KPIs in Analysis Services

by *Gogula G. Aryalingam*

http://sqlserveruniverse.com/content/SSAS0700105122008KPIsInAnalysisServices.aspx

KPIs or Key Performance Indicators are one of the most important entities in driving business decisions. It can be defined as a (quantifiable) measurement used to define and measure an organization's progress in achieving business goals. KPIs for an organization are decided upon by giving careful thought to how each goal and the progress towards these goals should be measured. Business processes throughout the organization would be taken into consideration, and KPIs decided for them. Sometimes KPIs would be created out of other KPIs, where the child KPIs give a weightage of themselves in order to calculate the parent KPI's value. Though decided upon at the very beginning of a business intelligence project, KPIs are almost always created towards the end of it.

SQL Server 2005 Analysis Services, allows for the creation of KPIs on its cubes. KPIs were not available with SQL Server 2000. This introduction of KPIs to SSAS gives solutionists more power in delivering their solutions, since they need not go for separate software like *Business Scorecard Manager* or *PerformancePoint Server* just to build KPIs. Of course these software can be used to build *upon* the SSAS solution for a much more effective solution.

This article takes a look at how to build and implement KPIs using SQL Server 2005 Analysis Services. A little knowledge of MDX is also required to effectively follow this article. Included with this article is a sample cube based on some fabricated *AdventureWorks* data, and can be downloaded from here. Follow included instructions to set up the cube.

KPI Terms used in SSAS

Value

The *value* is an MDX expression used to return the actual value of the KPI. It can be the result of a very simple expression like the value of a measure or something more complicated, such as a complex algorithm/calculation performed on a couple of measures.

E.g.: Suppose a organization deals with sales, and they need to measure their profit; they may define *ProfitPercentage* as a KPI. The *value* property of *ProfitPercentage* can be defined as *(Sales - Cost) / Cost* assuming *Sales* and *Cost* are measures in the cube.

Goal

The *goal* is an MDX expression used to specify the target value of the KPI. It specifies the value the business expects the KPI to have, in order to decide where they stand.

Status

The *status* is an MDX expression that indicates the state of the KPI at a given point in time. Usually the value for the status would be derived by using the *value* and *goal* properties of the KPI. The ideal values for the status would be a max of 1 (good) to a minimum of -1 (bad), while 0 indicates neutral status. The expression could return an infinite number of values within this range to indicate varying degrees of the status.

E.g.: Assuming that the *goal* is set to a constant value such as 0.25 for the *ProfitPercentage* KPI; we could agree upon status values for the ratio of Value:Goal, such that a result of 0.9 or higher to be 1 (good), a result of 0.8 or lower to be bad (-1) and a result between 0.9 and 0.8 to be neutral (0).

Status Indicator

The *status indicator* is a visual element which is used to present the status of the KPI. The display of the indicator is determined by the value returned by the *status* property of the KPI. Usually indicators are displayed as gauges, traffic lights or smileys.

Trend

The *trend* is an MDX expression that evaluates the value of a KPI across time. It can be expressed using any time

based criteria. Using this, the business user will be able to determine how the KPI's value has progressed over time.

E.g.: Continuing with the previous examples, we could now write an expression to compare the current *ProfitPercentage* KPI value with that of the value of the same period last year. We could agree upon a comparison technique which would rate a difference 0.25 or more as successful (1), less than -0.1 as 'falling behind' (-1) and values in between as 'no change in trend' (0).

Trend Indicator

The *trend indicator* is a visual element which is used to present the trend of the KPI. The display of this indicator is determined by the value returned by the *trend* property of the KPI.

Parent KPI

This property specifies which KPI acts as the parent for the current KPI. The parent KPI uses parts of its child KPIs' values to determine or calculate its own value. Also, the proper display of the child KPIs under the parent KPI can be facilitated in client applications using the *Parent KPI*.

Current Time Member

The *current time member* specifies the default time member for the KPI, for browsers to show. This too would be in the form of an MDX expression.

Weight

The *weight* is an MDX expression, which assigns a relative importance to the KPI. When the KPI is assigned to a Parent KPI, the weight comes into play by deciding how much (or what weightage) of its value should be passed to the parent.

Display Folder

The *display folder* specifies the folder in which the KPI will be stored when browsing.

Sample Scenario

Our sample will deal with two KPIs: *ProfitPercentage* and *SalesPerformance*. We shall also create a parent KPI called *SalesKPI* which would hold the these two.

Creating KPIs using BIDS

I guess by now you would have downloaded and setup the sample cube on your Analysis Services copy (setup instructions are available at the download location). Also, please note that you require the *AdventureWorks* database on your SQL Server 2005 database engine in order to process the cube.

1. Open the cube on BIDS:
    - Start Business Intelligence Development Studio
    - Go to: File -> Open -> Analysis Services Database...
    - In the *Connect to Database* dialog which opens, enter your Analysis Server name and select the *KPISample* database. Click on *OK*.
    - From the *Solution Explorer* window, open *Sales* cube.

2. Click on the *KPIs* tab

3. We shall first add a KPI named *ProfitPercentage* to the cube:
    - Click on the *New KPI* button or right-click on the *KPI Organizer* window and select *New KPI*.
    - In the KPI Designer which opens up, type in the name of the KPI as '*ProfitPercentage*'.

4. Add the measure to the *Sales Fact* measure group by selecting from the *Associated Measure Group* drop down.

5. Now let's add an expression to evaluate the *value* property of the KPI:
    - Type the following expression in the *Value Expression* box. You could also drag the measures from the *Metadata* tab in the *Calculation tools* window:

```
([Measures].[Sales] - [Measures].[Cost]) / [Measures].[Cost]
```

We have built the expression above using the logic that *ProfitPercentage* is calculated as the difference in Sales and Cost, divided by the Cost.

6. Now, let us set the *goal* of the KPI to a constant. Enter *0.25* in the *Goal Expression* box, which means the goal expects 25% more in sales than the cost.

7. Next we are on to the all important *status*. Here we shall write an expression to compare the *value* with the *goal*:
   - Before examining the actual expression let us take a look at a "pseudo-expression":

     ```
     CASE
         WHEN v/g >= 0.9 THEN 1
         WHEN v/g < 0.8 THEN -1
         ELSE 0
     END
     ```

     Here, *v* is the value, while *g* is the goal. A ratio of .9 for value:goal (*v/g*) would be considered good in this instance. This means that if the *value* makes up at least 90% of the goal then it would be considered as successful, hence the value 1 to indicate it. If *v/g* equals to less than 0.8 then the status will be considered as a failure (-1). And all those values do not fall into the above ranges will be considered neutral (0).

   - Now, enter the following MDX expression into the *Status expression* box:

     ```
     CASE
         WHEN KPIValue("ProfitPercentage") / KPIGoal("ProfitPercentage")
     >= .9 THEN 1
         WHEN KPIValue("ProfitPercentage") / KPIGoal("ProfitPercentage")
     < .8 THEN -1
         ELSE 0
     END
     ```

     Being the realization of the previous "pseudo-expression", the above expression replaces *v* with *KPIValue("ProfitPercentage")* and *g* with *KPIGoal("ProfitPercentage")*. The *KPIValue* function returns the value of the KPI's *value* property, while the *KPIGoal* function returns the value of the KPI's *goal* property. Both function take the name of the KPI through a character string parameter.

8. Select an appropriate image type from the *Status indicator* drop down to provide for the visuals of the *KPI status*.

9. The next property to be configured is the *trend*.
   - Once again, let us first look at a "pseudo-expression":

     ```
     CASE
         WHEN IsEmpty(x) THEN 0
         WHEN v - vx >= 0.25 THEN 1
         WHEN v - vx < -0.1 THEN -1
         ELSE 0
     END
     ```

     Here, *x* is the current time period of the previous year. For instance if the current selected time period is 20-May-2008, *x* will be 20-May-2007. *v* is the value of the KPI, which by default would return the value for the current time period. v_x is the value of the KPI for the same time period of the previous year.

     The expression first checks if *x* is empty, thereby resulting in 0 (or no change in trend). Then, it checks if the current period's value is an increase of at least 25%, thereby indicating 1 or a positive trend. If there's a decrease of more than 10% from last year it indicates a negative trend or -1. All other results will indicate no change in the trend (or 0).

   - Enter the following in the *Trend expression* textbox:

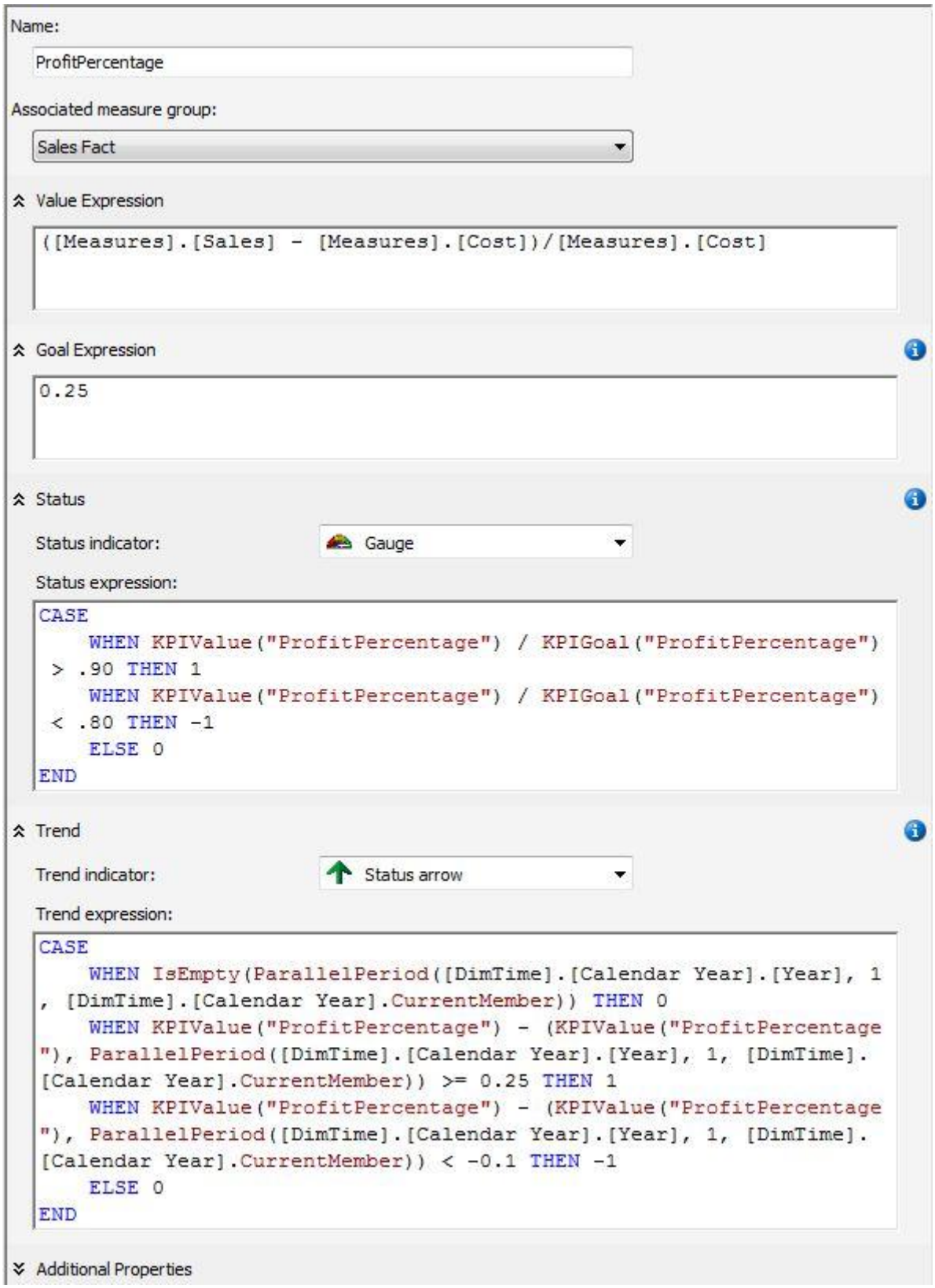
     ```
     CASE
     ```

```
    WHEN IsEmpty(ParallelPeriod([DimTime].[Calendar Year].[Year],
1, [DimTime].[Calendar Year].CurrentMember))
        THEN 0
    WHEN KPIValue("ProfitPercentage") -
(KPIValue("ProfitPercentage"), ParallelPeriod([DimTime].[Calendar
Year].[Year], 1, [DimTime].[Calendar Year].CurrentMember)) >= 0.25
        THEN 1
    WHEN KPIValue("ProfitPercentage") -
(KPIValue("ProfitPercentage"), ParallelPeriod([DimTime].[Calendar
Year].[Year], 1, [DimTime].[Calendar Year].CurrentMember)) < -0.1
        THEN -1
    ELSE 0
END
```

In the MDX expression above, *x* is replaced by *ParallelPeriod([DimTime].[Calendar Year].[Year], 1, [DimTime].[Calendar Year].CurrentMember)*. *ParallelPeriod* is a function which returns a member from a prior period in the same relative position as a specified member. The variable *v* is replaced by *KPIValue("ProfitPercentage")* and v_x is replaced by *(KPIValue("ProfitPercentage"), ParallelPeriod([DimTime].[Calendar Year].[Year], 1, [DimTime].[Calendar Year].CurrentMember))*.

10. Select an appropriate image type from the *Trend indicator* drop down to provide for the visual of the *KPI trend*.

    Here's a sample of what the KPI designer should look like:



11. Let us now move on the other KPI, *SalesPerformance*. Add this KPI using the steps described previously.
    - The expressions for the KPI properties are as follows. Add these to the appropriate expression textboxes:

value:	`([Measures].[Sales Quantity] - [Measures].[Target Quantity])/[Measures].[Target Quantity]`

goal:	`0.15`

status:	`CASE`
	`  WHEN KPIValue("SalesPerformance")/KPIGoal("SalesPerformance") >= 1`
	`1`
	`  WHEN KPIValue("SalesPerformance")/KPIGoal("SalesPerformance") < -1`
	`-1`
	`  ELSE 0`
	`END`

trend:	`CASE`
	`  WHEN IsEmpty(ParallelPeriod([DimTime].[Calendar Year].[Year], 1, [DimTime].[Calendar Year].CurrentMember))`
	`  THEN 0  WHEN KPIValue("SalesPercentage") - (KPIValue("SalesPercentage"), ParallelPeriod([DimTime].[Calendar Year].[Year], 1, [DimTime].[Calendar Year].CurrentMember)) >= 0.1`
	`  THEN 1`
	`  WHEN KPIValue("SalesPercentage") - (KPIValue("SalesPercentage"), ParallelPeriod([DimTime].[Calendar Year].[Year], 1, [DimTime].[Calen Year].CurrentMember)) < -0.1`
	`  THEN -1`
	`  ELSE 0`
	`END`

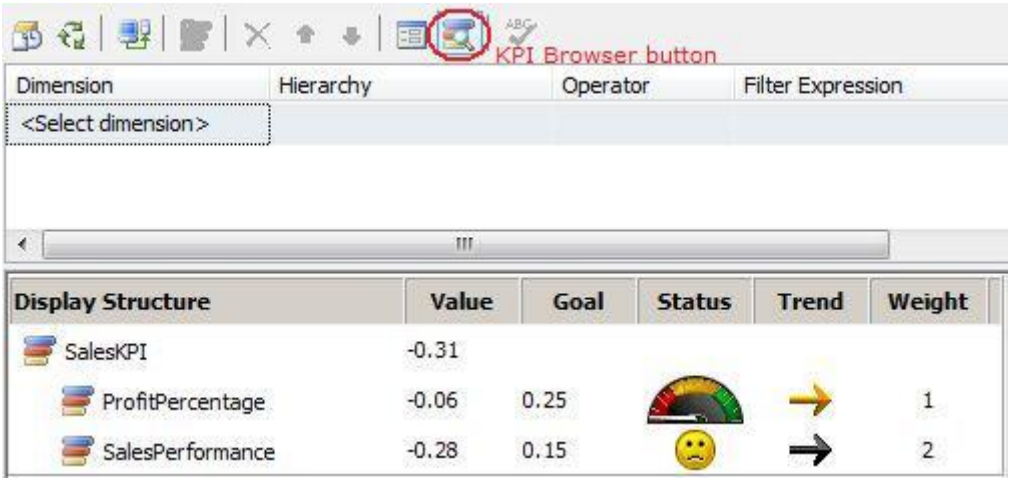
12. Select appropriate image types from the *Status indicator* and *Trend indicator* drop down boxes.

13. Let us now create the Parent KPI, *SalesKPI*. But beforehand we shall add weightages to the *ProfitPercentage* and *SalesPerformance* KPIs:
    - Add *1* and *2* to the *Weight* textboxes of the *ProfitPercentage* and *SalesPerformance* KPIs respectively.
    - Add a new KPI, name it *SalesKPI* and add the following expression to the *Value Expression* textbox:

      ```
      ((KPIValue("ProfitPercentage") * KPIWeight("ProfitPercentage")) +
      (KPIValue("SalesPerformance") * KPIWeight("SalesPerformance")))/2
      ```

      This expression specifies that the *SalesKPI* will amount to the average of the *ProfitPercentage* and *SalesPerformance* KPIs. Also, each KPI would be multiplied with their respective weightages before the average is calculated.

The KPIs are done! Next, process the cube. You will be able to view the KPIs using the built-in KPI Browser under the KPIs tab in BIDS. You could also use Office 2007 Excel.



Conclusion

KPIs are powerful means in finding out about the status of one's organization. The ability to provide for these means is now available with Analysis Services. Apart from what was given to you in this write-up, there is so much more in

how we can build KPIs.