IBM Cognos Proven Practices: IBM Cognos 10 Dynamic Query Mode Migration Scenarios

Product(s): IBM Cognos 10; Area of Interest: Upgrade/Migration

Daniel Wagemann
Cognos Proven Practices Advisor
IBM

Glen Seeds
Architect, Cognos OLAP Query Planner
IBM

Hamid Majdabadi
Team Lead, Cognos Platform Team
IBM

Maddy Knuth
Information Developer - IBM Cognos Analytic Applications
IBM

Rick Kenny
STSM - Cognos Data Access
IBM

Robyn Kinnaird
Editor

Ron Ferguson
Editor of technical documentation
IBM

Sylvie Savoie Evasuk
SAP BW QC
IBM

18 August 2011
This document describes various scenarios encountered when transitioning from Compatible to Dynamic Query Mode.

View more content in this series

Introduction

Purpose

This document describes various scenarios encountered when migrating from Compatible Query Mode to Dynamic Query Mode. Each scenario contains a scenario ID which can facilitate conversations with IBM Cognos Support by providing a common reference point.

Applicability

The document was validated using:

- IBM Cognos 10

Migrating from Compatible to Dynamic Query Mode

Although every effort was made to allow users to migrate reports effortlessly from Compatible Query Mode to Dynamic Query Mode, there may be scenarios where reports behave differently when migrated. These differences can be explained by the fact that Dynamic Query Mode applies clear behaviour rules consistent with the behaviour of related constructs to all applicable cases. In cases where these rules cannot be applied, the report fails with an error message identifying the underlying issue. The following sections present these scenarios in more detail, along with recommendations for resolving any differences.

Nonadjacent Nesting of Levels from the same Hierarchy

Document ID: DQM-D131-150041

OLAP

When using an OLAP-based package, a report with nonadjacent levels from the same hierarchy produces the following error:

XQE-PLN-0212 The report nests more than one level from the same hierarchy but they are not adjacent to each other. Please make levels from the same hierarchy adjacent.

The following case study illustrates the migration of a report in which the report author utilizes levels from the same hierarchy in a nonadjacent format. A crosstab projects the Sales region, Product line, and the Country respectively on the rows of the crosstab. The Years are projected
on the column. The Quantity is the measure of the crosstab. The report layout is depicted by the following image.

**Figure 1** A Crosstab with Sales region, Product line, Country on the rows and Years on the column. Quantity is the measure.

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Cells</th>
<th>Cell Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Sales region</td>
<td>#Product line</td>
<td>#Country</td>
</tr>
<tr>
<td>#Product line</td>
<td>#Country</td>
<td></td>
</tr>
</tbody>
</table>

The Sales region and Country are levels from the same hierarchy, where Sales region is located a level higher than Country within the Metadata view.

**Figure 2** IBM Cognos Report Studio metadata layout displaying Sales region and Country as levels

When executed in Compatible Query Mode, the report displays as anticipated with Sales region, Product line, and the Country respectively as the rows of the crosstab, the Years on the column and the Quantity as the measure.

**Figure 3** The projected crosstab executed in IBM Cognos Viewer using the Compatible Query Mode
Running the same report in Dynamic Query Mode produces the following error message:

XQE-PLN-0212 The report nests more than one level from the same hierarchy but they are not adjacent to each other. Please make levels from the same hierarchy adjacent.

In order to execute this report with no error, Dynamic Query Mode requires that levels from the same hierarchy be adjacent in the same hierarchical order. The example report above can be rearranged to run with Dynamic Query Mode by nesting the Country immediately next to the Sales region. This new nesting avoids the Sales region and Country levels from being split by the Product line level.

When this report is re-executed using Dynamic Query Mode, the report no longer produces an error. The report displays as anticipated with a nesting of Sales region, Country, and Product line as the rows of the crosstab, Years on the column edge and Quantity as the measure.

Figure 4 The report output using the Dynamic Query Mode with the level nesting order changed to match the hierarchical order of the hierarchy

Nesting Levels from the Same Hierarchy in a different Hierarchical order than Defined by the Metadata

Document ID: DQM-D132-150046

OLAP

When using an OLAP-based package, a report that projects levels from the same hierarchy in a different order than that defined by the metadata produces the following error when executed using Dynamic Query Mode:

XQE-PLN-0213 The report nests levels from hierarchies 'hierarchy_names' breaking the hierarchy level order. Please nest levels according to natural hierarchy level order.

The following case study illustrates the migration of a report in which the report author utilizes levels from the same hierarchy in a different order than that defined by the metadata. A crosstab projects Country and Sales region respectively on the rows. The Years are projected on the column, and Quantity is the measure. The report layout is depicted below.
Figure 5 A crosstab projects Country and Sales region respectively on the rows. The Years are projected on the column. The Quantity is the measure.

<table>
<thead>
<tr>
<th>Country</th>
<th>Sales region</th>
<th>Years</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Country#</td>
<td>#Sales region#</td>
<td>#Year#</td>
</tr>
<tr>
<td>#Sales region#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#Year#</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Sales region and Country are levels from the same hierarchy, where Sales region is located a level higher than Country within the Metadata view.

Figure 6 The metadata layout displaying Sales region and Country as levels

When this report is executed in Compatible Query Mode, the report displays Country and Sales Regions as the crosstab rows, Years as the column and Quantity as the measure.

Figure 7 IBM Cognos Viewer displaying the crosstab as Country and Sales Region nested as the rows, years as the columns and quantity as the measure

Running the same report in Dynamic Query Mode produces the following error message:

XQE-PLN-0213 The report nests levels from hierarchies 'Sales regions' breaking the hierarchy level order. Please nest levels according to natural hierarchy level order.

In order to execute this report with no error, Dynamic Query Mode requires levels nested in the same order as specified by the hierarchy. The levels nested on the rows of this crosstab example report can be rearranged to run using Dynamic Query Mode by nesting the Country under the Sales region to match the hierarchal order of the crosstab. With this new nesting in place the crosstab now projects Country and Sales region on the rows, the Years on the column, and Quantity is the measure as illustrated by the following image.
**Figure 8** Projected Crosstab with the Levels Rearranged to match the hierarchal order of the hierarchy

When this report is re-executed using Dynamic Query Mode, the report no longer produces an error. The report displays as anticipated with Sales region, Country, as the rows of the crosstab, the Years on the column and the Quantity as the measure.

**Figure 9** The report output with the Sales Region and Country nested on the row edge to match the hierarchal order of the hierarchy

---

**Same Hierarchy on Multiple Edges**

Document ID: **DQM-D134-150047**

**OLAP**

When using an OLAP-based package, a report that project levels from the same hierarchy on different edges of the crosstab, rows and columns, produces the following error:

XQE-PLN-0215 The report has levels from the same hierarchy on multiple edges. Please place levels from each of the following hierarchies on only one edge, 'hierarchy_names'.

The following case study outlines the migration of a report in which the report author utilizes levels from the same hierarchy on multiple edges. The following crosstab has the Country level as the columns and the Sales region as the rows, with Quantity as the measure. The Country level and Sales region come from the same hierarchy.

**Figure 10** Crosstab with Sales region projected on the row edge and Country projected on the column edge

When this report is executed in Compatible Query Mode, the report displays the Sales regions as the rows and the Country as the columns. The quantity measure data appears in the appropriate cells as a diagonal skew.
When this report is executed in Dynamic Query Mode, it produces the following error message:

XQE-PLN-0215 The report has levels from the same hierarchy on multiple edges. Please place levels from each of the following hierarchies on only one edge, 'level_names'.

In order to execute this report with no error, Dynamic Query Mode requires levels nested in the same order as specified by the hierarchy on a single edge. The example report can be rearranged to run with Dynamic Query Mode by nesting the Country under the Sales region on the rows and moving the Quantity measure to the columns. This crosstab projected is illustrated by the following image.

When this report is executed in either query mode, the report displays the Sales regions and country nested as the rows and the Quantity measure as the columns.

Using % of Each Column Total within IBM Cognos Analysis Studio

When using an OLAP-based package to create an Analysis Studio report where two levels of the same hierarchy are on the same edge and the option to show values as '% of Each Column Total'
has been selected, the outputs for each query mode appear differently. The following scenario illustrates the differences.

Consider the report below, which projects the Sales Region and Country nested on the rows and a set consisting of Accessories, Diagnostic Kits and Repair Kits on the columns. The measure is a % of column total calculation.

When this report is executed in Compatible Query Mode, it totals the percentages of each Country for the region to 100%.

Figure 14 Crosstab with Percentage of Children adding up to 100% for the Parent

<table>
<thead>
<tr>
<th>Quantity Sum (% of each column total)</th>
<th>Accessories</th>
<th>Diagnostic Kits</th>
<th>Repair Kits</th>
<th>All Products %</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMERICA CANADA</td>
<td>47.34%</td>
<td>45.23%</td>
<td>33.85%</td>
<td>49.00%</td>
</tr>
<tr>
<td>USA</td>
<td>52.66%</td>
<td>54.77%</td>
<td>45.15%</td>
<td>51.00%</td>
</tr>
<tr>
<td>AMERICA</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>ASIA</td>
<td>41.79%</td>
<td>68.53%</td>
<td>46.12%</td>
<td>50.35%</td>
</tr>
<tr>
<td>JAPAN</td>
<td>58.21%</td>
<td>31.45%</td>
<td>53.88%</td>
<td>49.65%</td>
</tr>
<tr>
<td>ASIA</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>EUROPE FRAANCE</td>
<td>47.35%</td>
<td>12.77%</td>
<td>30.04%</td>
<td>33.33%</td>
</tr>
<tr>
<td>GERMANY</td>
<td>22.59%</td>
<td>53.19%</td>
<td>35.70%</td>
<td>35.50%</td>
</tr>
<tr>
<td>ITALY</td>
<td>29.06%</td>
<td>34.04%</td>
<td>31.26%</td>
<td>31.17%</td>
</tr>
<tr>
<td>EUROPE</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>All</td>
<td>/0</td>
<td>/0</td>
<td>/0</td>
<td>/0</td>
</tr>
</tbody>
</table>

If this report is executed using Dynamic Query Mode, the percentage values for the measure cells change to become the percentage of total for all the Countries. Only when all the percentages are totaled for the entire report do they add up to 100%.

Figure 15 Crosstab with Percentage of Children adding up to 100% for All Parents

<table>
<thead>
<tr>
<th>Quantity Sum (% of each column total)</th>
<th>Accessories</th>
<th>Diagnostic Kits</th>
<th>Repair Kits</th>
<th>All Products %</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMERICA CANADA</td>
<td>21.97%</td>
<td>21.49%</td>
<td>21.31%</td>
<td>21.59%</td>
</tr>
<tr>
<td>USA</td>
<td>24.44%</td>
<td>26.02%</td>
<td>18.26%</td>
<td>22.47%</td>
</tr>
<tr>
<td>AMERICA</td>
<td>46.41%</td>
<td>47.51%</td>
<td>39.57%</td>
<td>44.06%</td>
</tr>
<tr>
<td>ASIA</td>
<td>8.13%</td>
<td>14.42%</td>
<td>12.08%</td>
<td>11.31%</td>
</tr>
<tr>
<td>JAPAN</td>
<td>11.32%</td>
<td>5.02%</td>
<td>14.10%</td>
<td>11.16%</td>
</tr>
<tr>
<td>ASIA</td>
<td>19.45%</td>
<td>21.04%</td>
<td>26.18%</td>
<td>22.47%</td>
</tr>
<tr>
<td>EUROPE FRAANCE</td>
<td>16.37%</td>
<td>4.01%</td>
<td>11.31%</td>
<td>11.16%</td>
</tr>
<tr>
<td>GERMANY</td>
<td>7.85%</td>
<td>16.73%</td>
<td>12.23%</td>
<td>11.89%</td>
</tr>
<tr>
<td>ITALY</td>
<td>9.02%</td>
<td>10.71%</td>
<td>10.71%</td>
<td>10.43%</td>
</tr>
<tr>
<td>EUROPE</td>
<td>34.14%</td>
<td>31.45%</td>
<td>34.29%</td>
<td>33.47%</td>
</tr>
<tr>
<td>All</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

In Dynamic Query Mode, this report displays the percentage of each value in relation to All the Countries, while in Compatible Query Mode, it returns the total for the children of the default member (typically root member).
Currently, the results returned by Dynamic Query Mode cannot be overridden by a report re-design. Should Compatible Query Mode results be required, this particular style of report would be excluded as a candidate for a move to Dynamic Query Mode.

**Slicers Using Value Expressions**

Document ID: DQM-D008-138775

**OLAP**

When using an OLAP-based package, a report that contains a slicer using a value expression produces the following error when executed:

XQE-PLN-0054 The query is not supported. The slicer contains at least one value expression. The slicer must have all set or member expressions.

Slicers are member sets. Therefore, a slicer must contain a member expression or a set expression.

Consider a crosstab report showing revenue for all product lines by retailer region. In the following example layout, the Report Author has placed Product line along the row edge and Retailer Region along the column edge, with Revenue as the measure.

**Figure 16 Crosstab report with Product line on the row edge, Retailer region on the column edge and Revenue as the measure**

![Crosstab report layout](image)

When executed, the report produces the same output in both Compatible Query Mode and Dynamic Query Mode, showing revenue in dollars and cents for each combination of Product line and Retailer region. In the following sample output, the cell at the intersection of Product line “Personal Accessories” and Retailer region “Americas”, for example, shows Revenue of 465,140,601.

**Figure 17 Crosstab showing revenue for Product lines by Retailer region**

![Crosstab showing revenue](image)

If the report author wants to see only the combined (total) revenue for Canada and United States, excluding revenue contribution from other countries, it might be tempting to use a slicer with the following member set expression:

\[[\text{Canada}] + \text{United States}\]
In Compatible Query Mode, validation of the slicer expression would result in the following warning message:

`OP-ERR-0233 Slicer Member Set expression="value(tuple([Canada])) + value(tuple([United States]))" is invalid. For anything other than a set of members from the data source, the results may not be what you expect.`

However, the report would execute without any errors and produce output, containing the desired combined revenue for Canada and United States. In the following example output, the Revenue for each combination of Product line and Retailer region is less than the corresponding value in the previous report output, which covered all countries. For example, the Revenue for the combination “Personal Accessories” and Retailer region ‘Americas’ is now 79,114,381 instead of 465,140,601.

**Figure 18 Crosstab showing combined revenue for report with slicer in Compatible Query Mode**

When performing an addition, both operands must be values and the result is always a value. In the example presented, the addition of the two members “Canada” and “United States” means “the value of member Canada + the value of member United States”. Since this is a value, not a member or set, it is not valid in a slicer. Running the modified report in the Dynamic Query Mode produces the following error message:

`XQE-PLN-0054 The query is not supported. The slicer contains at least one value expression. The slicer must have all set or member expressions.`

For this example, the appropriate solution is to create a slicer member set from the chosen members (Canada and United States) without an explicit calculation. The slicer member set expression then becomes:

`set([Canada], [United States])`

This slicer works as intended because the default rollup for the hierarchy containing the chosen members is Total. Therefore, the revenue in each cell of the crosstab is aggregated by summing the contributions from Canada and the United States.

When the report is executed with the modified slicer member set, the output again contains the combined (total) revenue for Canada and United States. The following output is identical to the previous sample output. For example, the Revenue for the combination “Personal Accessories” and Retailer region “Americas” is 79,114,381, as before.
An alternative solution is to replace the slicer by an explicit calculation nested on one of the edges of the crosstab. For example, the calculation can be nested along the column edge under Retailer region. Executing this report would produce output that is identical to the previous two sample outputs. Once again, the Revenue for the combination “Personal Accessories” and Sales region “Americas” is 79,114,381, as displayed in the following crosstab.

Although this solution requires a change to the report layout, it would be appropriate when the hierarchy containing the members used in the previous slicer does not have the desired default rollup (Total).

Summary Filter with Invalid Scope

When using an OLAP-based package, a report containing a summary filter with an invalid scope produces the following error:

XQE-PLN-0035 The query is not supported. It contains a summaryFilter with a scope that is not a projected dataItem.

Consider the following list report which consists of three columns, Product line, Region and Revenue. The Product line has been grouped and a summary total on Revenue has been added.
Figure 21 Layout for list report with Revenue by Product line and Region, grouped by Product line

In order to display only product lines with total revenue greater than three hundred million dollars, the report author adds a summary filter scoped on Product line and based on data item Revenue with the following condition:

\[ \text{Revenue} > 300000000 \]

If this report is executed in Compatible and Dynamic Query Mode, only Product lines with total revenue greater than three hundred million dollars are displayed.

Suppose the report author now changes the report to show the revenue for Product type instead of product line. The list report now consists of three columns, Product type, Region and Revenue. The Product type has been grouped and a summary total on Revenue has been added.

Figure 22 Layout for list report with Revenue by Product type and Region, grouped by Product type

Although the Product line has been replaced with the Product type query item, the summary filter is still in place and scoped on a now non-projected data item “Product line”.

If this report is validated using Compatible Query Mode, the following warning is presented:

OP-ERR-0210 The summary filter scope (Product line) found in the query (Query1) is not valid. It must refer to a projected query item. If it is projected, make sure that it is not a sibling to a query item of a different dimension.

Despite the warning, executing this report in Compatible Query Mode produces the expected results. The screen capture below displays a simple list with the Product type, Region and Revenue as columns. Product type has been grouped and a summary has been applied to Revenue. Only Product types that have combined Revenue greater than 300,000,000 are displayed.
Figure 23 IBM Cognos Viewer displaying a list report with Product types that have combined Revenue greater than 300,000,000

Executing the same report in Dynamic Query Mode produces the following error message:

XQE-PLN-0035 The query is not supported. It contains a summaryFilter with a scope that is not a projected dataItem.

In order to run this report in Dynamic Query Mode, the report author should determine whether the summary filter is still required or not. If not, the report can be fixed by removing the filter.

If the report author wants to filter summary values for Product type, the summary filter can be modified to set the scope accordingly, and the threshold value changed to something appropriate for the Product type level.

If the business requirement is to show revenue for Product type by region while only showing the rows of data for which the total revenue of the non-projected data item Product line is greater than $300,000,000, a summary filter is not appropriate. In this case, the report author can use a mandatory after-auto-aggregation detail filter with a filter expression such as the following:

Aggregate([Revenue] for [Product line]) > 300000000

Running the report with this filter would produce output identical to that of the original report in Compatible Query Mode. By using the for clause in the after-aggregation detail filter on a non-projected data item, the underlying query automatically projects and groups the item as the outermost column of the list. If the equivalent report were to be created from scratch, the list report would consist of the columns Product line, Product type, Region and Revenue. The Product line and Product type columns would have a grouping and summary applied. The layout for such a report is illustrated by the following image.
Figure 24 List report layout consisting of Product line, Product type, Region and Revenue

As Product line is a projected query item, the filter for this particular report would be the original summary filter of:

\[[Revenue] > 300000000\]

NOT Operator in a Detail Filter

When using an OLAP-based package, a report that uses a detail filter containing the NOT operator applied to a hierarchy produces the following error:

XQE-PLN-0190 Detail filters on multi-level member sets using the NOT operator are unsupported.

The following case study outlines the migration of a report in which the report author tries to exclude certain members of a level in a hierarchy. For this particular example, a report author has created a simple list report which consists of the Product line and Product type columns.

Figure 25 List report with two columns: Product line and Product type

When the report is executed, it produces a simple list report with the data for Product line and Product type as illustrated by the following image.
In order to filter out all Product types that start with the letter C, the Report Author creates a filter on the entire Products hierarchy (all levels) with the following expression:

\[ \text{not (caption([data_source].[Products].[Products]) starts with 'C')} \]

When executed in Compatible Query Mode, the report displays all Product types which do not start with the letter 'C'. This is because the filter applies only to the lowest projected level of the Products hierarchy, Product line is not filtered. Therefore, Camping Equipment still appears in the output. This is illustrated by the following image.

**Figure 27 The list output after the application of the filter with Product types that start with the letter 'C' removed**

If the same report is run in Dynamic Query Mode, the following error message appears:

\[ \text{XQE-PLN-0190 Detail filters on multi-level member sets using the NOT operator are unsupported.} \]

To execute this report in Dynamic Query Mode without errors, you can apply the filter to the Product type level using the following expression:

\[ \text{not (caption([Product type]) starts with 'C')} \]
The modified report runs in both Dynamic Query Mode and Compatible Query Mode, producing output identical to the output from Compatible Query Mode for the report with the initial filter expression.

**Using the NestedSet Function outside an Analytic Summary**

Document ID: **DQM-D027-157125**

**OLAP and DMR**

A report using the function `nestedSet` outside the within set clause of an analytic summary function (such as rank) produces the following error:

**XQE-PLN-0130** NestedSet is intended to be used only in the 'within set' parameter of the Rank function. In other contexts, it may not do what you expect.

In Dynamic Query Mode, use of the `nestedSet` function is restricted to the within set clause of analytic summary functions, such as the following:

- percentage
- percentile
- quantile
- quartile
- rank

Even when using the `nestedSet` with the functions above, results may not always be what you expect. For this reason, the `nestedSet` function is documented as intended for use by IBM Cognos Analysis Studio only. Using it in other studios is not recommended.

The following case study illustrates the migration of a report that uses the `nestedSet` function in an expression outside the `within set` clause of analytic summary function. For this example, the crosstab report consists of Quantity on the column edge and Product line on the outer row edge nested with the Top Two Products.

**Figure 28 Report layout with Quantity for Top Two Products nested within Product line**

<table>
<thead>
<tr>
<th>Product line</th>
<th>Top Two Products#</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#123#</td>
</tr>
<tr>
<td></td>
<td>#123#</td>
</tr>
<tr>
<td></td>
<td>#123#</td>
</tr>
<tr>
<td></td>
<td>#123#</td>
</tr>
</tbody>
</table>

The Top Two Products calculation consists of the following expression:

```
    NestedSet ( members ([Product line]), topCount ( descendants ( currentMember ( [data_source].[Products].[Products]), [data_source].[Products].[Products].[Product type]), 2, [Quantity]) )
```

IBM Cognos Proven Practices: IBM Cognos 10 Dynamic Query Mode Migration Scenarios
When the report is run in Compatible Query Mode, it displays the product types which have the highest and second highest quantities for each of the Product Lines. The following image provides a visual representation of this output.

**Figure 29 Output showing Quantity for the top two products in each Product line**

When this report is executed in Dynamic Query Mode, it produces the following error:

XQE-PLN-0130 NestedSet is intended to be used only in the 'within set' parameter of the Rank function. In other contexts, it may not do what you expect.

To run this report with no errors, Dynamic Query Mode requires that the NestedSet function be removed from the Top Two Products calculation, as shown in the following expression:

```sql
topCount ( descendants ( currentMember ( [data_source].[Products].[Products]), [data_source].[Products].[Products].[Product type] ), 2,[Quantity] )
```

The modified report runs in both Dynamic Query Mode and Compatible Query Mode, producing output identical to the output from the unmodified report run in Compatible Query Mode.

**Different Namespaces Based on the Same Dimension**

**Document ID:** DQM-D116-138602

When using a DMR package, a report containing items from dimensions in different namespaces that are based on the same underlying dimension will result in two distinct dimensions.

The following case study illustrates the migration of a report containing items from two different namespaces based on the same dimension. Consider a simple list report that contains only the Product line level from one namespace and the Product type level from another namespace. Both of the dimensions that these items are taken from are based on the same relational table, in which Product line and Product type are related.
Figure 30 Simple list report with two projected items from two dimensions in different namespaces

<table>
<thead>
<tr>
<th>Product line</th>
<th>Product type</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Product line&gt;</td>
<td>&lt;Product type&gt;</td>
</tr>
<tr>
<td>&lt;Product line&gt;</td>
<td>&lt;Product type&gt;</td>
</tr>
<tr>
<td>&lt;Product line&gt;</td>
<td>&lt;Product type&gt;</td>
</tr>
</tbody>
</table>

When executed in Compatible Query Mode, the report displays only the Product types for Camping Equipment. Compatible Query Mode treats the two dimensions from separate namespaces as if they came from the same dimension because they are based on the same relational table.

Figure 31 List report output with levels from two dimensions

If the report is executed using Dynamic Query Mode, it lists all possible Product types for each Product Line. Dynamic Query Mode treats the two dimensions from separate namespaces independently, thus maintaining the intent of the model.

Figure 32 List report with levels from two dimensions treated independently

If the report is modified to include Product line and Product type from the same dimension of a single namespace, the output from both Dynamic Query Mode and Compatible Query Mode is identical to the output from the original report executed in Compatible Query Mode.
Summary Filters in Master-Detail and Sectioned Crosstabs

When using OLAP or DMR based packages, reports which contain summary filters in a sectioned crosstab will produce the following error:

XQE-PLN-0159 Unsupported summary filter with a scope on a query with master detail relationship to itself.

Consider the following crosstab report in which the report author has nested Product line and Sales Region as the rows and Sales region as the column. Revenue has been chosen as the crosstab measure. A summary calculation has been applied to the Year totalling the Revenue for all the Sales regions. This crosstab is also illustrated in the following screen capture.

**Figure 33 Crosstab with Product line and Product type as rows, Sales region and Summary as columns, Revenue as the measure**

![Crosstab with Product line and Product type as rows, Sales region and Summary as columns, Revenue as the measure](image1.png)

The report author now decides to add a summary filter on Revenue and scoped on Product type, to remove from the output any summary values less than or equal to $300,000,000. The report author also decides to section the crosstab on Product line. The sectioning of the crosstab places the Product line query item into the header and generates a master-detail relationship. The following image displays these changes to the layout of the crosstab.

**Figure 34 Crosstab displaying Product type on the rows, Sales regions on the columns, Revenue as the measure and sectioned on Product line**

![Crosstab displaying Product type on the rows, Sales regions on the columns, Revenue as the measure and sectioned on Product line](image2.png)

When the report is run in Compatible Query Mode it displays a separate crosstab for each Product line within only the Product types whose total Revenue across all the Sales regions are greater than $300,000,000. The following image illustrates the report output in IBM Cognos Viewer.
Figure 35 IBM Cognos Viewer output showing the sectioned crosstab

If this same report is executed using Dynamic Query Mode, it will produce the following error:

XQE-PLN-0159 Unsupported summary filter with a scope on a query with master detail relationship to itself.

The error is caused by the master-detail relationship between the list section showing Product line and the crosstab. As both the master and the detail are using the same query, the summary filter applies to both the master and the detail. Since Product type, which is the scope of the summary filter, is projected only in the detail, the summary filter is invalid for the master.

In order to run this report without error using Dynamic Query Mode, the master-detail relationship between the list section and the crosstab will have to be split into two separate queries. In this scenario, the list object will need to be associated to a new master query which consists of only the Product line data item with no summary filter. The following screen capture displays the Report Studio Properties sheet of the List object with the new query association to the master query.

Figure 36 List properties displaying the Query association

With the list object associated to the master query without the summary filter, the report now executes using Dynamic Query Mode and returns the expected results.

Page Sets and List Summary Filters Sharing the Same Query

Document ID: DQM-D083-145498

OLAP and DMR

When using OLAP or DMR based packages, reports which contain a summary filter, a Page set and two lists sharing the same query produce the following error:
Unsupported summary filter with a scope on a query with master detail relationship to itself.

The following list consists of Quantity sold by Product type and Sales region. In addition to the groupings applied to Product type and Sales region, a summary calculation has been added to Quantity sold. A summary filter was added based on the data item Quantity sold and scoped on Product type, to remove from the output any summary values less than or equal to 3,000,000,000. This is also illustrated by the following screen capture.

**Figure 37 Quantity sold by Product type and Sales region**

<table>
<thead>
<tr>
<th>Product type</th>
<th>Sales region</th>
<th>Quantity sold</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Product type&gt;</td>
<td>&lt;Sales region&gt;</td>
<td>&lt;Quantity sold&gt;</td>
</tr>
<tr>
<td>&lt;Product type&gt;</td>
<td>&lt;Sales region&gt;</td>
<td>&lt;Quantity sold&gt;</td>
</tr>
<tr>
<td>&lt;Product type&gt;</td>
<td>&lt;Sales region&gt;</td>
<td>&lt;Quantity sold&gt;</td>
</tr>
<tr>
<td>Overall - Summary</td>
<td>&lt;Summary&gt;</td>
<td>&lt;Summary&gt;</td>
</tr>
</tbody>
</table>

In order to display one Product line per page, a page set is created within the Page explorer which groups on Product line. The following image gives a visual representation of the page set and the Product line grouping.

**Figure 38 Page set is grouped by Product line**

In addition to the first list, the report author decides to add a second list to display the Revenue by Product line and Product type. This second list is associated to the same query as the first. The following screen capture illustrates the report layout with the two stacked lists. The leftmost list still consists of Quantity sold by Product type and Sales region with the groupings on Product type and Sales region and a summary calculation on Quantity sold. The rightmost list consists of Product line, Product type and Revenue with a grouping applied to Product line and Product type.
Figure 39 Authored report with two lists using the same query

When the report is run in Compatible Query Mode it displays the two list objects side by side. The list on the left displays the Product type, Sales region and Quantity Sold while the list on the right displays the Product line, Product type and Revenue. Due to the page set grouping, the list on the right will only display one Product line per page. This is also illustrated by the following image.

Figure 40 IBM Cognos Viewer output for the two lists

If this same report is executed using Dynamic Query Mode, it will produce the following error:

XQE-PLN-0159 Unsupported summary filter with a scope on a query with master detail relationship to itself.

The error is caused by the master-detail relationship created internally by the Page Set grouping. As both the master and the detail are using the same query, the summary filter applies to both the master and the detail. Since Product type, which is the scope of the summary filter, is projected only in the detail, the summary filter is invalid for the master.

In order to run this report without error using Dynamic Query Mode the master-detail relationship between the two lists will have to be split into two separate queries. In this scenario, the right list object and the page set grouping will need to be associated to another query which consists of Product line, Product type and Revenue with no summary filter.

With the right list object and the page set grouping associated to a query without the summary filter, the report now executes using Dynamic Query Mode and returns the expected results.

Tuple Function with Value Expressions

Document ID: DQM-D016-13877
OLAP

When using OLAP based packages, reports which use value expressions as parameters to the tuple function produce the following error:

XQE-PLN-0093 The query is not supported. The argument at position 1 of 'tuple' from the data item 'Tuple_Revenue/Product cost,Camping Equipment' cannot be coerced into a 'Member' expression.

Within the following crosstab the report author projected the Year as the rows and a calculation which uses the tuple function on the columns. The expression for this calculation consists of tuple([Revenue Divided by Product Cost],[Camping Equipment]) where [Revenue Divided by Product Cost] is a separate data item with the expression [Revenue]/[Product cost]. This crosstab is also illustrated in the following screen capture.

Figure 41 Crosstab layout with Year for the rows and the tuple calculation for the columns

>When the report is validated using Compatible Query Mode it displays the following warning message.

OP-ERR-0182 Invalid parameter '[Revenue Divided by Product Cost]' provided for 'tuple' at position '1' for 'dataItem="Tuple Calculation"'. This parameter is 'Numeric' type but is expected to be one of the following: 'member'.

Despite the warning message when the report is executed using Compatible Query Mode, the crosstab displays with the years as the rows, and the resultant value of Revenue divided by Production cost for only Camping Equipment as the columns. The report output is illustrated by the following screen capture.

Figure 42 IBM Cognos Viewer output showing Years as rows and the tuple calculation as the columns

If this same report is executed using Dynamic Query Mode, it will produce the following error:
The query is not supported. The argument at position 1 of 'tuple' from the data item 'Tuple_Revenue/Product cost,Camping Equipment' cannot be coerced into a 'Member' expression.

The error is caused by passing something other than a member or a measure as the parameters to the tuple function.

In order to execute this report with no error in Dynamic Query Mode, the original tuple calculation should be replaced with the following calculation:

\[
\text{tuple}([\text{Revenue}], [\text{Camping Equipment}])/\text{tuple}([\text{Product cost}], [\text{Camping Equipment}])
\]

With this change the modified report now runs in both Dynamic Query Mode and Compatible Query Mode, producing identical output.

**Grouping on Attributes**

**Document ID:** DQM-D143-138738

DMR

When using DMR based packages, certain reports that group on attributes may produce different output.

The following list report consists of the Product size attribute and the Revenue measure. A detail filter has been applied on the Product size to only show product sizes which equal the word '100 ml'. This report layout is also illustrated by the following screen capture.

**Figure 43 List Object with the Product size attribute and Revenue**

When this report is executed using Compatible Query Mode, it returns one row for the 100 ml product size. The revenue value is the summary of all products of the 100 ml size. The report output is illustrated by the following screen capture.

**Figure 44 IBM Cognos Viewer output showing 1 record for the 100 ml Product size attribute**
If this same report is executed using Dynamic Query Mode, it returns multiple rows for the 100 ml product size. The revenue value is the detail value for each product that has a size of 100ml. The report output is illustrated by the following screen capture.

**Figure 45 IBM Cognos Viewer output showing 1 record for each member with the 100 ml Product size attribute**

The reason for the difference in output between the two query modes is that Compatible Query Mode applies relational semantics to the report as it does not contain any dimensional functions. With this approach, the report is auto summarized by the 100 ml product size. On the other hand, Dynamic Query Mode applies dimensional semantics as it would for an OLAP source. With this approach, grouping on an attribute is not possible. This means the output will display a single row of data for each Product detail member whose Product size is 100 ml.

For this scenario, the report cannot be altered to produce the same in both query modes. If the revenue is required to roll up for the product size, the model may need to be changed to include replace the attribute with a physical dimension within the model.

**Comparing Different Data Types within Expressions**

Document ID: DQM-D136-138736

OLAP and DMR

When using OLAP or DMR-based packages, reports with expressions that compare values with different data types will produce the following error in Dynamic Query Mode:

XQE-EXE-0044 Invalid to compare values with different data types.

In the following example, the report author creates a simple list report that contains only a projected data item called Year, which uses the Year level with the caption function as follows:

Caption([Sales (analysis)].[Time].[Time].[Year])

**Figure 46 A simple list report layout of the query item using the caption function**
The report author then applies a detail filter that compares a string to an integer, using the expression \([\text{Year}]=2004\).

When the report is executed in Compatible Query Mode, the report produces the following a list report with a single row for the 2004 year. This is also illustrated by the following image.

**Figure 47 Simple list report showing the Year result of a detail filter that specifies the value 2004**

![Simple list report showing the Year result of a detail filter that specifies the value 2004](image)

If the same report is executed using Dynamic Query Mode, the report produces the following error:

**XQE-EXE-0044** Invalid to compare values with different data types.

To execute this report without error in Dynamic Query Mode, the detail filter should be changed so that it compares values with the same data type. For this example, the detail filter would need to compare the Year level to a member from the Year level, as follows:

\([\text{Sales (analysis)}].[\text{Time}].[\text{Time}].[\text{Year}]=2004\)

With this change the modified report now runs in both Dynamic Query Mode and Compatible Query Mode, producing identical output.

**Vendor Specific Relational Functions in a DMR Report**

Document ID: **DQM-D144-140597**

When using a DMR package in Dynamic Query Mode, a report that contains an expression with any vendor specific relational function will produce an error message similar to the following.

**XQE-PLN-0098** The function 'function_name' is not supported.

All non-SAP BW vendor specific relational functions operate on relational query subjects or query items which are not available in OLAP-style reporting over relational data using a DMR package. Dynamic Query Mode does not allow a dimensional object, such as a member, as an argument to a vendor specific relational function. Therefore, vendor specific relational functions are not supported in this context.

In the following example, the report author creates a simple list report that contains the attributes called First name and Last name. A third column containing a query calculation called Full name...
was also added. The Full name calculation consists of the following expression using the IBM DB2 vendor specific relational function concat for the concatenation of strings.

```
concat([First name],concat(' ',[Last name]))
```

This report layout is illustrated by the following screen capture.

**Figure 48 List report layout showing First name and Last name attributes and the Full name calculation**

![List report layout](image)

When this report is executed in Compatible Query Mode it produces a list report with three columns. The first two columns display the first and last name of staff members while the third column displays the first and last name separated by a space. This output is also illustrated by the following screen capture.

**Figure 49 List report output with First name, Last name and Full Name as displayed in IBM Cognos Viewer**

![List report output](image)

When this report is executed in Dynamic Query Mode, the report produces the following error message:

```
XQE-PLN-0098 The function 'concat' is not supported.
```

To execute this report without error in Dynamic Query Mode, the Full name calculation should be changed so that it uses the non vendor specific functions found within the Common Functions or Operators folders of the IBM Cognos Report Studio Expression Editor available components section. In this particular scenario, the calculation can be changed to the expression: `[First name] || ' ' || [Last name]`.

With this change the modified report now runs in both Dynamic Query Mode and Compatible Query Mode, producing identical output.
Non Consecutive Same Hierarchy Nesting in a List Report

Document ID: DQM-D145-144729

OLAP and DMR

When using an OLAP or DMR package for a list report, grouped levels from different hierarchies must not be interspersed. If they are the report will produce the following error when executed using Dynamic Query Mode:

XQE-PLN-0187 Some grouped dataItems of hierarchy <hierarchyName> are not consecutive. Check dataItems <dataItemName> and <dataItemName> of query <QueryName>.

Consider the following list report in which the report author has projected the levels Year, Retailer and Month and grouped on each of the columns. Year and Month are levels from the same hierarchy while Retailer, placed between them, is from a different hierarchy. This is also illustrated by the following screen capture.

**Figure 50 A list report projecting the levels Year and Month separated by the Retailer level from a different hierarchy**

When executed in Compatible Query Mode, the report displays the Year, Retailer and Month as columns. This is illustrated by the following screen capture.

**Figure 51 List report showing the Year, Retailer and Month as columns**

When this report is executed in Dynamic Query Mode, the report produces the following error message:

XQE-PLN-0187 Some grouped dataItems of hierarchy 'Time' are not consecutive. Check dataItems 'Month' and 'Year' of query 'Query1'.

In order to execute this report with no error, Dynamic Query Mode requires two columns which project sets from the same hierarchy appear in consecutive groups. The columns Retailer and Month will have to be ungrouped, rearranged in order and then grouped again for the report to execute in Dynamic Query Mode. The following image shows the report layout after rearranging the columns and re-doing the grouping. The list column order is now Year, Month and Retailer.

**Figure 52 List report with columns Retailer and Month rearranged so that the levels Year and Month now appear in consecutive groups**

With this change; the modified report now runs in Dynamic Query Mode but produces a different output. The following image illustrates the change in the list output. The list report now displays the Year and Month in grouped format with all the Retailers listed for each Month.

**Figure 53 List report showing the Year, Month and Retailer as columns**

**Index of Error Codes**

- XQE-PLN-0035
- XQE-EXE-0044
- XQE-PLN-0054
- XQE-PLN-0093
- XQE-PLN-0098
- XQE-PLN-0130
- XQE-PLN-0159
XQE-PLN-0187
XQE-PLN-0190
XQE-PLN-0212
XQE-PLN-0213
XQE-PLN-0215
About the authors

Daniel Wagemann

Daniel Wagemann is an IBM Cognos Proven Practice Advisor for Business Analytics in Canada. In his 11 years working with the IBM Cognos product Suite, he has established a vast understanding of all areas of an IBM Cognos deployment. His areas of expertise include course development, technical writing, consulting and customer support. His work can be found within almost all areas of the Proven Practices Site.

Glen Seeds

Glen Seeds is an architect in the Cognos BI Platform team. His role covers all aspects of dimensional queries, with focus on aligning query engine semantics to the needs of BI authors. In the 6 of his 14 years with the company in this function, he has made significant contributions to product documentation and training material, to guide authors in creating reports that deliver surprise-free results against dimensional data.

Hamid Majdabadi

Hamid Majdabadi is a team lead in the IBM Cognos platform team, responsible for quality assurance of the IBM Cognos 10 Dynamic Query Engine, primarily in TM1 and Microsoft infrastructure and integration. His technical interests are mostly in OLAP and integration of TM1 and Microsoft SQL Server Analysis Services with IBM Cognos BI and reporting. Hamid has over 12 years experience in quality assurance.

Maddy Knuth

Maddy Knuth is an IBM Cognos Information Developer working on IBM Cognos product documentation in Canada. She is currently working on an assortment of user documents, including the IBM Cognos Analytic Applications documentation. Her expertise is in task-based user documentation, and she is passionate about grammar and innovation in customer documentation.
Rick Kenny

Rick Kenny is a Senior Technical Staff Member at IBM Cognos, specializing in query languages. Currently, he is leading efforts related to the adoption of the Dynamic Query Mode and the definition of a formal specification for the IBM Cognos query language. During his 6 years at Cognos, he has led the Cognos Architecture Council and various cross-product architecture initiatives. Rick is a veteran of the software industry, with over 25 years experience building enterprise systems for clients and as product, covering fields as diverse as information retrieval, remote asset tracking and literacy training.

Robyn Kinnaird

Robyn Kinnaird is a technical editor at IBM. She has worked as an editor and technical writer for over 20 years. Robyn has written and developed documentation and technical literature for the telecommunications, medical, personal productivity software, networking, and business analytics software industries. She has a Bachelor's degree in English Language and Writing and is a member of the Editor's Association of Canada.

Ron Ferguson

Ron Ferguson is a technical editor for IBM Cognos products. Ron has worked as a writer and editor in the software business for 20 years and has been with Cognos for seven years.

Sylvie Savoie Evasuk

Sylvie Savoie Evasuk has been a Software Quality Assurance Engineer within the Cognos Lab for 16 years. She has worked with legacy Cognos products (Architect, Cognos Query, Powerhouse) and has been involved in the software development of newer products like: Report Studio, Query Studio, Analysis Studio, Framework Manager, PowerPlay Studio and Cognos 8 Go! Dashboard. She obtained her degree from St Francis Xavier University with honors in Psychology, followed by a one-year intensive Computer Science diploma. She understands both the customer and design perspective of Cognos products. Sylvie presently holds the position of Team Lead for the Platform Test Organization.