

# DeltaMaster clicks!

## 11/2008

Greetings, fellow data analysts!

There are certain times each year when not much really happens in the world – and, accordingly, there is very little to report. Unfortunately, however, most newspapers are published every day and if they lack serious, pulsating topics, most simply print bizarre, insignificant stories that shouldn't be reported in the first place. In times like those, haven't you already wished that you could just postpone your subscription until something newsworthy happens again?

In companies, we rarely experience a drought of news because something is always happening. Nevertheless, we often hear requests for a system that could monitor a given situation and only inform users when something specific has happened, for example, when revenues have dropped beyond a predefined limit. Did we say 'could'? With *DeltaMaster* you can! In fact, its exception reporting functionality has become extremely sophisticated with the last 5.3.6 release. Learn more in this edition of *clicks!*.

Best regards,

Your Bissantz & Company team

### **Bissantz Campus**

Following the reorganization and extension of our training programs, the Bissantz Campus is now open for the 2008 Fall/Winter semester! Business Intelligence professionals can choose from a vast selection of courses for *DeltaMaster* and Microsoft SQL Server.  
[www.bissantz-campus.de](http://www.bissantz-campus.de)

### **DeltaMaster in XING**

In the online network XING, we have recently started a new group where you can discuss issues and exchange experiences with fellow *DeltaMaster* users, ask questions to our experts as well as receive helpful tips.  
[www.xing.com/net/deltamaster](http://www.xing.com/net/deltamaster)

### **DeltaMaster@Work**

**November 27<sup>th</sup> 2008, Nuremberg**  
Create reports that say something!  
[www.bissantz.de/dm@w](http://www.bissantz.de/dm@w)

### **Archive**

[www.bissantz.de/clicks/en](http://www.bissantz.de/clicks/en)



### **Standardization in reporting: The battle continues in 2009**

In our seminars series with Dr. Rolf Hichert, we made a strong case for factual, serious, simple reporting and standardization through unified notations. The images above were taken at our last seminar in 2008...but we'll be back next year! So mark your 2009 calendars today for February 18<sup>th</sup> and October 1<sup>st</sup> as well as a special event hosted in Berlin on May 25<sup>th</sup>.

## Tip of the month *Exception reporting with ReportServer*

In addition to standard reports, you can use *DeltaMaster* in combination with *ReportServer* to report exceptional circumstances. To use this feature, you simply define the exception criteria (e.g. a threshold for variances) in the analysis session. *ReportServer* will then monitor these criteria routinely, and when the conditions have been fulfilled, it will notify the designated users, for example, in an e-mail. This concept is commonly referred to as exception reporting, alerting or monitoring.

In *DeltaMaster clicks! 6/2007*, we introduced a trick to implement exception reporting using MDX in the *Report generator*. Although you can still use this approach, the exception reporting features in *DeltaMaster 5.3.6* offer a more powerful, flexible and simple alternative.

## Alotting tasks

If you want to receive reports when specific events take place, you will first need to:

- § Define the conditions in which you wish to receive an alert (i.e. a report) or not. This takes place in the *DeltaMaster* analysis session.
- § Ensure that the conditions are monitored regularly. This is done in a *ReportServer* job, which can be automatically executed in regular intervals. *DeltaMaster* should only generate and distribute a report when these conditions have been fulfilled.

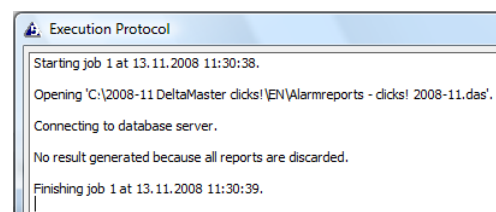
In each report, you can determine if *ReportServer* should always generate the report (this is the default setting) or only when certain criteria have been met. This configuration is stored in the analysis session (.das file).

## The procedure

In each *ReportServer Job*, a .das file serves as the template. When the session or the selected *Folder* is processed, *ReportServer* calculates and searches all reports to determine which ones should be added to the job results or not.

- § If a report is not activated for exception reporting (default setting), it will automatically be transferred into the result.
- § If a report is activated for exception reporting, *ReportServer* will first check the defined condition. If this condition has been met, *ReportServer* will include the report in the job result; if not, the *ReportServer* will ignore the report (i.e. not add it to the job results).

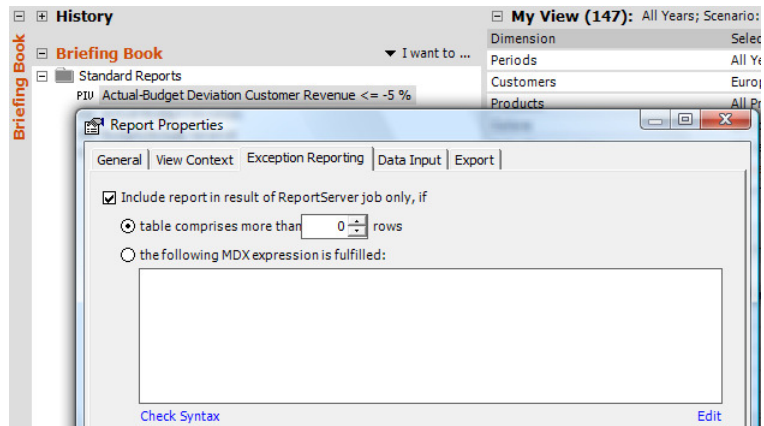
At the end of the job processing, *ReportServer* will output the results in the desired format and the desired distribution type. And if no report has been entered into the result after all of these tests, there is nothing left for *ReportServer* to do; it did its service and can remain idle for the time being. That is exception reporting.



In this case, the job will end with the message: “No result generated because all reports are discarded.” Please note that this is a status message – not an error message – because no result was intended.

### Conditions in report properties

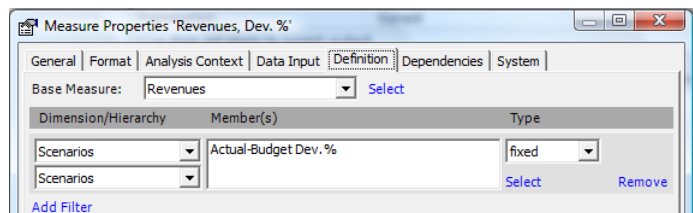
Under *Report properties* (context menu), you can define the conditions for adding a report to the result. In general, *DeltaMaster* can either use the number of data records contained in a report or a user-defined MDX expression to define exception conditions.



### Exception criteria: Report length

The first option – based on the number of rows – is designed for pivot tables or primarily tabular report types such as *Ranking* or *PowerSearch*.

One common use for alerts is a (numerical) filter. For example, we have created a user-defined measure that calculates the budget-actual revenue variance as a percentage.

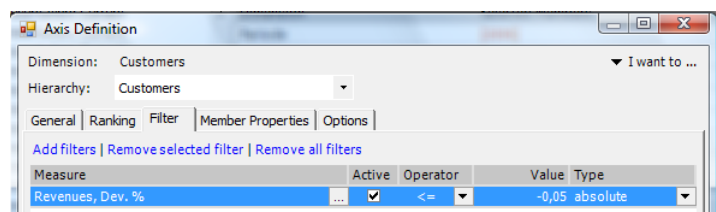


If you do not use any other filters, you will probably receive a list similar to the one on your right. Since the relative variances are still very small, the situation displayed in this list may not look very threatening to some. But we won't go off on a tangent about the tradeoffs of relative, absolute and weighted variances this time (see *DeltaMaster clicks!* 6/2008).

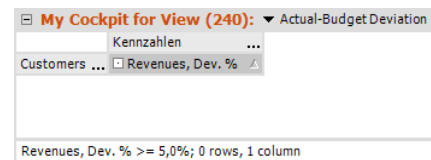
Customers	Kennzahlen
Arizona	-2,4%
Maine	-2,2%
Connecticut	-2,2%
New York	-1,8%
Arkansas	-1,6%
New York	-1,0%
New Hampshire	-0,9%
Idaho	-0,8%
Pennsylvania	-0,7%
California	-0,7%
Illinois	-0,6%
Colorado	-0,3%
Massachusetts	-0,3%
Colorado	-0,3%
South Dakota	-0,1%
California	0,0%
Kansas	0,2%
Wyoming	0,4%
Wisconsin	0,4%

Let's assume that you only want to receive an alert when a budget variance exceeds 5 %.

Using a *Filter* in the *Axis definition*, we can quickly implement this requirement in the pivot table.



The screenshot on your right displays the results. Since the current data contains no variances that are “worse” than 5%, the result contains no records (i.e. it remains empty).



If the valid condition were “*Only include the report in the ReportServer job when the table contains more than 0 rows*”, ReportServer would not generate a report with the data displayed here. If we executed that same job for a different time frame, however, a few of the regions could have numbers that are more than 5% off budget. In this case, the table would be longer, the condition would be fulfilled, and the report would be listed in the job result. In other words, it would be generated and distributed.

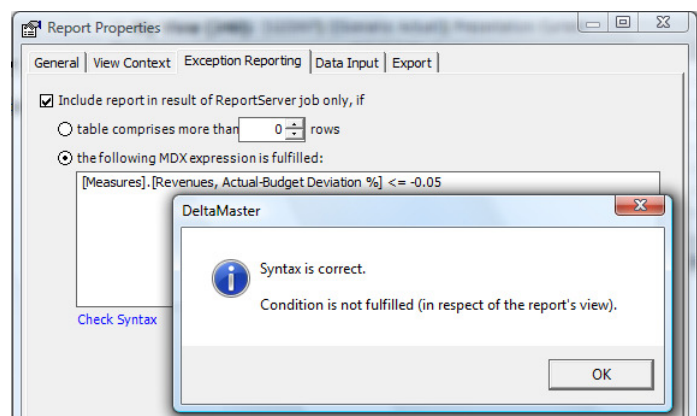
In general, if you use the number of records as a condition for exception reporting in pivot tables, you should activate the *Suppress empty rows* option (e.g. in the *Axis definition*, context menu or *I want to...* menu). Otherwise, a table could contain rows even when a query result has no values, for example, as part of the *Member selection*. In this case, ReportServer would not omit this table because its suppression rules are based on the existing rows – not if values actually exist or not. Please note, however, that ReportServer only takes data rows – not headlines - into account.

### Exception criteria: MDX conditions

Alternatively, you can define an MDX expression to create criteria that do not depend on the report or its length. In Flexreports, combination cockpits or other report types in which it is hard to determine the number of rows, you can only define *MDX conditions*.

If the result returns the expression ‘true’, the report will be added into the result of the ReportServer job. If it is ‘false’, it will not.

To edit the expression easily, you can click on the *Modify* link on the bottom right-hand side of the MDX editor; you can also use this same functionality for calculated sets, user-defined measures, etc. In the screenshot on your right, we have defined the 5 percent threshold for the total revenue variance in the report view. This definition could also apply for a geo analysis or other detailed reports that only should be presented when the overall situation requires it.



In other areas that support MDX entries, *DeltaMaster* immediately uses the expression. In exception reporting, however, it saves this expression for later use with ReportServer. As a result, you can also use the *Check syntax* option to verify if the expression is formally correct and see (for control purposes) if the condition has been fulfilled in the saved report view or not. Of course, when the job is processed later, the view will probably differ because a *Report update* or a *Report generator* will also have taken effect.

## Creating a job in ReportServer

Besides that, you do not need to change any other settings in ReportServer to run exception reports. The settings under *Report properties* will determine if it should treat a situation as an exception or not. As a result, you simply create a new job for the prepared analysis session.

Most likely, you will use 'mail' as the *Distribution type* to give recipients instant, direct notice in the case that something should happen. Since *ReportServer Office* now also supports HTML (see *DeltaMaster deltas! 5.3.6, feature #12*), you could also display these results on a PDA, Blackberry or other mobile devices.

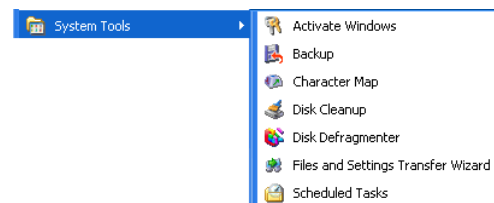
If you have prepared several briefing books for exception reporting, it is helpful to bundle these into their own job group. This way, you only need to make a single entry in your scheduler application (see below) to monitor all reports in the same interval. Simply enter the same name (e.g. 'exception reporting') for each of these jobs in the *Job group* field, which is located on the far-right side of the job definition.

Job active	Job Group	Synchronize
<input checked="" type="checkbox"/>	Exception Reporting	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	Exception Reporting	<input checked="" type="checkbox"/>

## Managing time for event monitoring

The job which monitors the threshold or other conditions is now set. It needs to be executed in regular intervals (e.g. automatically once a day) so that *DeltaMaster* can effectively monitor the defined situation. To do this, simply use the same time-planning mechanisms that you would for any other job that must be executed regularly, for example, a specialized scheduling tool (e.g. SQL Server Agent in Microsoft SQL Server), the Windows command called 'at', or *Scheduled Tasks* in your Windows control panel.

When naming the program that should be executed, select the complete path to ReportServer and add the ID of the job or job group that you would like to run.



For a job with the *JobID 5*, the expression could look like this:

```
"C:\Program Files\DeltaMaster 5.3\ReportServer.exe" 5
```

For the entire job group, the expression could be:

```
"C:\Program Files\DeltaMaster 5.3\ReportServer.exe" "Exception Reporting"
```

If there are empty spaces in the path or the group names, please use parentheses in both cases.

## Partially or not at all

In addition to suppressing the number of reports you receive, you can also use the described functions to reduce the size of your briefing books. Since the exception conditions apply to each report, you can automatically weed out some reports, for example, when numbers are missing or the users lack the required access rights. Let's assume, for example, you have saved the reports 'Budget, Domestic' and

'Budget, International' in the same report folder. If you haven't started your international planning process (and, of course, there is no data in the database), you can use exception reporting to automatically exclude the international report so that it doesn't even appear in the briefing book. This same idea applies for different access rights. If a report does not return values with the access rights for a certain user, it will also automatically suppress that report.

If you want to base your entire reporting on exceptions, please ensure that all reports in your briefing book (or in the folders which you have selected in the ReportServer job) are activated for exception reporting. Otherwise, *DeltaMaster* will generate and distribute these reports because the result of the jobs will never be 'empty'.

### *Closing words of caution*

As we have stressed several times before, we are hesitant about exception reporting because its core purpose is to withhold information. The information systems expert Norbert Szyperski already warned of these problems back in 1978:

"Fixed thresholds are dangerous in combination with the fiction of management by exception. Managers should be curious, in other words, should proactively seek out new relationships among information and not just dose off attentively like a person monitoring a switchboard." (translated from Mertens/Griese, *Integrierte Informationsverarbeitung 2*, Wiesbaden 2002)

As an alternative, the business intelligence pioneer Nicolas Bissantz suggests: "Design standard reports that are so attractive and information dense that the readers enjoy receiving them and would never even consider trying to suppress them." (<http://blog.bissantz.com/variances-2>)

### *Questions? Comments?*

Just contact your Bissantz team for more information!