMBR("Scenario")),"|"),@CONCATENATE
                 (@CONCATENATE (@NAME(@CURRMBR("Year")),"|"),@CONCATENATE (@CONCATENATE (@NAME(@CURRMBR("Period")),"|"),
                @CalcMgrDateToString("StartMonth","MM-dd-yyyy")))))), @_false);
                )

ENDFIX
```



```
NoMarket|NoProduct|Actual|FY10|Jan|09-11-2001
NoMarket|NoProduct|Actual|FY11|Jan|06-28-2012
NoMarket|NoProduct|Actual|FY12|Jan|10-15-2014
NoMarket|NoProduct|Actual|FY13|Jan|03-04-2016
NoMarket|NoProduct|Actual|FY14|Jan|10-18-2004
```

Notes @CalcMgrLogText(filename,msg,printdate)

@CalcMgrLogText always appends the file. You'll have to remove the file before running the calc.

@CalcMgrDaysBetween

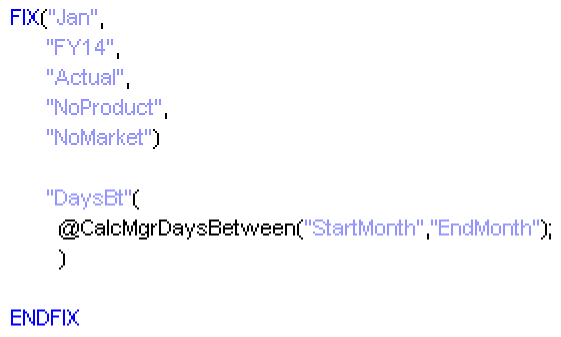
Syntax

@CalcMgrDaysBetween(fromDate,toDate)

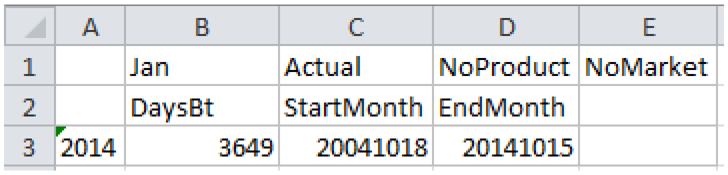
Parameter	Description
fromdate	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
todate	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Example

In this example, we are checking the days elapsed between StartMonth and EndMonth



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "DaysBt"(
     @CalcMgrDaysBetween("StartMonth","EndMonth");
     )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		DaysBt	StartMonth	EndMonth	
3	2014	3649	20041018	20141015	

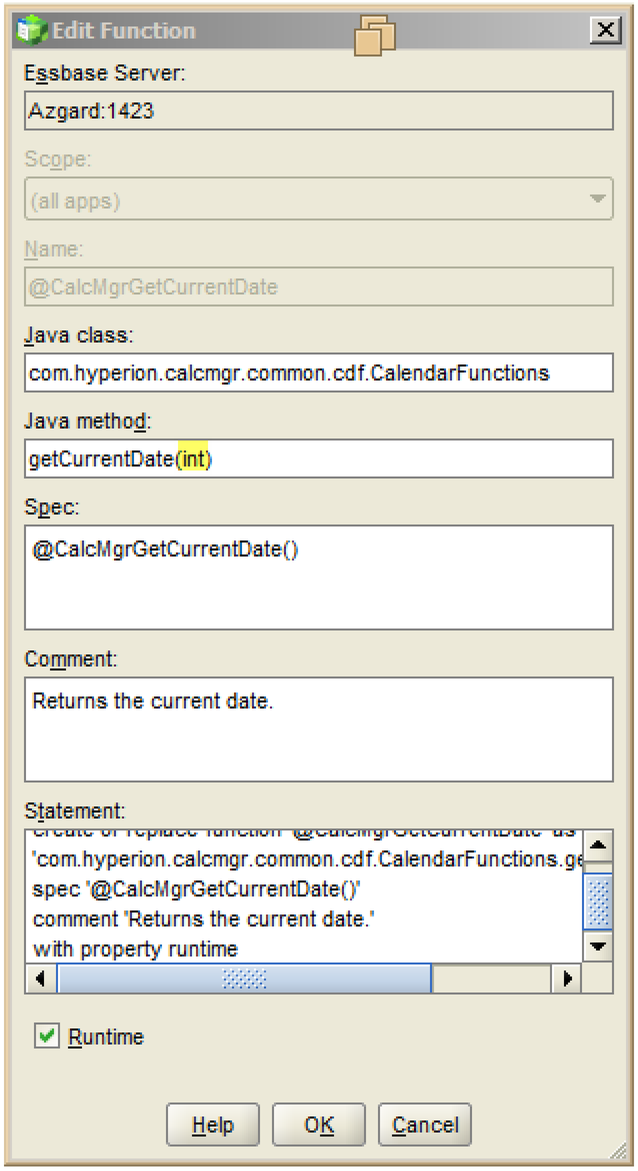
@CalcMgrGetCurrentDate

Syntax

@CalcMgrGetCurrentDate()

Notes

To use this function, you'll need to register this correctly; the Java method used in this function is incorrect. The method asks to provide an integer value for this to work, which I believe is a bit awkward.



Azgard:1423

(all apps)

@CalcMgrGetCurrentDate

com.hyperion.calcmgr.common.cdf.CalendarFunctions

getCurrentDate(int)

@CalcMgrGetCurrentDate()

Returns the current date.

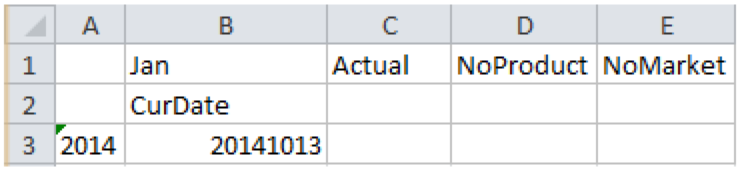
```
create or replace function @CalcMgrGetCurrentDate as
'com.hyperion.calcmgr.common.cdf.CalendarFunctions.ge
spec '@CalcMgrGetCurrentDate()'
comment 'Returns the current date.'
with property runtime
```

Remove “**int**” from the function and save it.

Example

In this example,e we are populating CurDate with the time of execution time stamp (yyyymmdd)

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "CurDate"(
      @CalcMgrGetCurrentDate ();
      )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		CurDate			
3	2014	20141013			

@CalcMgrGetCustomDate

Syntax

@CalcMgrGetCustomDate(year,month,day)

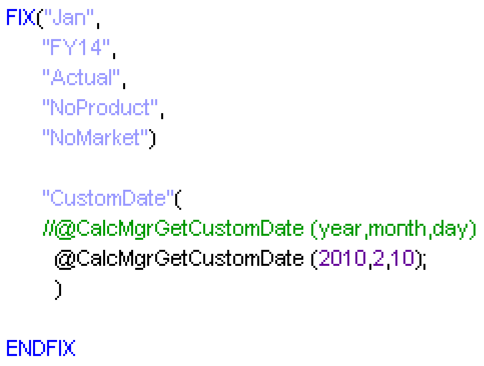
Parameter	Description
year	Integer, valid year in yyyy format.
month	Integer, valid month in mm format.
day	Integer, valid month in dd format.

Notes

If you enter wrong values in month and days, the function will perform an auto correction.

Example

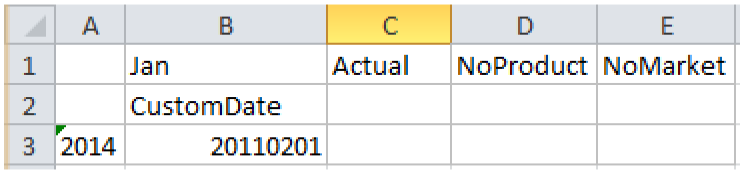
1.



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "CustomDate"(
    //@CalcMgrGetCustomDate (year,month,day)
     @CalcMgrGetCustomDate (2010,2,10);
     )

ENDFIX
```

2. If the values in months and days are wrong, the function will do an auto correction.

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "CustomDate"(
    //@CalcMgrGetCustomDate (year,month,day)
     @CalcMgrGetCustomDate (2010,13,32);
     )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		CustomDate			
3	2014	20110201			

You entered 13 for month, so it'll do the math and make it 01 (Jan) and add 1 to the year 2011. Now we all know that there are 31 days in Jan and you entered 32 and the end result is 2011 Feb 1st = 20110201.

@CalcGetDay

Syntax

@CalcGetDay(date)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

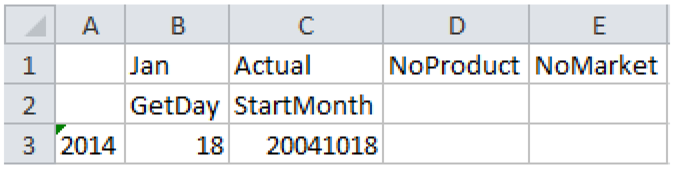
Notes

This function can be used together with @CalcMgrGetCurrentDate to populate last refresh times.

Example

In this example, we are getting the day when the Product was introduced to the Market.

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "GetDay"(
     @CalcMgrGetDay ("StartMonth");
     )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		GetDay	StartMonth		
3	2014	18	20041018		

@CalcMgrGetMaxDaysInMonth

Syntax

@CalcMgrGetMaxDaysInMonth(date)

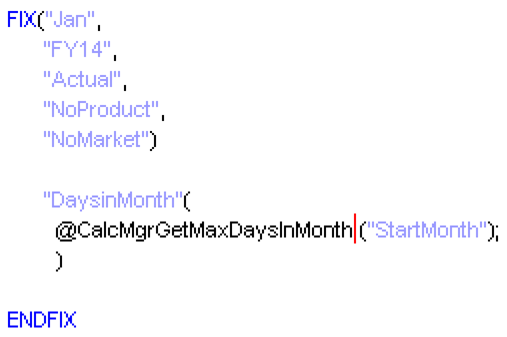
Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Notes

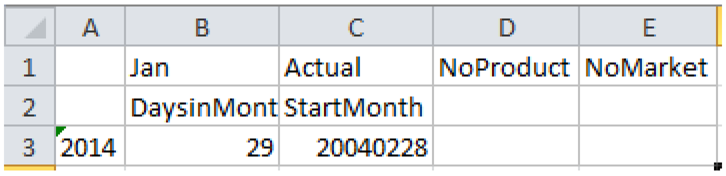
Gives you the maximum number of days in a month, and yes it checks whether the year is a leap year.

Example

In this example, we got 29 days as 2004 is a leap year.



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "DaysinMonth"(
     @CalcMgrGetMaxDaysInMonth("StartMonth");
     )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		DaysinMont	StartMonth		
3	2014	29	20040228		

@CalcMgrGetMonth

Syntax

@CalcMgrGetMonth(date)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning

Parameter	Description
	date yyyymmdd.

Notes

This function works similar to @CalcMgrGetDay function; only difference is that the function returns the month. This function behaves like @DATEPART(date, DP_MONTH)

@CalcMgrGetWeekOfMonth

Syntax

@CalcMgrGetWeekOfMonth (date)

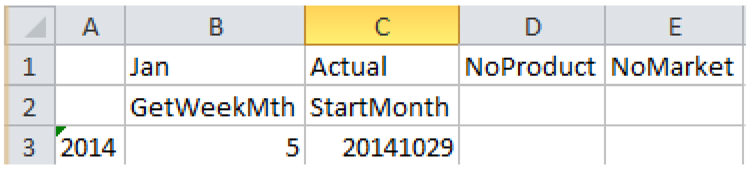
Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Notes

This function returns number of the week in month (1 ~ 5)

Example

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "GetWeekMth"(
      @CalcMgrGetWeekOfMonth("StartMonth");
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		GetWeekMth	StartMonth		
3	2014	5	20141029		

@CalcMgrGetWeekOfYear

Syntax

@CalcMgrGetWeekOfYear (date)

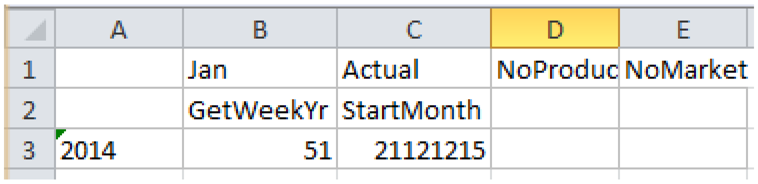
Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Notes

This function works similar to @CalcMgrGetDay function; the only difference is that the function returns the week of the year. This function behaves like @DATEPART(date, DP_WEEK) - returns the week of the year for the input date (1 to 54)

Example

```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "GetWeekYr"(
      @CalcMgrGetWeekOfYear("StartMonth");
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduc	NoMarket
2		GetWeekYr	StartMonth		
3	2014	51	21121215		

@CalcGetYear

Syntax

@CalcGetYear (date)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Notes

This function works similar to @CalcMgrGetDay function; the only difference is that the function returns the year. This function behaves like @DATEPART(date, DP_YEAR)

@CalcMgrIsLeapYear

Syntax

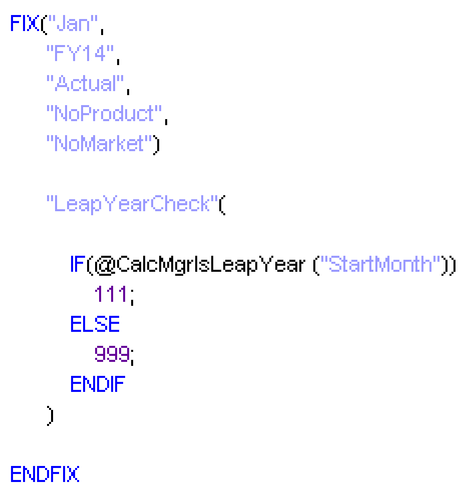
@ CalcMgrIsLeapYear ()

Notes

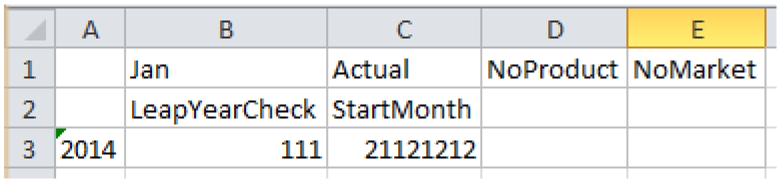
This Boolean function returns true of false.

Example

In this example, we are checking whether the year of the StartMonth is a leap year or not.



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "LeapYearCheck"(

       IF(@CalcMgrIsLeapYear ("StartMonth"))
          111;
       ELSE
          999;
       ENDIF
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		LeapYearCheck	StartMonth		
3	2014	111	21121212		

@CalcMgrRollDay

Syntax

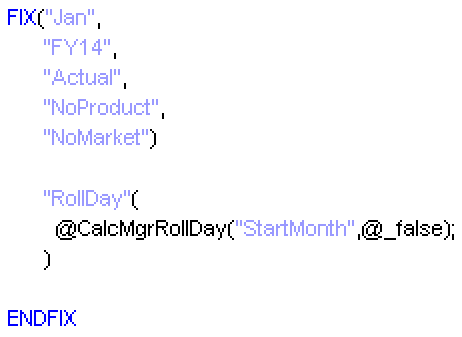
@CalcMgrRollDay(date,up)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
up	Boolean, Use @_True to rollup or @_False to roll down

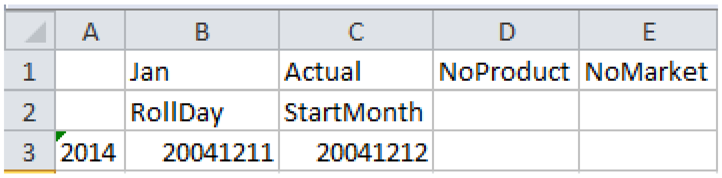
Notes

I'm not sure whether this function is used to add days; in my testing, I found this to add or go back one day.

Example



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "RollDay"(
     @CalcMgrRollDay("StartMonth",@_false);
    )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		RollDay	StartMonth		
3	2014	20041211	20041212		

@CalcMgrRollMonth

Syntax

@CalcMgrRollMonth (date,up)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
up	Boolean, Use @_True to rollup or @_False to roll down

Notes

This function works similar to @CalcMgrRollDay.

@CalcMgrRollYear

Syntax

@CalcMgrRollYear (date,up)

Parameter	Description
date	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Parameter	Description
up	Boolean, Use @_True to rollup or @_False to roll down

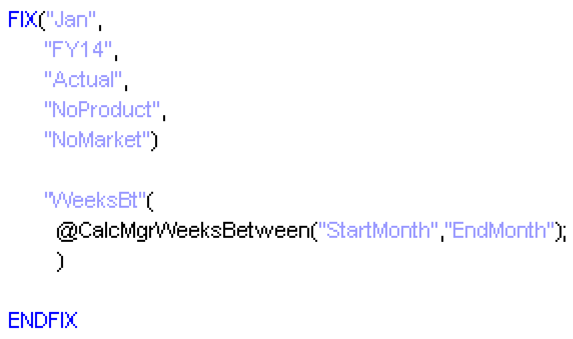
Notes
This function works similar to @CalcMgrRollDay.

@CalcMgrWeeksBetween

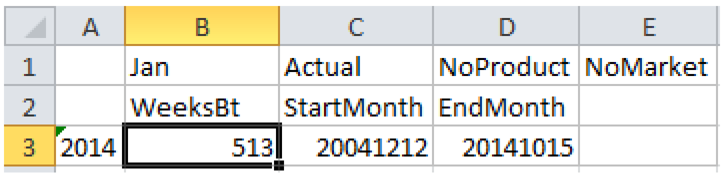
Syntax
@CalcMgrWeeksBetween (fromDate,toDate)

Parameter	Description
fromdate	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.
todate	Any valid single member name or member combination, or a function that returns a Planning date yyyymmdd.

Example
In this example we are checking the weeks elapsed between StartMonth and EndMonth



```
FIX("Jan",
    "FY14",
    "Actual",
    "NoProduct",
    "NoMarket")

    "WeeksBt"(
     @CalcMgrWeeksBetween("StartMonth","EndMonth");
     )

ENDFIX
```



	A	B	C	D	E
1		Jan	Actual	NoProduct	NoMarket
2		WeeksBt	StartMonth	EndMonth	
3	2014	513	20041212	20141015	

Comments

These are some really great time functions that can help Planning developers; you don't need to do the math to get days, year, et al.

Two functions that are missing from the DATEPART are:

- DP_WEEKDAY returns the week day of the input date. (1 - Sunday, 2 - Monday, ... 6 - Saturday).
- DP_DAYOFYEAR returns the day of the year numbering (1 to 366).

Improvements

I believe some of the functions are redundant such as:

- Add functions
- Get functions

➢ I would like to see them replaced with a single function where you can mention what to return (similar to DATEPART, DATEDIFF functions).

➢ I would like to see GetCurrentDate function expanded to provide hours, minutes, and seconds information so it can be used to provide the database refresh time.

I expect to keep my Sample Calendar application. If someone asks, "Tell me the date after 5 weeks," I don't need to flip the calendar and start counting. Just run my calc and show off ☺.