QUICK START MANUAL

OLATION V20

PARIS

OLATION®

VISIONARY INTELLIGENCE

QUICK START MANUAL

Information in this document and in documents associated with other PARIS Technologies Training materials is subject to change without notice, and does not represent a commitment on the part of PARIS Technologies. No part of this manual may be reproduced or transmitted in any form, or by any means whatsoever, without the written permission of PARIS Technologies International, Inc.

Windows is a trademark of the Microsoft Corporation.

Microsoft Excel is a trademark of the Microsoft Development Corporation.

This page has been left blank intentionally.

Table of Contents

1. Introduction to OLATION® 1

2. Demo Overview 1

3. Create a New OLATION® Database 2

4. Create the SALES Cube 4

4.1. Create Dimensions from Relational Tables 5
4.1.1. Create the Account Dimension 5
4.1.2. Create the Version Dimension 14
4.1.3. Create the Month Dimension 15
4.1.4. Create the Region Dimension 17
4.2. Create a Cube from a Relational Table - SALES Cube 19
4.2.1. Assign the Measure Member 21
4.2.2. Arrange the Dimension Order 22
4.2.3. Define Relationships 23
4.2.4. Define Cube Setting: Disable READ-ONLY Setting 24
4.2.5. Save the Cube 24

5. Viewing the OLATION Data 26

5.1. Create the Dynamic Excel Front End–-PowerExcel Slice 26
5.2. Viewing OLATION Data via PowerOLAP's OLATION Linked Cube Feature 34
5.2.1. Create a New PowerOLAP Database 34
5.2.2. Create a New OLATION Linked Cube 35
5.2.3. Viewing the OLATION Metadata Structure built in PowerOLAP 37
5.2.4. Viewing the Fact Data - Create a PowerOLAP Slice View 40
5.2.5. Arranging the Slice View 41
5.2.6. Create the Dynamic Excel Front End – PowerOLAP Slice to Excel 44

6. OLATION® and Real-Time Application Results 46

6.1. Pull Data Updates from the Relational Source down to the OLATION Database 46
6.2. Write-back (for Planning) to the Source Relational Database 48

7. Define Formula in OLATION® 53

7.1. Define the Cube Formula Statement 55
7.2. Checking the Cube Formula Results 57

8. Create the PRODUCT_SALES Cube 60

8.1. Pre-Work: Prepare the Product Factdata SQL Table 60
8.1.1. Create the Product_Factdata SQL Table in SQL Server Studio 60
8.1.2. Modify the Product_Factdata SQL Table to Include Products Column 62
8.2. Create a Multidimensional Dimension - *Product* Dimension 63
8.2.1. Add Dimension Members 65
8.2.2. Define Aliases 66
8.2.3. Define Properties 67
8.2.4. Define a Subset 68
8.2.5. Configure Other Dimension Settings - Settings Tab 69

8.2.6.Save the Dimension 69
8.3. Create the PRODUCT_SALES Cube 71
8.3.1.Assign the Measure Member 73
8.3.2.Arrange the Dimension Order 74
8.3.3.Define Relationships 74
8.3.4.Define Cube Setting: Disable READ-ONLY Setting....75
8.3.5.Save the Cube....76
8.4. View the PRODUCT_SALES Cube 77
8.5. Create a Cross-Cube Formula for the PRODUCT_SALES Cube....79
8.5.1.Viewing the Results of the Cross-Cube Formula 82

OLATION® Quick Start Manual

1. Introduction to OLATION®

The Olation® Quick Start Manual is designed to introduce users to the fundamentals of the Olation® application by using step-by-step procedures to create an Olation® model. PARIS Technologies recommends that new Olation® users familiarize themselves with Introduction to Olation section of the Olation User Manual as a prerequisite to this manual. This Quick Start manual also assumes that that product is installed, with prerequisites in place to use the product.

By working through this Quick Start Manual, you will learn about the elemental functions of the Olation application: how to create an Olation database; how to create Dimensions from relational tables as well as create a multidimensional type of Dimension; how to define Dimension attributes; how to create or edit the Dimension hierarchy; how to create a Cube and define Cube attributes, and; how to create simple internal Cube and cross-cube formulas. Note that—especially if you are an Administrator, you will not only build cubes (business models) by the steps described here but will undoubtedly need to understand the concepts and follow the steps covered in these pages because they convey how to use, customize and advance your models. These skills will enable you to vastly increase the potential business uses and benefits of the product at your organization.

2. Overview

The exercises in this Module serve as a quick tutorial on how to build simple Olation Models which will cover topics such as:

- Basic Modeling in Olation*
- Defining Dimension and Cube Attributes
- Creating simple Cube and Cross-Cube Formulas
- Dynamic Writeback Capability of an Olation Model

*Note: the underlying SQL database *Using_Olation* is utilized in this manual, so it will be necessary to install/access it in order to follow the step-by-step exercises. Furthermore: this database has been organized in such a way as to quickly enable you to build a cube from relational tables. In a production system, you will likely need to reorder/remake tables with a view in mind of the cube(s) you intend to create in Olation. As well, this database does not feature indexed tables, which you will more likely encounter in a real-world scenario.

3. Create a New OLATION® Database

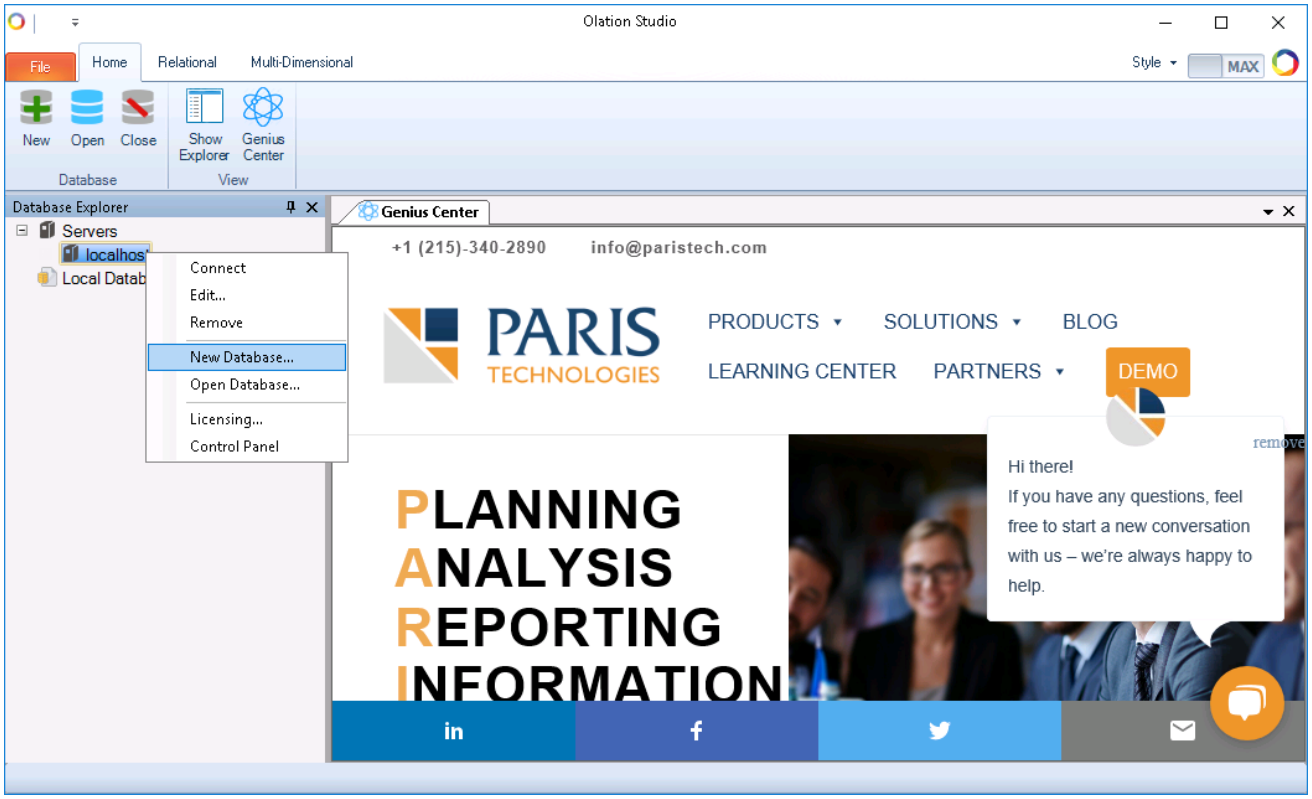
This section is concerned with creating an Olation database from a source relational database. The Olation database and its components will be built from logic in the source database and will demonstrate the dynamic connection between Olation and the source database.

IMPORTANT: For this exercise, we will create an Olation database hosted on an Olation Server working with a local instance of SQL Server. We will be using the **USING_OLATION** SQL database to build our database and its corresponding Dimensions, Cubes and attributes.

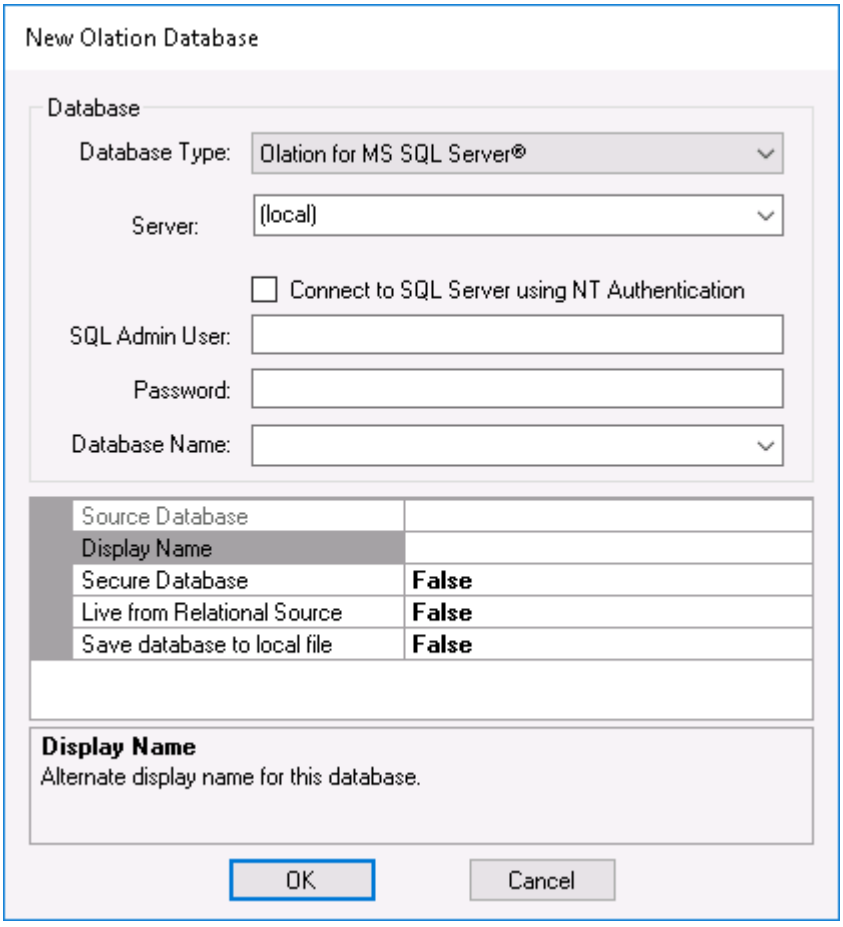
To create a new OLATION® Database:

1. Launch OLATION® Studio.
2. In the Database Explorer, expand **Servers**, then right-click on **localhost** (this will be our preferred Olation Server for this demo) and select **Connect**, to establish a connection to the Server.

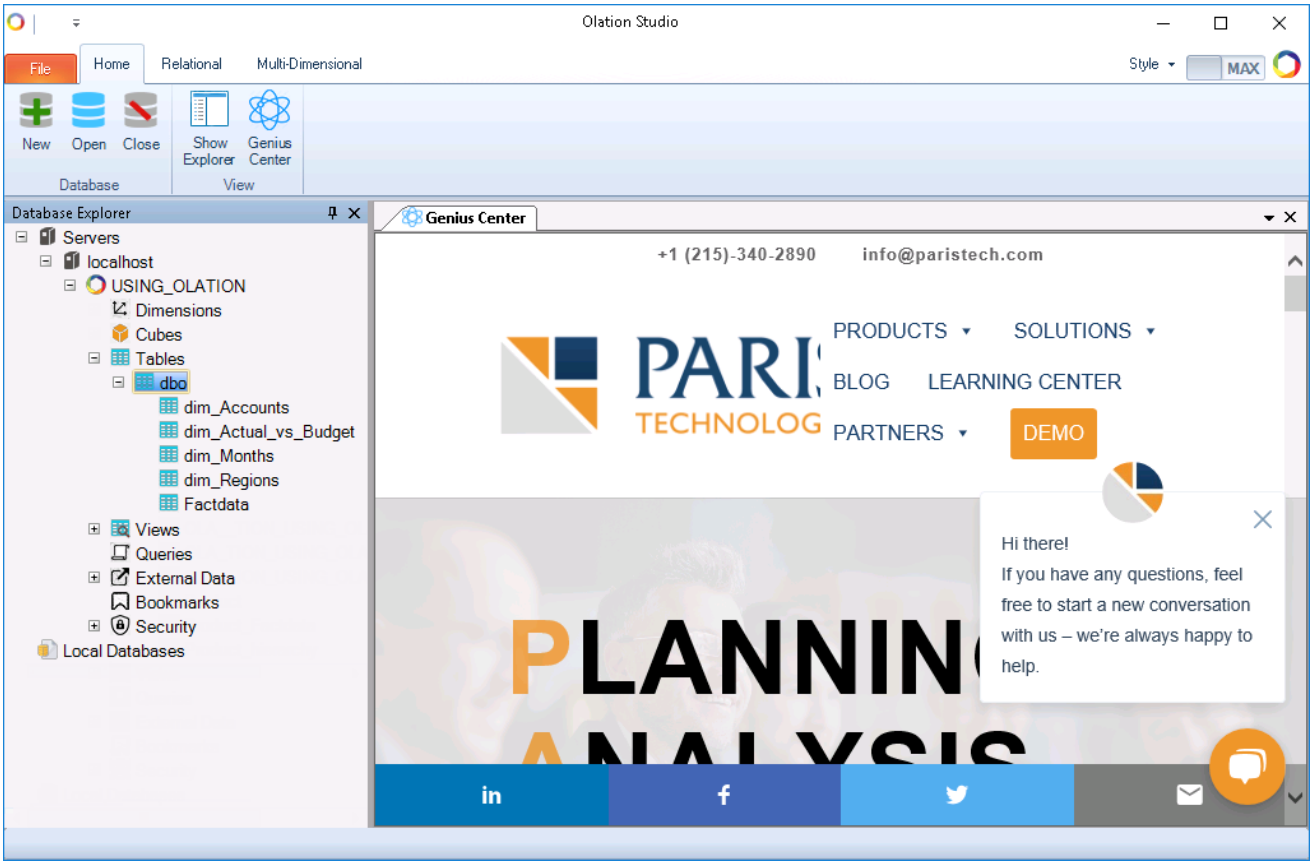
   Note: If you wish to connect to a different Olation Server, you will be required to register that Olation Server.
   To register a server: go to Database Explorer→right-click on Servers→ select Register Server.
3. Next, right-click on the preferred Olation Server (**localhost**) and select **New Database** from the options.



This displays the New Olation Database dialog box.



4. Select a **Database Type** from the drop-down list, i.e., **Olation for MS SQL Server®**. (as shown above).
5. Specify a SQL Server—in this case, **(local)** for **localhost** is being used.
6. Enable the **Connect to SQL Server using NT Authentication** checkbox option. Alternatively, you can disable the option and enter instead a <**SQL Admin User**> and <**Password**>.
7. Click on the **Database Name** drop-down and select the preferred source SQL database for the Olation model you are about to create. Following this example, select **USING_OLATION**.
8. In the **Display Name** field, type the <**name of new Olation Database**>.
   Here is where you specify the preferred name for your Olation database. By default, this field will take the name of the selected Database. For our demo keep the default name, i.e., **USING_OLATION**.
9. There are 3 additional settings (Secure Database; Live from Relational Source; Save database to local file) that you can configure for your Olation database. For now, leave the default settings. [NOTE: additional Olation features and capabilities not covered in this Quick Start manual can be found in the Olation User Manual or the product Help file.]
10. Click **OK**.
    This will create and initialize an Olation database. The new Olation database will appear in the Olation's Database Explorer, as in the following image.



Note that the Tables shown are the same as those that can be seen in SQL Server (*dim_Accounts*, *dim_Actual_vs_Budget*, *dim_Months*, *dim_Regions* and *Factdata*). It is worth noting that Views and Queries from SQL Server, and External Data, can be reached as well.

4. Create the SALES Cube

Now, with the database is created, we can begin building Dimensions, Dimension components, and the Cube itself, all built from these relational tables.

Important: Please note that specific images and figures may be different from the data set you are working on. This exercise is meant to serve as a guide on the steps to be followed to help you understand how to use OLATION but still allow you to work with your own data set.

For this section, the objective is to create a SALES Cube that will be composed of the Dimensions *Account*, *Version*, *Month* and *Region*. We will also write a simple cube formula to handle basic calculations across specific intersections within the Cube. Following that, we will create a view out of this Olation Cube and

demonstrate the writeback capability between a front-end client (e.g., PowerExcel or PowerOLAP) and the source relational database (in this example the MS SQL Database).

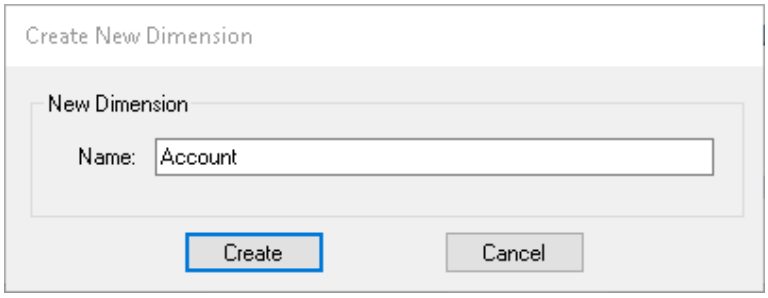
4.1. Create Dimensions from Relational Tables

Dimensions are lists of related terms used to organize data and are used to construct Cubes, the modeling structures used for planning, analytics and reporting on data.

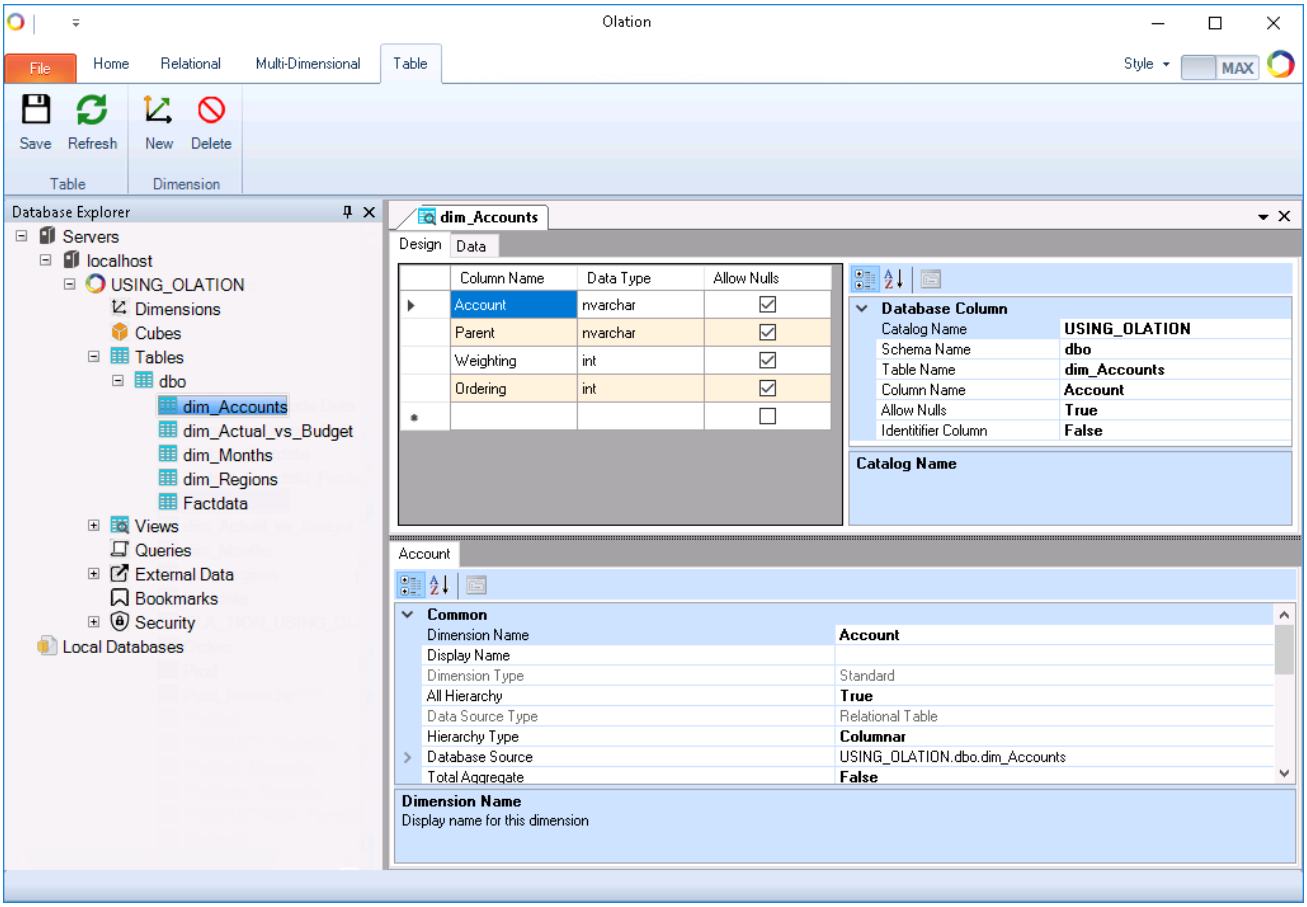
The first step is to build Dimensions. One of the significant features of OLATION® is that it permits a user to create a Dimension using common relational database components: tables, queries and views. From relational tables, we define the component Members and configure other attributes for Dimensions. Again, as mentioned earlier, we will be creating four Dimensions from the existing SQL tables: **Account**, **Version**, **Month** and **Region**.

4.1.1. Create the Account Dimension

1. Go to the Database Explorer pane (left section of the Olation window), expand on **Tables** and then expand on **dbo**. Select the table '**dim_Accounts**'; right-click and select **New Dimension**.
2. In the **Create a New Dimension** dialog box, type a <**name for your Dimensio**n>. You can give the Dimension any name you want. For this exercise, keep the name **Account** for the Dimension.



3. Click **Create**.
   This opens the Table window on the right. This also opens the **Table Tab** along the OLATION ribbon.



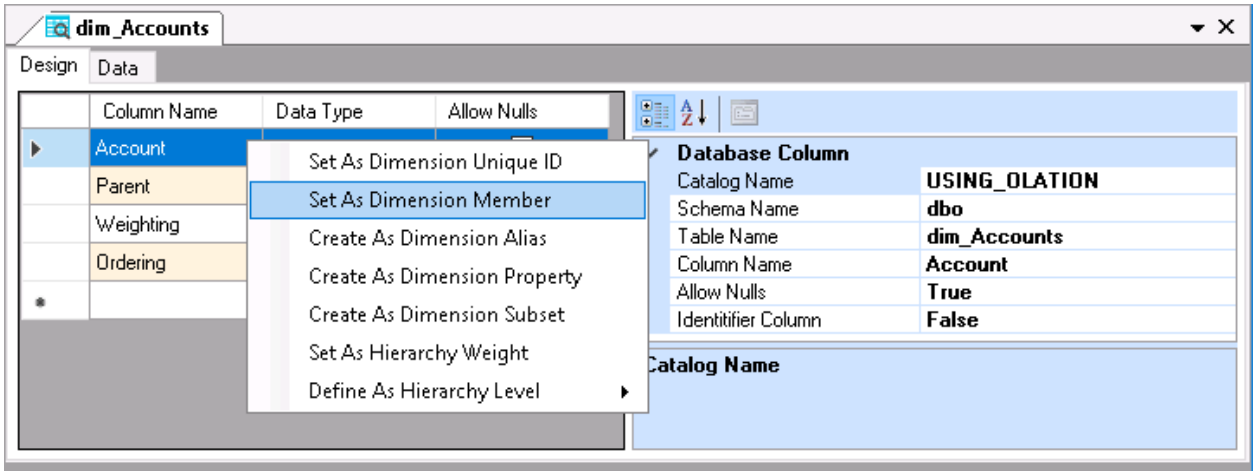
The Table Tab (per above image) includes definitions of the SLQ table being accessed—top left (Design/Data tabs), top right (Database Column), and properties of the Account dimension (bottom half, Account tab). Using these in concert will enable you to build the Dimension according to desired specifications.

About Dimension Members

Dimensions are composed of Detail and Aggregate Member types. Detail Members "add up" to Aggregate Members. Aggregate Members are "parent" Members when they aggregate a group of "child" Members. For example, you may create a Dimension named *Month*, whose *First Quarter* member can be an Aggregate Member of *January*, *February*, and *March*, each of them a Detail Member. Note that child Members can also themselves be an Aggregate.

We will next add Dimension Members to the *Account* dimension, and create a logical Aggregate.

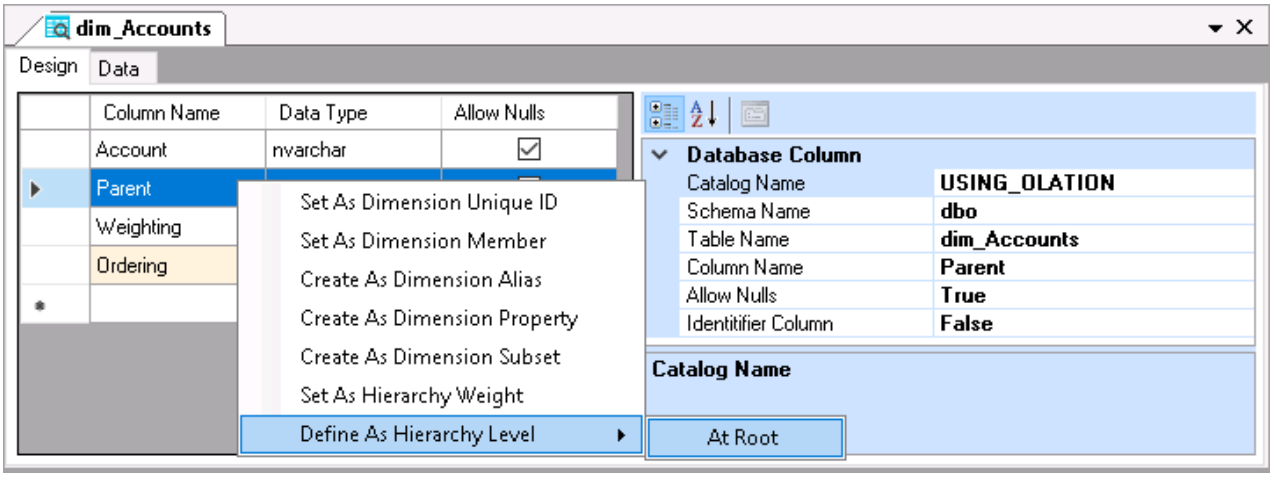
1. In the **Design Tab**, select the top row, where '**Account**' appears under Column Name.
2. Right-click and then select **Set as Dimension Member**, as shown below.



Define Hierarchy

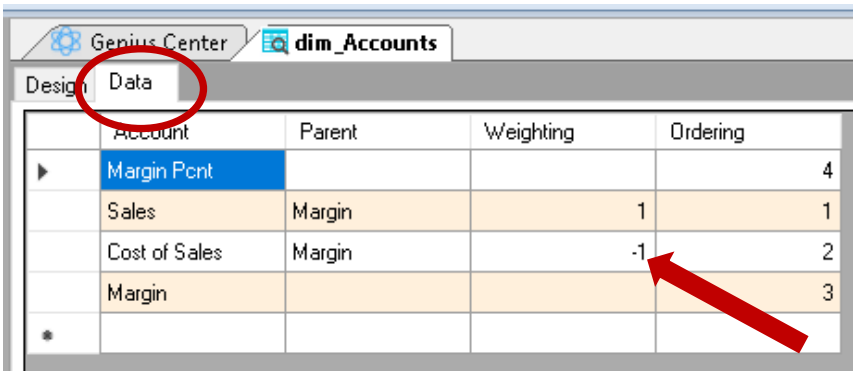
Parent-child relationships exist in the underlying *dim_Accounts* table, ensuring that an accurate calculation occurs when *Sales* and *Cost of Sales* "hierarchize" as *Margin*. To create this hierarchy:

1. In the **Design Tab**, select the row corresponding with the Column Name '**Parent**'.
2. Right-click and then select **Define As Hierarchy Level**, then select **At Root**. (At Root means that this will be the highest-level Aggregate Member.)



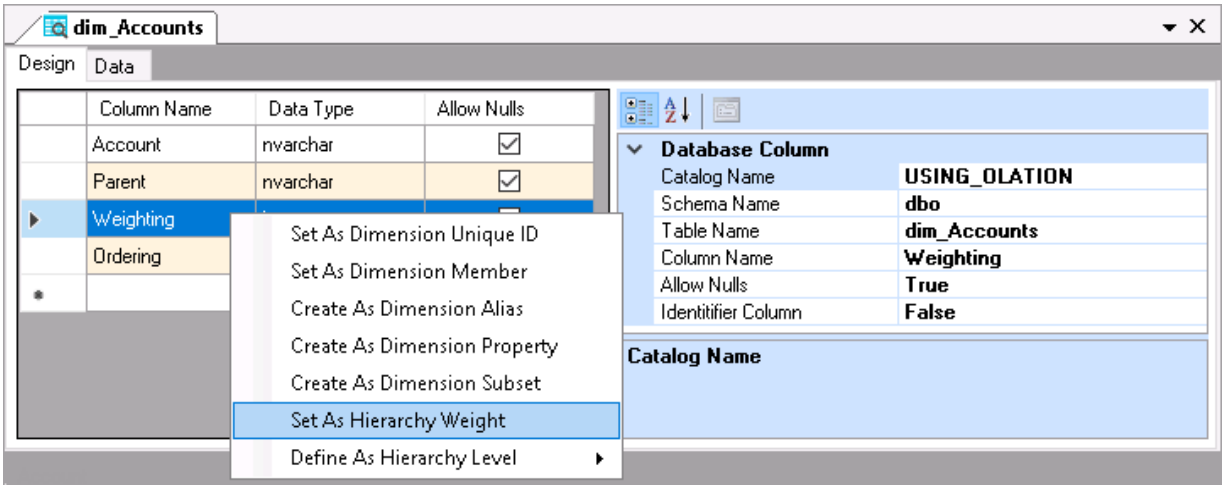
Define Hierarchy Weight

1. First, go to the **Data Tab** (circled in the following image): note that in the Cost of Sales has a "**-1**" Weighting (see arrow).



To ensure that *Cost of Sales* (as a negative value) and *Sales* (as a positive value) calculate correctly:

2. Back in the **Design Tab**, select the row with Column Name '**Weighting**'.
3. Right-click and then select **Set as Hierarchy Weight** (see next image).



Note the bottom half of the "dim_Accounts" pane: here are selections for various attributes or properties that you can choose to further define the Dimension.

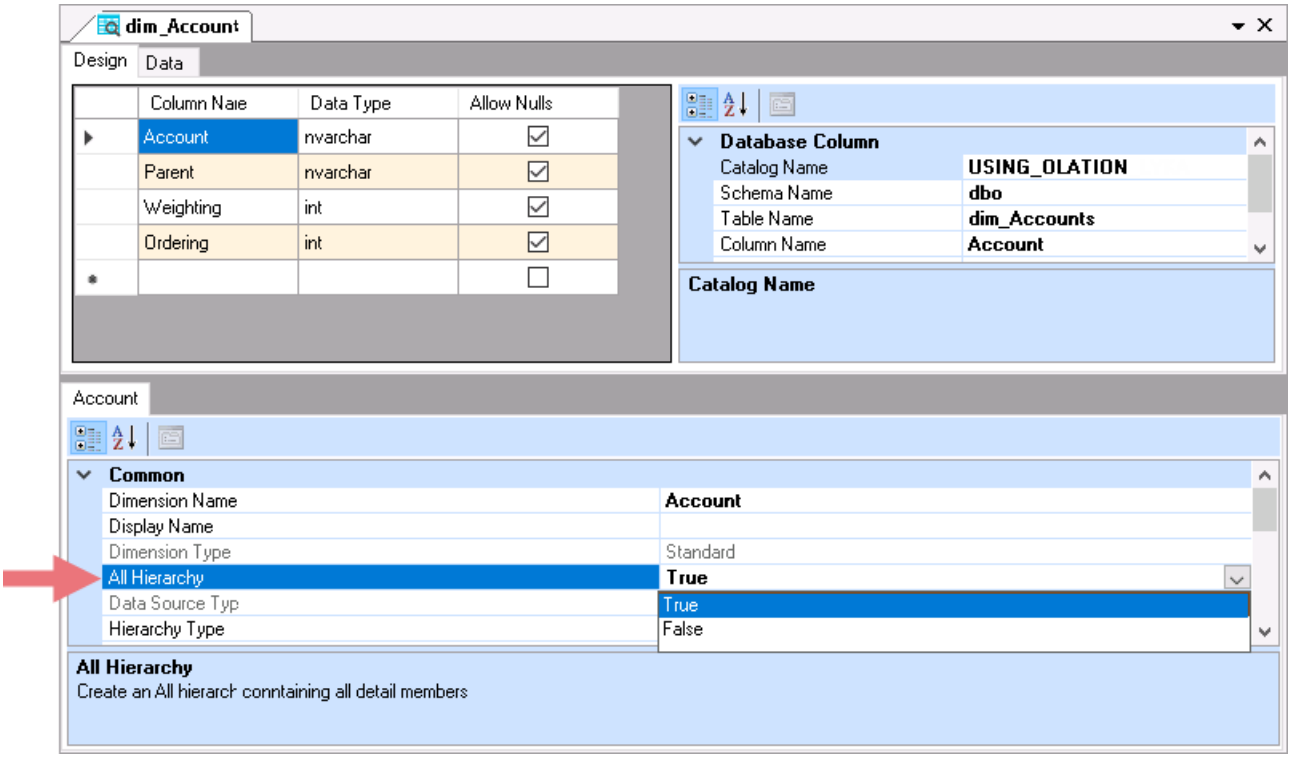
The first consideration we will give to this Dimension is defining its hierarchy (or hierarchies).

The next 2 selections will explain further about Hierarchies and Aggregates—specifically as they relate to the Dimension property selections that can be made in the bottom half of the right-hand pane (in this case, the *Account* dimension):

Enable/Disable the All Hierarchy option

1. At the bottom half of the window of the Account Tab, scroll down to see the **All Hierarchy** option.
2. Click the drop-down button that appears on the far right and ensure that **False** is selected.
   Note: Alternatively, you can double-click on the **All Hierarchy** field to change selections from *True* to *False* and vice versa.

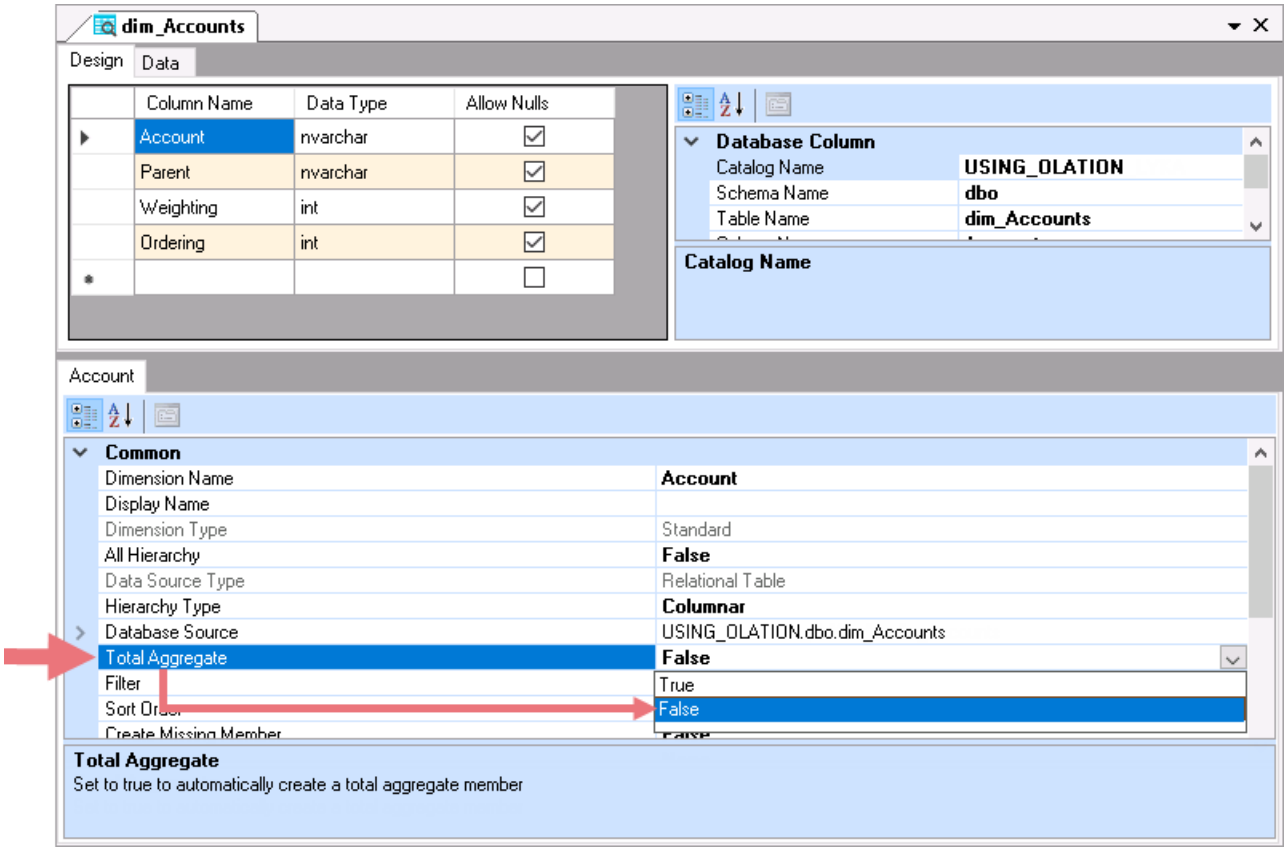
The 'All Hierarchy' option: A selection of **True** creates a Member at the root called "All", adding every Detail Member under it. (For this example, there is no logical need for an *All* Dimension member.)



Enable/Disable the Total Aggregate option

1. At the bottom half of the window scroll down to see the **Total Aggregate** option.
2. Click the drop-down button that appears on the far right and select **False**.
   Note: Alternatively, you can double-click on the **Total Aggregate** field to the change selection from *True* to *False*.

Note: The 'Total Aggregate' option: A selection of **True** creates a Total Member(s) that sits at the top of each defined Hierarchy. (For this example, there is no logical need for such a calculation.)



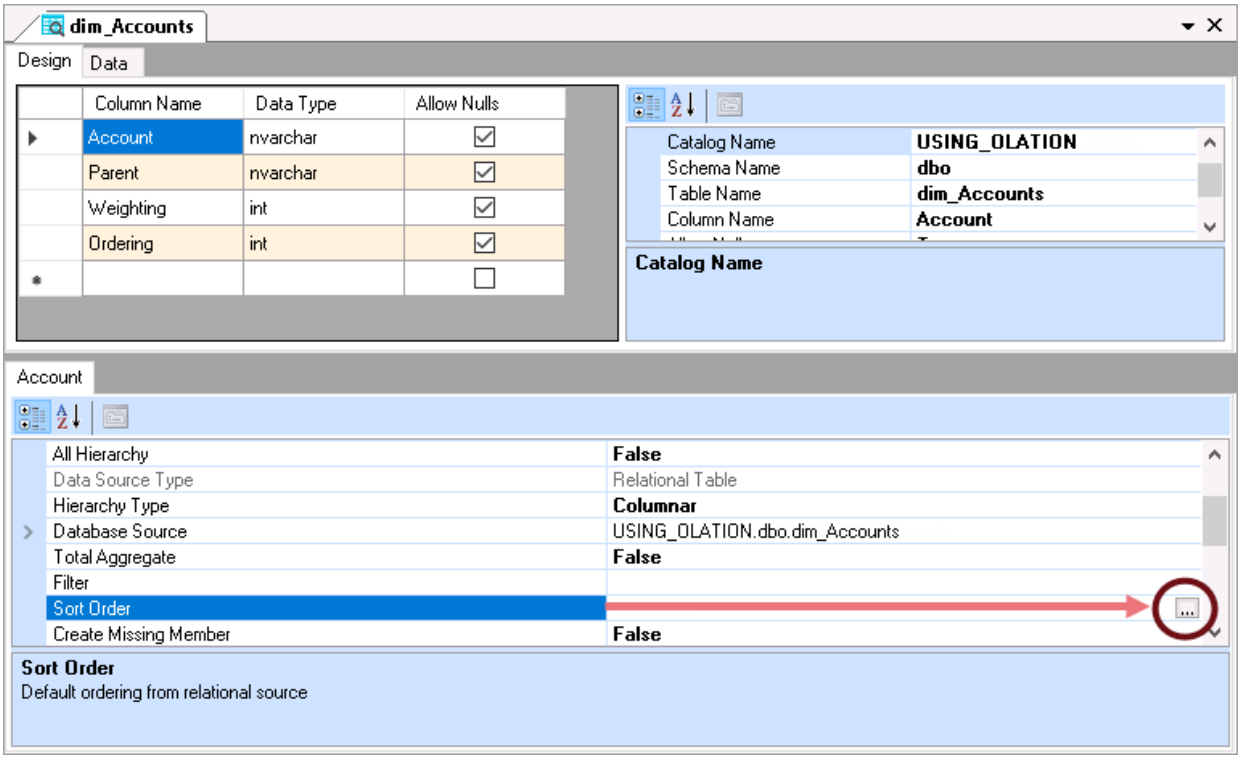
Defining a Sort Order

You can define the order of the Members as they appear listed within a Dimension—for example, you may want them arranged in an ascending or descending order; or you may also arrange the list based on a sequence defined within a reference column.

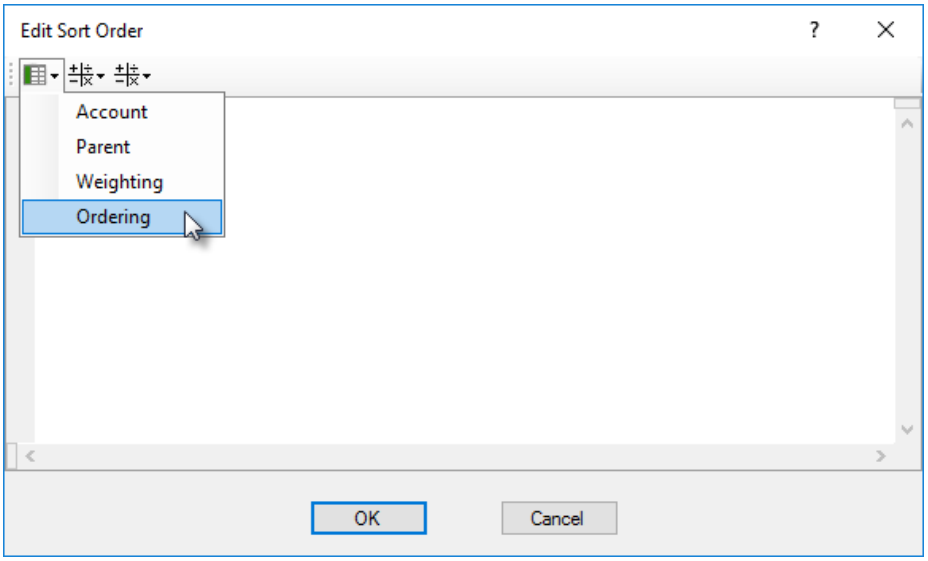
To define the sort order:

1. At the bottom half of the Table window, select the **Sort Order** option. An ellipsis button will appear on the far right (see next image).

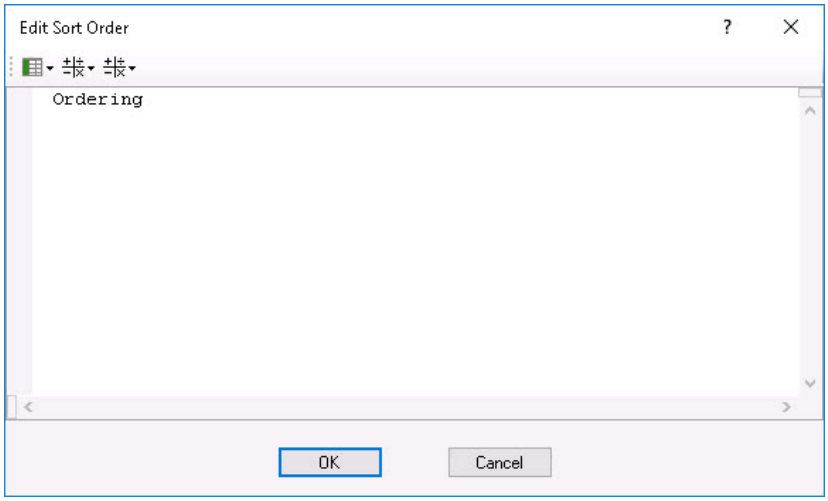
2. Click on that **ellipsis button** to bring up the **Edit Sort Order** dialog box (next page, second image).



3. In the Edit Sort Order dialog, click the **Measure Columns** icon and select **Ordering**.



4. Click **OK** to close the dialog (which appears below).

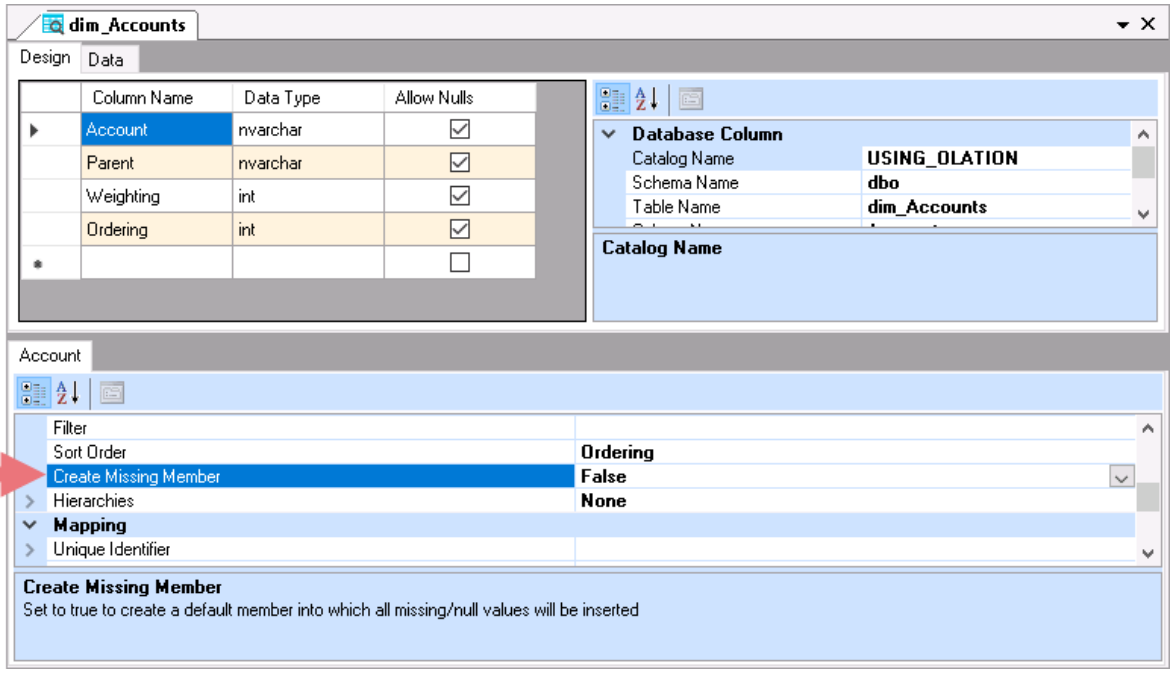


Enable/Disable Create Missing Member

The Create Missing Member option allows you to create a default Member in which values are placed for "missing" Dimension members (i.e., transaction records that have no Member indicated in transactional tables). For this Dimension, turn off the Create Missing Member option:

1. Go to the bottom half of the screen and select the **Create Missing Member** option.
2. Click the drop-down button that appears on the far right and select **False**.
   Note: Alternatively, you can double-click on the **Create Missing Member** field to change the selection from *True* to *False*.

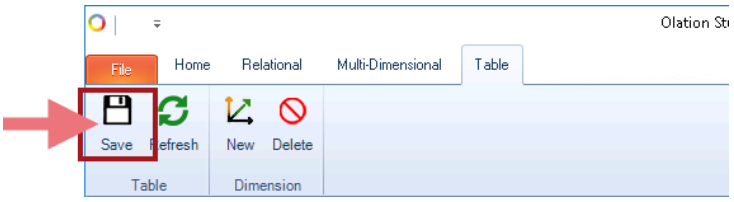
   Note: If you enable this option (set to 'True')–see the additional row (**Missing Member**) displayed directly below –you can enter in the textbox the preferred name for those missing Member[s].



Save the Dimension

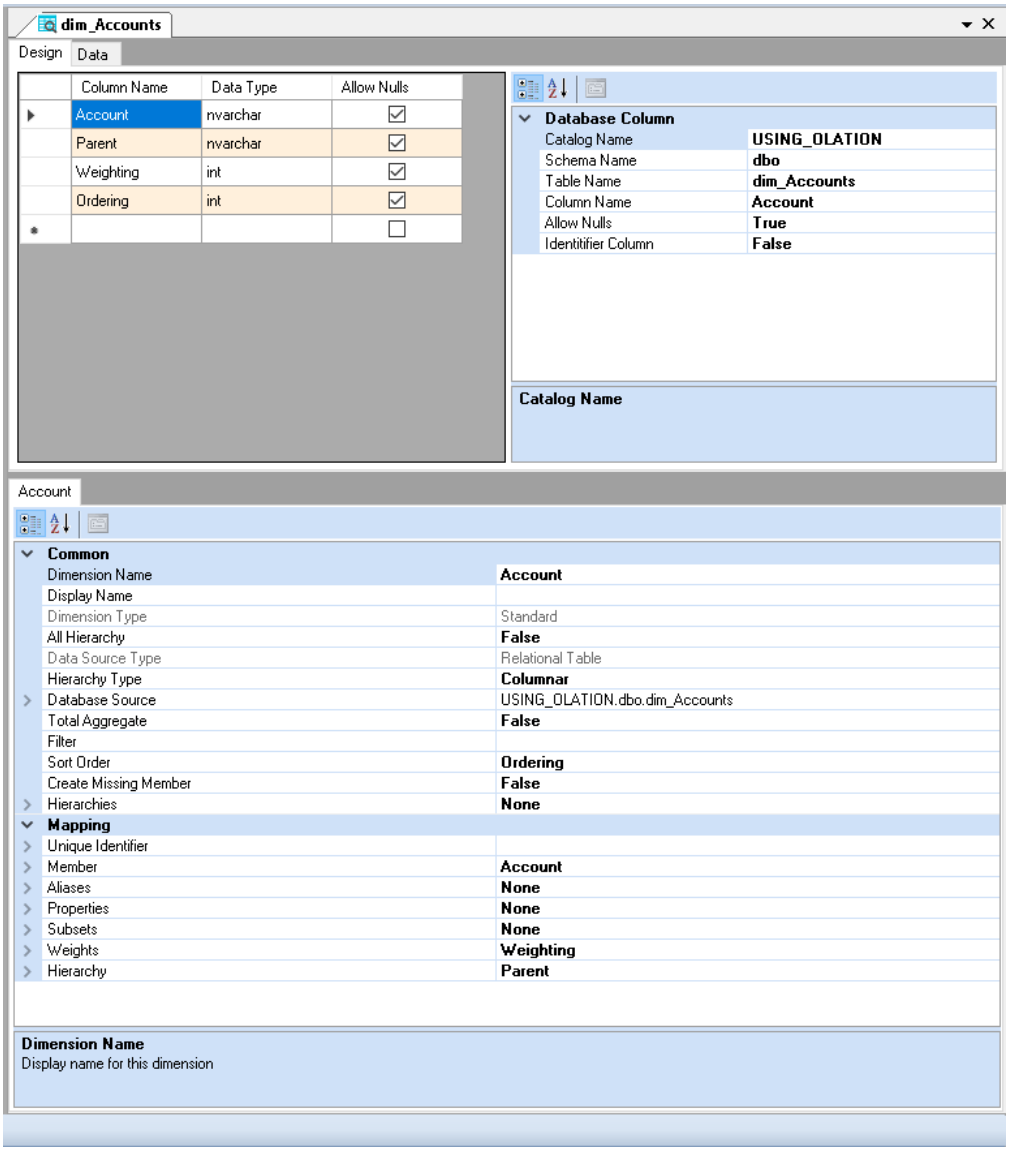
To save the Dimension:

1. Go to the Olation ribbon: in the **Table tab**, click on the **Save** icon.



NOTE, on the left, in the Database Explorer window, when you expand under Dimensions, the newly created **Account** dimension now exists.

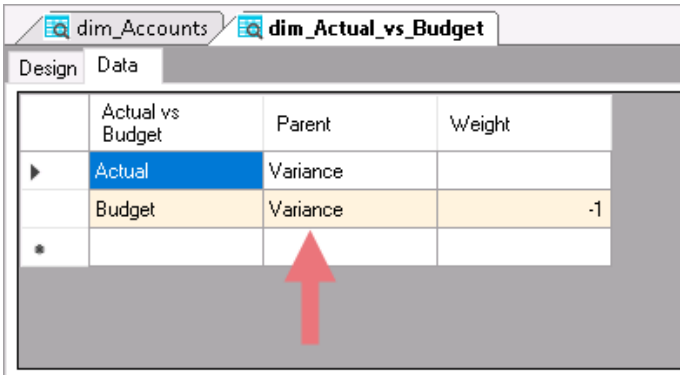
The “dim_Accounts” pane, when completed, will look as follows.



The next steps will concern creating the Dimensions *Version*, *Month* and *Region*.

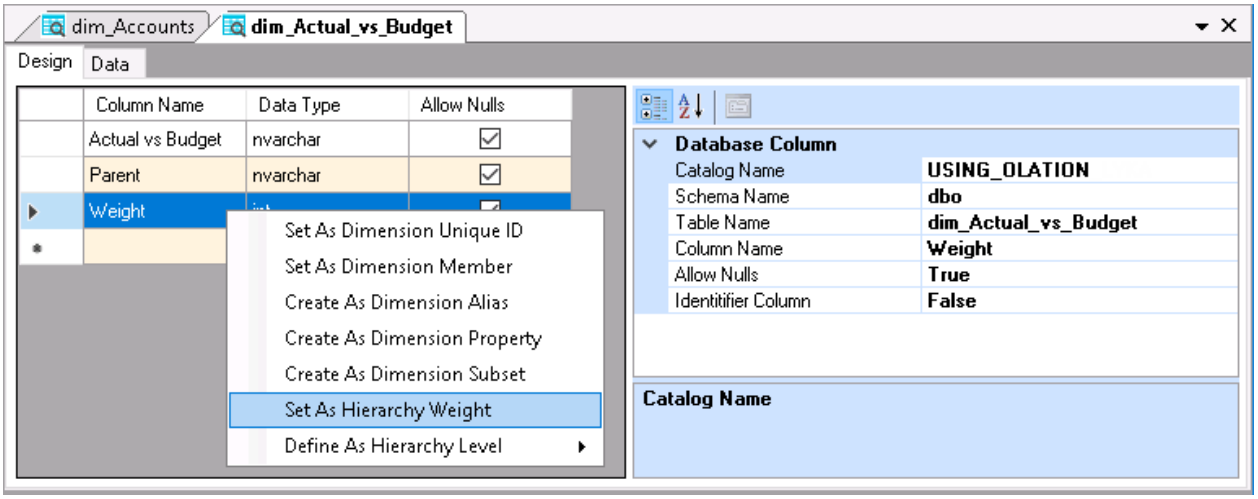
4.1.2. Create the Version Dimension

1. In the Database Explorer pane, select **dim_Actual_vs_Budget**; right-click on it and select **New Dimension**.
2. In the **Create New Dimension** dialog box, enter **Version** in the textbox and click **Create**.
3. In the **Design Tab**, select the row corresponding to the **Actual vs Budget** column name; right-click on it and select **Set As Dimension Member**.
4. Define a hierarchy for the *Version* dimension:
   For this example, we want to create a hierarchy whereby *Variance* has *Sales* and *Cost of Sales* as child members. To do this, go to the **Design Tab** and right-click on the row beginning **Parent**; select **Define As Hierarchy Level**, then select **At Root**.



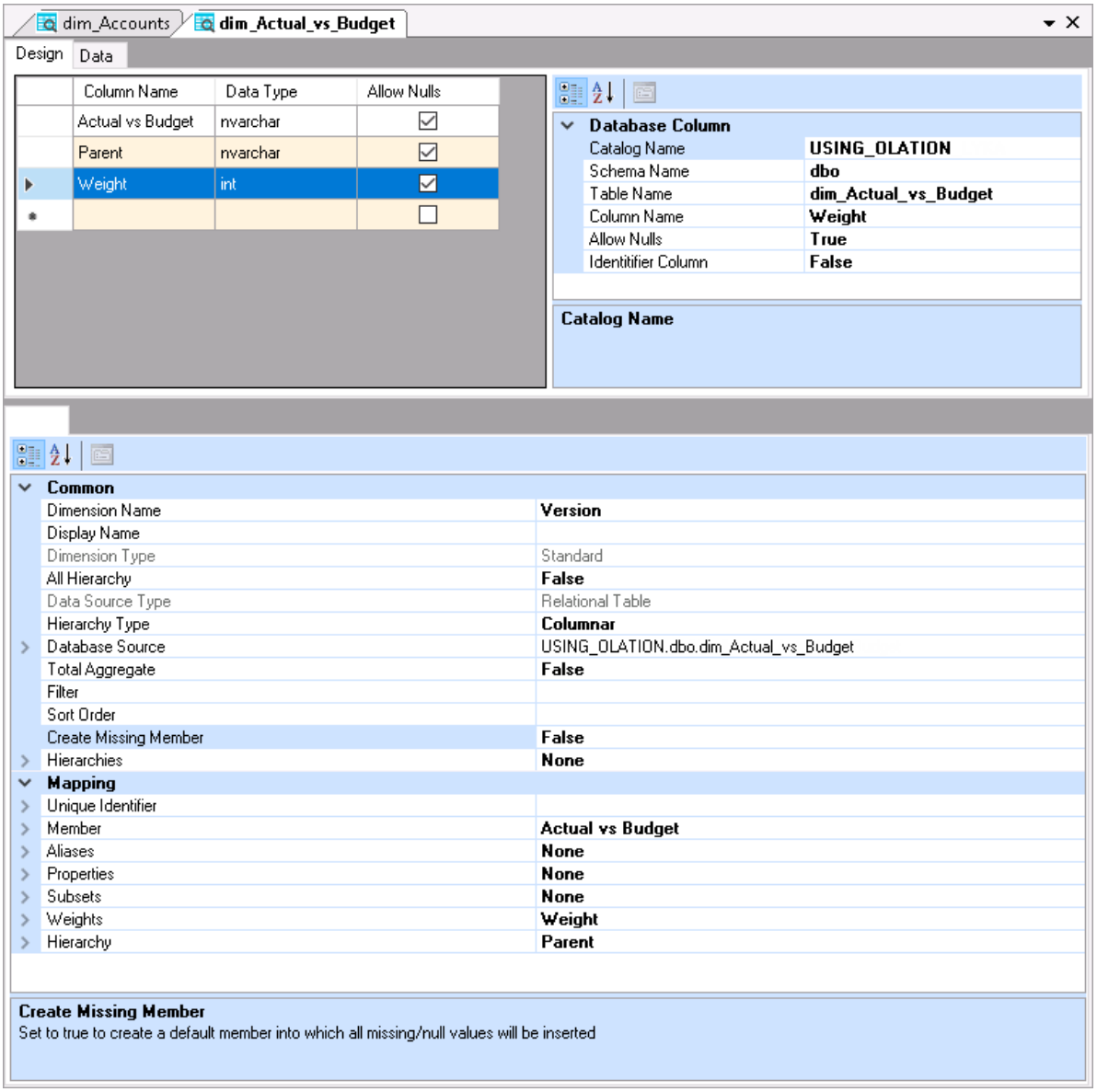
Note: The screenshot above shows, in the Data Tab, the *Actual vs Budget* column. This column contains *Actual* and *Budget* as Members. The 'Parent' column contains *Variance*. By setting this column as the root (which you did above), *Variance* will calculate as *Actual* minus *Budget*.

5. Set the hierarchy weight for the *Version* dimension: back in the **Design Tab**, select the row corresponding to the **Weight** column name; right-click on it and select **Set As Hierarchy Weight** (see next image).



6. Disable the 'All Hierarchy' option: locate the **All Hierarchy** setting and select **False**.
7. Disable the 'Total Aggregate': locate the **Total Aggregate** setting and select **False**.

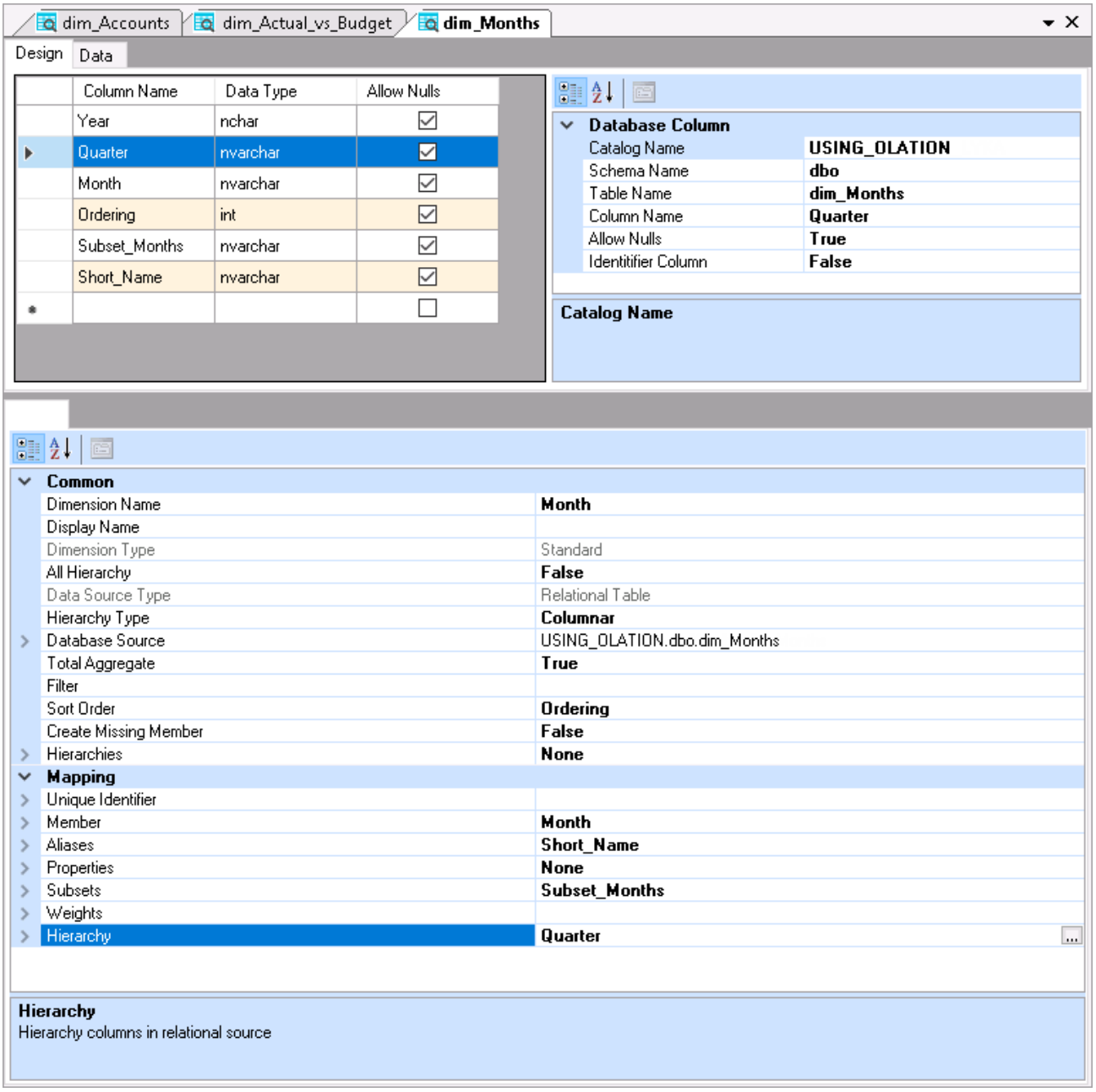
8. Disable the 'Create Missing Member' option: locate the **Create Missing Member** setting and select **False**.
9. Save the **Version** dimension: again, click the **Save Table** icon.
   The **Version** dimension now appears under **Dimensions** in the Database Explorer.



4.1.3. Create the Month Dimension

1. In the Database Explorer pane, select **dim_Months**; right-click on it and select **New Dimension**.
2. In the **Create New Dimension** dialog box, enter **Month** in the textbox and click **Create**.

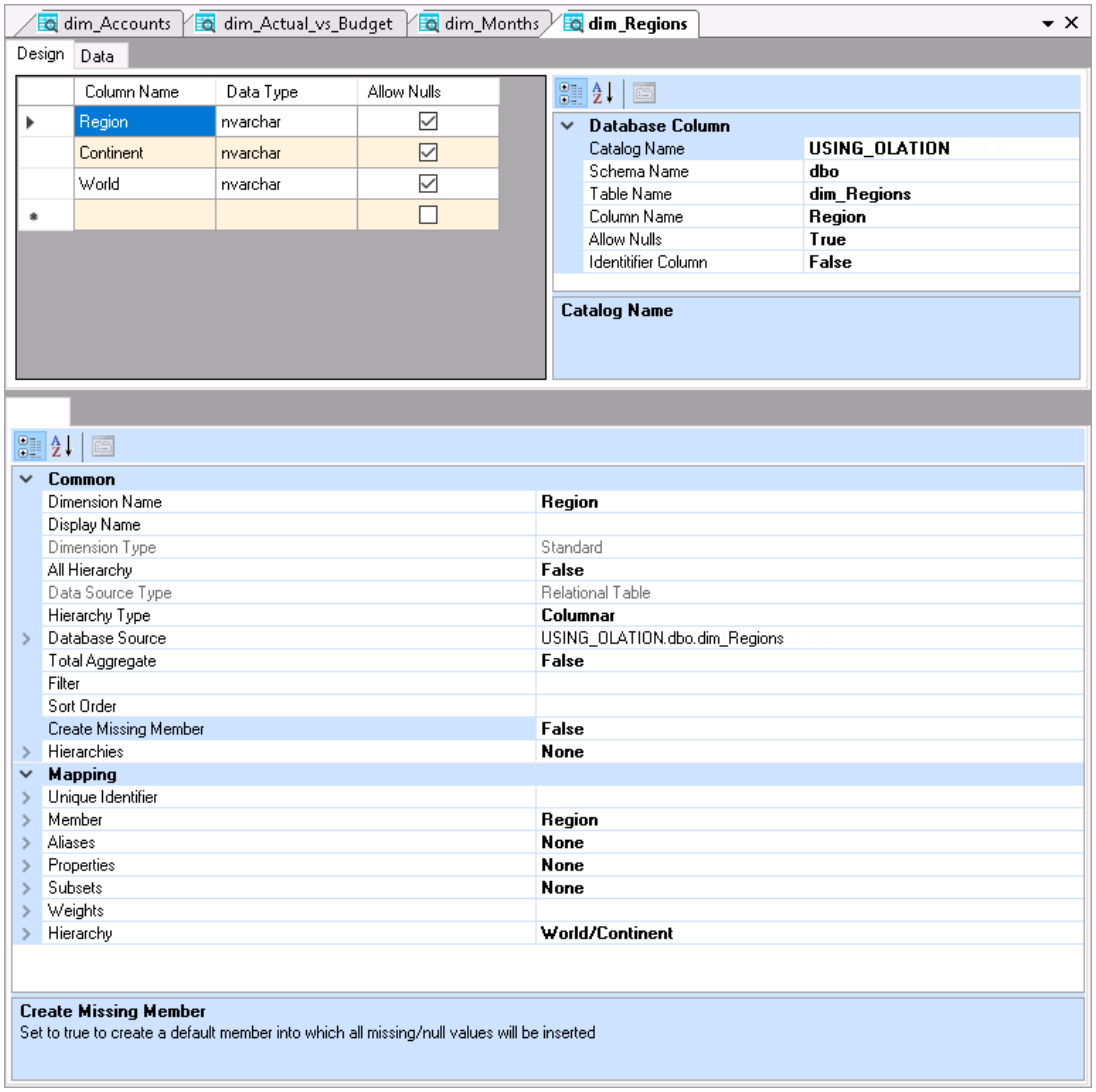
3. In the **Design Tab**, select the row corresponding to the **Month** column name; right-click on it and select **Set As Dimension Member**.
4. Begin defining the hierarchy for the Month dimension: right-click on the row corresponding to the **Quarter** column name; select **Define As Hierarchy Level**, then select **At Root**.
5. Define an Alias for the Month dimension: for this example, we want to set the *Short_Name* column as the alias of the Month dimension, so that it can display the month names in their shortened or abbreviated form.
   To do this, right-click on the row corresponding to the **Short_Name** column name; select **Create As Dimension Alias**.
6. Define a Subset for the Month dimension: for this example, we want to define a Subset, which is a custom list of Members.
   To do this, right-click on the row corresponding to the **Subset_Months** column name; select **Create As Dimension Subset**.
7. Disable the 'All Hierarchy' option: on the **Property Grid** locate the **All Hierarchy** option and select **False**. (There will be no need to add all months, as they will be aggregated in the following selection.)
8. We want to enable the 'Total Aggregate' option by creating a Total of all 'Quarters'.
   To do this, go to the **Property Grid** on the right and locate the **Total Aggregate** setting, then select **True**.
9. Disable the 'Create Missing Member' option:
   To do this, go to the **Property Grid** on the right and locate the **Create Missing Member** setting then select **False**.
10. Here we will do something different: we want to preserve the "Ordering" that has been set up in the Table, so that *January* is followed by *February*, *March*, etc. when the Dimension is built.
    - Go to the **Property Grid** at the bottom then locate and select the **Sort Order** option.
    - Click on the **ellipsis button** corresponding to it on the far right to bring up the **Edit Sort Order** dialog box.
    - Click the **Measure Columns** icon and select **Ordering** from the options.
    - Click **OK**.
11. Save the **Month** dimension—it will also appear under **Dimensions** in the Database Explorer pane. Under the dim_Months tab, the Dimension and its attributes will appear as in the following image.



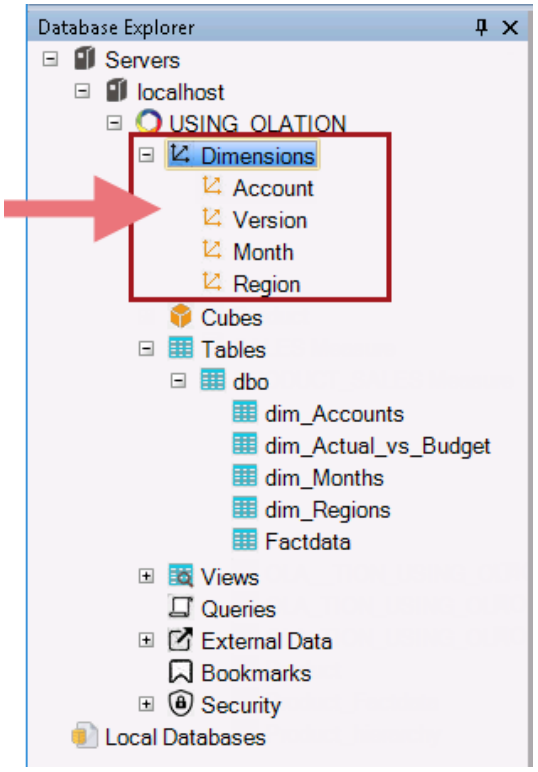
4.1.4. **Create the Region Dimension**

1. In the Database Explorer pane, select **dim_Regions**; right-click on it and select **New Dimension**.
2. In the **Create New Dimension** dialog box, enter **Region** in the textbox and click **Create**.

3. In the **Design** tab, select the row corresponding to the **Region** column name; right-click on it and select **Set As Dimension Member**.
4. Define the hierarchy for the Region dimension, with *World* as the highest-level parent; with the continents as direct children, and; with the regions as the leaf level Members:
    - Right-click on the row corresponding to the **World** column name, select **Define as Hierarchy Level**, then **At Root**.
    - Right-click on the row corresponding to the **Continent** column name; select **Define As Hierarchy Level**, and choose **Sub-Level of World**.
5. Disable the 'All Hierarchy' option, i.e., set to **False**.
6. Disable the 'Total Aggregate' option, i.e., set to **False**.
7. Disable the 'Create Missing Member' option, i.e., set to **False**.
8. Save the **Region** dimension—it will also appear under **Dimensions** in the Database Explorer pane. Under the dim_Regions tab, the Dimension and its attributes will appear as in the following image.



This completes the creation of the four Dimensions: *Account*, *Version*, *Month* and *Region*. At this point, you see all four Dimensions listed in the Database Explorer.



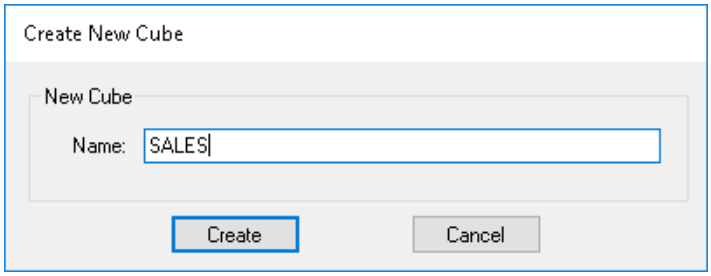
4.2. Create a Cube from a Relational Table - SALES Cube

We proceed to the all-important step of creating a Cube. For this exercise, we will create a Cube called SALES, composed of the Dimensions we created previously: *Account*, *Version*, *Month* and *Region*.

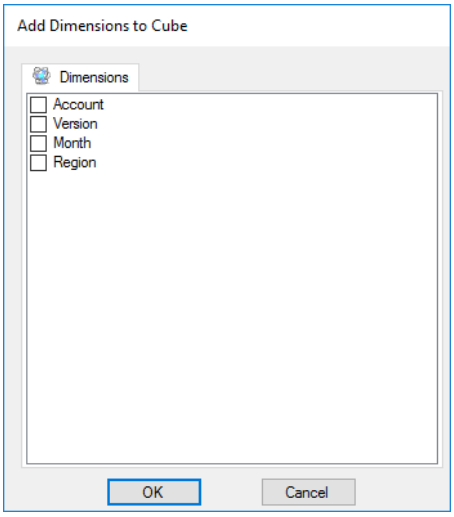
For this exercise, we will be creating the SALES Cube from the SQL table called *dbo.Factdata*, which should contain all the transactional data.

To create the SALES Cube:

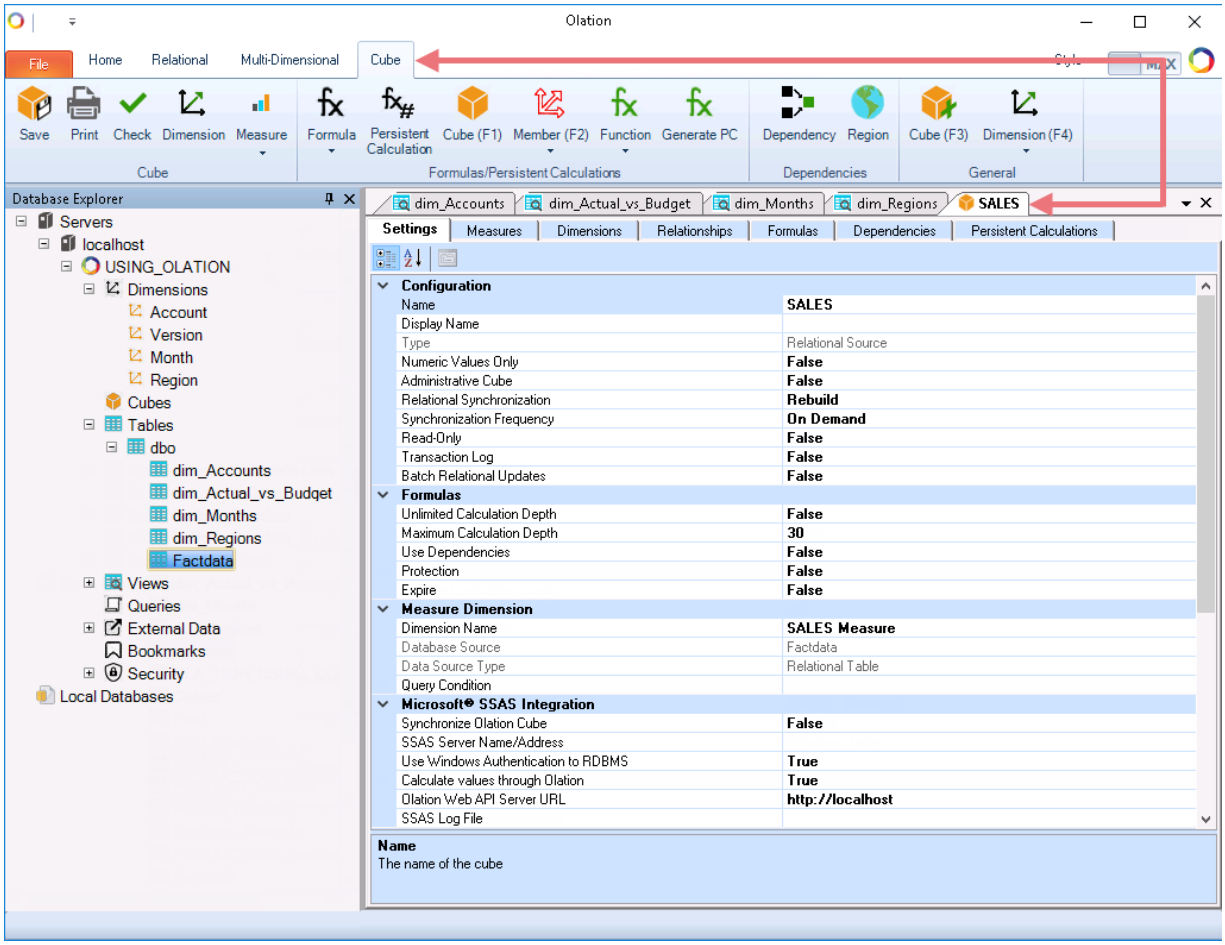
1. Among the **Tables** in Database Explorer, select **Factdata**; right-click on it and select **New Cube**. The **Create New Cube** dialog box appears.
2. In the dialog box, type a <**name for your cube**>. For this exercise, type **SALES** as the cube name. Note that you can give the Cube any name you want.



3. Click **Create**. The **Add Dimensions to Cube** dialog box appears.



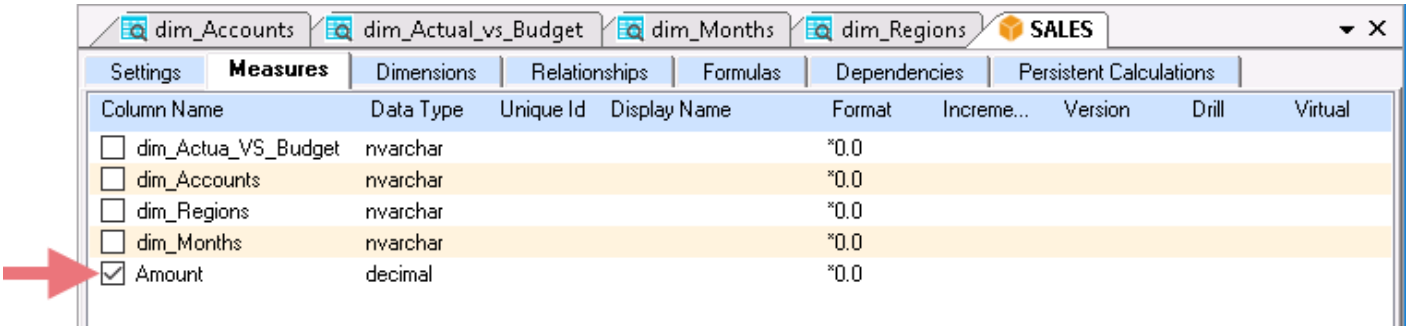
4. Select the Dimensions to be included into the Cube by checking the corresponding checkboxes. In this example select all four Dimensions: **Account**, **Version**, **Month**, and **Region**.
5. Click **OK**. This opens the **Cube Definition Window** on the right, under the SALES Tab. Note also that a **Cube Tab** also appears along the Olation ribbon.



4.2.1. Assign the Measure Member

The Measure Member will provide the numbers at the intersections of the Dimensions within the Cube—the “factdata” that the model will provide for reporting, analytics and planning purposes.

1. In the **Sales Cube Definition Window**, go to the **Measures Tab**.
2. Select from the list which item(s) you want to designate as Measures enabling the checkbox(es). For this exercise check the **Amount** checkbox.

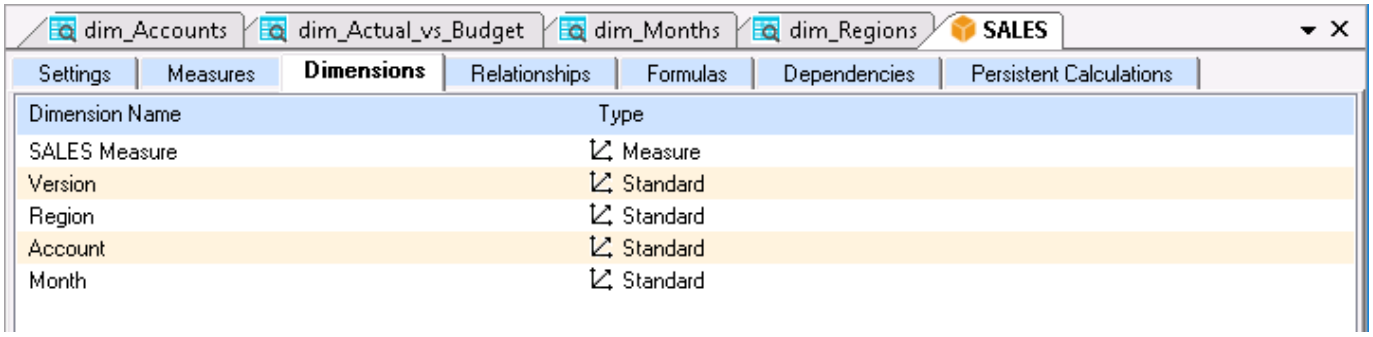


4.2.2. Arrange the Dimension Order

You can arrange the order of the Dimensions in Olation so that when you report in the preferred client/viewer, it will by default appear in the order you specified. In the Dimensions Tab of the Cube Definition Window, the bottom-most Dimension goes to the Rows, the second to the last Dimension goes to the Columns, while all other Dimensions at the top go to the Filter section.

To define the order of the Dimensions:

1. In the **Sales Cube Definition Window**, go to the **Dimensions Tab**.
2. Drag and drop the Dimensions in the following order as shown in the image below:



4.2.3. Define Relationships

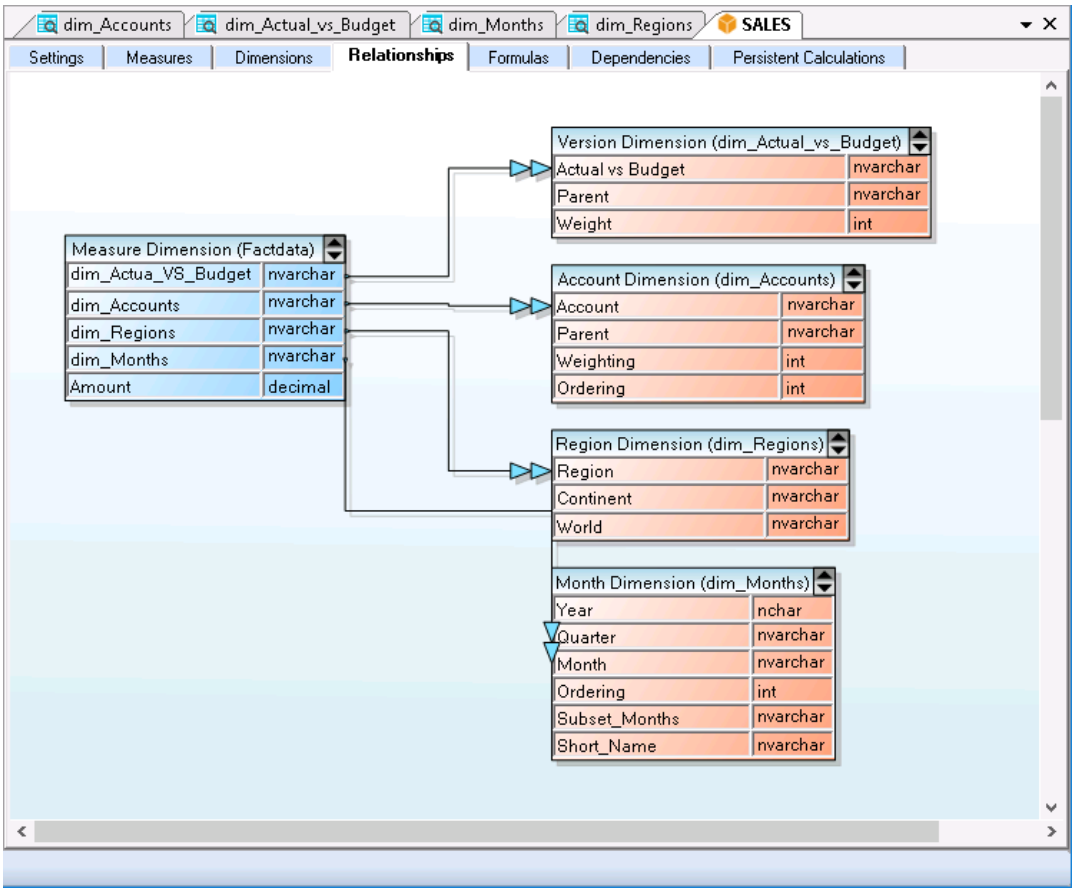
We will next create relationships between the relational tables and the Measure table. The objective here is to create a link between the data or columns of the Measure table with the corresponding columns of other tables, views or Dimensions.

1. Go to the **Relationships Tab**. This will display the Dimension tables and the Measure table.

   The four Dimension tables—*Version*, *Account*, *Region* and *Month*—appear. Following this example, you will need to link all four Dimension tables to the *Measure* table. To create a link, click on a column name from the Measure table on the left then drag and drop to the appropriate column name on the relational or Dimension table on the right. You will see an arrow that links one part of the Measure table to the corresponding place on the other relational table.

2. Click on **dim_Actual_Vs_Budget** from the ***Measure Dimension*** table on the left; drag and drop to **Actual vs Budget** in the ***Version Dimension*** table on the right.
3. Click on **dim_Accounts** from the ***Measure Dimension*** table on the left; drag and drop to **Account** in the ***Account Dimension*** table on the right.
4. Click on **dim_Regions** from the ***Measure Dimension*** table on the left; drag and drop to **Region** in the ***Region Dimension*** table on the right.
5. Click on **dim_Months** from the ***Measure Dimension*** table on the left; drag and drop to **Month** in the ***Month Dimension*** table on the right.

   The completed Relationships Tab will look as in the below image.



Notice that tables are color-coded in such a way that the Measure table is always Blue, other Dimension tables are Orange, and—though one was not created here—Date dimensions appear Green. Inserting another intermediate table will bring it up in this tab as another color.

4.2.4. Define Cube Setting: Disable READ-ONLY Setting

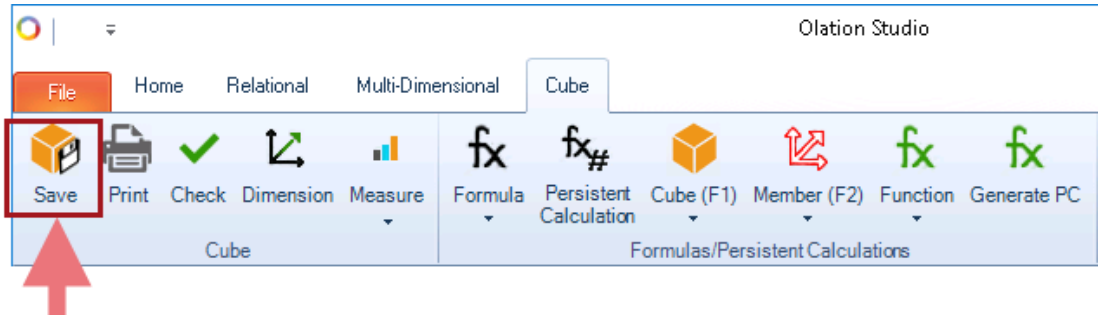
For this exercise, we want to disable the Read-Only setting to allow the SALES cube to have the write-back capability between a front-end and the Olation cube:

1. In the **Sales Cube Definition Window**, go to the **Settings (first) Tab**.
2. In the Configuration section, locate the **Read-Only** setting, click on the corresponding drop-down button and select **False**.

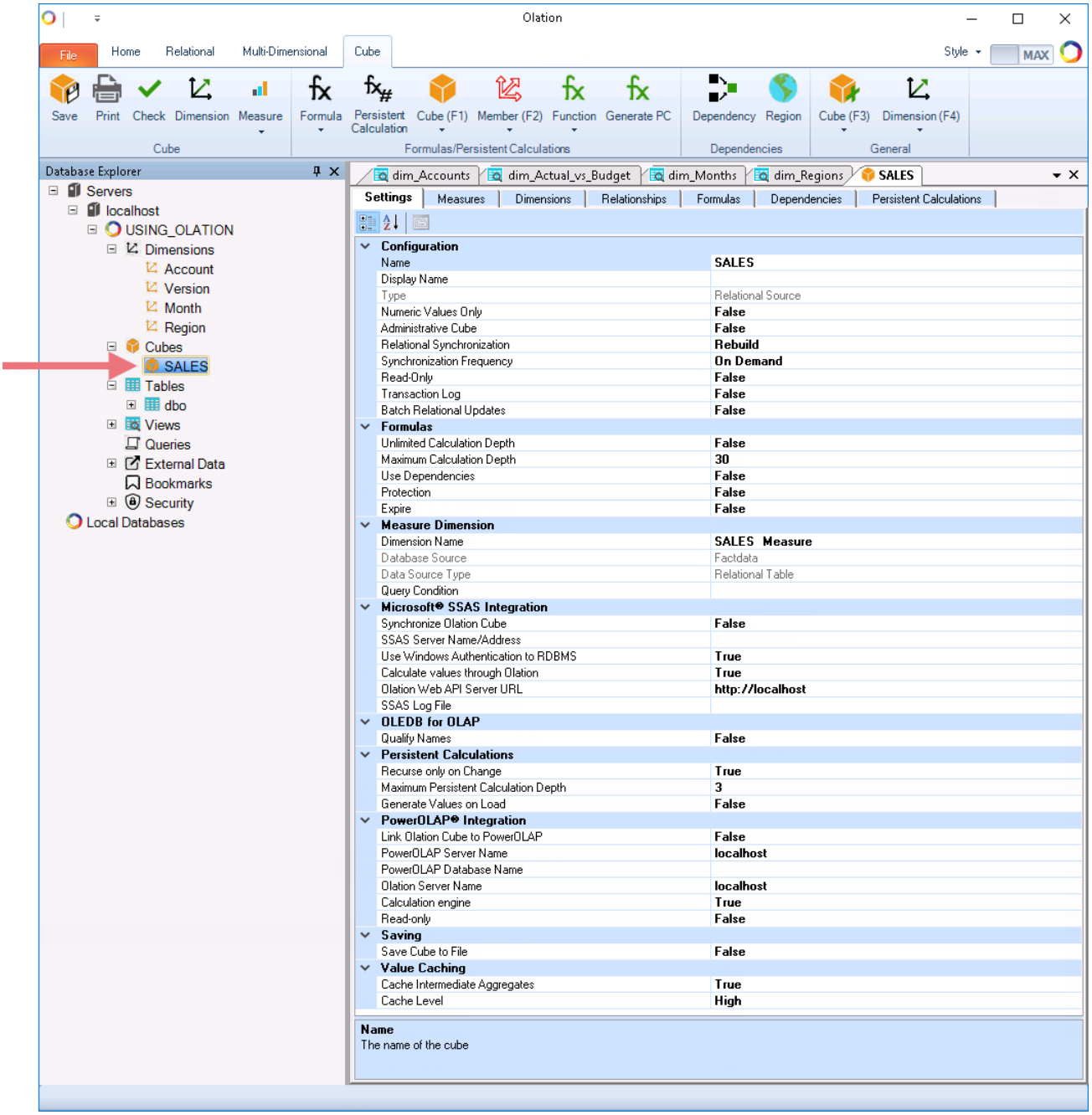
4.2.5. Save the Cube

To save the Cube:

1. Go to the **Cube Tab** on the Olation ribbon and click the **Save** icon.



Once saved, the Sales cube is listed under Cubes in the Database Explorer.



5. Viewing the OLATION Data

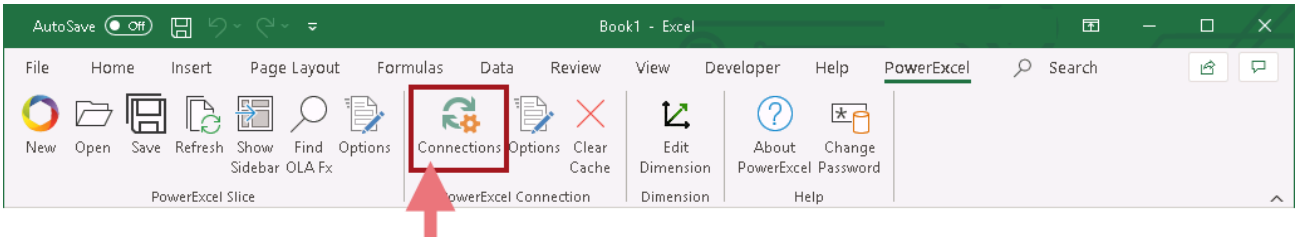
Olation® supports many clients as front ends or browsers of data, providing users a wide selection when working with an OLATION model.

5.1. Create the Dynamic Excel Front End –- PowerExcel Slice

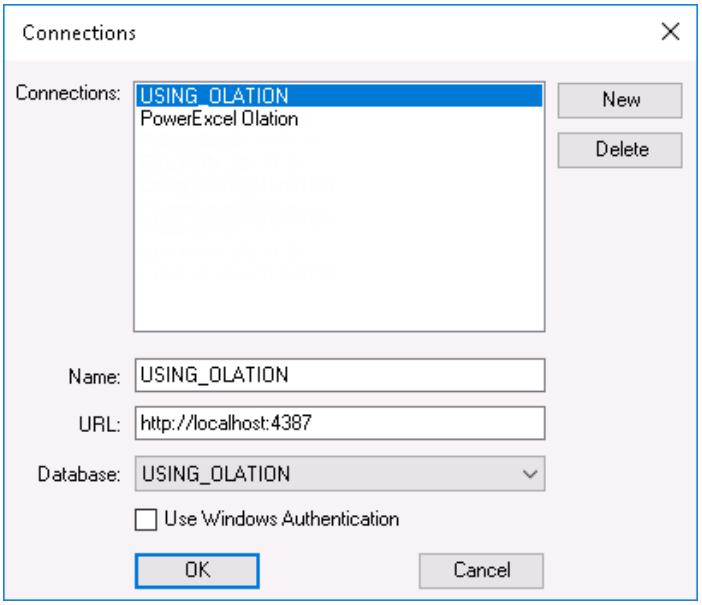
For this exercise, we will use PowerExcel as the front-end client to view and access the Olation Sales cube.

To create a PowerExcel Slice:

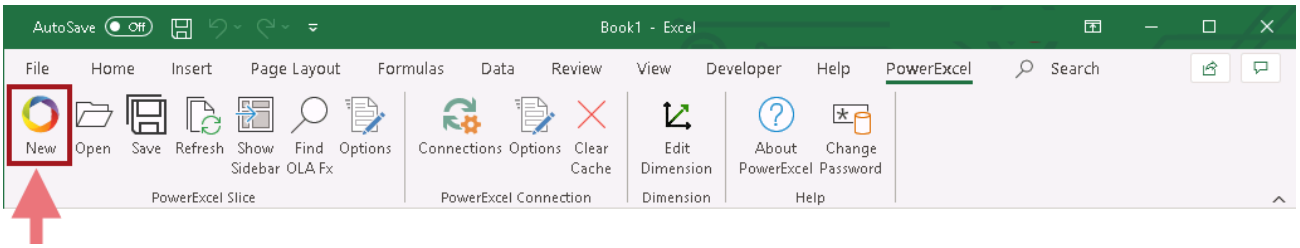
1. Launch the **Excel** application and go to the **PowerExcel Tab**.
2. First, create a '**PowerExcel connection**'.
   To do this:
   - In the PowerExcel Connections control group, click the **Connections** icon.



   - In the Connections dialog that appears, click **New**.
   - In the **Name** field, enter the <**name of the PowerExcel connection**>.
     For example, name this as **USING_OLATION PXL**.
   - In the URL field, enter the <**correct URL**>.
     Note: This URL will be the URL of the Olation Server where the source Olation database is currently running/opened. (Note that localhost and the correct port are used in this example.)
   - Click on the **Database drop-down** and select the source Olation database (i.e., **USING_OLATION**)—see next image.
     Important: The source Olation database must be running/opened in an Olation server so it can be accessible via PowerExcel.
   - Click **OK**.
     The connection is established; assuming all remains the same, these steps will not need to be repeated to create Slices from the selected database from this point forward.



3. Create a **PowerExcel Slice**.
   To do this:
   - In the **PowerExcel Tab** of the Excel ribbon, go to the PowerExcel Slice control group and click the **New** or **New PowerExcel Slice icon**.



   - In the **PowerExcel sidebar** that appears on the right-hand side of Excel select the <**correct PowerExcel connection**>. Following this example, select **USING_OLATION PXL**.
   - Select the <**correct Cube**>. For this example, select **SALES**.

4. Re-arrange the Dimensions in your PowerExcel Slice.
   Drag and drop the Dimensions along the Filter, Column and Row sections of the PowerExcel sidebar.

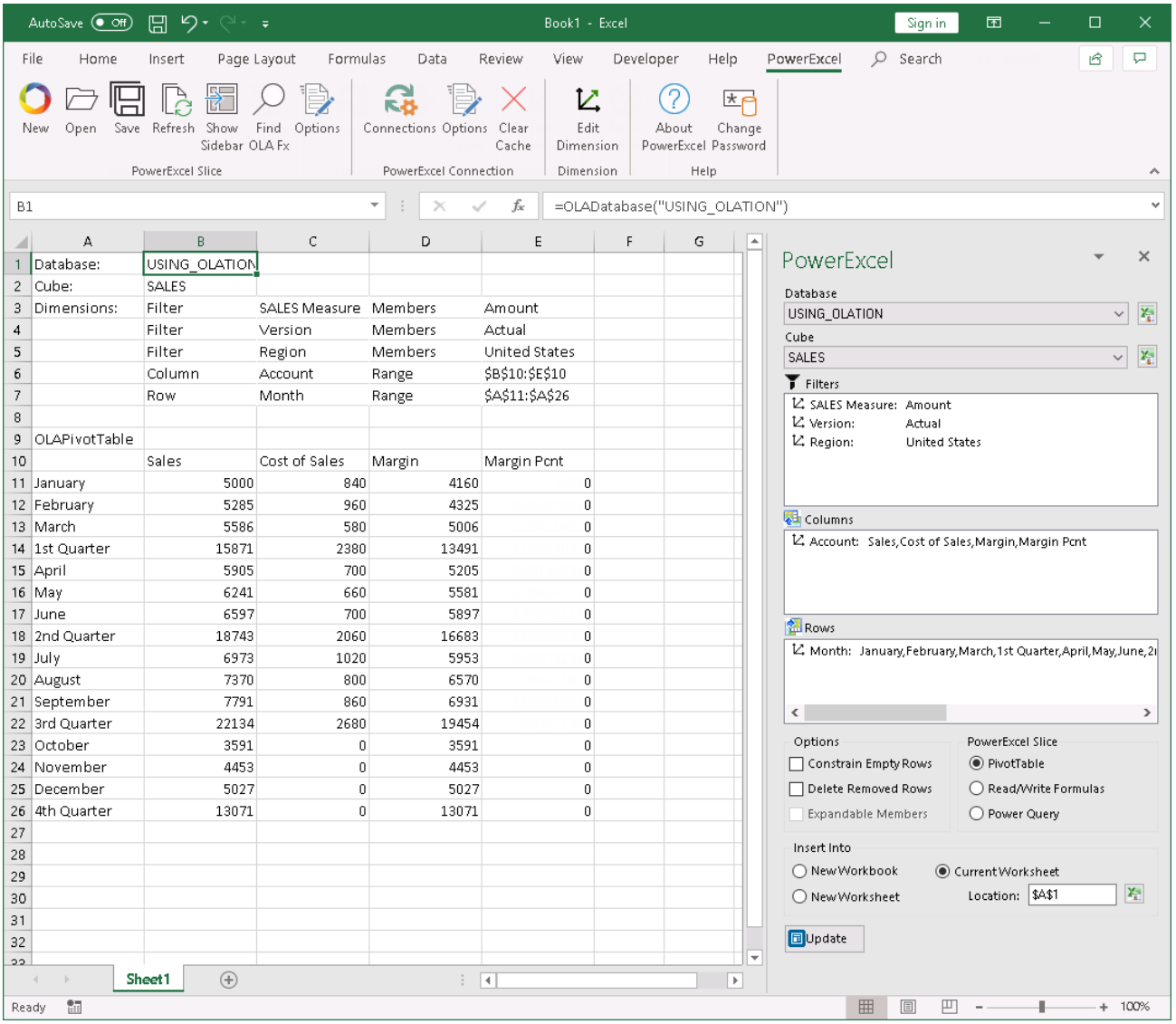
   Arrange the Dimensions as follows:

FILTER:
SALES Measure
Version
Region

COLUMNS:
Account

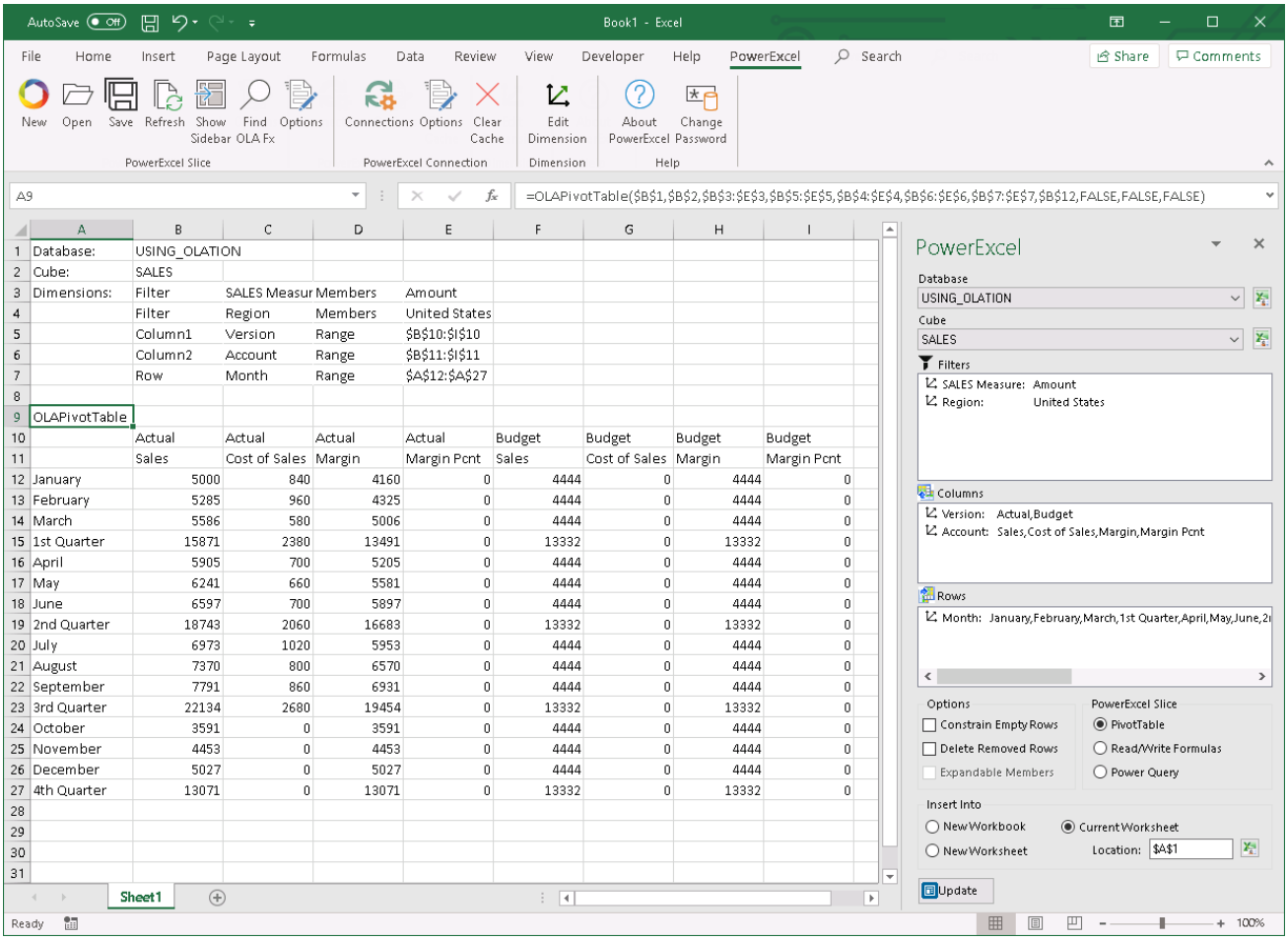
ROWS:
Month

5. Set the display Members for the Dimensions along the Filter section.
   To do this:
   - In the Filter section of the PowerExcel sidebar, double-click on a Dimension.
   - Clear/Delete the Members appearing on the right-hand pane of the Select Members dialog. Then drag and drop the preferred display Member from the left-hand side to the right-hand side of the dialog.
     Note: The Member appearing on the right-hand side of the Select Members dialog will be set as the display Member. In this example (see next image), *Actual* is the Member for the *Version* dimension, and *United States* is the Member for the *Region* dimension.
     Note: You can make use of the toolbar buttons located at the upper portion of the Select Members dialog to configure your display Member.
   - Click the **green checkmark** (OK button) to commit the changes.
6. Set the display Members for the Column/Row section.
   To do this:
   - In the Column or Row section of the PowerExcel sidebar, double-click on a Dimension.
   - Clear/Delete the Members appearing on the right-hand pane of the Select Members dialog. Then drag and drop the preferred display Member/s from the left-hand side to the right-hand side of the dialog.
     Note: The Members appearing on the right-hand side of the Select Members dialog will be displayed along the Columns or Rows of the resulting PowerExcel Slice.
     Note: You can make use of the toolbar buttons located at the upper portion of the Select Members dialog to configure the display Members for the columns and rows.
   - Define the order that you want the Members to appear by dragging and dropping them on top or below each other.
     In this example (see next image) *Sales*, *Cost of Sales*, *Margin,* and *Margin Pcnt* are shown in Columns. And *January*, *February*, *March*, *1st Quarter*…. etc., to *4th Quarter* and *Total Quarter* are in Rows
   - Click the **green checkmark** (OK button) to commit the changes.
7. Select a PowerExcel Slice type, e.g., **PivotTable**.
8. Select where you want to insert the PowerExcel Slice, i.e., whether in a New Workbook, New Worksheet or Current Worksheet. Following our example, select **Current Worksheet**.
9. Define the starting cell where the PowerExcel will be generated into.
   Following this example, select the **cell A1** as the starting point.
10. Click the **Insert** button. The resulting PowerExcel Slice will appear.
    Note: If you configure your Olation Cube to have a write-back capability and you have the access rights (assuming you are accessing a secured Olation database), then you will be able to enter values in your PowerExcel Slice and push those updates back to the Cube. Bear in mind, however, that you can write only on Detail (non-Aggregate) intersections.



You can also ‘Stack’ or ‘Nest’ Dimensions.
For this example:

- In the PowerExcel sidebar, drag the **Version** dimension on the **Columns** section just above the *Account* dimension.
- Next, double-click on the **Version** dimension to bring up the Select Members dialog and select **Actual** and **Budget** as display Members.
- Click the **green checkmark button** to go back to the PowerExcel Slice then click the **Update** button. The Slice will look as follows:



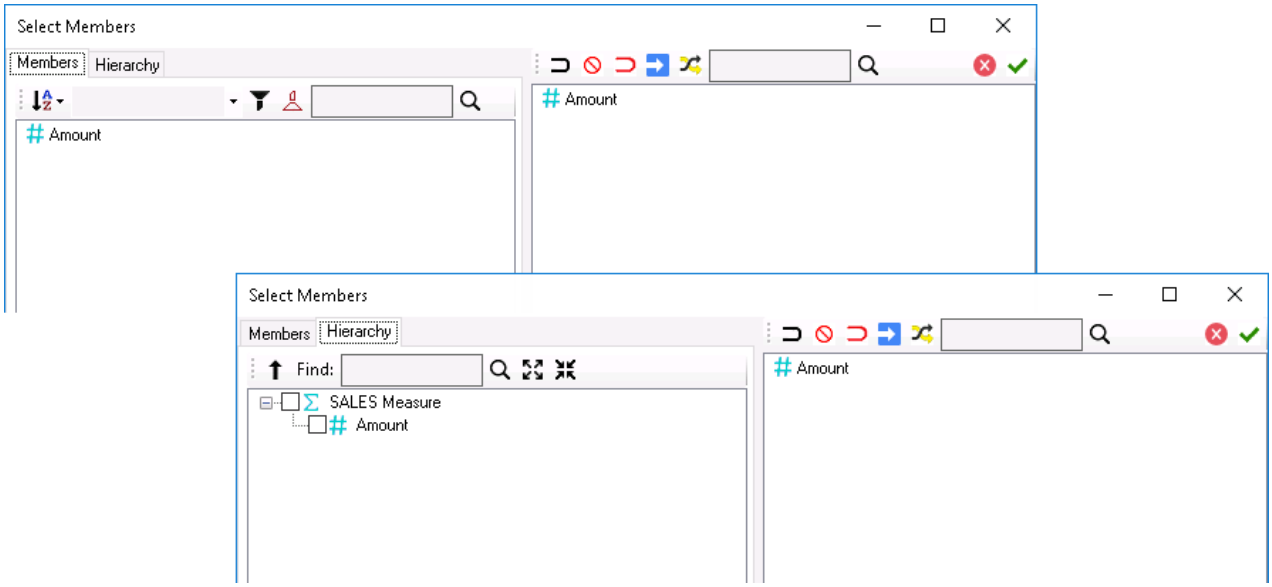
	A	B	C	D	E	F	G	H	I
1	Database:	USING_OLATION							
2	Cube:	SALES							
3	Dimensions:	Filter	SALES Measur	Members	Amount				
4		Filter	Region	Members	United States				
5		Column1	Version	Range	$B$10:$I$10				
6		Column2	Account	Range	$B$11:$I$11				
7		Row	Month	Range	$A$12:$A$27				
8									
9	OLAPivotTable								
10		Actual	Actual	Actual	Actual	Budget	Budget	Budget	Budget
11		Sales	Cost of Sales	Margin	Margin Pcnt	Sales	Cost of Sales	Margin	Margin Pcnt
12	January	5000	840	4160	0	4444	0	4444	0
13	February	5285	960	4325	0	4444	0	4444	0
14	March	5586	580	5006	0	4444	0	4444	0
15	1st Quarter	15871	2380	13491	0	13332	0	13332	0
16	April	5905	700	5205	0	4444	0	4444	0
17	May	6241	660	5581	0	4444	0	4444	0
18	June	6597	700	5897	0	4444	0	4444	0
19	2nd Quarter	18743	2060	16683	0	13332	0	13332	0
20	July	6973	1020	5953	0	4444	0	4444	0
21	August	7370	800	6570	0	4444	0	4444	0
22	September	7791	860	6931	0	4444	0	4444	0
23	3rd Quarter	22134	2680	19454	0	13332	0	13332	0
24	October	3591	0	3591	0	4444	0	4444	0
25	November	4453	0	4453	0	4444	0	4444	0
26	December	5027	0	5027	0	4444	0	4444	0
27	4th Quarter	13071	0	13071	0	13332	0	13332	0
28									
29									
30									
31									

11. Revert to the previous PowerExcel Slice orientation by moving the **Version** dimension back to the Filter area and selecting **Actual** as the display Member.

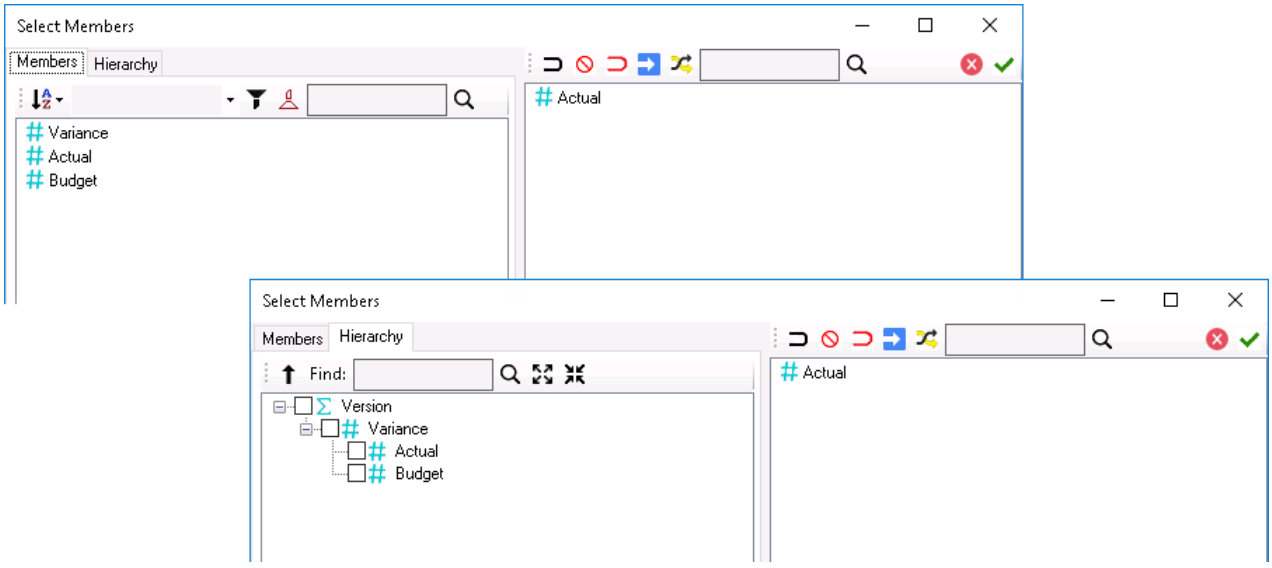
By inspecting each Dimension (to inspect, double-click on these Dimensions along the PowerExcel sidebar to bring up the Select Members dialog), we see the following Dimensions and their corresponding Members: (Note that the images show both the Members and Hierarchy Tabs.)

IMPORTANT: If the Dimensions are placed along the Filter section of the PowerExcel Slice, when you bring up the Select Members dialog you will only see the Members and Hierarchy Tabs.

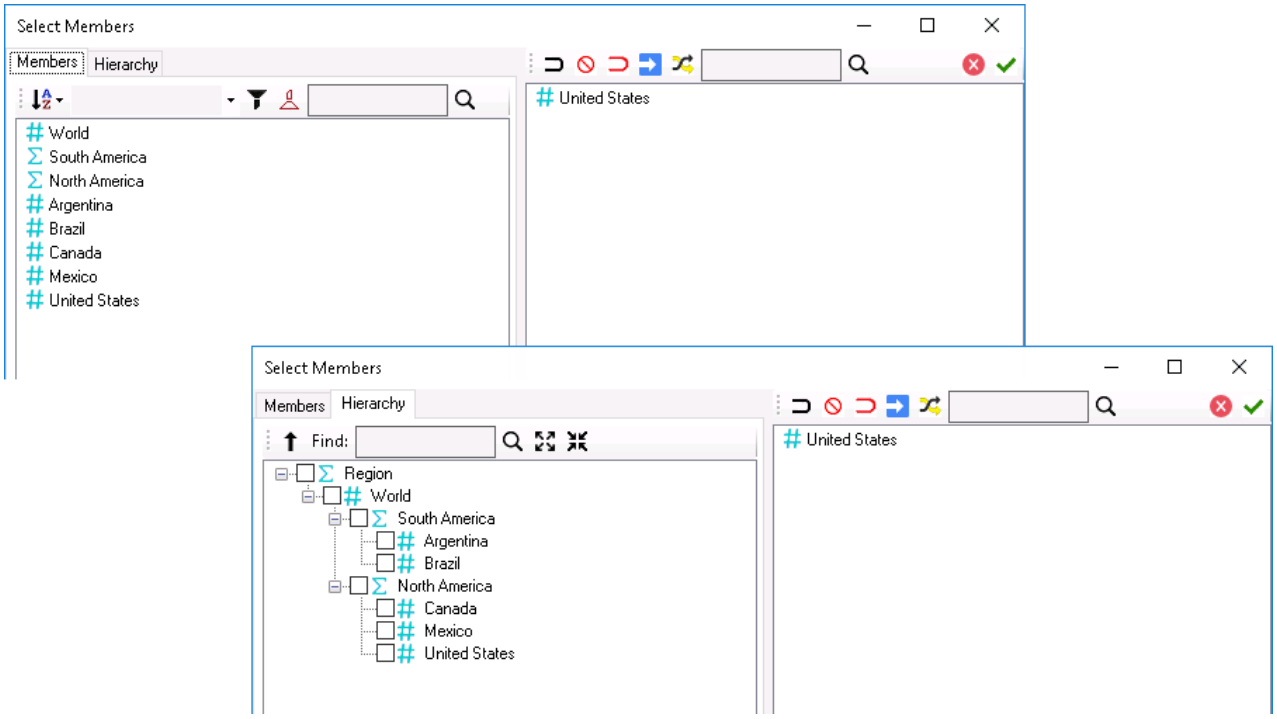
If the Dimensions are placed along the Rows or Columns sections of the PowerExcel Slice, when you bring up the Select Members dialog, you will see the Members, Hierarchy and Subsets Tabs displayed.



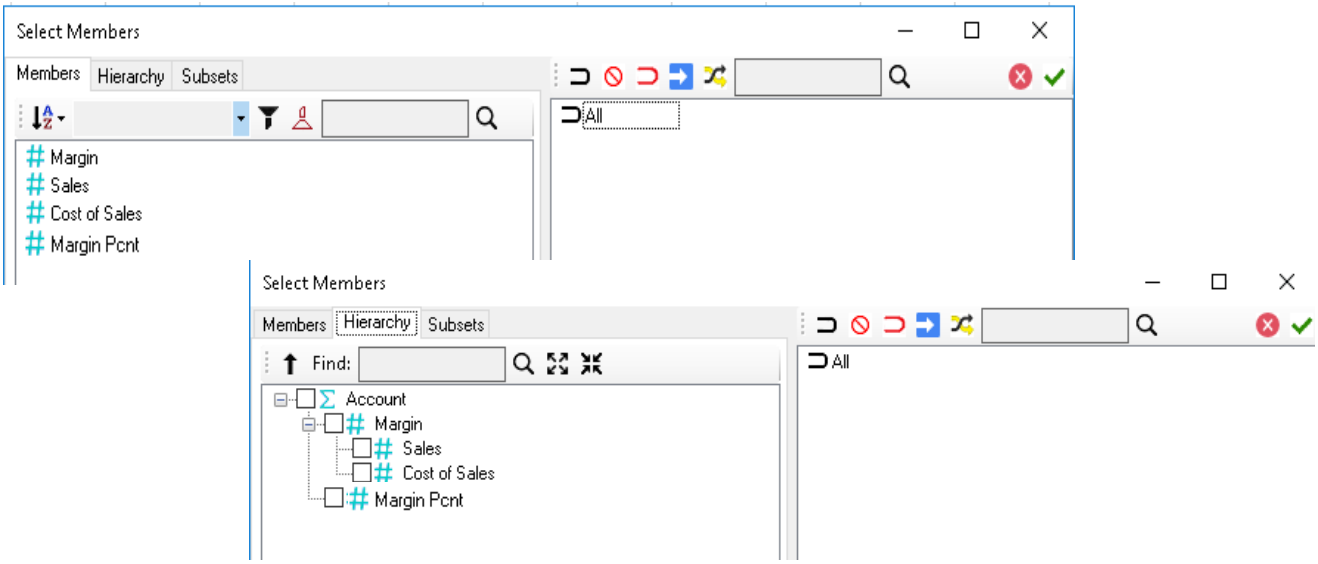
Dimension: **SALES Measure**



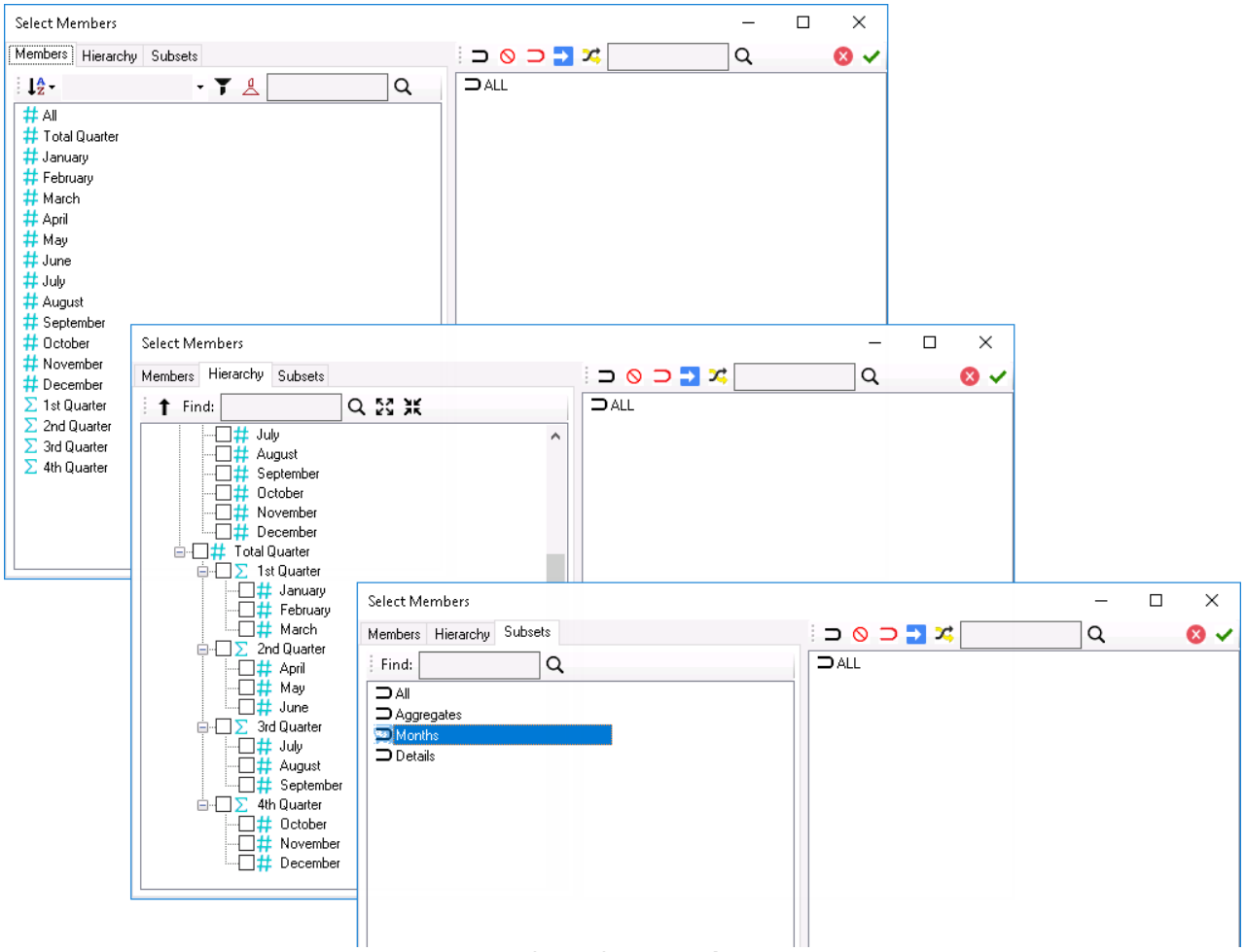
Dimension: **Version**



Dimension: **Region**



Dimension: **Account**



Dimension: **Month**

5.2. Viewing OLATION Data via PowerOLAP's OLATION Linked Cube Feature

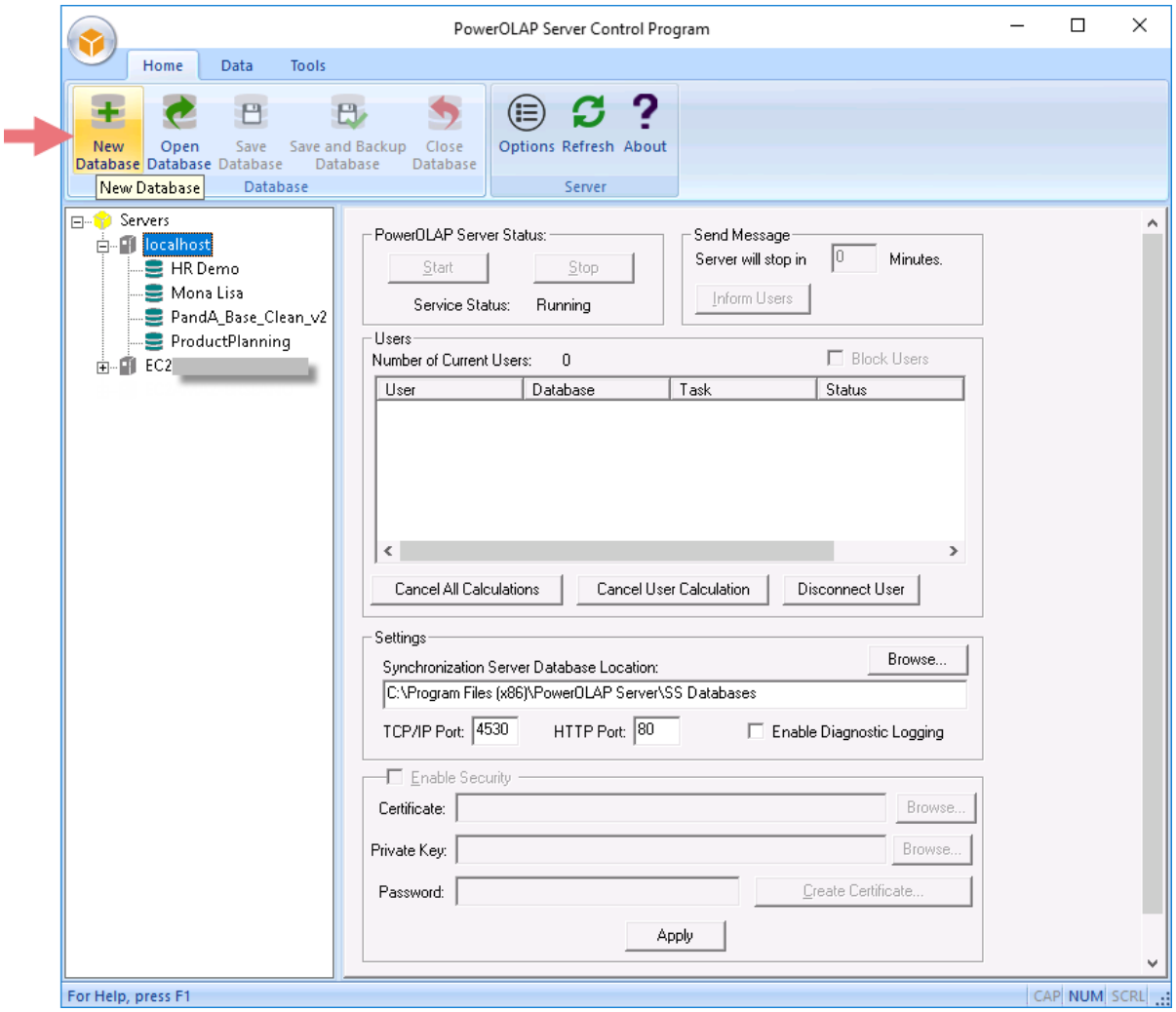
PowerOLAP—along with PowerExcel—is a PARIS Technologies product that also provides front-end capabilities to work with an Olation model. This section presupposes some familiarity with the use of PowerOLAP—including the first few steps that follow.

5.2.1. Create a New PowerOLAP Database

For this demo, we will create a New PowerOLAP database and load it on the PowerOLAP MDB Server.

To create a new PowerOLAP Database (directly loaded on the PowerOLAP Server):

1. First, launch the **PowerOLAP Server** and select a **server** where you want to create your new PowerOLAP database.
   For this example, let us just choose the **localhost**.



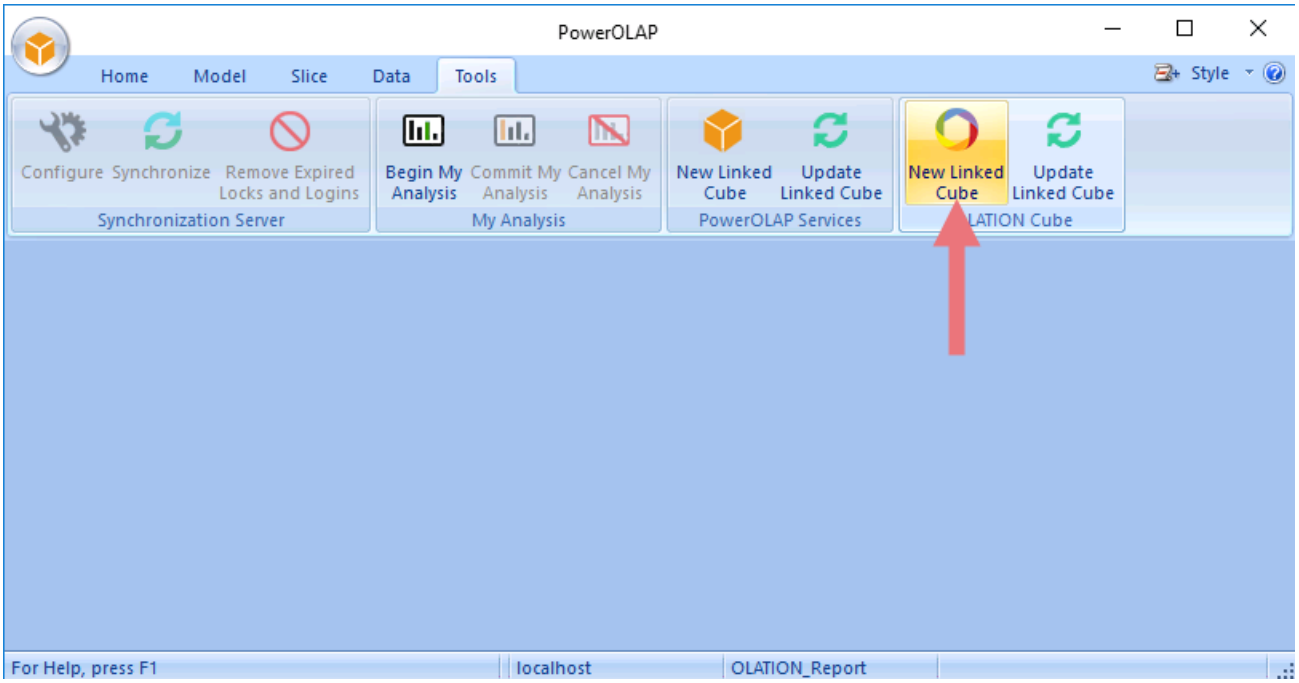
2. Click on the **New Database** command to create a new database—for this example we will call it **OLATION_Report**.

3. Next, launch the **PowerOLAP Modeler** and open the database you just created.
   To do this: click on the **File icon**→**Open Database** command→In the Open Database dialog, click on the server drop-down and select the correct server (**localhost**)→pick the correct database (**OLATION_Report**)→click **OK**.

5.2.2. Create a New OLATION Linked Cube

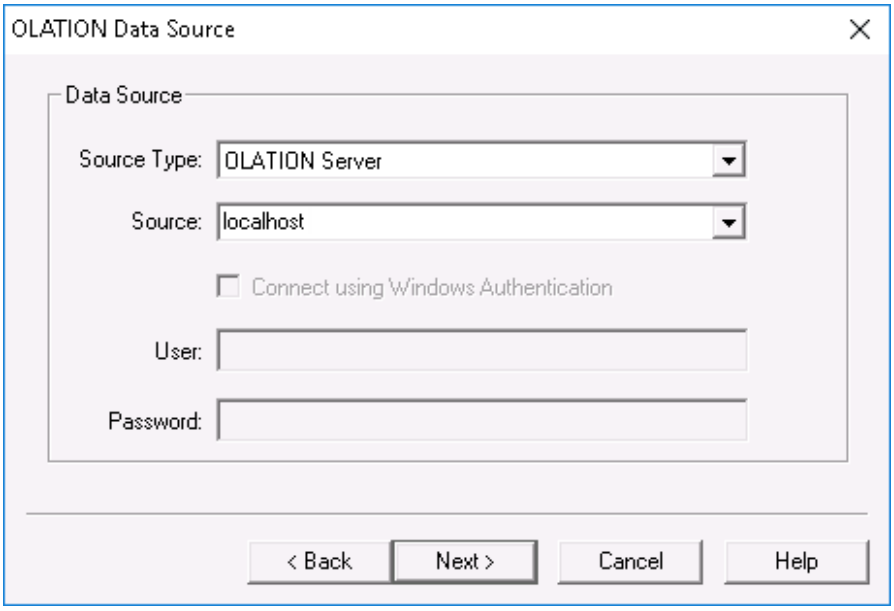
Now that the OLATION_Report database is opened in the PowerOLAP Modeler client, you can proceed to create the Olation Linked Cube:

1. In the PowerOLAP ribbon, go to the **Tools Tab**; then in the OLATION Cube control group select the **New Linked Cube** command icon.
   This will bring up the OLATION Data Source dialog where the user specifies a valid provider to be used to allow PowerOLAP to communicate to a specified database.



Fill out the required fields in the **Olation Data Source** dialog box:

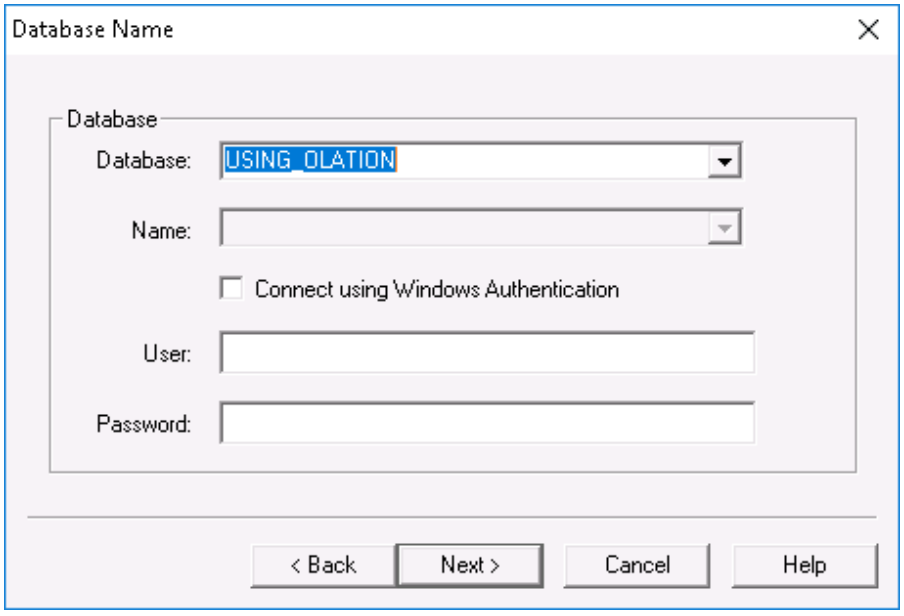
2. Select a **Source Type**.
   Since in the example, we are trying to access a local Olation database, choose **OLATION Server**.
3. In the **Source** field, type in the appropriate source. Following this exercise, enter **local** or **localhost**.



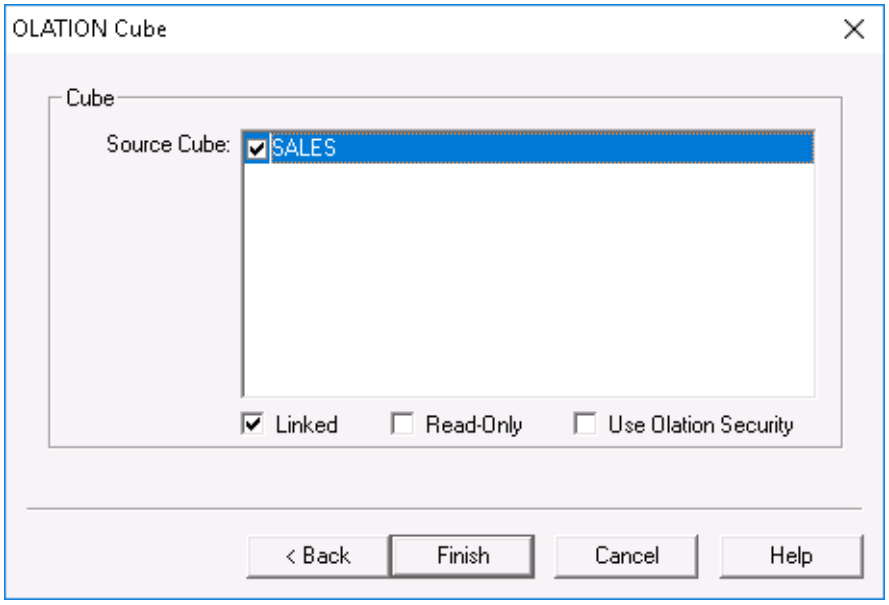
4. Click **Next**.

Fill out the required fields in the succeeding **Database Name** dialog box:

5. Click on the **Database drop-down** and select the name of the Olation database you want to connect to. For this example, select **USING_OLATION**.
6. Choose how to connect to a Secured Olation Database. Connect either via *Windows Authentication* or by providing a valid *User* and *Password*. For this exercise, since the source Olation database has no Security, we can skip this step.



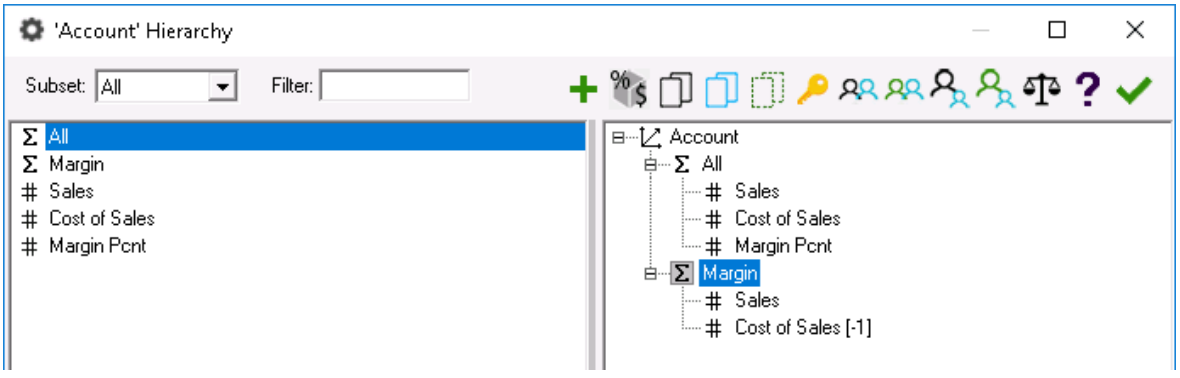
7. Click **Next**. The OLATION Cube dialog appears.



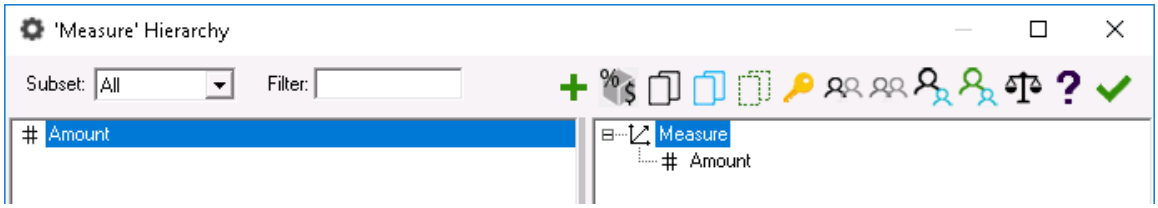
8. In the **OLATION Cube** dialog box, select **SALES** as the source Cube.
9. Take notice of the three checkboxes at the bottom: *Linked*, *Read-Only* and *Use Olation Security*. For this exercise, since we want to allow write-back and we are accessing an unsecured Olation database, enable the **Linked** checkbox option.
10. Click **Finish**—a message appears indicating that the Olation cube was successfully built.
11. Click **OK**.

5.2.3. Viewing the OLATION Metadata Structure built in PowerOLAP

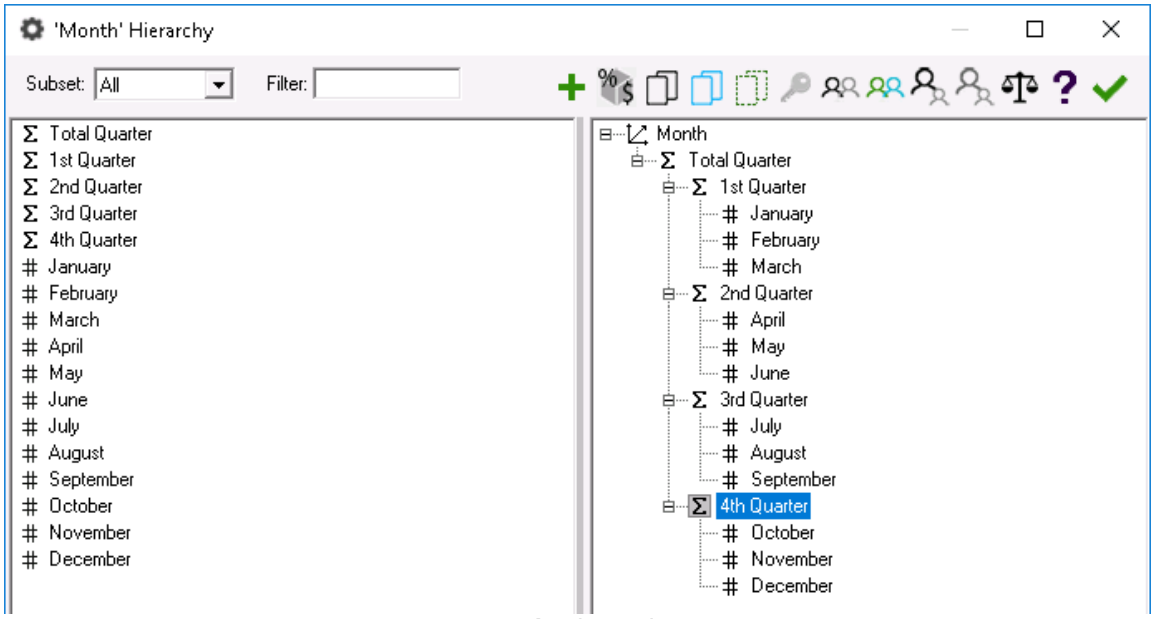
We will next view the Cube and Dimension (the metadata structure) built in PowerOLAP by creating the Linked Cube from Olation.



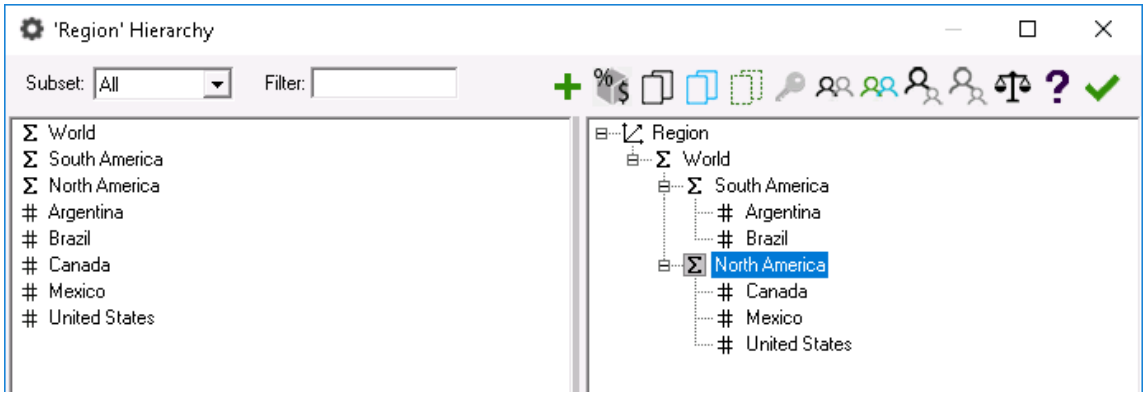
Account Dimension



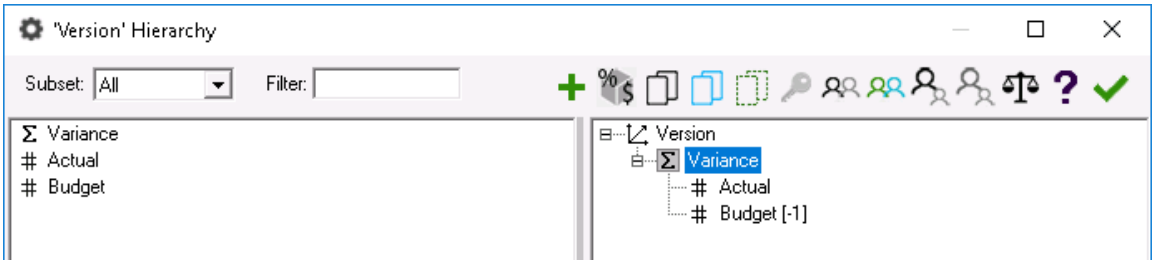
Measure Dimension



Month Dimension

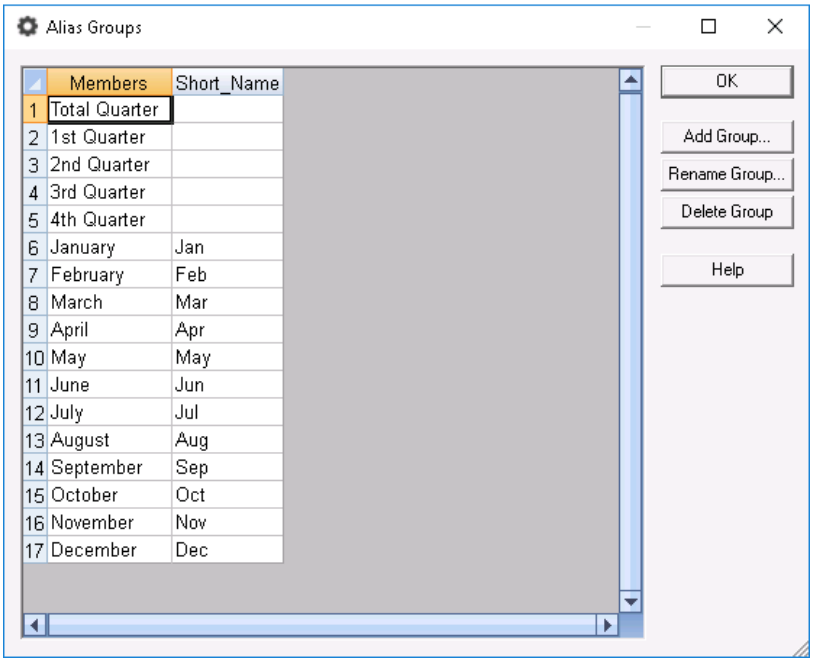


Region Dimension

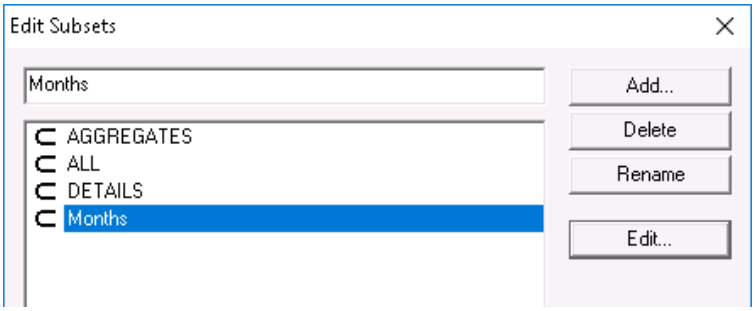


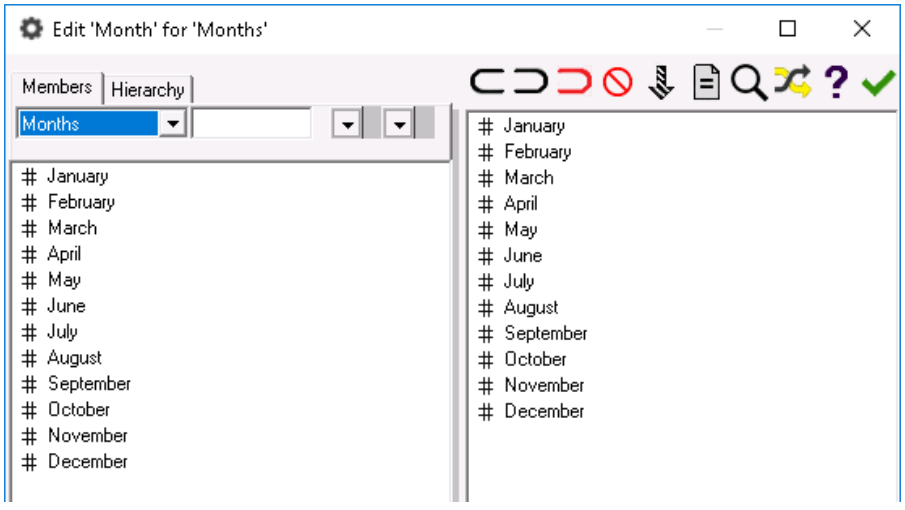
Version Dimension

You can also view the **Alias** and **Subset** Groups that you have configured for the *Month* dimension.

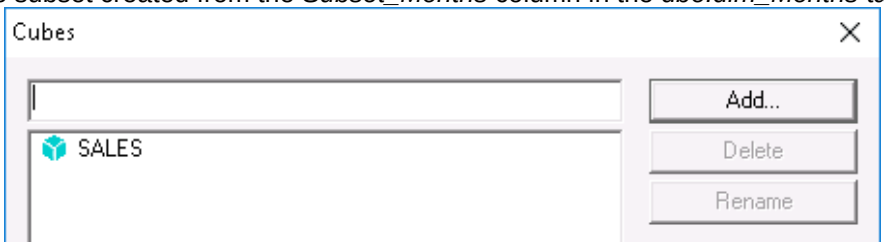


The alias created from the *Short_Name* column in the *dbo.dim_Months* table





The subset created from the *Subset_Months* column in the *dbo.dim_Months* table

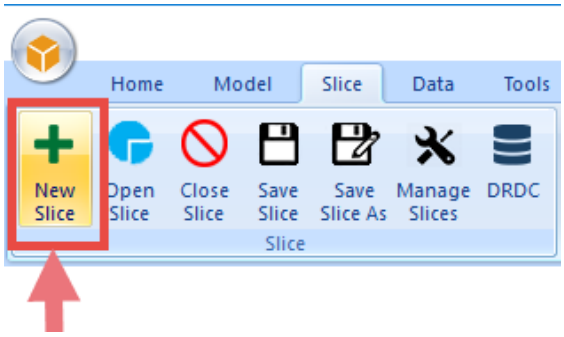


The SALES Olation Cube built in PowerOLAP

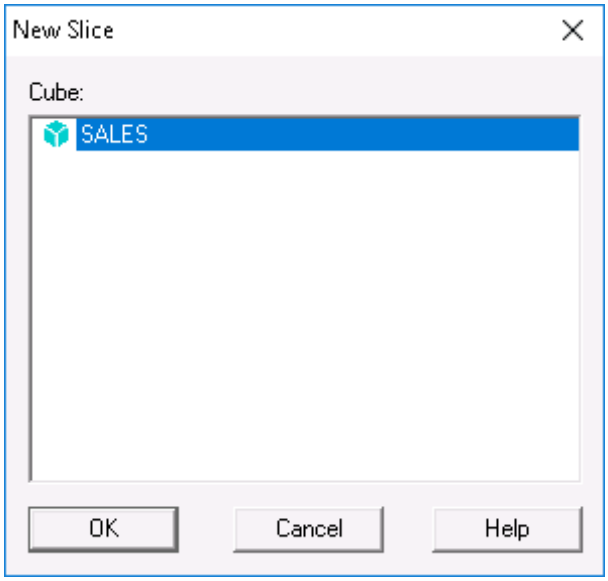
5.2.4. Viewing the Fact Data - Create a PowerOLAP Slice View

Next we will create a PowerOLAP Slice to see the transactional data coming from the Olation cube.

1. In the PowerOLAP ribbon, go to the **Slice Tab**, and in the Slice control group select the **New Slice** command icon.

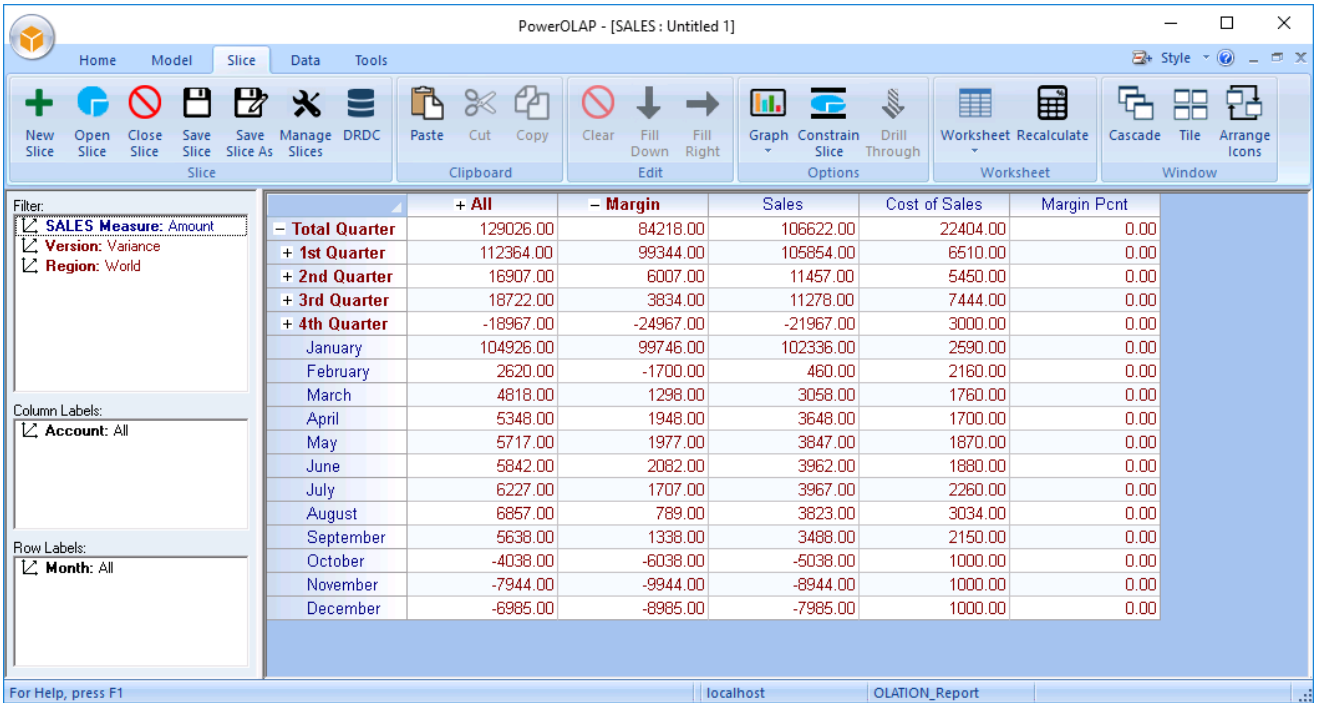


The **New Slice** dialog appears, displaying existing Cubes in the database—in this case, the SALES cube.



2. With the **SALES** cube selected, click **OK**. Press the **F9** key or click the **Recalculate** button to show data in the Slice.

   Note: You can also re-arrange the Slice or change the Filter Members as per your preference.



5.2.5. Arranging the Slice View

1. You can rearrange the Slice by dragging Dimensions along the Filter, Column and Row panes. You can also change the page Members to be displayed, and specify Members to be displayed along the columns and rows.

   Next, to re-arrange the Slice view and then create a dynamic Excel front end:

   - Go to the **Filter section** and double-click on the Dimension whose display Member you want to change. For example, double-click on the **Version** Dimension; in the **Edit Slice** dialog, go to the right-hand pane and double-click on the preferred display Member.
     Alternatively, you can also select the new display Member on the right-hand pane and click the **Set Page Member** icon that can be found along the toolbar at the top.

     Note: If the preferred Member is not in the right-hand pane, just drag and drop the appropriate Member from the Member list on the left-hand pane of the Edit Slice dialog box to the right-hand pane. Bear in mind that the Member marked in yellow is the currently selected display Member.

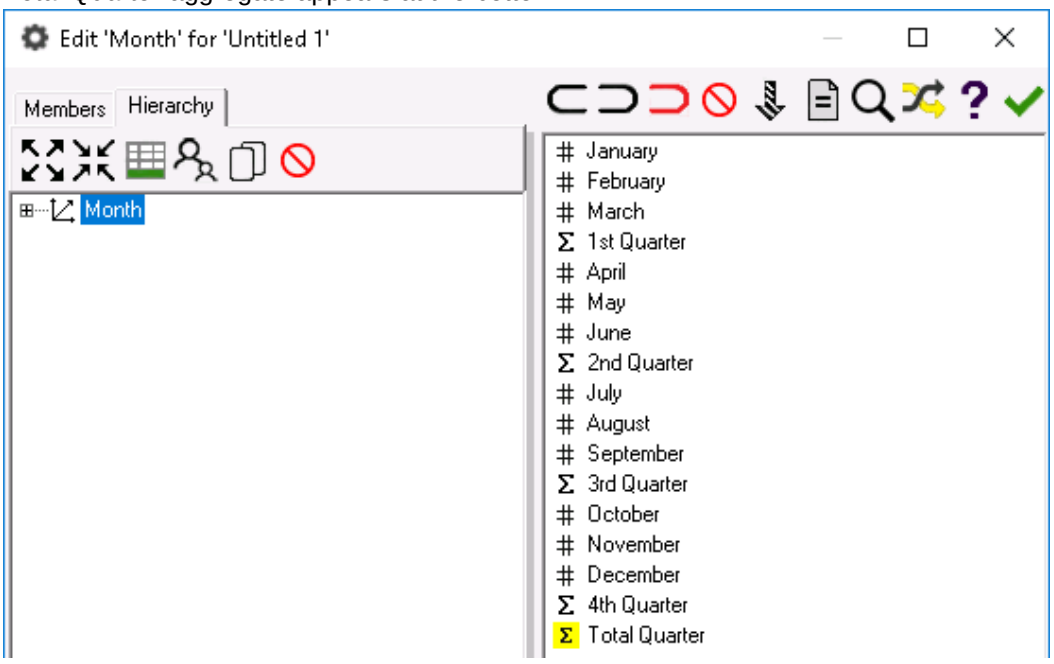
Make your Filter selections as follows:

FILTER:
SALES Measure: Amount
Version: Actual
Region: United States

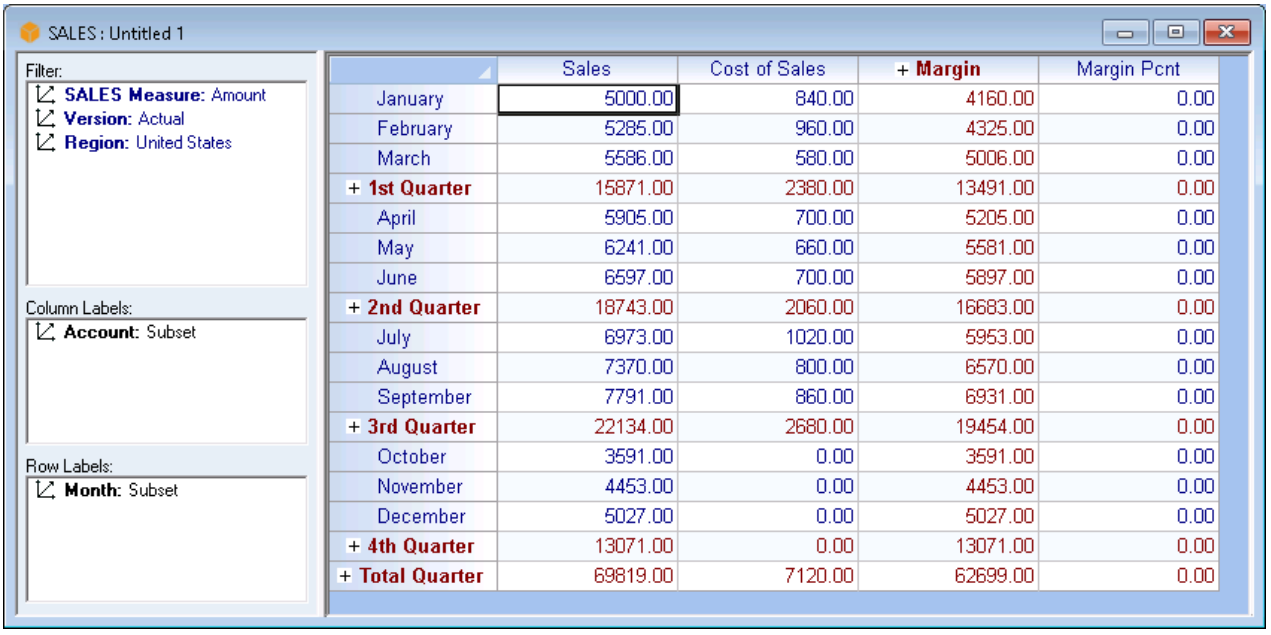
- Next, change the display Members for the 'Account' dimension along the Columns so that only *Sales*, *Cost of Sales*, *Margin*, and *Margin Pcnt* appears.
  Go to **Column Labels section** and double-click on **Account**. In the **Edit Slice** dialog that appears, include in the right-hand pane only those Members you want to see. In this case, make sure that only **Sales**, **Cost of Sales**, **Margin**, and **Margin Pcnt** are included, arranging them in that order. Click the **green checkmark** (OK button) when done.

  Note: You can use the toolbar located at the top of the right-hand pane of this dialog, or you can also use drag and drop or the delete key to add or remove Members to the display list. You can also drag and drop Members into any order you wish.

- Go to the **Row Labels section** and double-click on **Month**. In the **Edit Slice** dialog, order the Members along the Rows so that months appear above their respective quarters and the *Total Quarter* aggregate appears at the bottom:

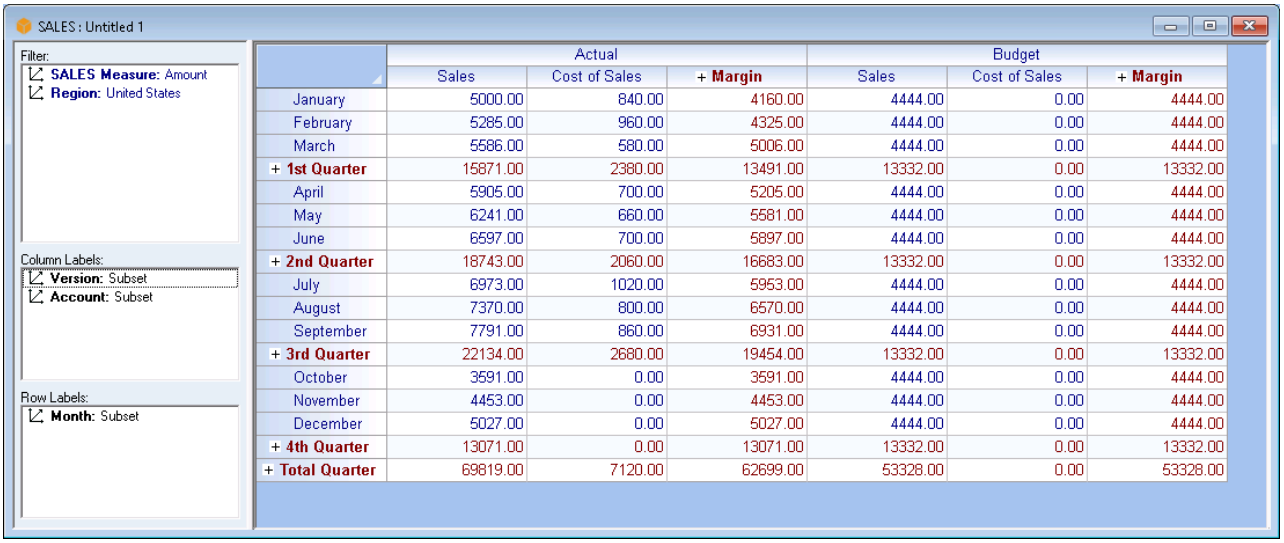


- Click the **green checkmark** (OK button) when done to go back to the Slice.
  Note: If you want to move around Dimensions, you can drag and drop them accordingly along the Filter, Column and Row areas.
- Click the **Recalculate** button along the Slice Tab of the PowerOLAP ribbon or press the **F9** key to refresh the Slice view.
  The reconfigured Slice looks as follows:



	Sales	Cost of Sales	+ Margin	Margin Pcnt
January	5000.00	840.00	4160.00	0.00
February	5285.00	960.00	4325.00	0.00
March	5586.00	580.00	5006.00	0.00
+ 1st Quarter	15871.00	2380.00	13491.00	0.00
April	5905.00	700.00	5205.00	0.00
May	6241.00	660.00	5581.00	0.00
June	6597.00	700.00	5897.00	0.00
+ 2nd Quarter	18743.00	2060.00	16683.00	0.00
July	6973.00	1020.00	5953.00	0.00
August	7370.00	800.00	6570.00	0.00
September	7791.00	860.00	6931.00	0.00
+ 3rd Quarter	22134.00	2680.00	19454.00	0.00
October	3591.00	0.00	3591.00	0.00
November	4453.00	0.00	4453.00	0.00
December	5027.00	0.00	5027.00	0.00
+ 4th Quarter	13071.00	0.00	13071.00	0.00
+ Total Quarter	69819.00	7120.00	62699.00	0.00

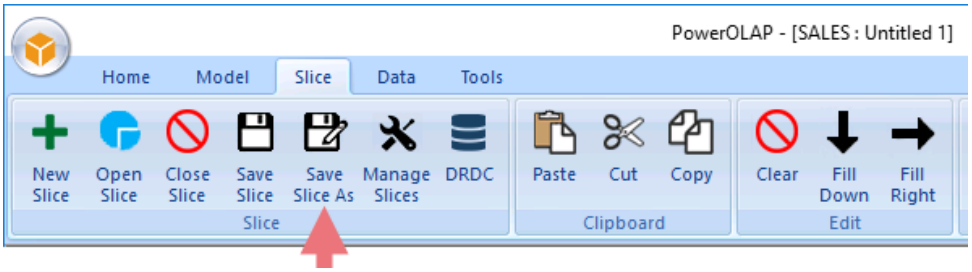
Note: You can also 'Nest' or 'Stack' Dimensions in a Slice view. An example of a 'Nested Dimension' is shown below, with *Version* and *Account* along the Columns.



	Actual			Budget		
	Sales	Cost of Sales	+ Margin	Sales	Cost of Sales	+ Margin
January	5000.00	840.00	4160.00	4444.00	0.00	4444.00
February	5285.00	960.00	4325.00	4444.00	0.00	4444.00
March	5586.00	580.00	5006.00	4444.00	0.00	4444.00
+ 1st Quarter	15871.00	2380.00	13491.00	13332.00	0.00	13332.00
April	5905.00	700.00	5205.00	4444.00	0.00	4444.00
May	6241.00	660.00	5581.00	4444.00	0.00	4444.00
June	6597.00	700.00	5897.00	4444.00	0.00	4444.00
+ 2nd Quarter	18743.00	2060.00	16683.00	13332.00	0.00	13332.00
July	6973.00	1020.00	5953.00	4444.00	0.00	4444.00
August	7370.00	800.00	6570.00	4444.00	0.00	4444.00
September	7791.00	860.00	6931.00	4444.00	0.00	4444.00
+ 3rd Quarter	22134.00	2680.00	19454.00	13332.00	0.00	13332.00
October	3591.00	0.00	3591.00	4444.00	0.00	4444.00
November	4453.00	0.00	4453.00	4444.00	0.00	4444.00
December	5027.00	0.00	5027.00	4444.00	0.00	4444.00
+ 4th Quarter	13071.00	0.00	13071.00	13332.00	0.00	13332.00
+ Total Quarter	69819.00	7120.00	62699.00	53328.00	0.00	53328.00

2. Going back to our previously reconfigured Slice (before a demonstration of 'nesting' Dimensions), save the Slice:

   - With the Slice opened and active (meaning that this should be the currently selected Slice, in case several Slices are opened), go to the **Slice Tab** and click the **Save Slice As** command.



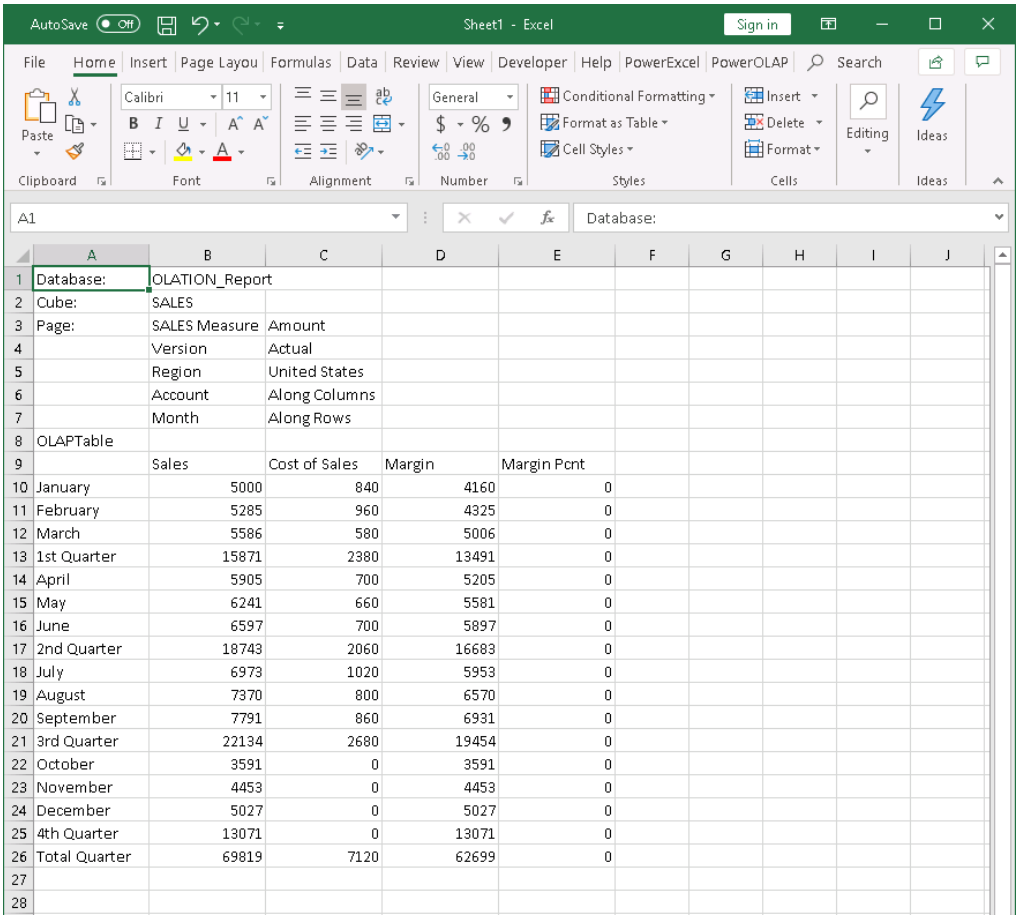
- In the Save Slice As dialog, enter the <**name of the Slice**>., e.g., **Comparative Margin per Quarter**.
- Click **OK**.

5.2.6. Create the Dynamic Excel Front End –PowerOLAP Slice to Excel

We will next create a dynamic, bi-directional Excel front end.

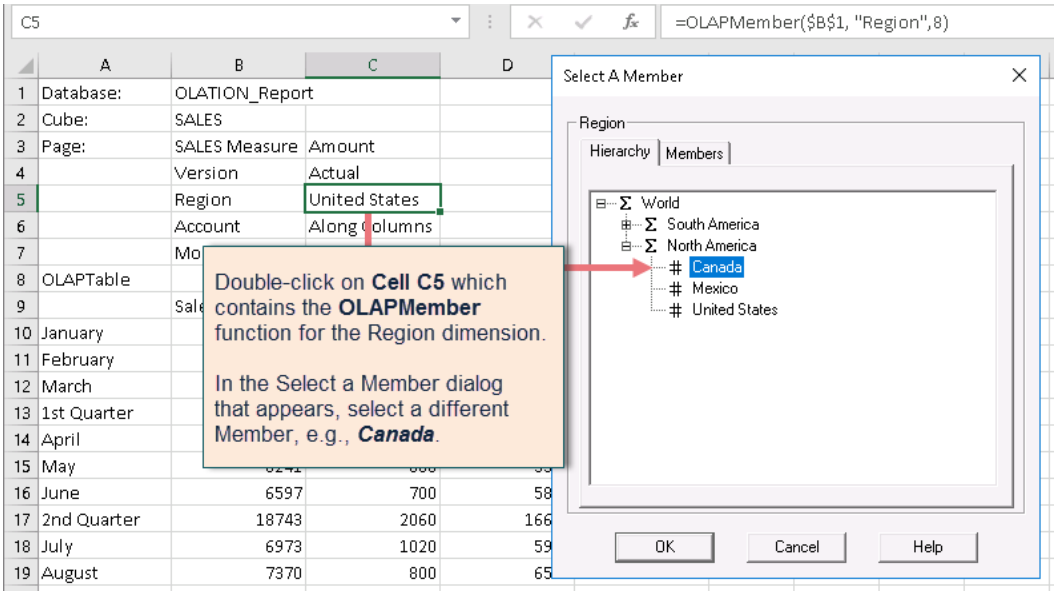
1. With the Slice opened and active, go to the PowerOLAP ribbon, **Slice Tab**, click on the **Worksheet drop-down** and select **OLAPTable** as the Excel connection type.
   Or you can simply press the **F8** key.

   The following spreadsheet will be generated.



	A	B	C	D	E
1	Database:	OLATION_Report			
2	Cube:	SALES			
3	Page:	SALES Measure	Amount		
4		Version	Actual		
5		Region	United States		
6		Account	Along Columns		
7		Month	Along Rows		
8	OLAPTable				
9		Sales	Cost of Sales	Margin	Margin Pcnt
10	January	5000	840	4160	0
11	February	5285	960	4325	0
12	March	5586	580	5006	0
13	1st Quarter	15871	2380	13491	0
14	April	5905	700	5205	0
15	May	6241	660	5581	0
16	June	6597	700	5897	0
17	2nd Quarter	18743	2060	16683	0
18	July	6973	1020	5953	0
19	August	7370	800	6570	0
20	September	7791	860	6931	0
21	3rd Quarter	22134	2680	19454	0
22	October	3591	0	3591	0
23	November	4453	0	4453	0
24	December	5027	0	5027	0
25	4th Quarter	13071	0	13071	0
26	Total Quarter	69819	7120	62699	0
27					
28					

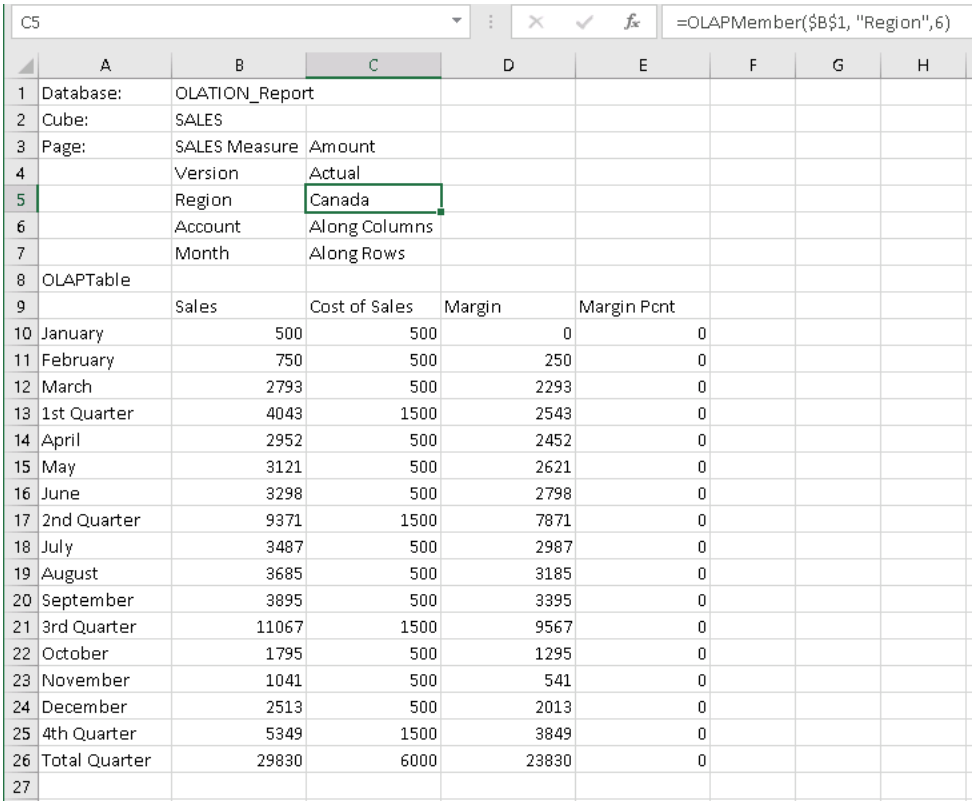
2. Now, to show data for a different region, e.g., *Canada*,
   double-click on cell **C5** (this contains the OLAPMember connection reference for the *Region* dimension).



	A	B	C	D
1	Database:	OLATION_Report		
2	Cube:	SALES		
3	Page:	SALES Measure	Amount	
4		Version	Actual	
5		Region	United States	
6		Account	Along Columns	
7		Mo		
8	OLAPTable			
9		Sal		
10	January			
11	February			
12	March			
13	1st Quarter			
14	April			
15	May			
16	June	6597	700	58
17	2nd Quarter	18743	2060	166
18	July	6973	1020	59
19	August	7370	800	65

3. In the Select a Member dialog that appears, select **Canada** then click **OK**.

4. Back in the Excel worksheet, press **F9** to refresh values.
   Notice how the values have changed to reflect data for *Canada Actuals*.



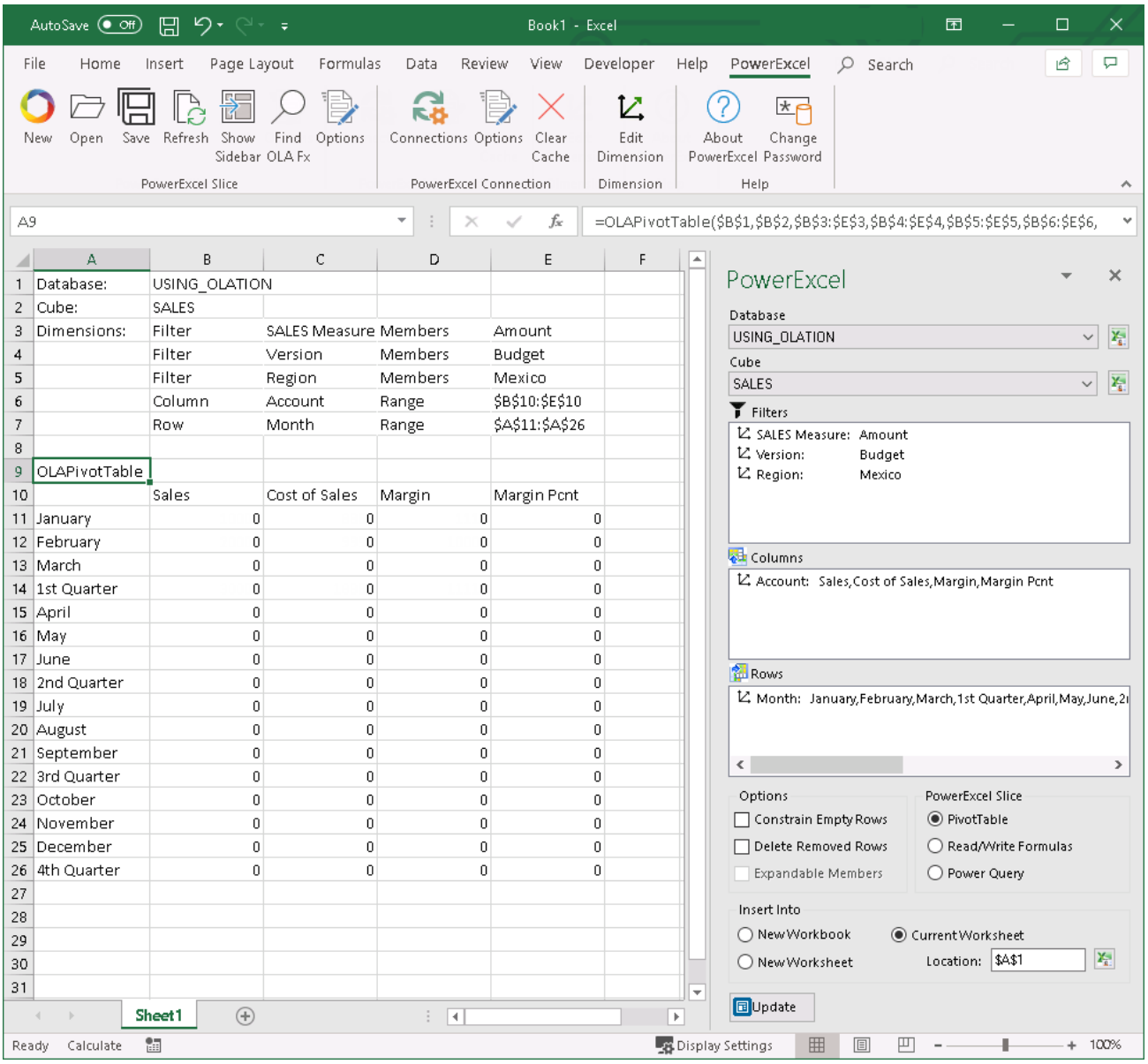
	A	B	C	D	E	F	G	H
1	Database:	OLATION_Report						
2	Cube:	SALES						
3	Page:	SALES Measure	Amount					
4		Version	Actual					
5		Region	Canada					
6		Account	Along Columns					
7		Month	Along Rows					
8	OLAPTable							
9		Sales	Cost of Sales	Margin	Margin Pcnt			
10	January	500	500	0	0			
11	February	750	500	250	0			
12	March	2793	500	2293	0			
13	1st Quarter	4043	1500	2543	0			
14	April	2952	500	2452	0			
15	May	3121	500	2621	0			
16	June	3298	500	2798	0			
17	2nd Quarter	9371	1500	7871	0			
18	July	3487	500	2987	0			
19	August	3685	500	3185	0			
20	September	3895	500	3395	0			
21	3rd Quarter	11067	1500	9567	0			
22	October	1795	500	1295	0			
23	November	1041	500	541	0			
24	December	2513	500	2013	0			
25	4th Quarter	5349	1500	3849	0			
26	Total Quarter	29830	6000	23830	0			
27								

6. OLATION® and Real-Time Application Results

We are now prepared for our next step—to show the true dynamism of OLATION®, working with Excel and PowerExcel, and, importantly, the "real, real time" applications that can be created—for example, for Sales planning.

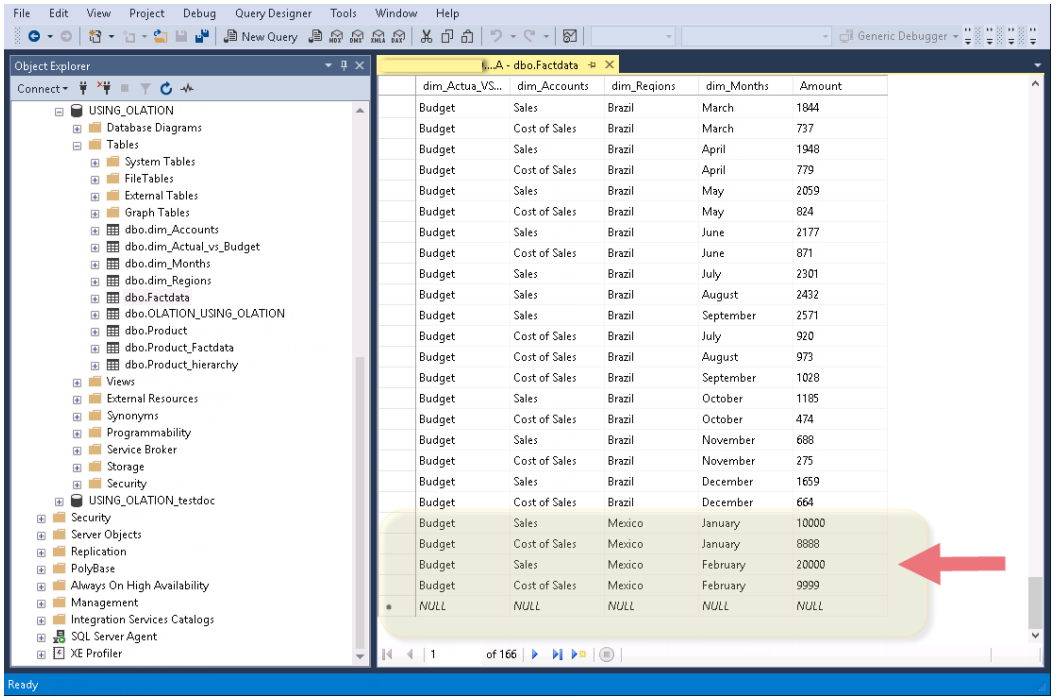
6.1. Pull Data Updates from the Relational Source down to the OLATION Database

This section concerns data updates from a source relational database, which will be reflected into an Olation model. The screenshot below shows a PowerExcel report previously created out of the Olation database called USING_OLATION. Note that there is no Budget data (all *zeros*) for the region *Mexico*.

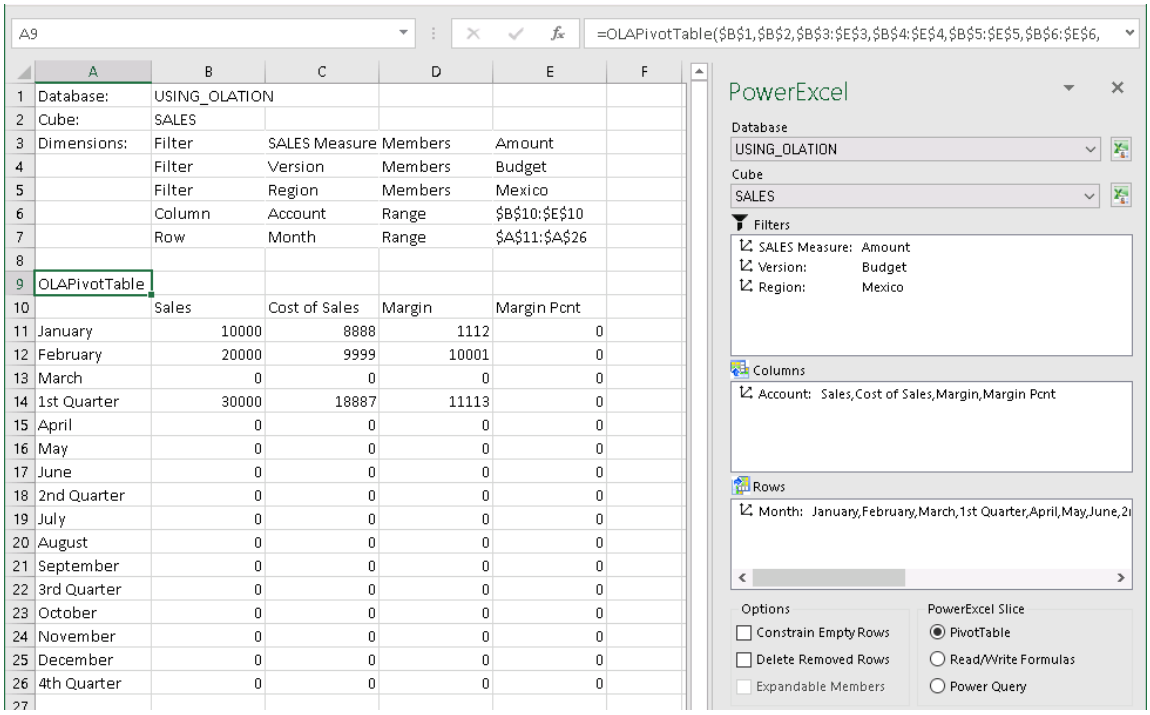


	A	B	C	D	E	F
1	Database:	USING_OLATION				
2	Cube:	SALES				
3	Dimensions:	Filter	SALES Measure	Members	Amount	
4		Filter	Version	Members	Budget	
5		Filter	Region	Members	Mexico	
6		Column	Account	Range	$B$10:$E$10	
7		Row	Month	Range	$A$11:$A$26	
8						
9	OLAPivotTable					
10		Sales	Cost of Sales	Margin	Margin Pcnt	
11	January	0	0	0	0	
12	February	0	0	0	0	
13	March	0	0	0	0	
14	1st Quarter	0	0	0	0	
15	April	0	0	0	0	
16	May	0	0	0	0	
17	June	0	0	0	0	
18	2nd Quarter	0	0	0	0	
19	July	0	0	0	0	
20	August	0	0	0	0	
21	September	0	0	0	0	
22	3rd Quarter	0	0	0	0	
23	October	0	0	0	0	
24	November	0	0	0	0	
25	December	0	0	0	0	
26	4th Quarter	0	0	0	0	
27						
28						
29						
30						
31						

Assume that there are new transactional records added into the SQL database relational source—for example, Budget figures for *Mexico* in *January* and *February*, as in the succeeding screenshot (see highlighted records):



Once you refresh or update the Olation model, the PowerExcel report (or other front-end client) will show the values reflected there—as in the following image:



	A	B	C	D	E	F
1	Database:	USING_OLATION				
2	Cube:	SALES				
3	Dimensions:	Filter	SALES Measure	Members	Amount	
4		Filter	Version	Members	Budget	
5		Filter	Region	Members	Mexico	
6		Column	Account	Range	$B$10:$E$10	
7		Row	Month	Range	$A$11:$A$26	
8						
9	OLAPivotTable					
10		Sales	Cost of Sales	Margin	Margin Pcnt	
11	January	10000	8888	1112	0	
12	February	20000	9999	10001	0	
13	March	0	0	0	0	
14	1st Quarter	30000	18887	11113	0	
15	April	0	0	0	0	
16	May	0	0	0	0	
17	June	0	0	0	0	
18	2nd Quarter	0	0	0	0	
19	July	0	0	0	0	
20	August	0	0	0	0	
21	September	0	0	0	0	
22	3rd Quarter	0	0	0	0	
23	October	0	0	0	0	
24	November	0	0	0	0	
25	December	0	0	0	0	
26	4th Quarter	0	0	0	0	
27						

6.2. Write-back (for Planning) to the Source Relational Database

For this exercise we will enter data in a dynamic front end—PowerExcel, in this example—and see how it is written back to the source USING_OLATION SQL relational database.

Assume that you are a Salesperson in Brazil and you need to enter Budget figures based on Prior Year Actuals data; the objective is to increase our Sales by 10%.

Additionally, we want to assess our *Prior Year Actual performance* and get the Average Cost Ratio, and then increase it by 2% to cover the expected increase in raw materials. We will then multiply our Actuals data by the new Cost Ratio to get our new Cost of Sales Budget figures.

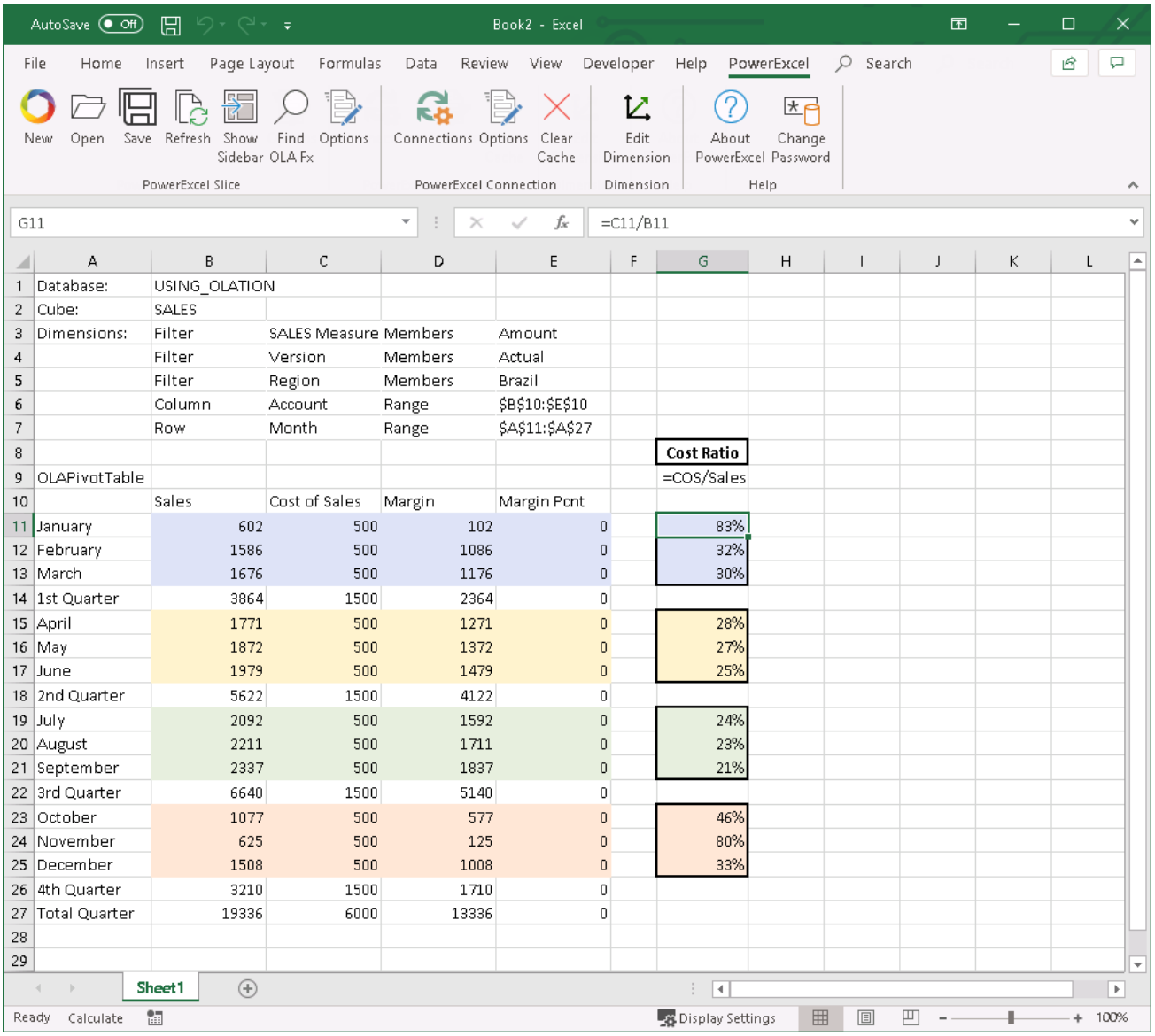
First create a formula in Excel to calculate the Cost Ratio by month. Go to Column G and define a formula where:

COST RATIO = Cost of Sales / Sales

IMPORTANT: For this exercise, we will make use of the PowerExcel Slice report we have created to show customer calculations that check the results from the Olation model. It is worth noting that you can always remove these calculations that are added in Excel.

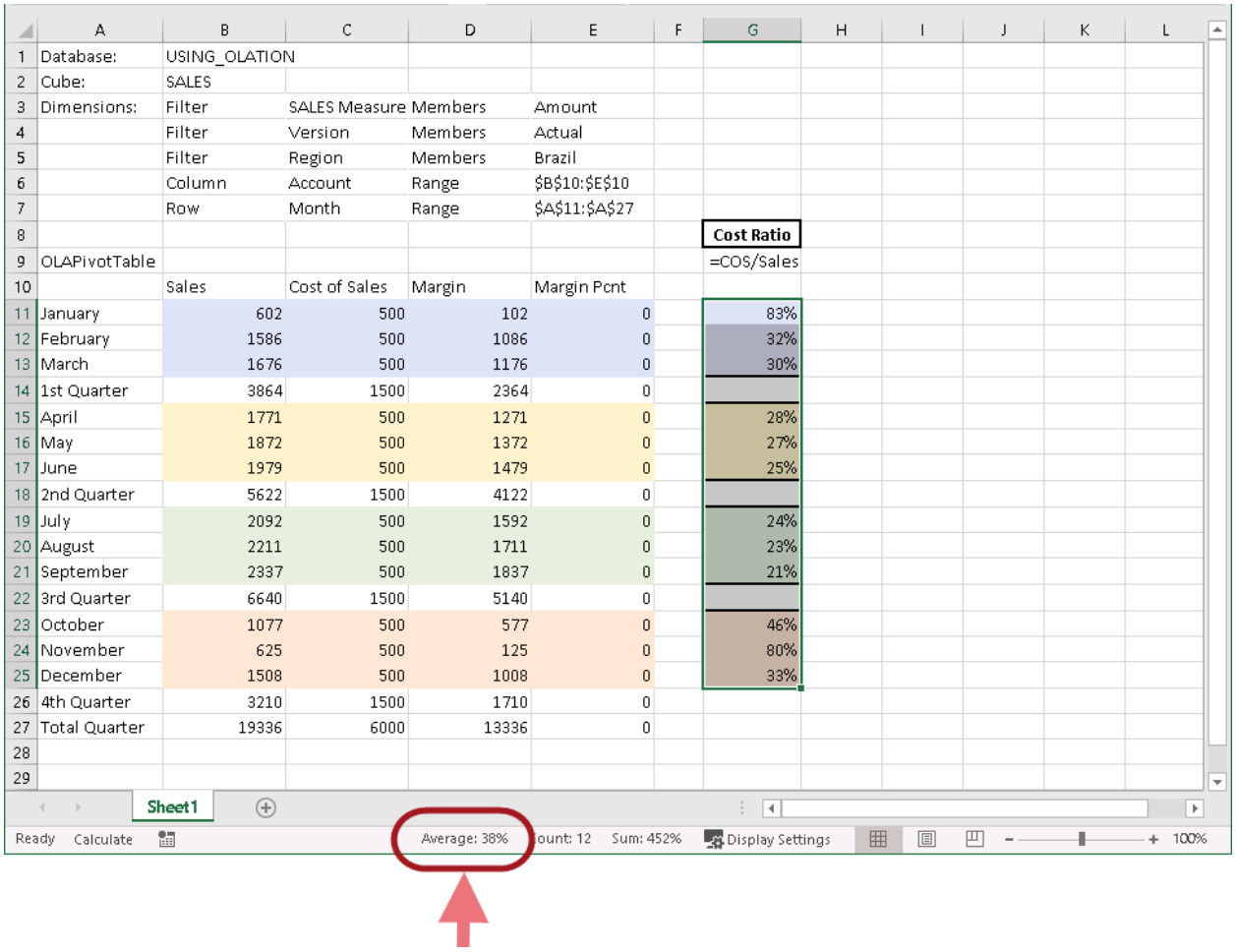
1. Go to cell **G11** and enter the formula **=C11/B11**
   whereby:
   C11 – is the Cost of Sales for the month of January; and
   B11 – is the Sales for the month of January
2. Copy this formula across all individual months *February* to *December*.
   Delete formula for aggregate Members: Quarters and Total Quarters.
3. Press the **F9** key to refresh values.

   The Excel report will appear as follows:



	A	B	C	D	E	F	G	H	I	J	K	L
1	Database:	USING_OLATION										
2	Cube:	SALES										
3	Dimensions:	Filter	SALES Measure	Members	Amount							
4		Filter	Version	Members	Actual							
5		Filter	Region	Members	Brazil							
6		Column	Account	Range	$B$10:$E$10							
7		Row	Month	Range	$A$11:$A$27							
8							Cost Ratio					
9	OLAPivotTable						=COS/Sales					
10		Sales	Cost of Sales	Margin	Margin Pcnt							
11	January	602	500	102	0		83%					
12	February	1586	500	1086	0		32%					
13	March	1676	500	1176	0		30%					
14	1st Quarter	3864	1500	2364	0							
15	April	1771	500	1271	0		28%					
16	May	1872	500	1372	0		27%					
17	June	1979	500	1479	0		25%					
18	2nd Quarter	5622	1500	4122	0							
19	July	2092	500	1592	0		24%					
20	August	2211	500	1711	0		23%					
21	September	2337	500	1837	0		21%					
22	3rd Quarter	6640	1500	5140	0							
23	October	1077	500	577	0		46%					
24	November	625	500	125	0		80%					
25	December	1508	500	1008	0		33%					
26	4th Quarter	3210	1500	1710	0							
27	Total Quarter	19336	6000	13336	0							
28												
29												

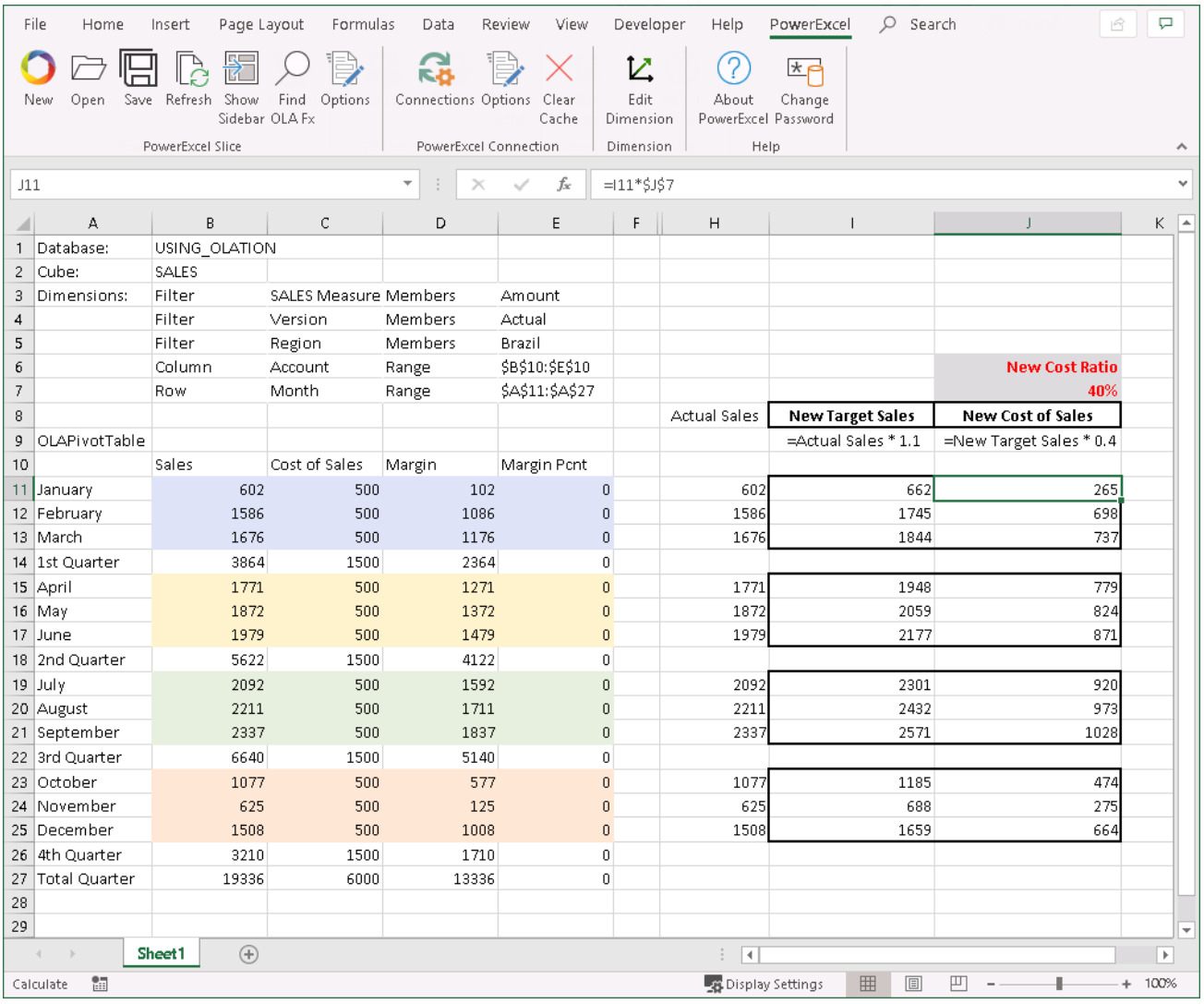
4. Add the Cost Ratio for all months and divide it by 12. This will give you an Average Cost Ratio of about 38%.
   TIP: There is an average value found that can be found at the bottom of the Excel window (circled in the following image) that averages the highlighted values.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Database:	USING_OLATION										
2	Cube:	SALES										
3	Dimensions:	Filter	SALES Measure	Members	Amount							
4		Filter	Version	Members	Actual							
5		Filter	Region	Members	Brazil							
6		Column	Account	Range	$B$10:$E$10							
7		Row	Month	Range	$A$11:$A$27							
8							Cost Ratio					
9	OLAPivotTable						=COS/Sales					
10		Sales	Cost of Sales	Margin	Margin Pcnt							
11	January	602	500	102	0		83%					
12	February	1586	500	1086	0		32%					
13	March	1676	500	1176	0		30%					
14	1st Quarter	3864	1500	2364	0							
15	April	1771	500	1271	0		28%					
16	May	1872	500	1372	0		27%					
17	June	1979	500	1479	0		25%					
18	2nd Quarter	5622	1500	4122	0							
19	July	2092	500	1592	0		24%					
20	August	2211	500	1711	0		23%					
21	September	2337	500	1837	0		21%					
22	3rd Quarter	6640	1500	5140	0							
23	October	1077	500	577	0		46%					
24	November	625	500	125	0		80%					
25	December	1508	500	1008	0		33%					
26	4th Quarter	3210	1500	1710	0							
27	Total Quarter	19336	6000	13336	0							
28												
29												

5. Now that we have identified the Average Cost of 38%, we will increase it by 2%, i.e., making budgeted Cost of Sales is 40%. Therefore, **type 40%** in **cell J7**. We can now delete data or hide column G.
6. Next, as we intend to calculate new targeted Sales figures (110% of Last Year's Actual Sales): copy the *Sales values* for individual months and **Paste Special as Values** to Column H. Again, remove values for aggregate months.
7. In **Column I** create a formula that will calculate the new target *Sales Amount* per month, which is 110% of each month Sales values.
   To do this:
   - Enter the formula: **=H11*1.1** in cell **I11**.
   - Copy and paste the formula to cells **I12 to I25**.
   - Press **F9** to refresh values.
   - In **Column J**, create a formula that will compute for the budgeted Cost of Sales by multiplying **New Target Sales** values (in column I) to 40% (cell J7).
   - Enter the formula: **=I11*$J$7** in **J11**. (Note: Make J7 an absolute reference, as indicated.)
     To summarize:
     I11 – is the New Target Sales for the month of January; and
     $J$7 – is the cell containing the New Cost Ratio.

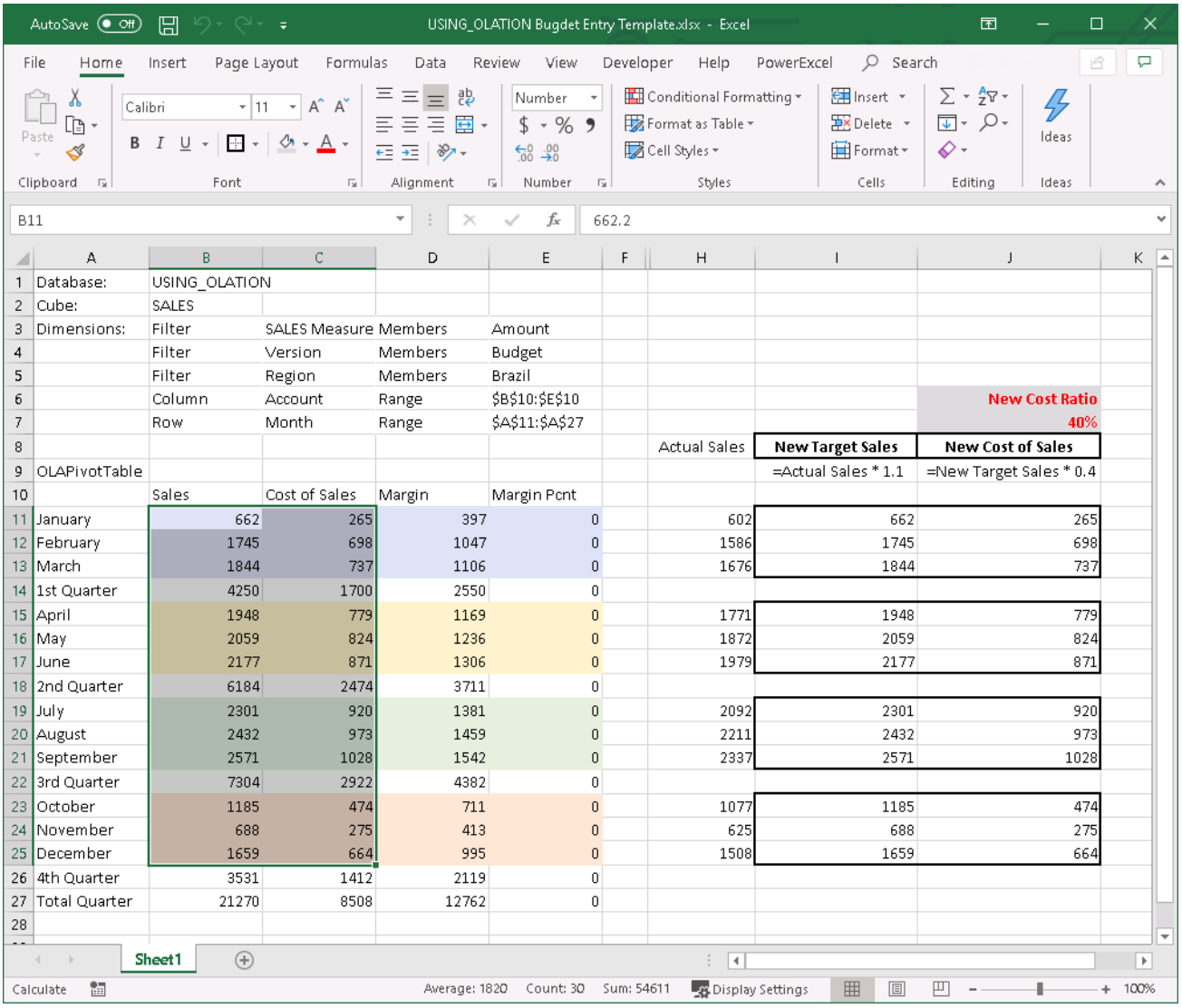
- Copy and paste the formula to cells **J12 to J25**.
- Press **F9** to refresh values—see results in the following image:



	A	B	C	D	E	F	H	I	J	K
1	Database:	USING_OLATION								
2	Cube:	SALES								
3	Dimensions:	Filter	SALES Measure	Members	Amount					
4		Filter	Version	Members	Actual					
5		Filter	Region	Members	Brazil					
6		Column	Account	Range	$B$10:$E$10				New Cost Ratio	
7		Row	Month	Range	$A$11:$A$27				40%	
8							Actual Sales	New Target Sales	New Cost of Sales	
9	OLAPivotTable							=Actual Sales * 1.1	=New Target Sales * 0.4	
10		Sales	Cost of Sales	Margin	Margin Pcnt					
11	January	602	500	102	0		602	662	265	
12	February	1586	500	1086	0		1586	1745	698	
13	March	1676	500	1176	0		1676	1844	737	
14	1st Quarter	3864	1500	2364	0					
15	April	1771	500	1271	0		1771	1948	779	
16	May	1872	500	1372	0		1872	2059	824	
17	June	1979	500	1479	0		1979	2177	871	
18	2nd Quarter	5622	1500	4122	0					
19	July	2092	500	1592	0		2092	2301	920	
20	August	2211	500	1711	0		2211	2432	973	
21	September	2337	500	1837	0		2337	2571	1028	
22	3rd Quarter	6640	1500	5140	0					
23	October	1077	500	577	0		1077	1185	474	
24	November	625	500	125	0		625	688	275	
25	December	1508	500	1008	0		1508	1659	664	
26	4th Quarter	3210	1500	1710	0					
27	Total Quarter	19336	6000	13336	0					
28										
29										

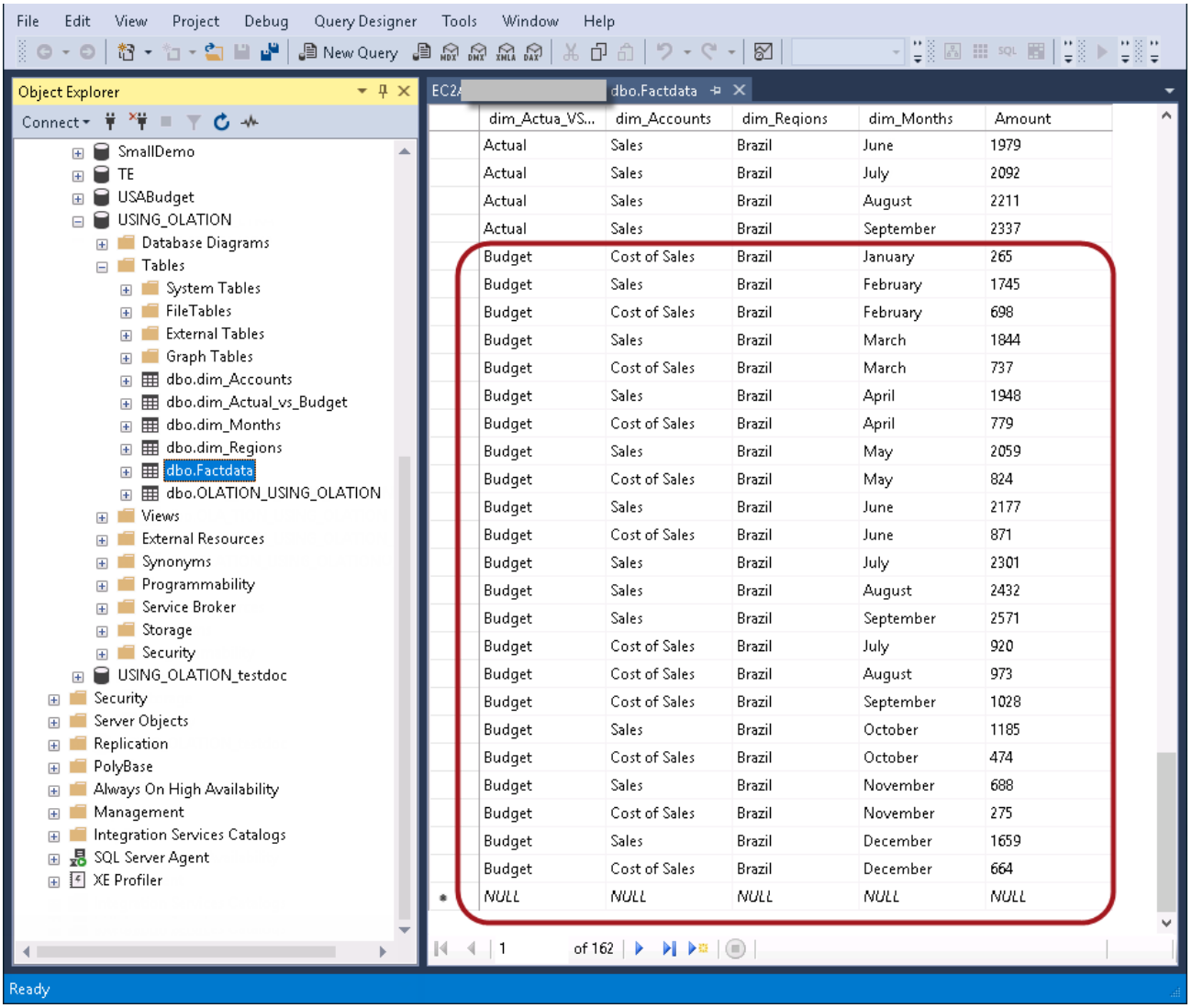
8. You can apply Excel formatting to this worksheet so you can reuse it in the future. For this example, save this as **USING_OLATION Budget Entry Template**.
9. To make the spreadsheet true to its name, we want it to show Budget values: to do this, double-click on cell **E4** and change the *Version* to **Budget**. Press the **F9** key to refresh values. We now have the Budget Sales and Budget Cost of Sales figures for Brazil—all *zero* values at this point.
10. Next, to enter those values into the USING_OLATION database: you can manually enter the numbers, but an easier method is to copy the computed *New Target Sales* and *Budgeted Cost of Sales* figures.
    - Highlight the range of cells that contains the new values (in the example, **I11 to J25**), copy the values and then put your cursor on cell **B11** then select **Paste as Values**.

11. Press **F9** to refresh values.
    Upon refreshing the values, notice how, dynamically, the figures appear and also aggregate for *Quarters*, *Total Quarter* and *Margin*.



	A	B	C	D	E	F	H	I	J	K
1	Database:	USING_OLATION								
2	Cube:	SALES								
3	Dimensions:	Filter	SALES Measure	Members	Amount					
4		Filter	Version	Members	Budget					
5		Filter	Region	Members	Brazil					
6		Column	Account	Range	$B$10:$E$10				New Cost Ratio	
7		Row	Month	Range	$A$11:$A$27				40%	
8							Actual Sales	New Target Sales	New Cost of Sales	
9	OLAPivotTable							=Actual Sales * 1.1	=New Target Sales * 0.4	
10		Sales	Cost of Sales	Margin	Margin Pcnt					
11	January	662	265	397	0		602	662	265	
12	February	1745	698	1047	0		1586	1745	698	
13	March	1844	737	1106	0		1676	1844	737	
14	1st Quarter	4250	1700	2550	0					
15	April	1948	779	1169	0		1771	1948	779	
16	May	2059	824	1236	0		1872	2059	824	
17	June	2177	871	1306	0		1979	2177	871	
18	2nd Quarter	6184	2474	3711	0					
19	July	2301	920	1381	0		2092	2301	920	
20	August	2432	973	1459	0		2211	2432	973	
21	September	2571	1028	1542	0		2337	2571	1028	
22	3rd Quarter	7304	2922	4382	0					
23	October	1185	474	711	0		1077	1185	474	
24	November	688	275	413	0		625	688	275	
25	December	1659	664	995	0		1508	1659	664	
26	4th Quarter	3531	1412	2119	0					
27	Total Quarter	21270	8508	12762	0					
28										

12. Next, we can see that the real-time result of the Budget entries have propagated dynamically through the application—next, utilize SQL Server Management Studio to access the newly added data (which should be reflected in the *dbo.Factdata* SQL table) of the USING_OLATION SQL database. See below, showing the new *Budget Sales*, *Cost of Sales*, and *Margin* values for *Brazil* are added into the fact data table.



7. Define Formula in OLATION®

We will next cover how to define formulas within an Olation Cube. For this exercise, we will create a simple Internal Cube Formula that computes the value of **Margin Pcnt** (abbreviation for *Margin Percent*).

When writing a Cube formula in Olation, be aware of the correct formula syntax and terminologies:

Left Hand Side Expression (LHS) = Right Hand Side Expression (RHS);

Parameters explained below:

Parameter	Description
Left Hand Side or **LHS**	The defined expression or range where the calculation "goes"
Right Hand Side or **RHS**	Contains a constant value, a mathematical expression, and may include Cube Functions and a Cube Reference
Semi-colon (**;**)	Indicates the end of a particular Formula

It is also important to understand the SYNTAX for each side (LHS and RHS).

Left Hand Side (LHS) or RANGE Reference

Qualifier{"Dimension Name.Member Name"}

Qualifier	Determines in what areas or intersections within the Cube the value be returned. The different types of Qualifiers are: • All And {} • Aggregates And {} • Details And {}
{"Dimension Name.Member Name"}	This indicates the specific Dimension and Member that will be populated by the resulting values.

The QUALIFIERS

Below is a brief explanation of what the qualifiers are:

- **All And {}** - This qualifier indicates that LHS Range will apply to all intersections and cells within the cube, both for Aggregate and Detail level cells.

- **Aggregates And {}** - This qualifier indicates that LHS Range will apply only to intersections or cells that are on an Aggregate level.
- **Details And {}** - This qualifier indicates that LHS Range will apply only to intersections or cells which are on a Detail level, or the lowest level Members.

Right Hand Side (RHS) or CUBE Reference (where applicable)

```
"Cube Name".["Dimension Name.Member Name"]
```

"Cube Name".	This specifies the source Cube where the data or range from the RHS are coming from. For Internal Cube Formulas, the Cube name is the same as the name of the active Cube. For cross-cube formulas, the Cube name is that of a different Cube.
{"Dimension Name.Member Name"}	This specifies the specific Dimension and Member or range from within the source Cube.

Note: You can write a COMMENT within the Formula Editor, and these 'Comments' will be skipped or not read by Olation when parsing through the formula syntax.
To write a comment, use the symbols:

- **/*** - marks the start of the comment
- ***/** - marks the end of the comment
- **//** - indicates that the whole line will be read as a comment

7.1. Define the Cube Formula Statement

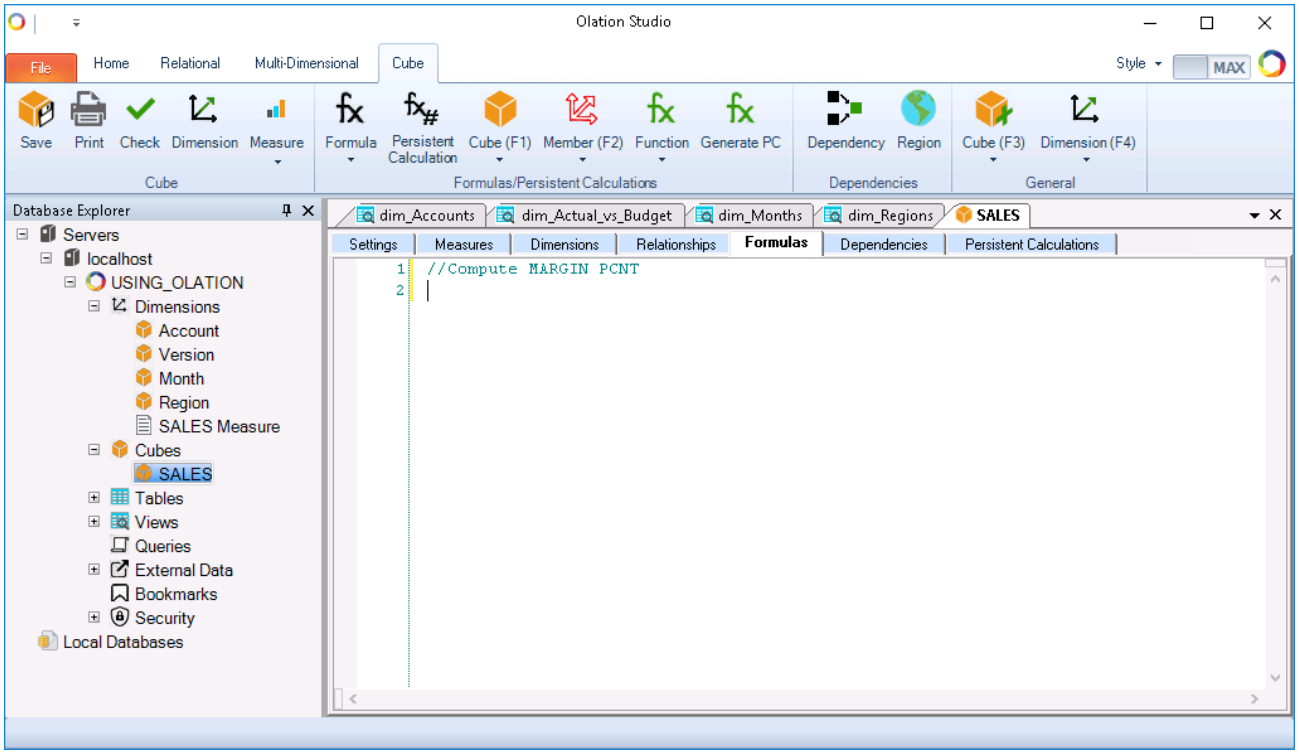
Note: As earlier mentioned, we will define an internal Cube formula. We will add this formula to the SALES Cube of the USING_OLATION database. The formula to compute for Margin Pcnt is:

```
MARGIN PCNT = Margin / Sales
```

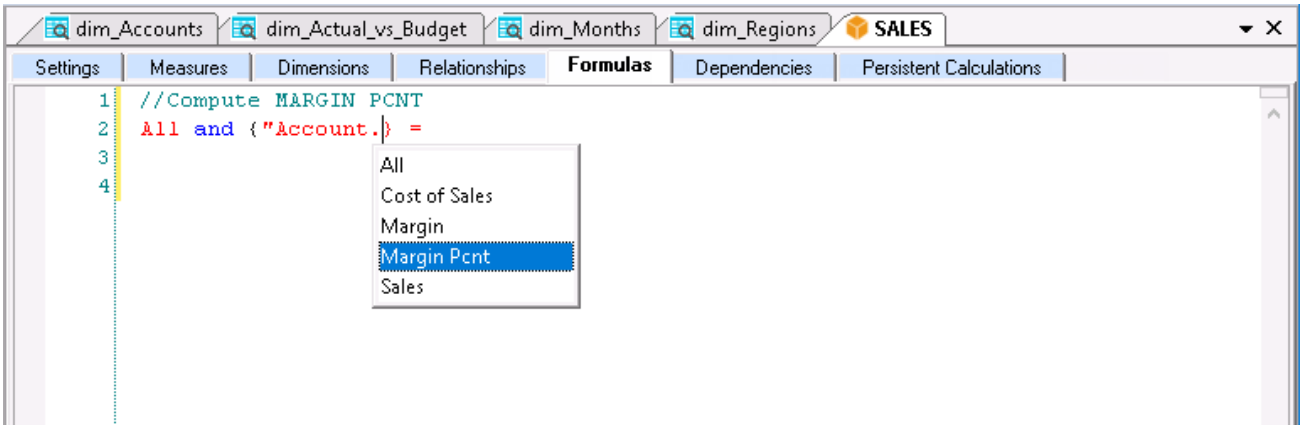
To define the formula:

1. Go back to the OLATION® Studio and, to edit the **SALES** cube, double-click on it.
   Note: Alternatively, you can also **right-click on the appropriate Cube** and select **Edit Cube**.

2. Go to the **Formulas Tab** (shown in the next image, with the Comment *//Compute MARGIN PCNT*)

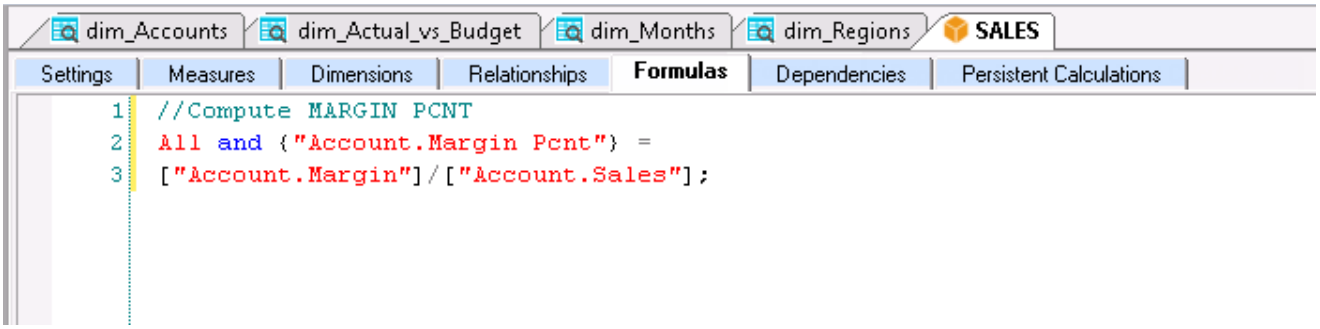


3. Write the **LHS Expression**.
   You can make use of the command buttons along the Cube Tab of the Olation ribbon in writing your formula: click the **Formula button** and select **All And {}** as the qualifier; double-click on **Account** as the Dimension, and; double-click on **Margin Pcnt** as Member.



4. Move your cursor after the **equals symbol** (**=**). Here you will define the RHS expression of your formula. Recall that the formula for computing *Margin Pcnt* is: **Margin / Sales**.

5. Write the **RHS Expression**. Again, you can make use of the command buttons along the Cube Tab, in this case the Cube button:
   - (On the **Cube button**) double-click on **SALES** as the Cube; double-click to select **Account** as the Dimension and double-click to select **Margin** as Member. This will be the numerator.
   - Outside the rightmost square bracket, enter the **forward slash symbol** (*/*) to denote division as the operation.
   - On the **Cube** button select **SALES** as the Cube; double-click to select **Account** as the Dimension and double-click to select **Sales** as Member. This will be our denominator.
6. Outside the rightmost square bracket, type a **semi-colon** (;) to indicate the end of the formula. The formula will appear as follows:



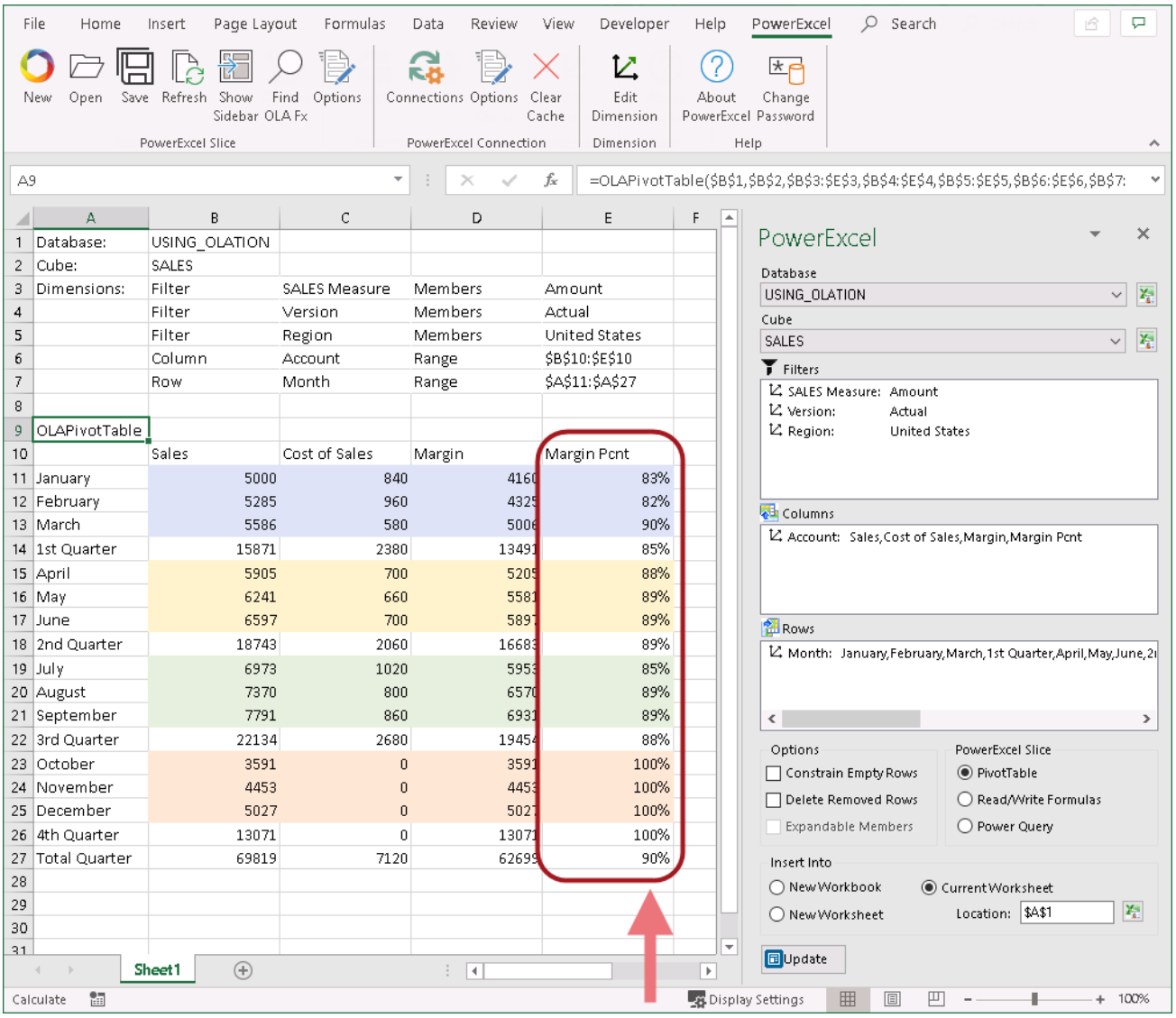
7. Click the **Check** or **Check Syntax button** (located along the Cube Tab of the Olation ribbon) to verify that the formula is free from syntax errors. You should get a prompt that says, 'The formula syntax is correct'.
8. Save the **Cube**.

7.2. Checking the Cube Formula Results

Having defined the Cube formula for the Account *Margin Pcnt*, we can check the results in any front end that provides a view of that account.

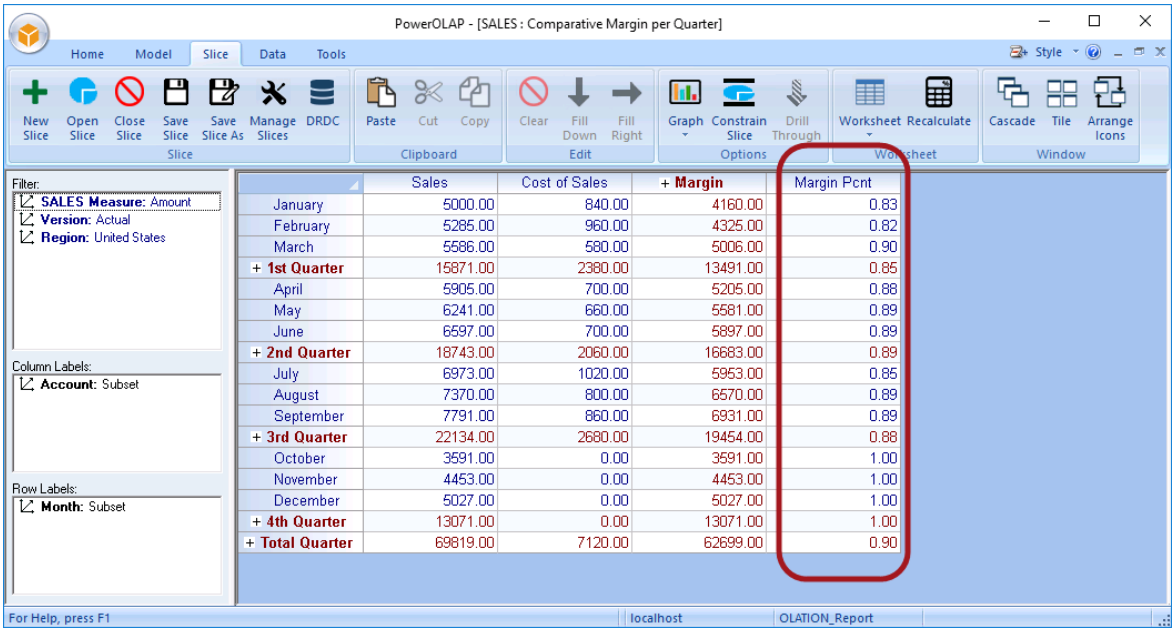
[**Note:** Make sure to refresh or recalculate your Slice or Excel view before seeing results.]

In the POWEREXCEL SLICE (see following image)



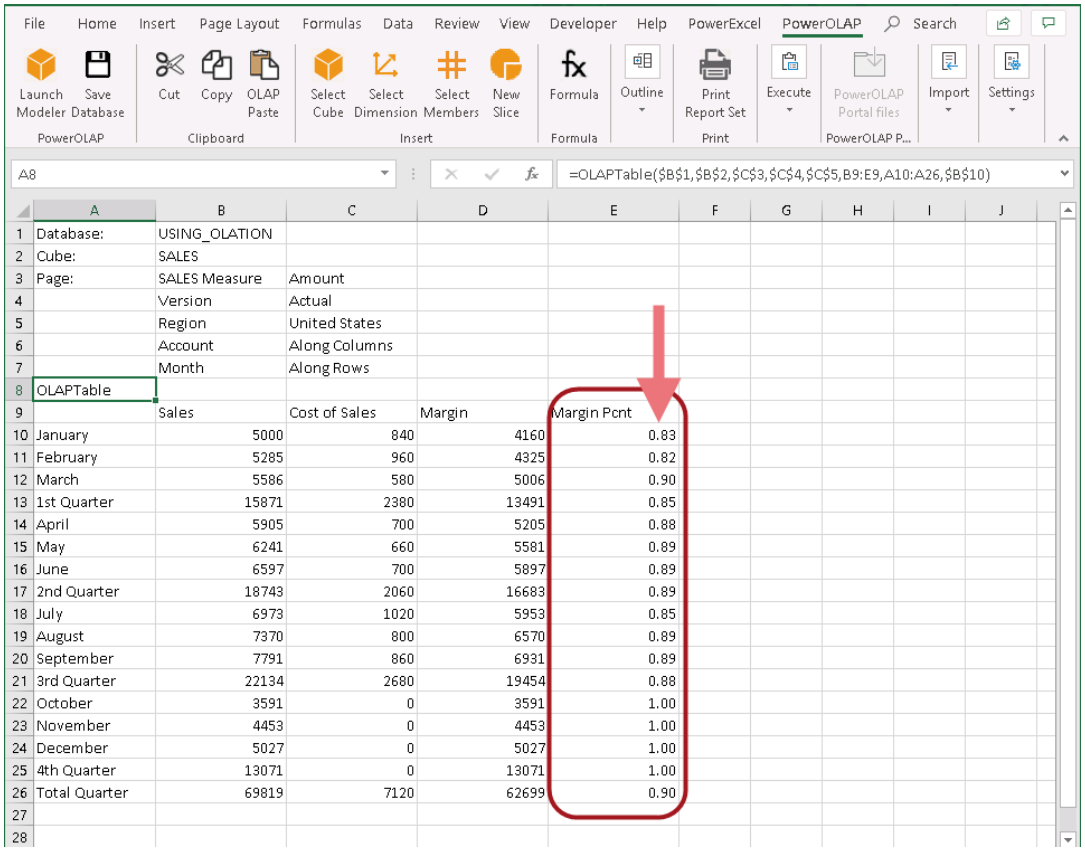
[Note: You can keep this Slice open or save it as an Excel file to your desktop, or wherever you wish. Further on in this manual we will want to use it to make a comparison to a Slice that we create in a new Cube.]

In the POWEROLAP SLICE (see following image)



A screenshot of the Comparative Margin per Quarter Slice
(the PowerOLAP Slice we created in earlier topics)

In the POWEROLAP SLICE TO EXCEL



8. Create the PRODUCT_SALES Cube

In this section, the objective will be to create a 5-dimensional *PRODUCT_SALES* Cube composed of the four relational-table – based Dimensions created previously (*Account*, *Version*, *Month*, *Region*) and a new *Product* dimension. The new Dimension will *not* be based on a pre-existing relational table, pointing up Olation's ability to add new/independent "meta data" logic to a model; we will reference this "non-relational – based" dimension as being "multidimensional" in nature. [It is worth noting that, in the end—as will be shown—the creation of the "multidimensional" data will result Olation creating relational tables; so, in effect, these dimensions do have a consequent basis in relational table logic.]

We will also demonstrate the use of a Cross-Cube Formula in the second Cube (PRODUCT_SALES) that will pull in fact data from the Cube created previously (*SALES* Cube).

Important: Please note that specific figures may be different from the data set you are working on. This exercise is intended to server as a guide to help you understand how to use OLATION so that you can work with your own data set.

8.1. Pre-Work: Prepare the Product Factdata SQL Table

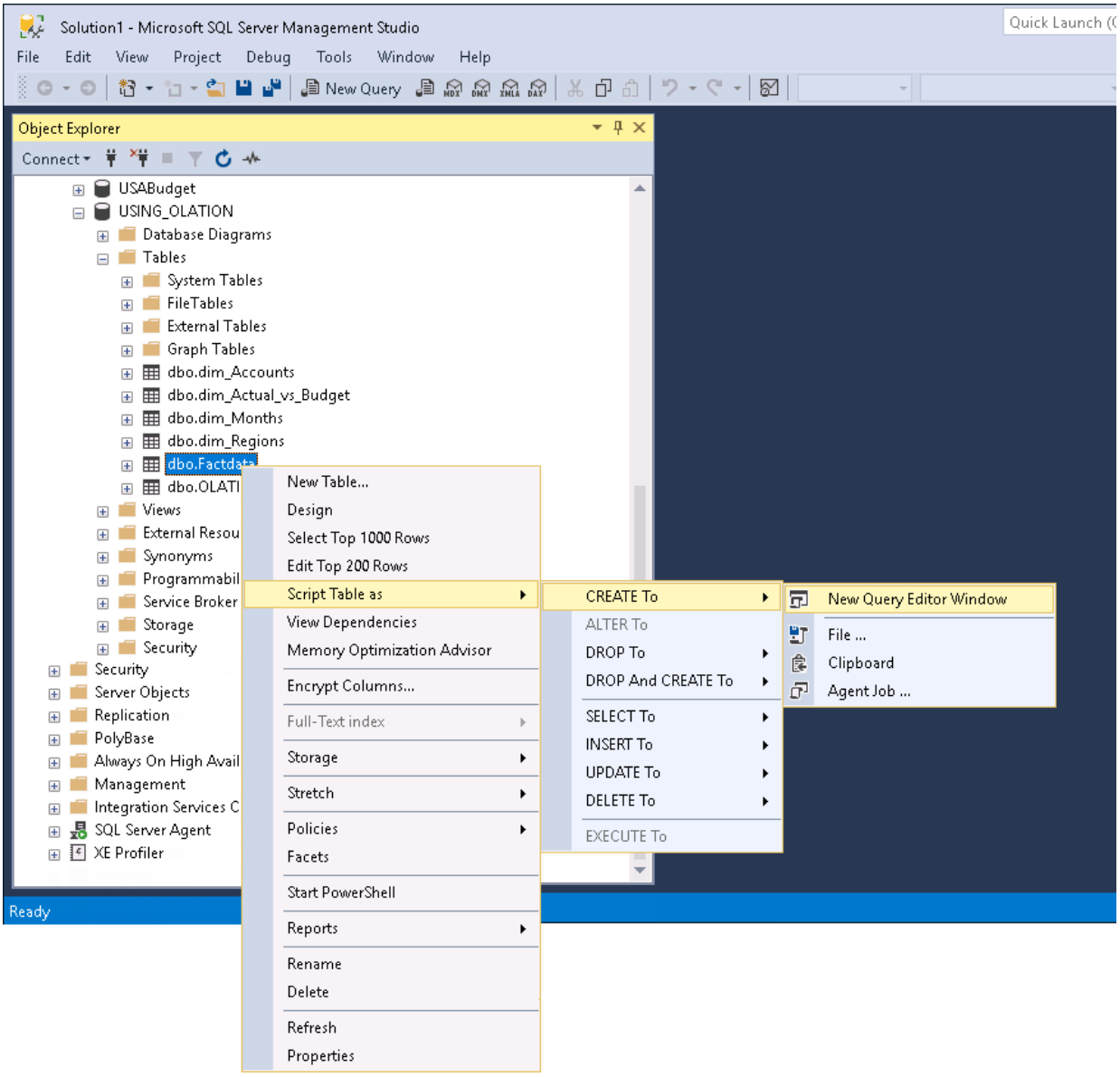
Before proceeding to create the new *PRODUCT_SALES* Cube, we will first need to prepare a necessary SQL table, i.e., a *dbo.Product_Factdata* SQL table. We need this table to be able to define the relationship between all our Dimension and a Measures table.

Note: If the Product_Factdata table is not yet created into the USING_OLATION SQL Database, you need to go to the SQL Server Management Studio and create that table from there. Then make sure to modify this factdata table to include an additional *Products column*.

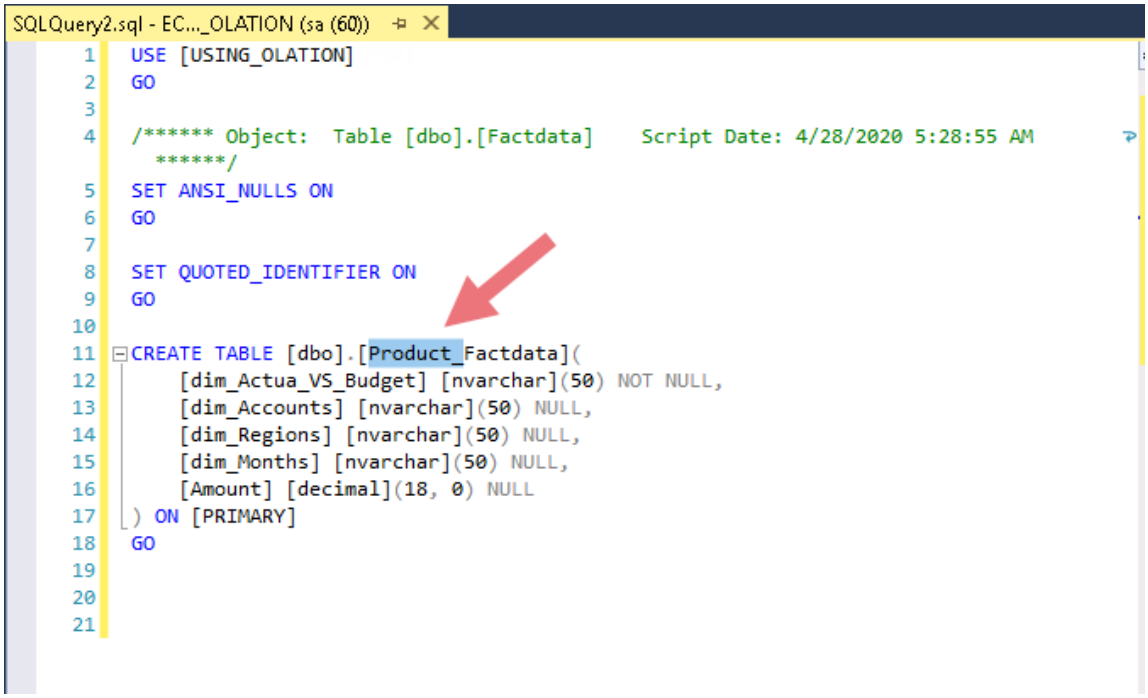
8.1.1. Create the Product_Factdata SQL Table in SQL Server Studio

To create the Product_Factdata SQL table (a copy of the Factdata SQL table):

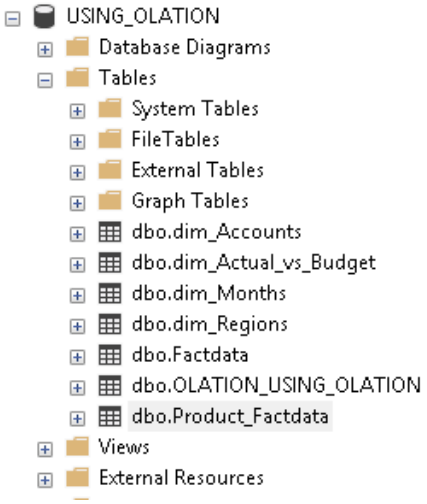
1. Launch **SQL Server Management Studio**.
2. In SQL Server's Object Explorer pane, expand **Databases** and locate the source Olation database, i.e., **USING_OLATION**.
3. Expand the **USING_OLATION** database, expand **Tables** and locate the **dbo.Factdata** table.
4. Right-click on the **dbo.Factdata** table, select **Script Table as**, **CREATE To** and choose **New Query Editor Window** (see next image).



5. In the New Query Editor Window that appears, **edit the script** by changing the name of the *dbo.Factdata* table to **dbo.Product_Factdata** as shown in the following image.
   The highlighted text in blue is typed in to rename the table name (see where arrow points).



6. Click **Execute button** or press the **F5** key to execute the query.
   You should see a Message that says 'Commands completed successfully'.
7. Refresh the **USING_OLATION** SQL database.
   You will see the *dbo.Product_Factdata* table in the Object Explorer pane of the SQL database, as shown in the following image.



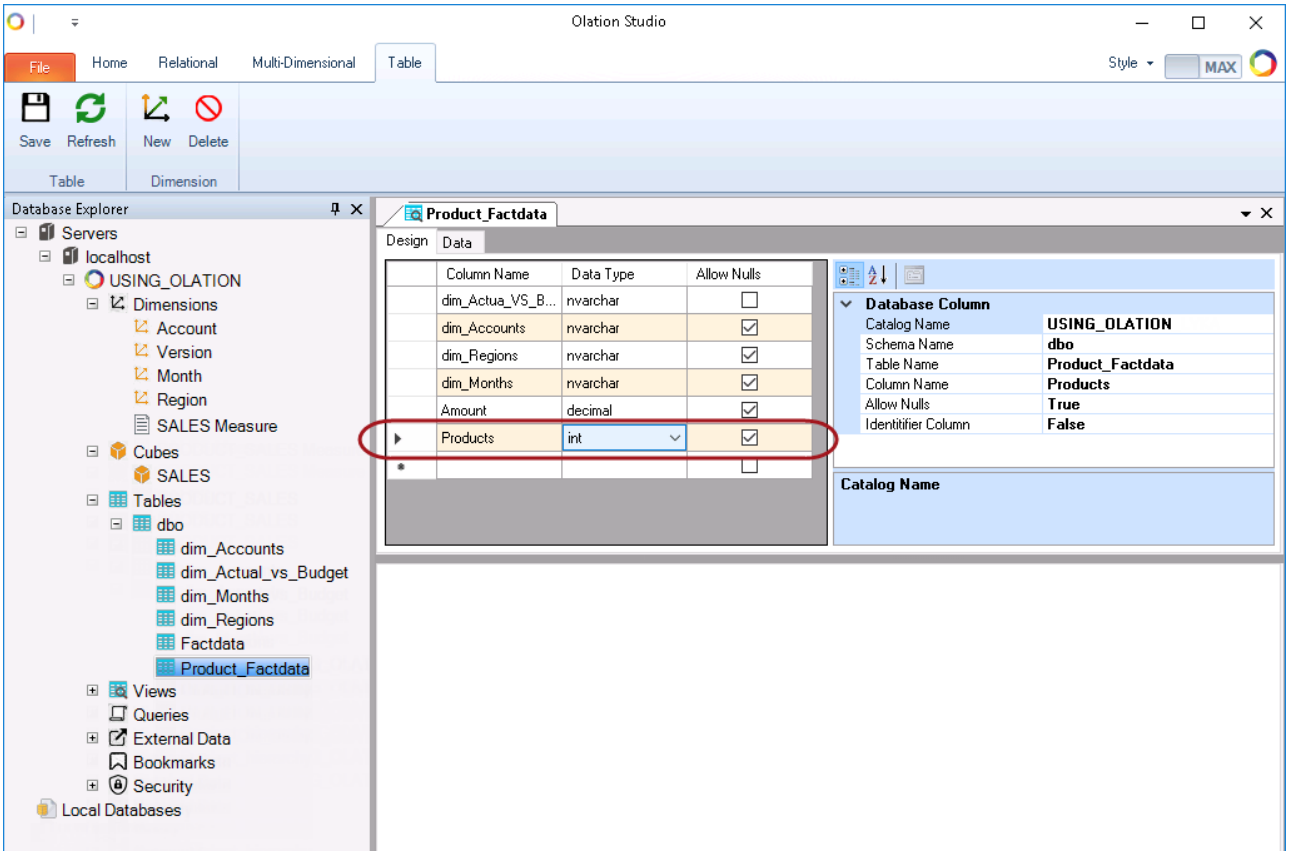
8.1.2. Modify the Product_Factdata SQL Table to Include Products Column

There are several ways to add a new column into an SQL database table, commonly performed from SQL Server Management Studio.

However, in this exercise, we will demonstrate how to do it from the Olation Studio.

To do this:

1. Launch OLATION® Studio and open the correct database (**USING_OLATION**).
2. In Database Explorer, expand the **USING_OLATION** database, expand **Tables**, then locate and right-click on the **Product_Factdata** table.
   Note: If this database was already open when you created the dbo.Product_Factdata table, simply refresh the Tables in Olation to ensure that all tables are shown.
3. Right-click on **Product_Factdata** and select **Edit Table**.
   A Table view of the *Product_Factdata* table will appear on the right.
4. Go to the last row and in the Column Name field, type **Products,** and for the Data Type click on the drop-down and select **int** (abbreviation for integer)—see next image.



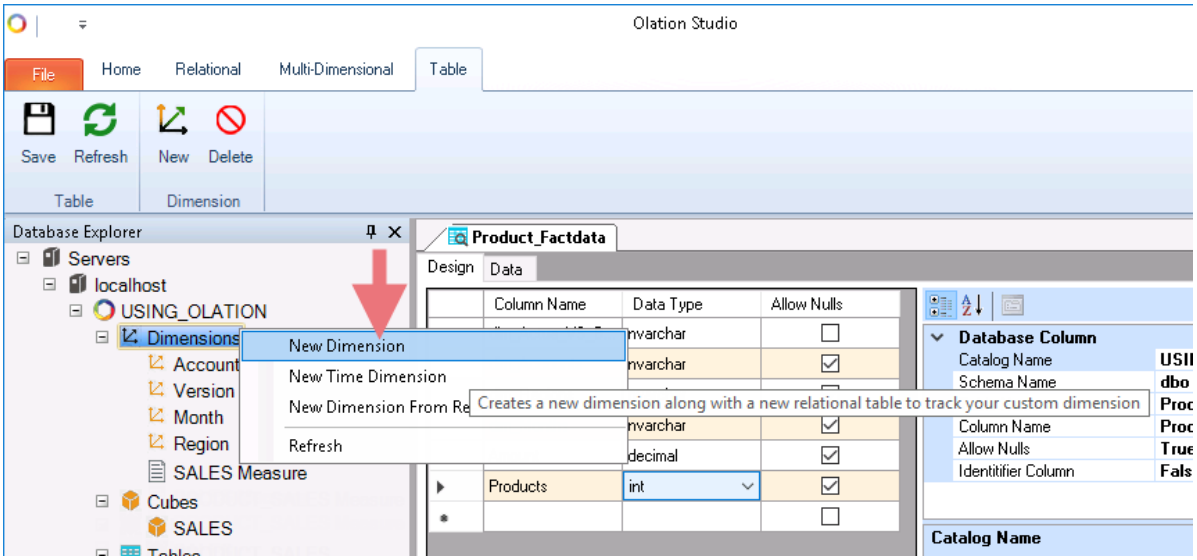
5. Click the **Save icon** in the **Table Tab** of the Olation ribbon.
   The *Product_Factdata* table has now been modified to include the additional *Products column* that we need.

8.2. Create a Multidimensional Dimension - *Product* Dimension

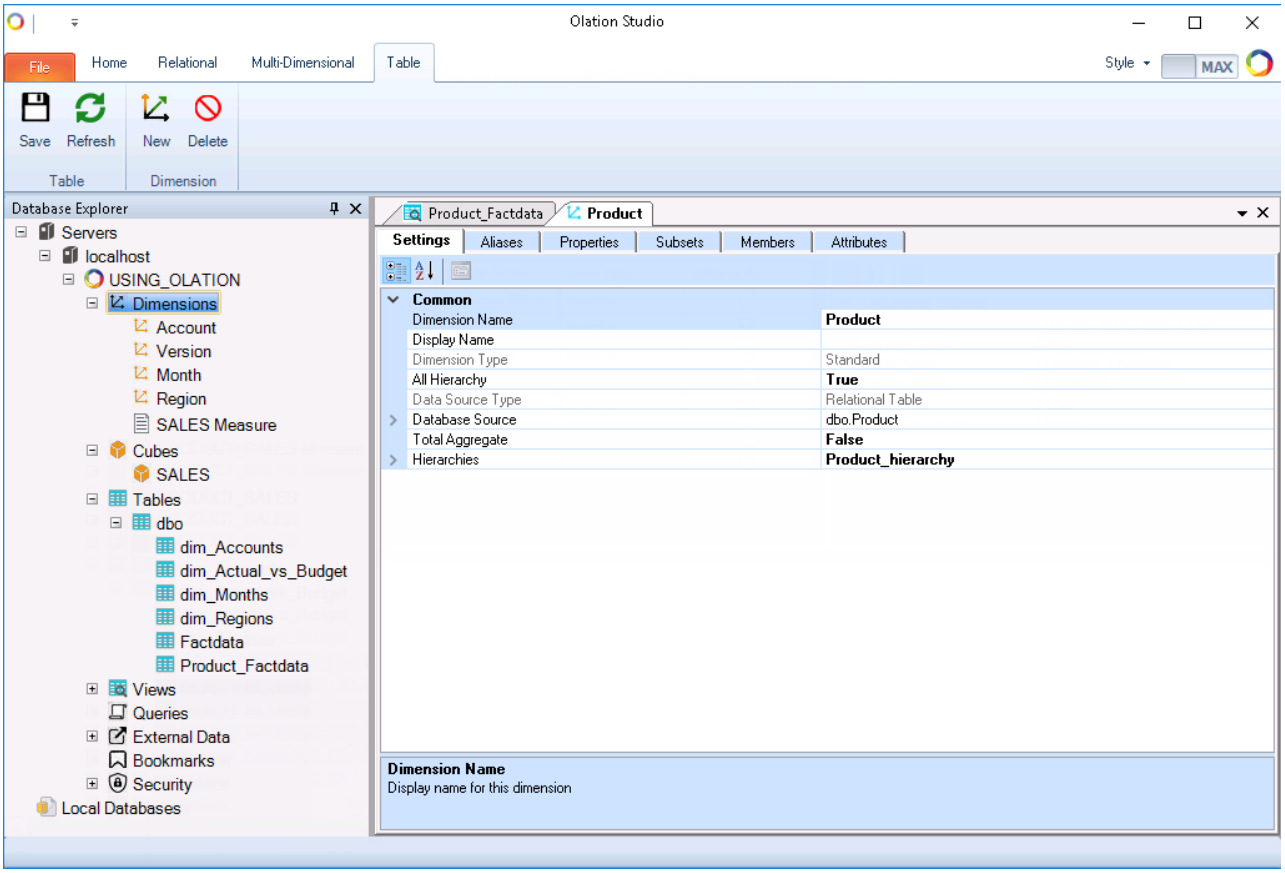
As previously mentioned, we will use the same Dimensions we created in Exercise 1 for our new Cube, i.e., the Dimensions *Account*, *Version*, *Month* and *Region*. We will only need to create an additional Dimension, a multidimensional Dimension type, called **Product**.

To create the *Product* Dimension:

1. In OLATION, expand the **USING_OLATION** database; right-click on **Dimensions** and select **New Dimension**.



2. In the **Create New Dimension** dialog, type **Product,** and click **Create**.
The Dimension Editing window for the Product dimension appears on the right.



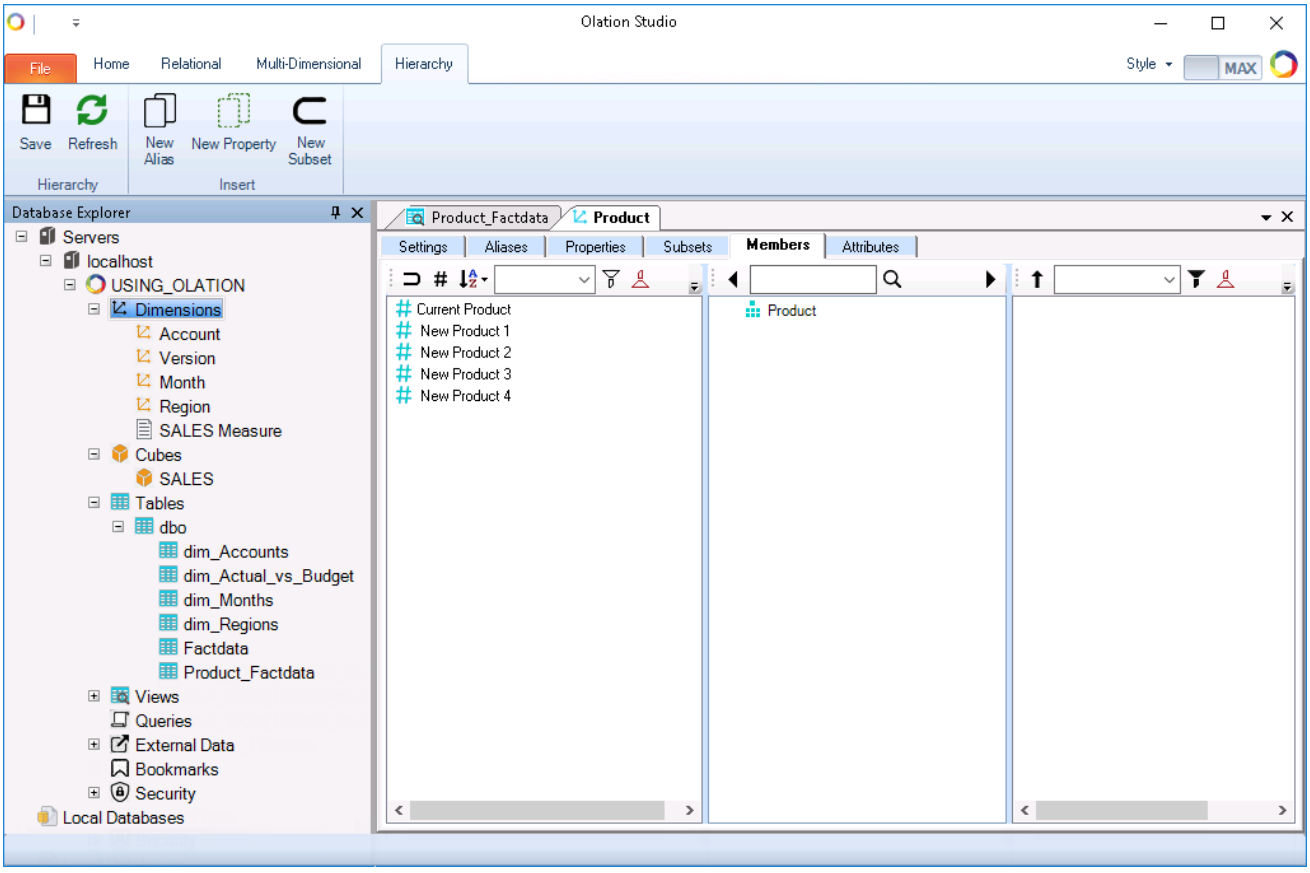
8.2.1. Add Dimension Members

We will begin by adding the Members to the Product dimension. To add Members to the Product dimension:

1. Go to the **Members Tab**. Here, we will be adding the following Members:

   MEMBERS:
   Current Product
   New Product 1
   New Product 2
   New Product 3
   New Product 4

2. In the **Member List pane** (left-hand pane), click **Add Member icon** (# icon). In the textbox that appears, type the <**member name**>, i.e., **Current Product**, then press the **Enter** key. **Note:** You can keep adding Members by clicking on the **Add Member icon** and typing in the Member name; alternatively, you can press **CTRL**+**Enter** keys after adding a Member. When completed, the Members Tab will look as follows:



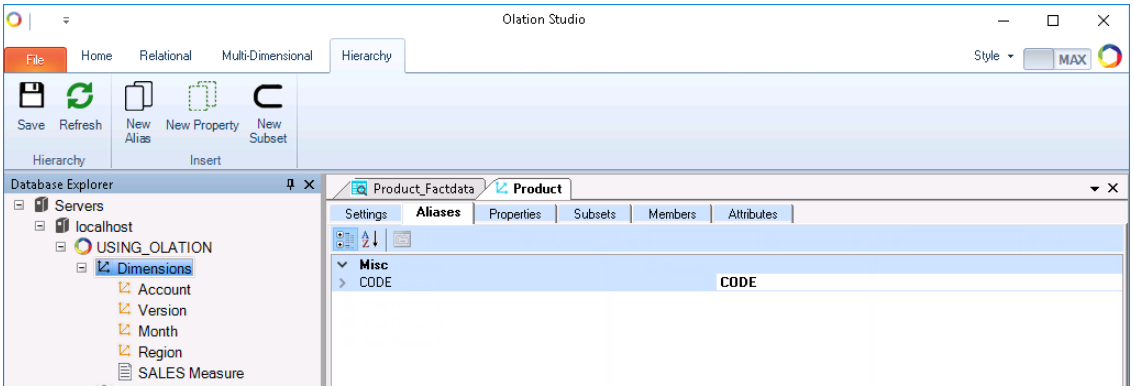
Next, you will define additional Dimension attributes, namely:

- **ALIAS**
  You will create an Alias Group called *Codes*, to assign Product Codes for each Member.

- **PROPERTIES**
  You will classify each product item according to whether it is sold in a specific season (Season) or whether it is sold regularly, regardless of season (All-year-round).
- **SUBSET**
  You will create a Subset group that includes only New Products.

8.2.2. Define Aliases

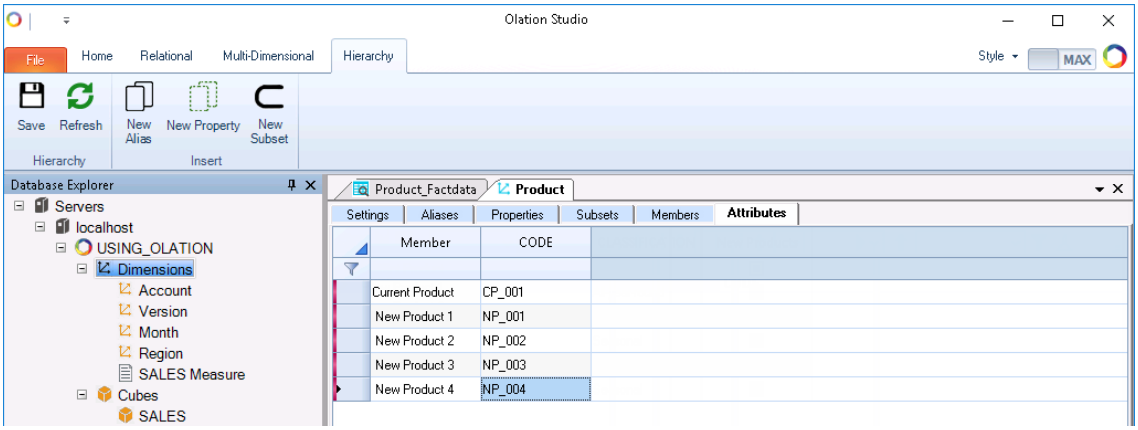
1. In the Dimension Editing Window for *Product*, go to the **Aliases Tab**.
   Note: Before you can define Aliases, you must first create the Alias Group.
2. Right-click on the Aliases window and select **New Alias**.
3. In the Create New Alias dialog, type the <**alias group name**>–in this example, **CODE**.
4. Click **Create**. The Alias group *CODE* appears in this window, as in the following image.



Next, define Aliases and assign them to the respective Members in the *Product* Dimension.

Note: It is important to note that Aliases, like Members, must be unique within the Dimension. So, once you assign an Alias to a Member, you cannot assign it to any other Member within that Dimension.

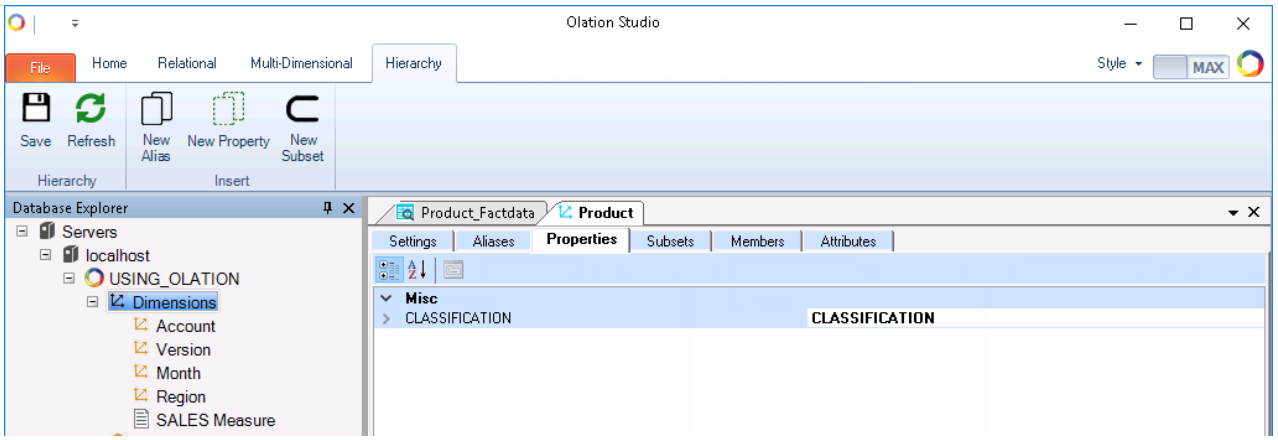
5. Go to the **Attributes Tab**.
6. In the column *CODE*, create the Aliases for each Member, as shown in the following image:



8.2.3. Define Properties

Next, define Properties for product items that identify the type of season each product sells in.

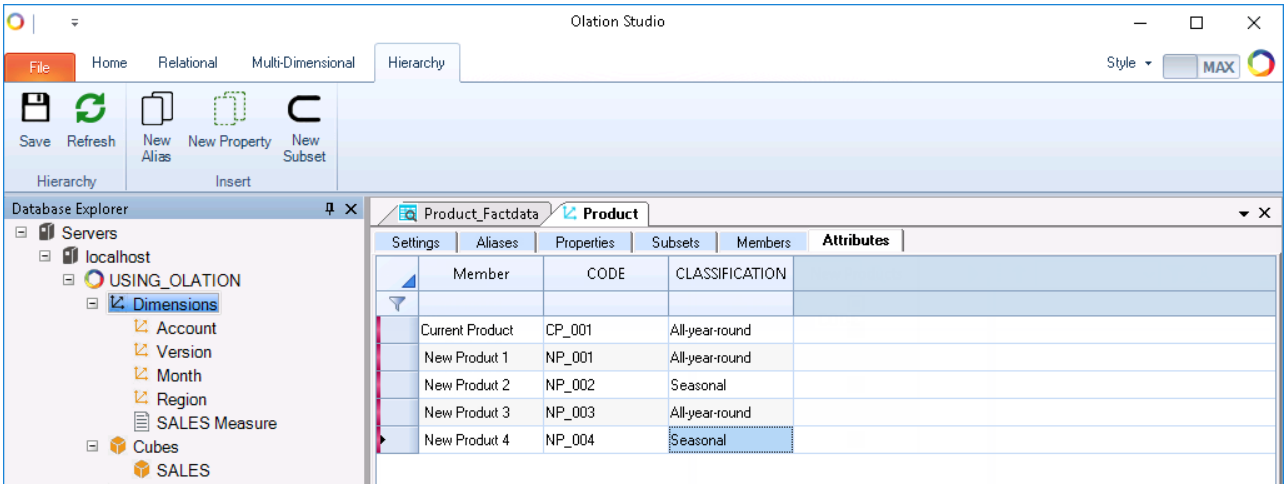
1. In the Dimension Editing Window for the *Product* dimension, go to the **Properties Tab**.
   Note: Before you can define Properties, you must first create the Property Group.
2. Right-click on the Properties window and select **New Property**.
3. In the Create New Property dialog that appears, type the <**property group name**>--for this example, type **CLASSIFICATION**.
4. Click **Create**. The Property group called CLASSIFICATION appears listed in this window.



Next, define the Property and assign them to respective Members within the *Product* dimension.

Note: Unlike Aliases, a single Property can be assigned to multiple Dimension members since it is only used for annotation purposes.

5. Go to the **Attributes** Tab.
6. In the Column CLASSIFICATION, create the Property for each Member, as shown in the following image:

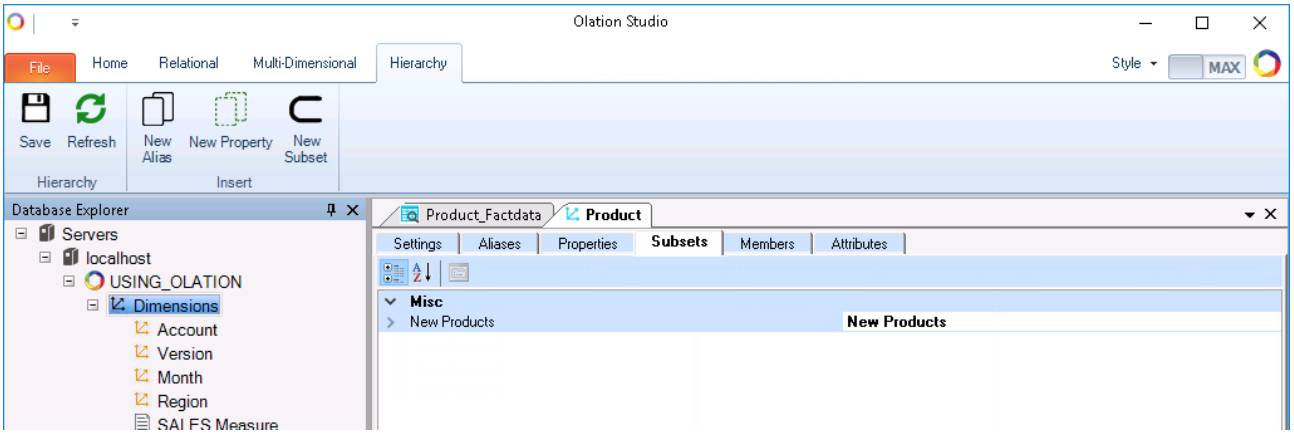


8.2.4. Define a Subset

Subsets are particularly helpful when you often use a specific set of Dimension Members for reporting, analytical and planning activities. Instead of repetitively picking these Members each time you create a new view of the data, you can select a Subset—the custom set of Members—and obtain results for its Members, all in a group, immediately.

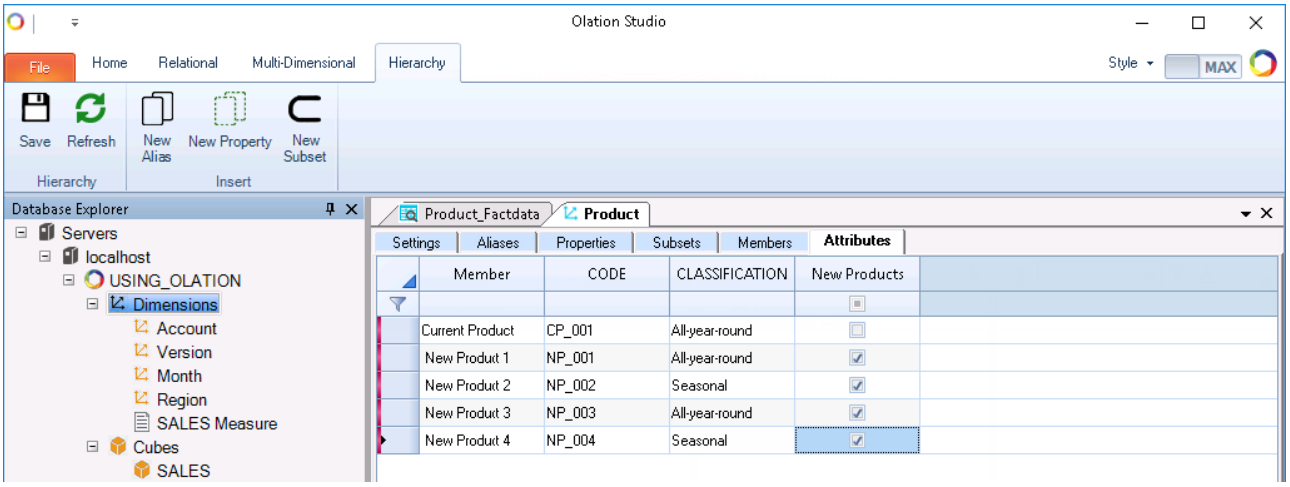
For this exercise, we will create a subset group called *New Products*.

1. In the Dimension Editing Window for the *Product* dimension, go to the **Subsets Tab**.
   Note: You need to create a Subset Group first, then, in the Attributes Tab, select the Members that will comprise the Subset.
2. Right-click on the Subsets window and select **New Subset**.
3. In the Create New Subset dialog that appears, type the <**subset group name**>-- for this exercise, type **New Products**.
4. Click **Create**. The Subset group *New Products* now appears listed in this window:



Next, select the Members for this Subset group:

5. Go to the **Attributes** Tab. Tick the corresponding checkboxes of Members to be included in the Subset group. In this case, check the boxes corresponding to **New Product 1** to **New Product 4**.
   The Attributes Tab will look as follows:



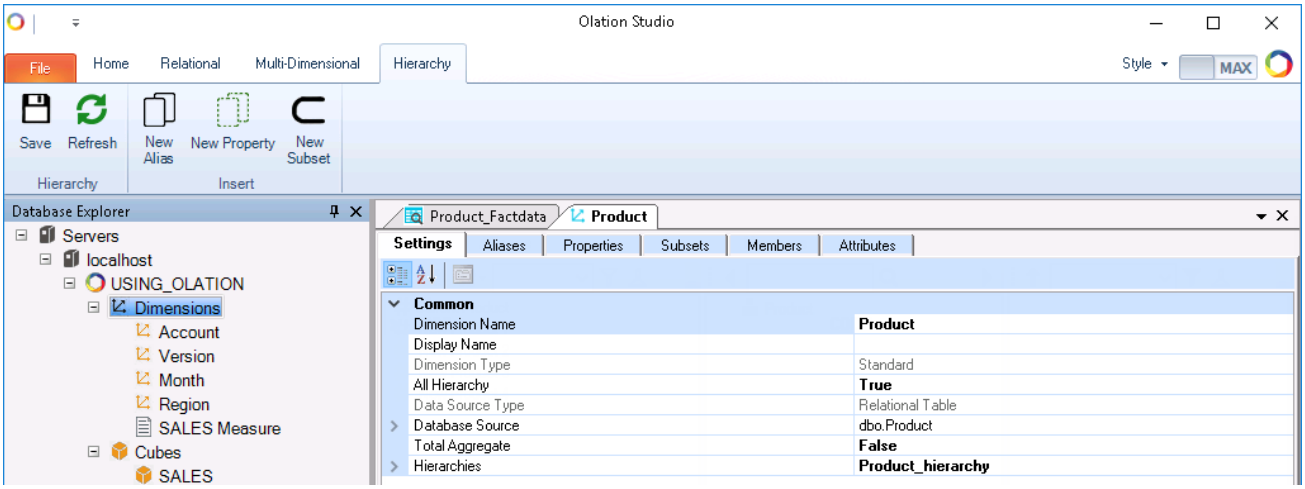
8.2.5. Configure Other Dimension Settings - Settings Tab

For this exercise, we will enable the All Hierarchy option and disable and Total Aggregate option.

1. In the Dimension Editing Window for *Product* Dimension, go to the **Settings Tab**.
2. Locate the **All Hierarchy** option and set to **True**.

   Locate the **Total Aggregate** option and set to **False**.

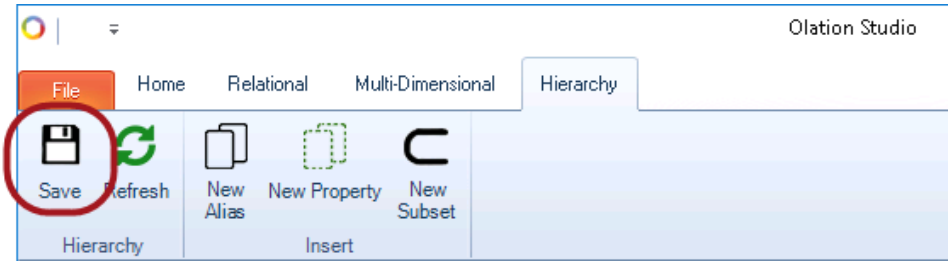
When completed, the Settings Tab will look as follows:



8.2.6. Save the Dimension

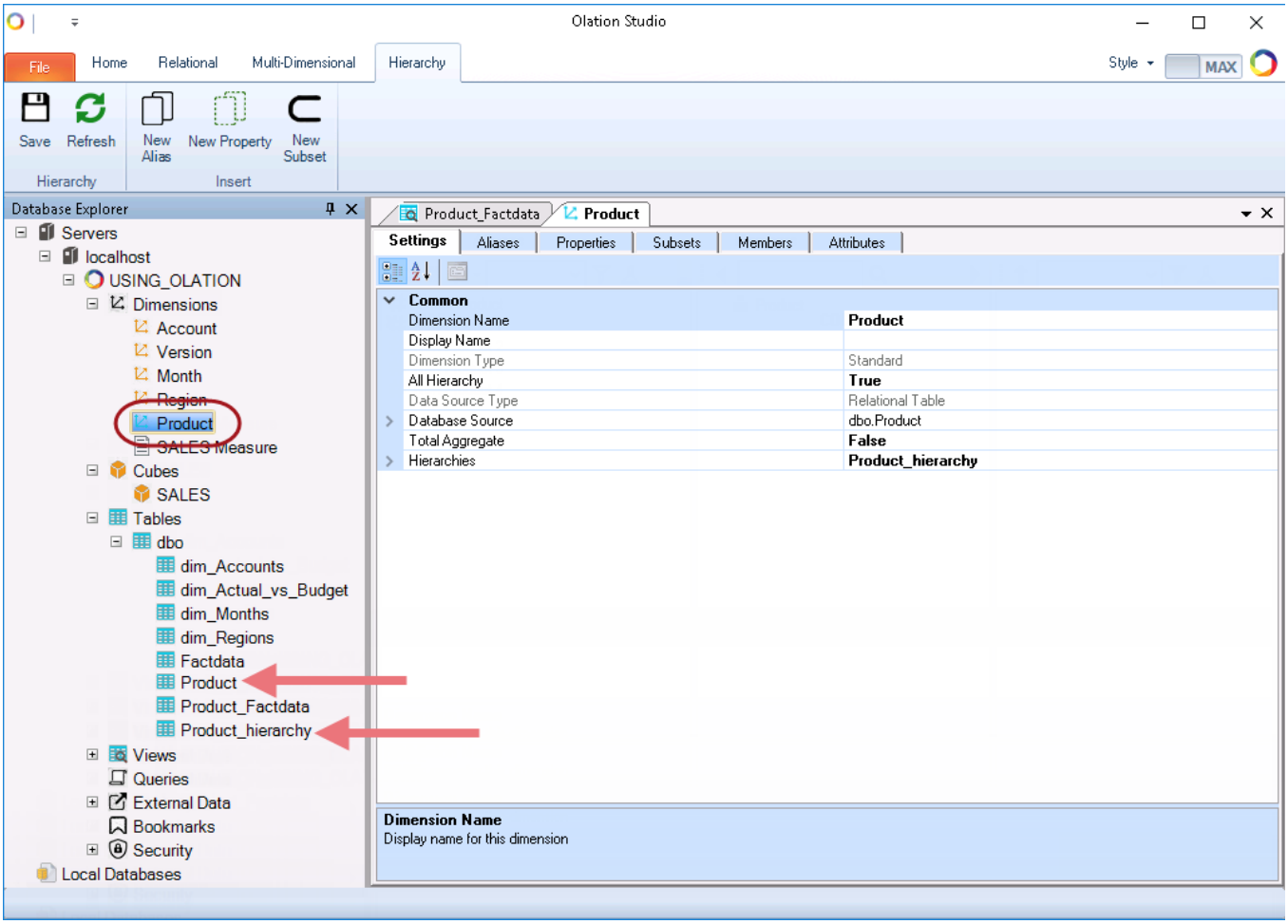
After completing the creation of the Dimensions and defining its attributes:

1. Go to the **Hierarchy Tab** on the Olation ribbon.
2. Click the **Save** command.



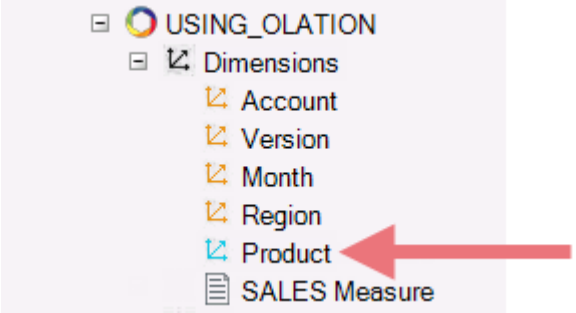
In the Database Explorer pane, note the newly created Dimension, i.e., **Product** (see circled in next image).

Refresh Tables: note also the two additional tables: **Product** and **Product_hierarchy** (see arrows in the next image. Indeed, Olation creates these tables as a result of adding them as (multidimensional) Dimensions in Olation Studio.



Notice (following image) finally that the icon indicator for the *Product* dimension is blue, which is different from the other Dimensions (they are orange). This is indicative of the nature of the

Dimension. Whereas the first four Dimensions were created from a relational source (relational table), the *Product* dimension is of a multidimensional data type.

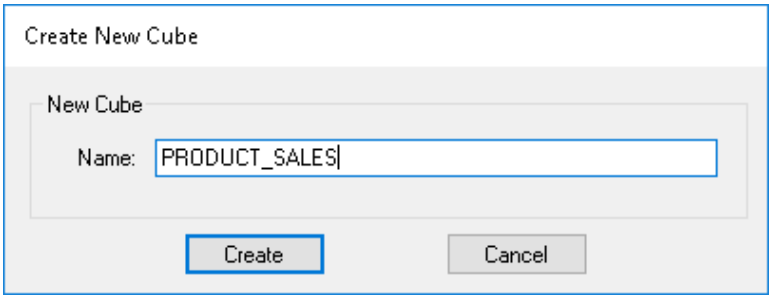


8.3. Create the PRODUCT_SALES Cube

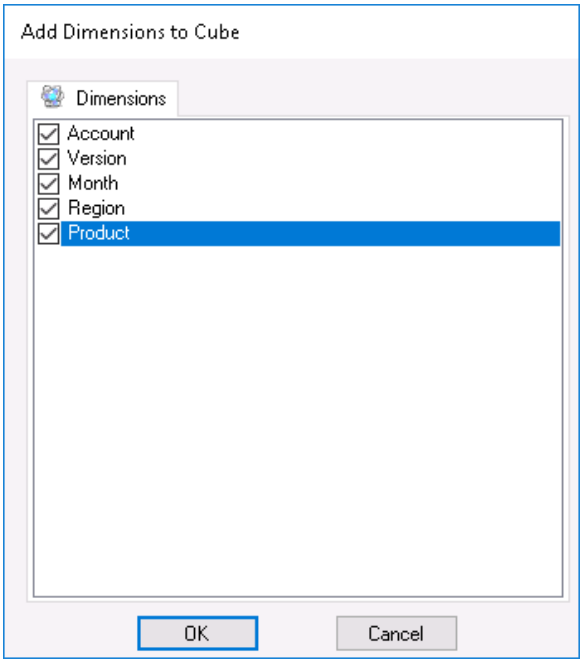
With five Dimensions now existing in Olation—and having created the *Product_Factdata* table in SQL Server, we are now ready to create the PRODUCT_SALES Cube. (Note that if you view the Product_Factdata table in SQL Studio, you will see that it does not have any records, i.e., no transactional values.)

To create the PRODUCT_SALES Cube:

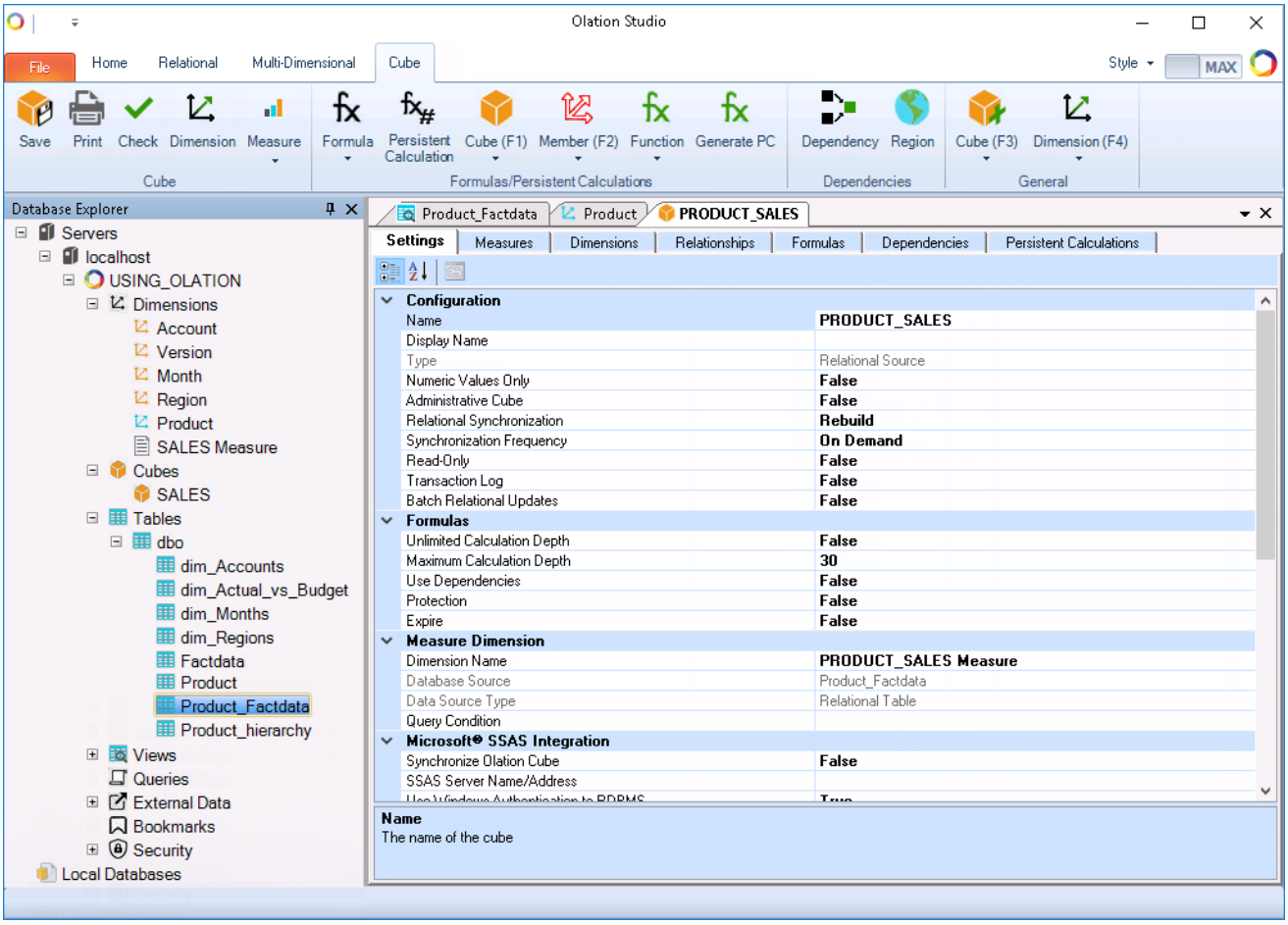
1. Under **Tables** in Database Explorer: right-click on **Product_Factdata** and select **New Cube**. The **Create New Cube** dialog box appears.
2. In the dialog box, type a <**name for your cube**>.
   For this exercise, type **PRODUCT_SALES** as the Cube name.



3. Click **Create**. The **Add Dimensions to Cube** dialog box appears.

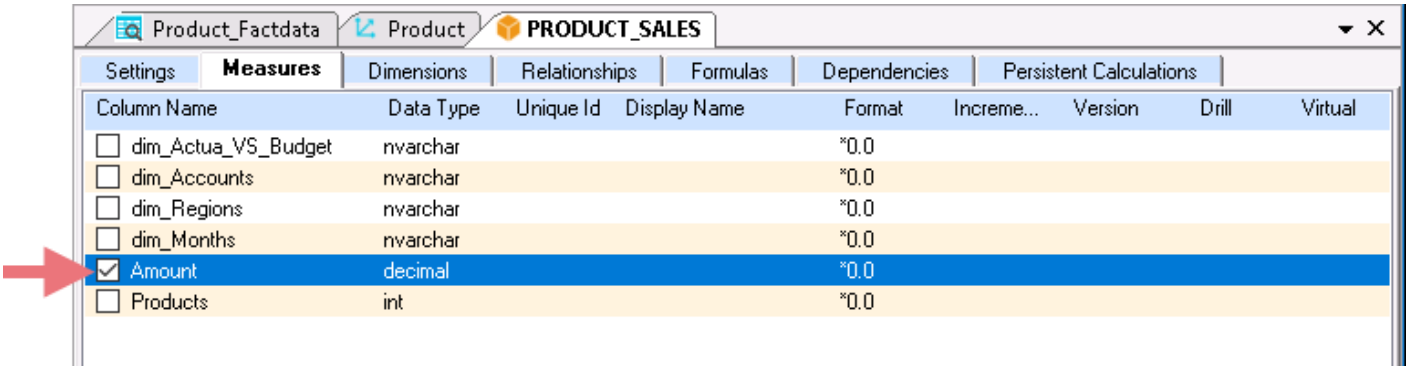


4. Select the Dimensions to be included into the Cube by checking the corresponding checkboxes. (In this example select all five dimensions: **Account**, **Version**, **Month**, **Region** and **Product**, as shown in the image above.)
5. Click **OK**. This opens the **Cube Definition Window** on the right.



8.3.1. **Assign the Measure Member**

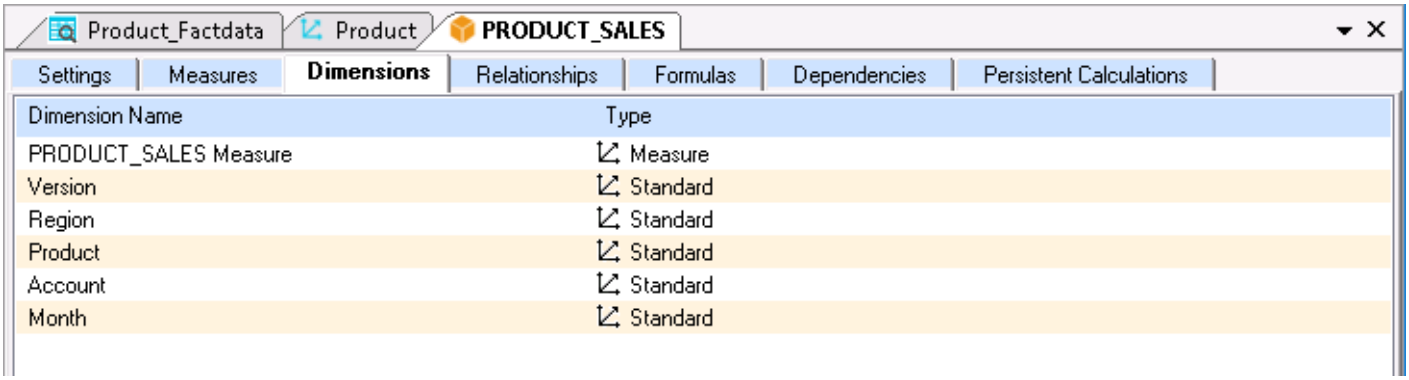
1. In the **PRODUCT_SALES Cube Definition Window**, go to the **Measures** tab.
2. Select from the list which item(s) you want to designate as Measures. For this exercise, check the **Amount** checkbox.



8.3.2. Arrange the Dimension Order

To define the order of the Dimensions:

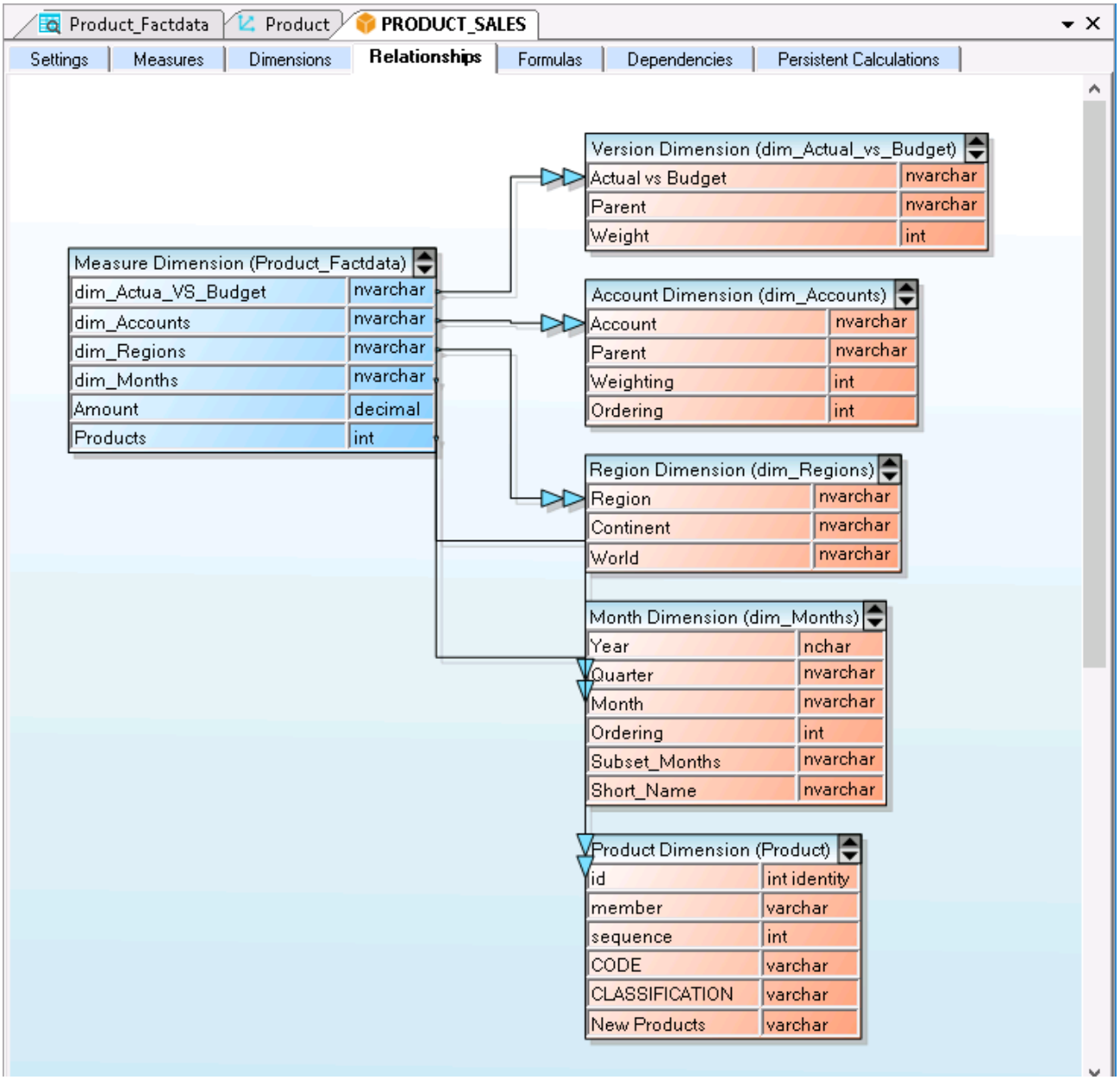
1. In the **Sales Cube Definition Window**, go to the **Dimensions** Tab.
2. Drag and drop the Dimensions in the following order as shown in the image below:



8.3.3. Define Relationships

These steps will be similar to how relationships were defined in the SALES Cube. The only difference is the need to link an additional Dimension Table (i.e., *Product* Dimension) to the Measure Table—thus, go to the **Relationships Tab** and link the five (5) Dimension Tables (*Account*, *Version*, *Month*, *Region* and *Product*) to the Measure Table (*Product_Factdata*):

1. Click on **dim_Actual_Vs_Budget** in the ***Measure Dimension*** table on the left; drag and drop to **Actual vs Budget** of the ***Version Dimension*** table on the right.
2. Click on **dim_Accounts** in the ***Measure Dimension*** table; drag and drop to **Account** in the ***Account Dimension*** table on the right.
3. Click on **dim_Regions** in the ***Measure Dimension*** table; drag and drop to **Region** column in the ***Region Dimension*** table on the right.
4. Click on **dim_Months** in the ***Measure Dimension*** table on the left; drag and drop to **Month** in the ***Month Dimension*** table on the right.
5. Click on **Products** in the ***Measure Dimension*** table; drag and drop to the **id** column in the ***Product Dimension*** table on the right.



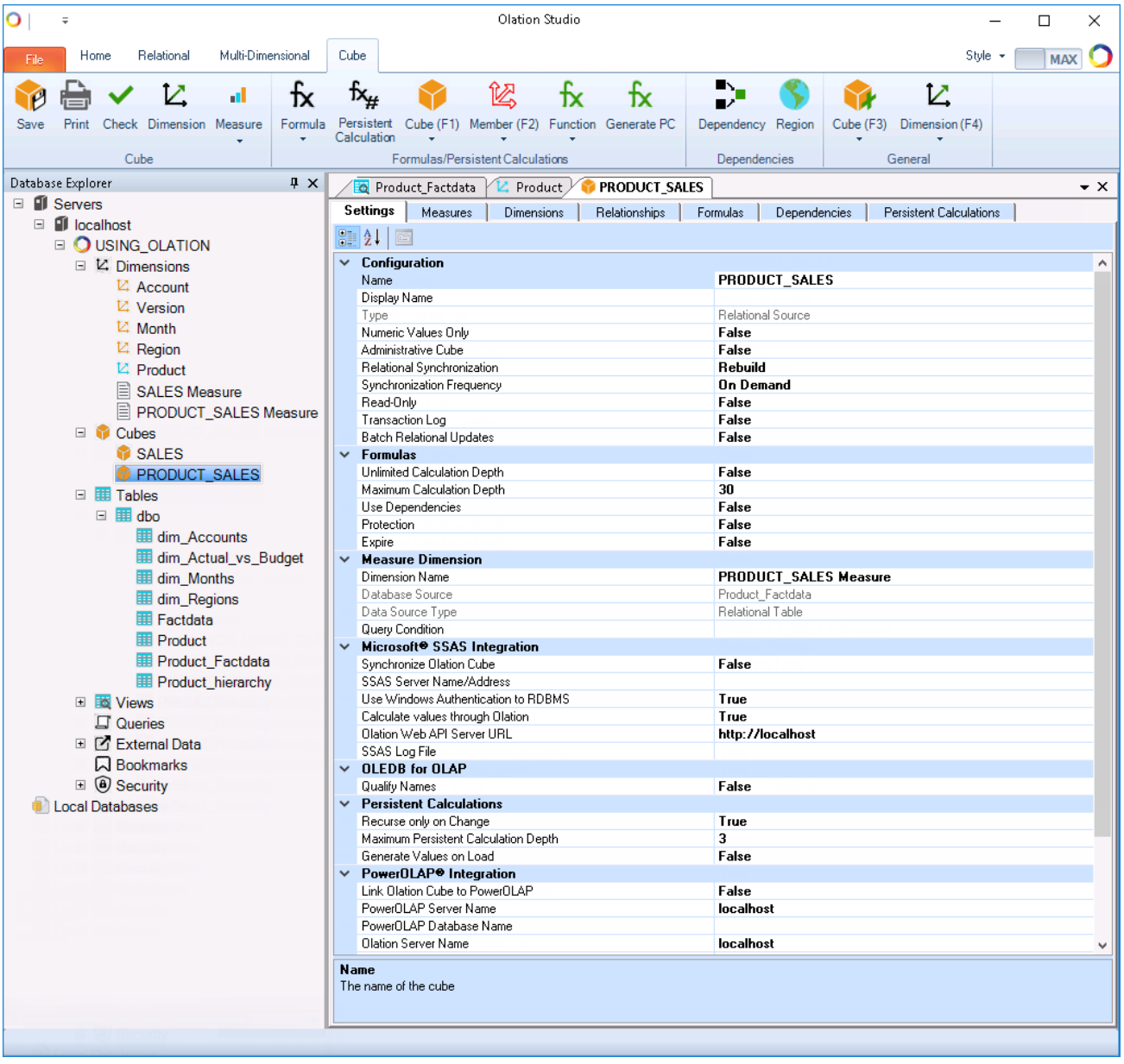
8.3.4. Define Cube Setting: Disable READ-ONLY Setting

For this new Cube—as with the *SALES* cube—we want to configure it to have write-back capability, so we will disable the Read-Only setting. To do this:

1. In the **PRODUCT_Sales Cube Definition Window**, go to the **Settings** tab.
2. In the Configuration section, locate the **Read-Only setting**, and set this to **False**.

8.3.5. **Save the Cube**

1. Go to the **Cube Tab** of the Olation ribbon.
2. Click the **Save Cube** icon.
   Note: The Cube Tab becomes visible when you bring up the **Cube Definition Window**.
   The **PROUDCT_SALES** Cube now appears listed under Cubes in Database Explorer.

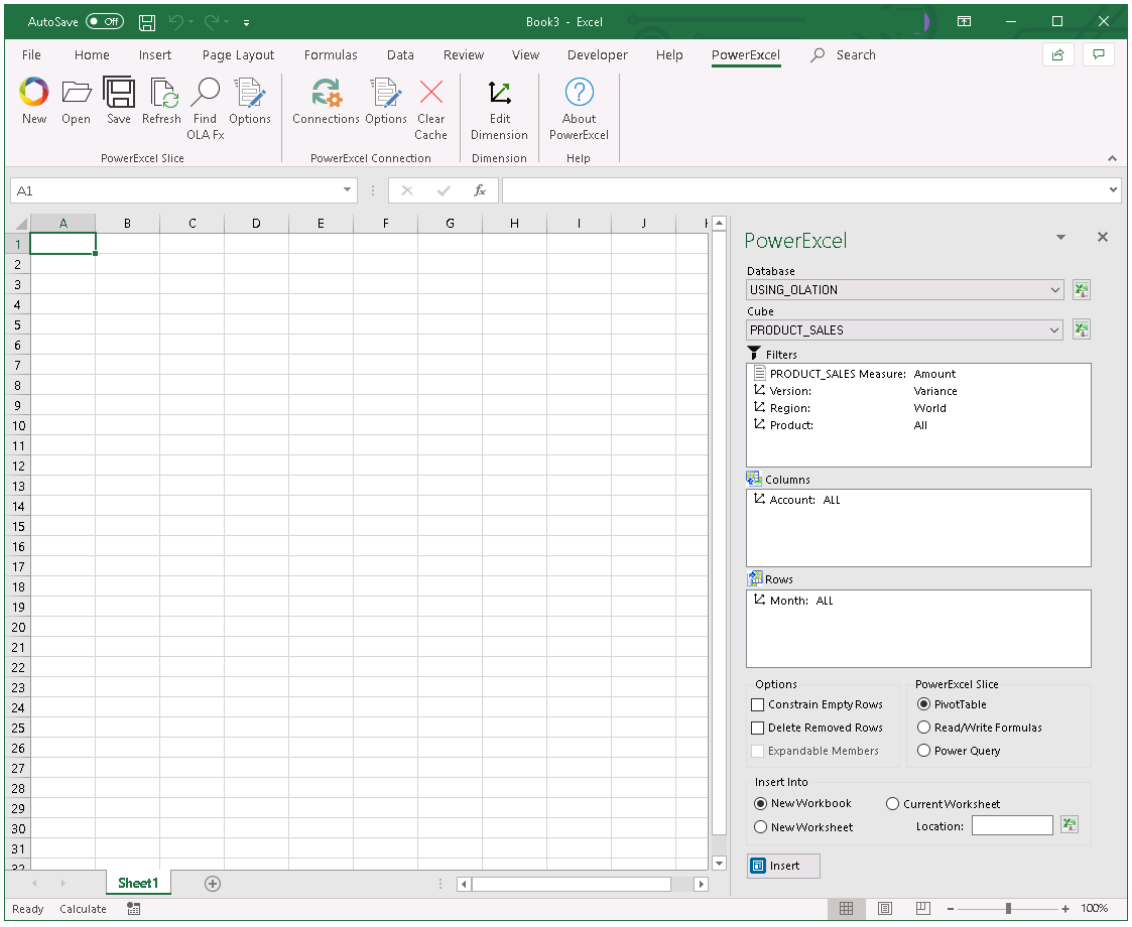


8.4. View the PRODUCT_SALES Cube

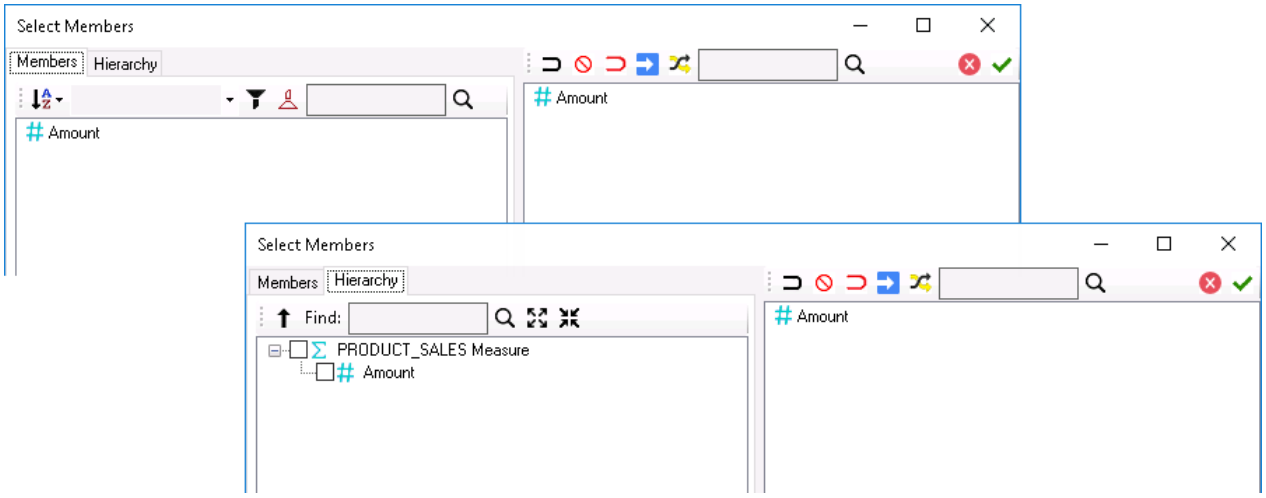
We will next explore the *PRODUCT_SALES* cube to view the metadata structure that we just built. Remember that there is NO FACT DATA yet for this Cube since the relational table from which it was created from (**dbo.Product_Factdata**) has no transactional values/records. Once again, for this exercise, we will view this (second) Cube using PowerExcel as the front-end client.

To view the PRODUCT_SALES Cube:

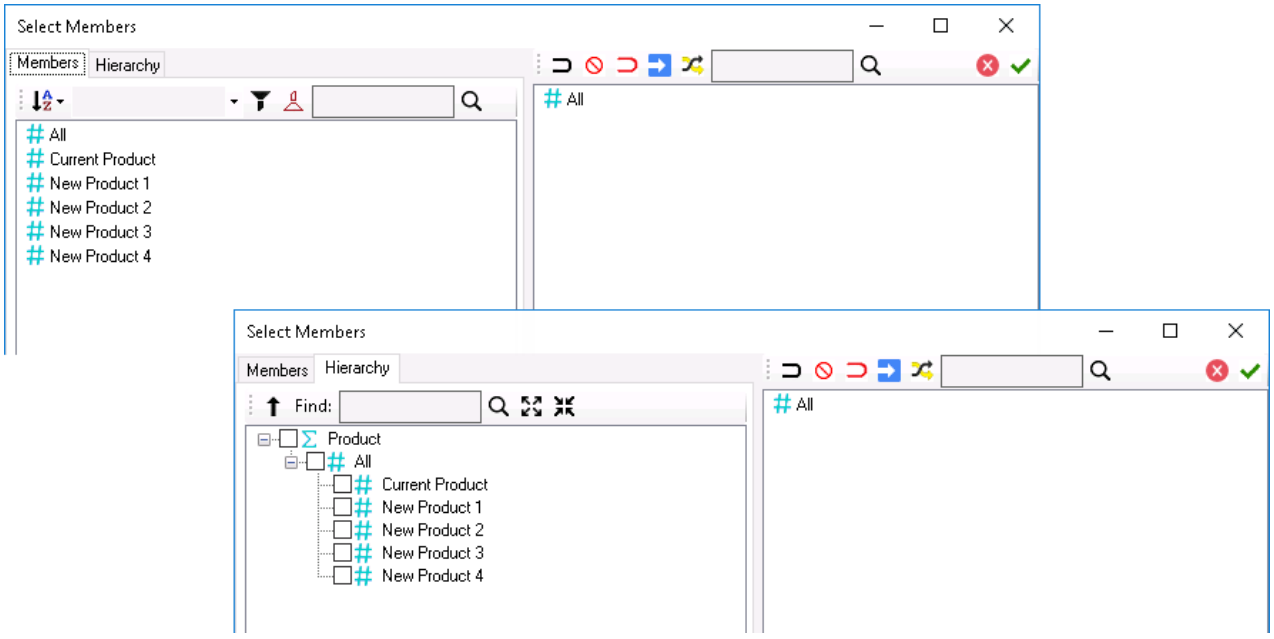
1. Go to Excel and click the **PowerExcel Tab** from the Excel ribbon.
2. Create a new PowerExcel Slice:
   - In the PowerExcel Tab, click the **New** icon.
     This will display the PowerExcel sidebar on the right-hand area of the Excel worksheet.
   - In the PowerExcel sidebar, click on the **Database drop-down** and select the correct PowerExcel Database connection, i.e., **USING_OLATION**.
     Note: Since we have previously configured this database connection in an earlier exercise, we will already see it listed in the drop-down menu.
   - Click on the **Cube drop-down** and select the correct Cube, i.e., **PRODUCT_SALES**.
     The five Dimensions of the selected Cube appear in the Filters, Columns and Row boxes.



- Check the Dimension Members and attributes of the two newly created Dimensions, **PRODUCT_SALES** Measure, and **Product** by double-clicking on them.



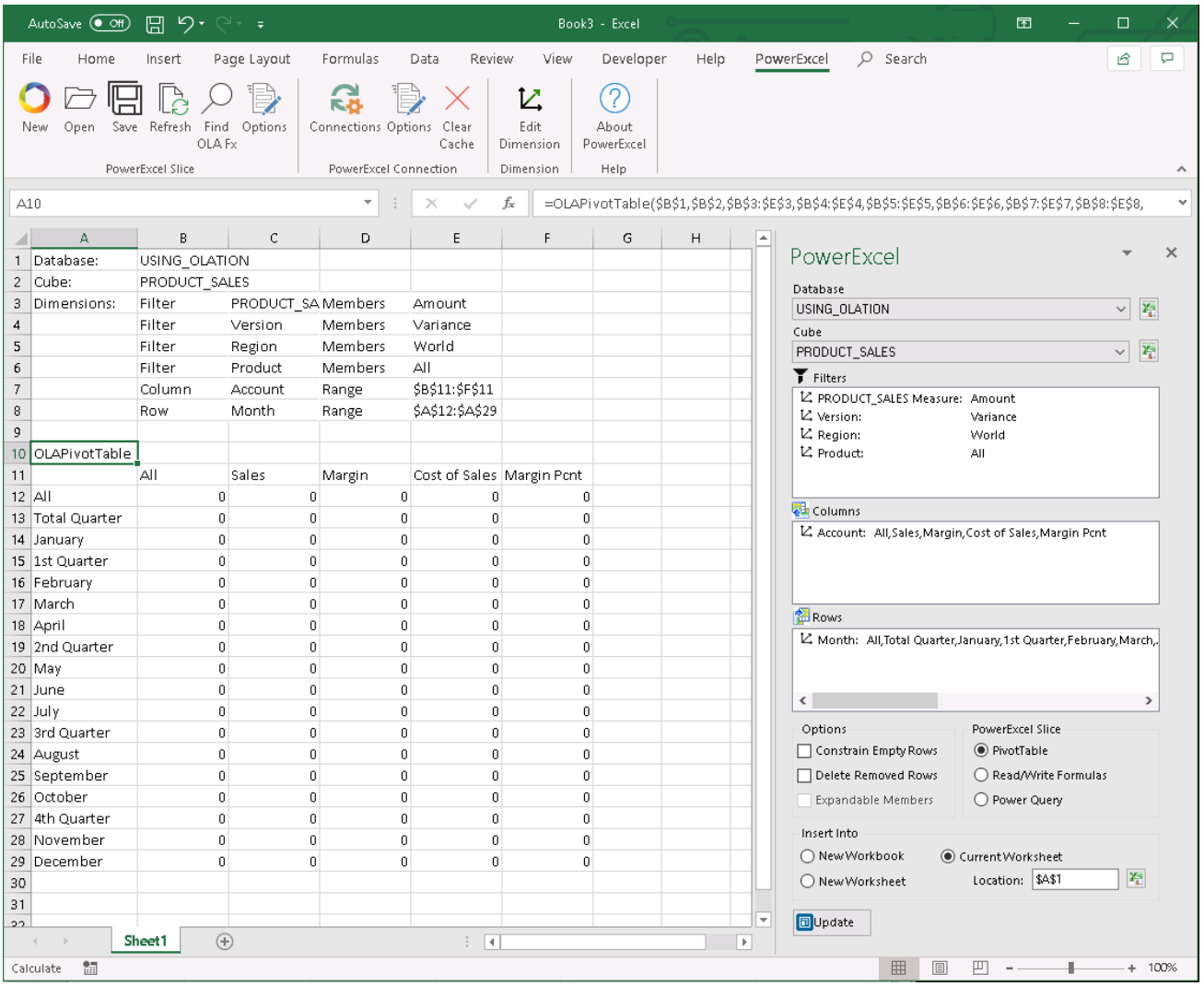
Dimension: **PRODUCT_SALES Measure**



Dimension: **Product Measure**

3. Next, to create a PowerExcel Slice:
   - Leave the default Slice arrangement.
   - Pick **PivotTable** as the PowerExcel Slice type.
   - Select to insert into the **Current Worksheet** starting at **cell A1**. Click the **Insert** button. The PowerExcel Slice will appear as follows (it is worth mentioning that this is a kind of

"default" Slice, as we made no selections along Filters, Columns or Rows):
Note: As you can see, the PowerExcel Slice currently has NO FACT DATA. What has been built so far is just the metadata structure of the Cube.



8.5. Create a Cross-Cube Formula for the PRODUCT_SALES Cube

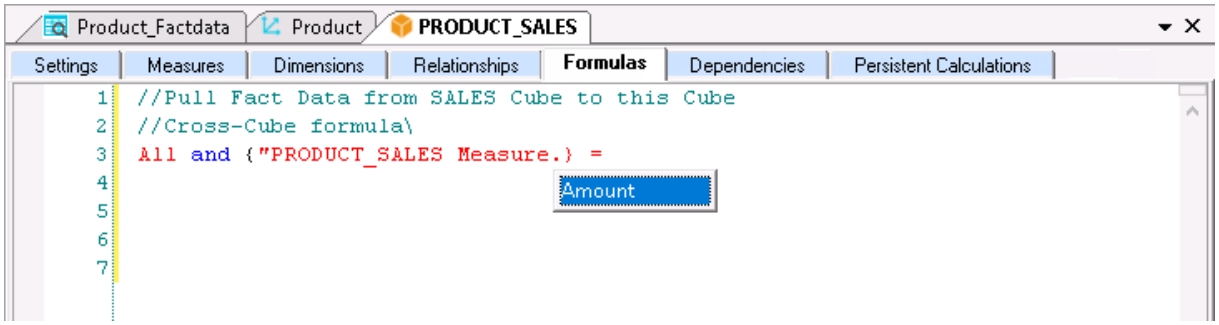
We next want to populate the newly created Cube with fact data: we will pull the fact data from the first Cube (created earlier, the *SALES* Cube) into specific intersections of the *PRODUCT_SALES* cube by using a Cross-Cube Formula.

Because there is a difference in the dimensionality of the two Cubes—the second Cube has an additional Dimension (*Product*), we will need to define the specific intersections where the data will go.

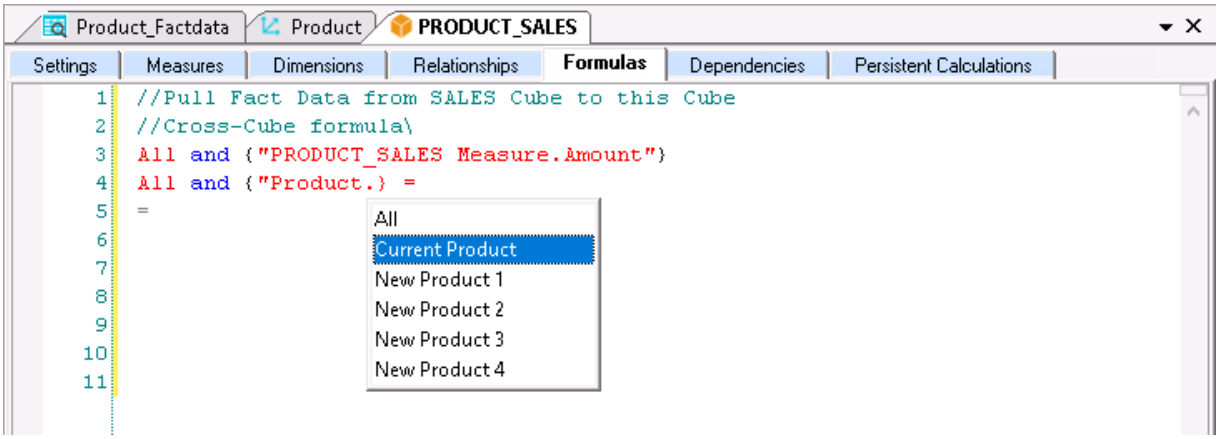
First, let's make an assumption for the sake of this exercise: in the past, our example company carried one product (*Current Product*) but now wants to launch 4 additional products (*New Product 1*, *New Product 2*, *New Product 3* and *New Product 4*). The Finance Director wants to use *Current Product* data as the basis to create analyses and plans for the new products. Our objective, therefore, will be to pull in data from the *SALES* CUBE, targeting results to the **Current Product** Member of the *Product* Dimension in the *PRODUCT_SALES* cube.

To create the Cross-Cube formula for the PRODUCT_SALES Cube:

1. In Olation Studio double-click on the **PRODUCT_SALES** Cube.
2. To define the formula, go to the **Formulas Tab**
   In the formula pane, you can type in a descriptive comment to identify what it is for. Make use of the symbols ***//*** or; ***/**** and ****/***.
3. Write the **LHS Expression (**aka **Range Reference)**. You can use the command buttons along the Cube Tab of the Olation ribbon:
   - Click on the **Formula button** and select **All And {}** as the qualifier; double-click to select **PRODUCT_SALES Measure** as the Dimension and double-click for **Amount** as Member (as in the following image).

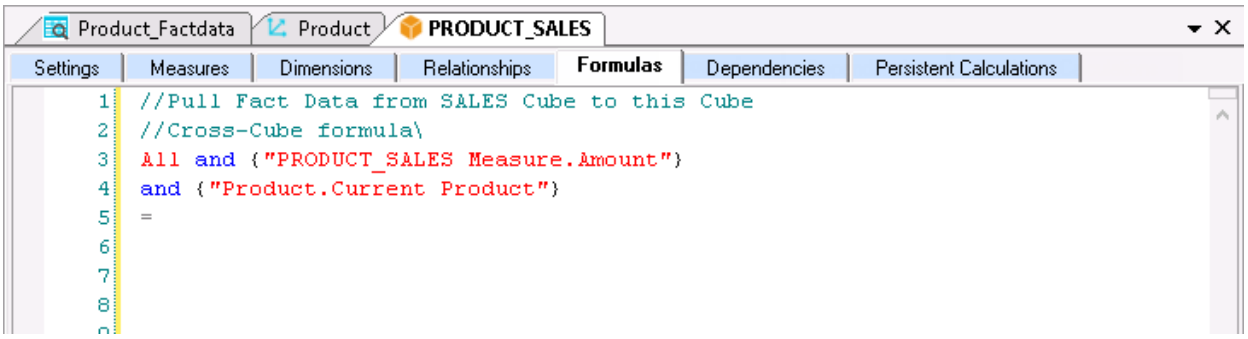


   - Place your cursor on the line below.
     Once again, click on the **Formula button** and select **All And {}** as the qualifier; select **Product** as the Dimension and **Current Product** as Member.

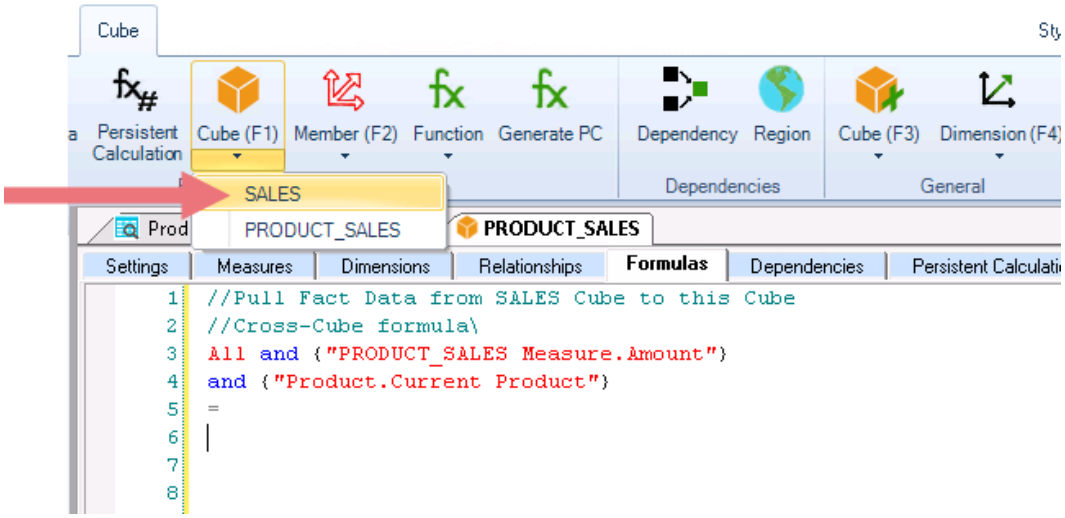


   - Delete **All** at the beginning of the second expression.

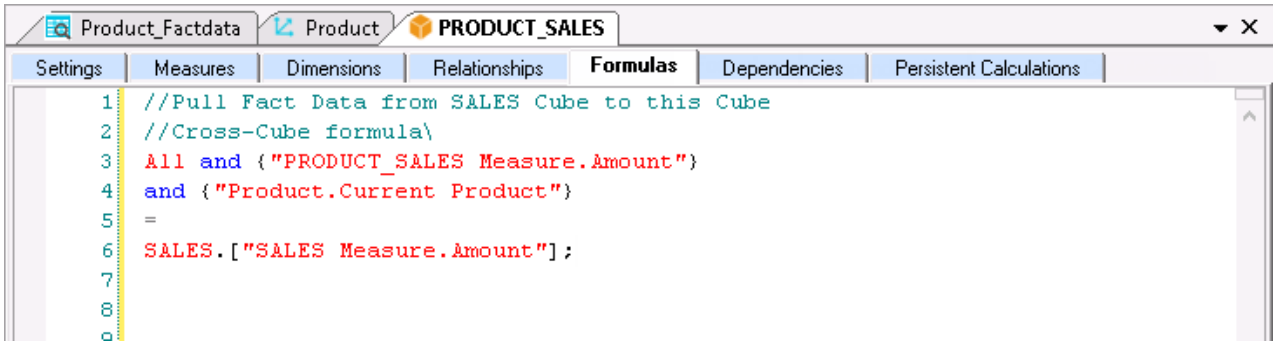
- Delete the first **equals symbol** (**"="**), to the right on the top line, so the formula looks as follows:



4. Move your cursor after the **equals symbol** (**=**).
5. To write the **RHS Expression (**aka **Cube Reference)**.
   Click on the **Cube button** and select the **SALES** Cube; double-click to select **SALES Measure** as the Dimension and double-click for **Amount** as Member.



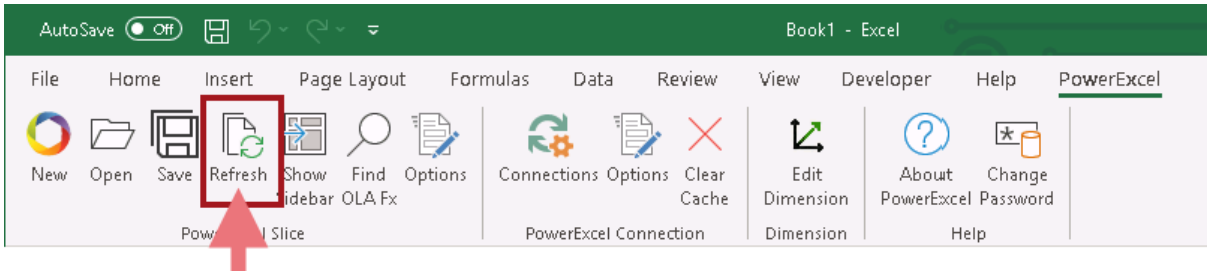
6. Type a **semi-colon** (;) to indicate the end of the formula.
   The formula will appear as follows (you may hit **Enter** between lines—it will not affect the formula):



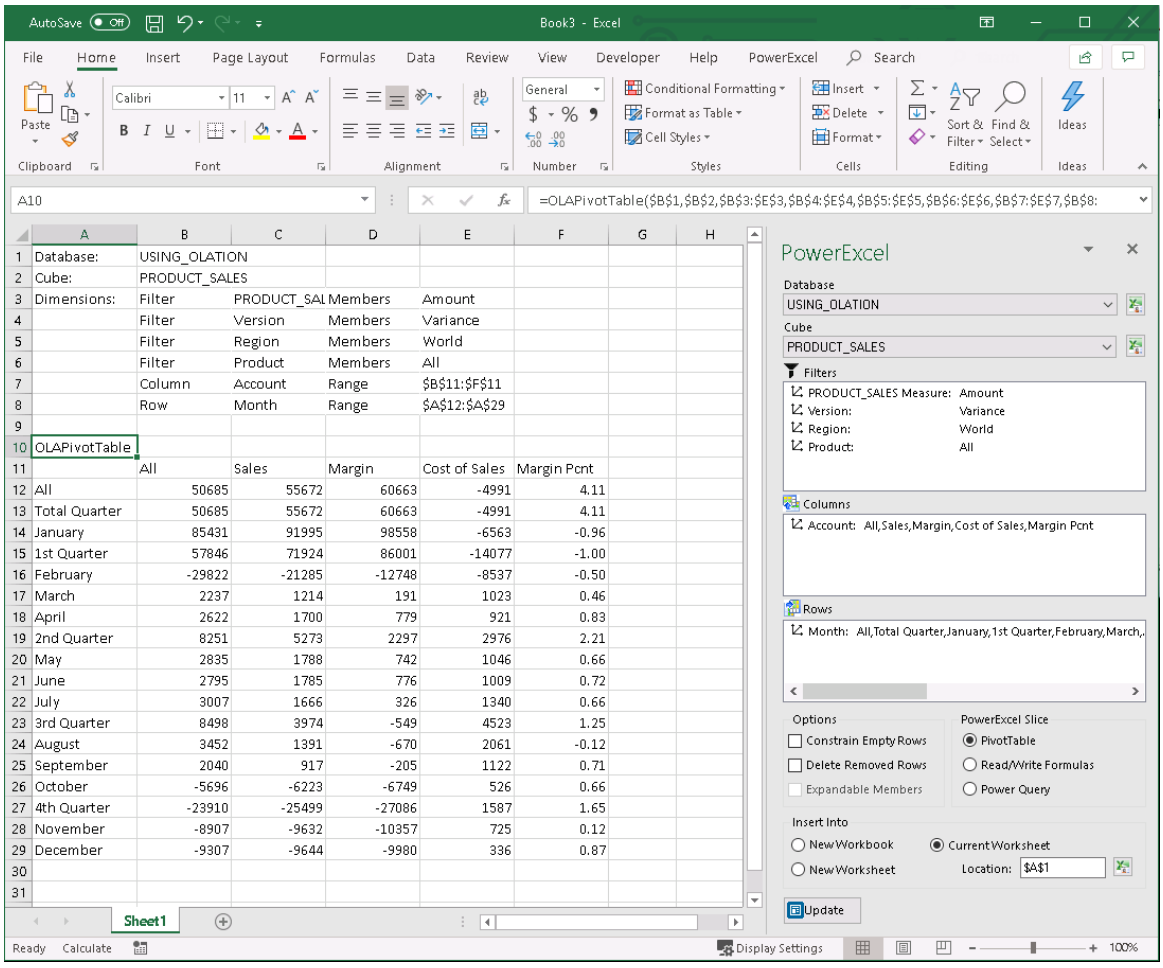
7. Click the **Check icon** (the green checkmark), aka **Check Syntax button** to verify that the formula is free from syntax errors. You will get a prompt that says, 'The formula syntax is correct'.
8. Save the **Cube**.

8.5.1. Viewing the Results of the Cross-Cube Formula

1. Go back to the PowerExcel Slice we have created for the *PRODUCT_SALES* Cube.
2. To update the PowerExcel Slice data, click on the **Refresh icon** found on the PowerExcel Tab. You can also just simply press **F9**.
   Note: Make sure that your PowerExcel Database connection is active.



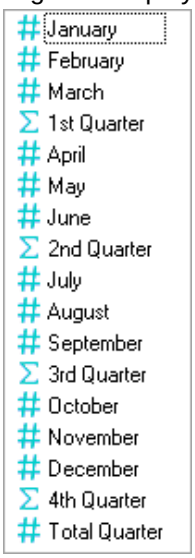
You can now see data come into the Slice.



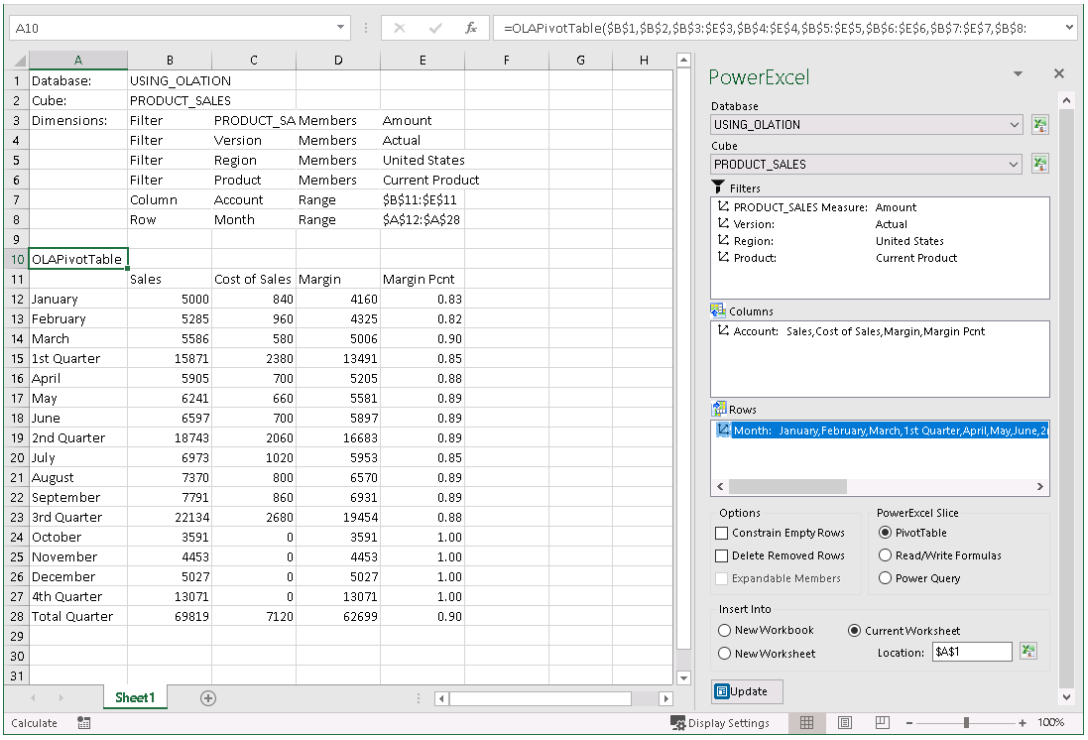
3. Given that we created a Cross-Cube Formula to get data from the *SALES* cube and push it into the specific Product Member—*Current Product*—of the *PRODUCT_SALES* cube, we can make a comparison of two Slices.

First, change the latest Slice like so:

- In the **Filters** box for the Slice (from the *PRODUCT_SALES* cube), select the following:
  PRODUCT_SALES Measure: **Amount**
  Version: **Actual**
  Region: **United States**
  Product: **Current Product**
- Change the display Members along the columns for the *Account* Dimension to only show: **Sales**, **Cost of Sales**, **Margin** and **Margin Pcnt**.
- Change the display Members along the rows to show the following:

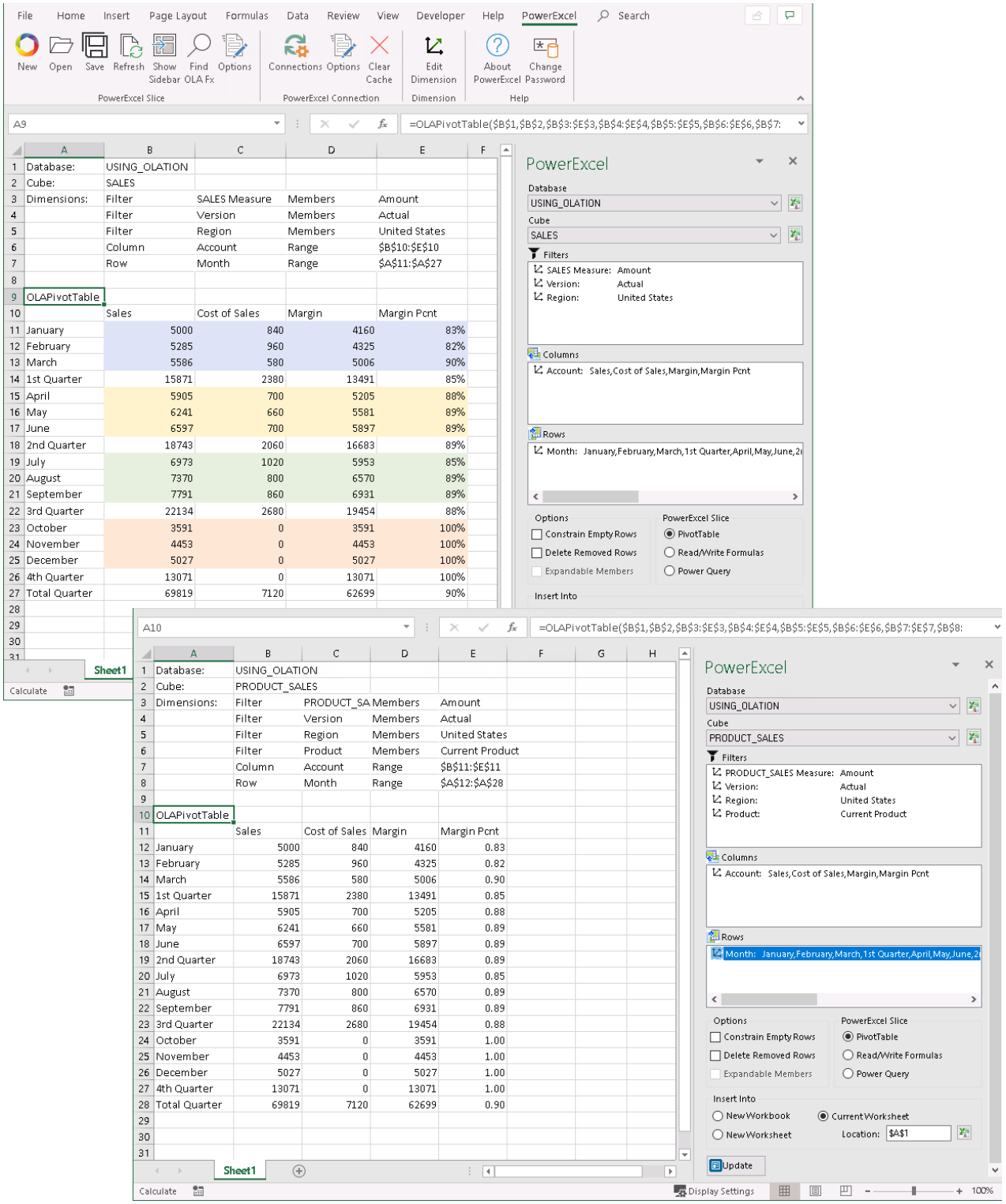


The PowerExcel Slice will update to show the following:

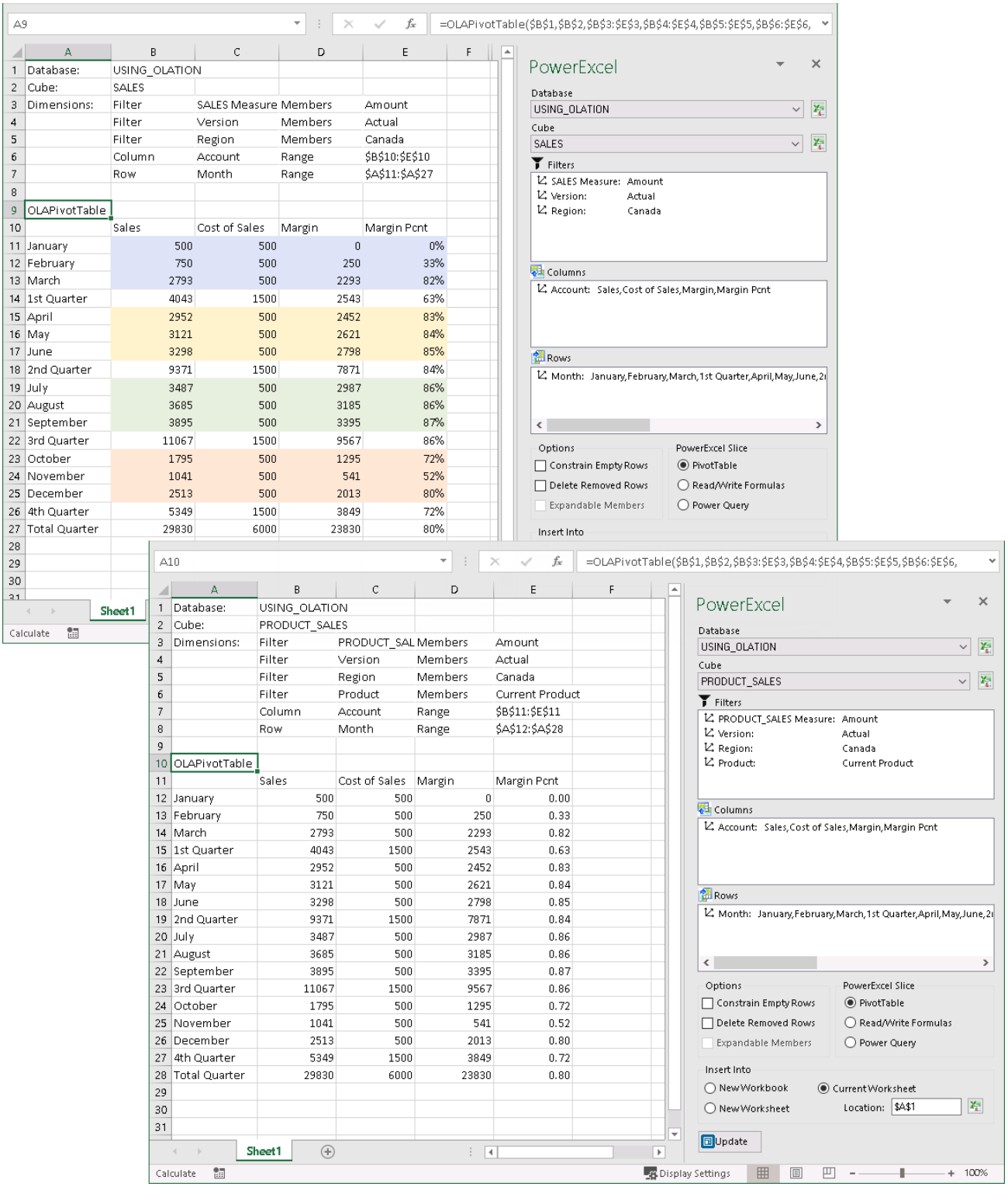


	A	B	C	D	E
1	Database:	USING_OLATION			
2	Cube:	PRODUCT_SALES			
3	Dimensions:	Filter	PRODUCT_SA	Members	Amount
4		Filter	Version	Members	Actual
5		Filter	Region	Members	United States
6		Filter	Product	Members	Current Product
7		Column	Account	Range	$B$11:$E$11
8		Row	Month	Range	$A$12:$A$28
9					
10	OLAPivotTable				
11		Sales	Cost of Sales	Margin	Margin Pcnt
12	January	5000	840	4160	0.83
13	February	5285	960	4325	0.82
14	March	5586	580	5006	0.90
15	1st Quarter	15871	2380	13491	0.85
16	April	5905	700	5205	0.88
17	May	6241	660	5581	0.89
18	June	6597	700	5897	0.89
19	2nd Quarter	18743	2060	16683	0.89
20	July	6973	1020	5953	0.85
21	August	7370	800	6570	0.89
22	September	7791	860	6931	0.89
23	3rd Quarter	22134	2680	19454	0.88
24	October	3591	0	3591	1.00
25	November	4453	0	4453	1.00
26	December	5027	0	5027	1.00
27	4th Quarter	13071	0	13071	1.00
28	Total Quarter	69819	7120	62699	0.90
29					
30					
31					

4. Next, show (or open) the Slice created earlier from the *SALES* cube to do a side by side comparison with this new PowerExcel Slice from the PRODUCT_SALES cube.
   Notice that, indeed, the data matches across the two Cube, as shown in the images below:



Slices from the *SALES* cube and the *PRODUCT_SALES* cube showing *United States*, *Actuals*.

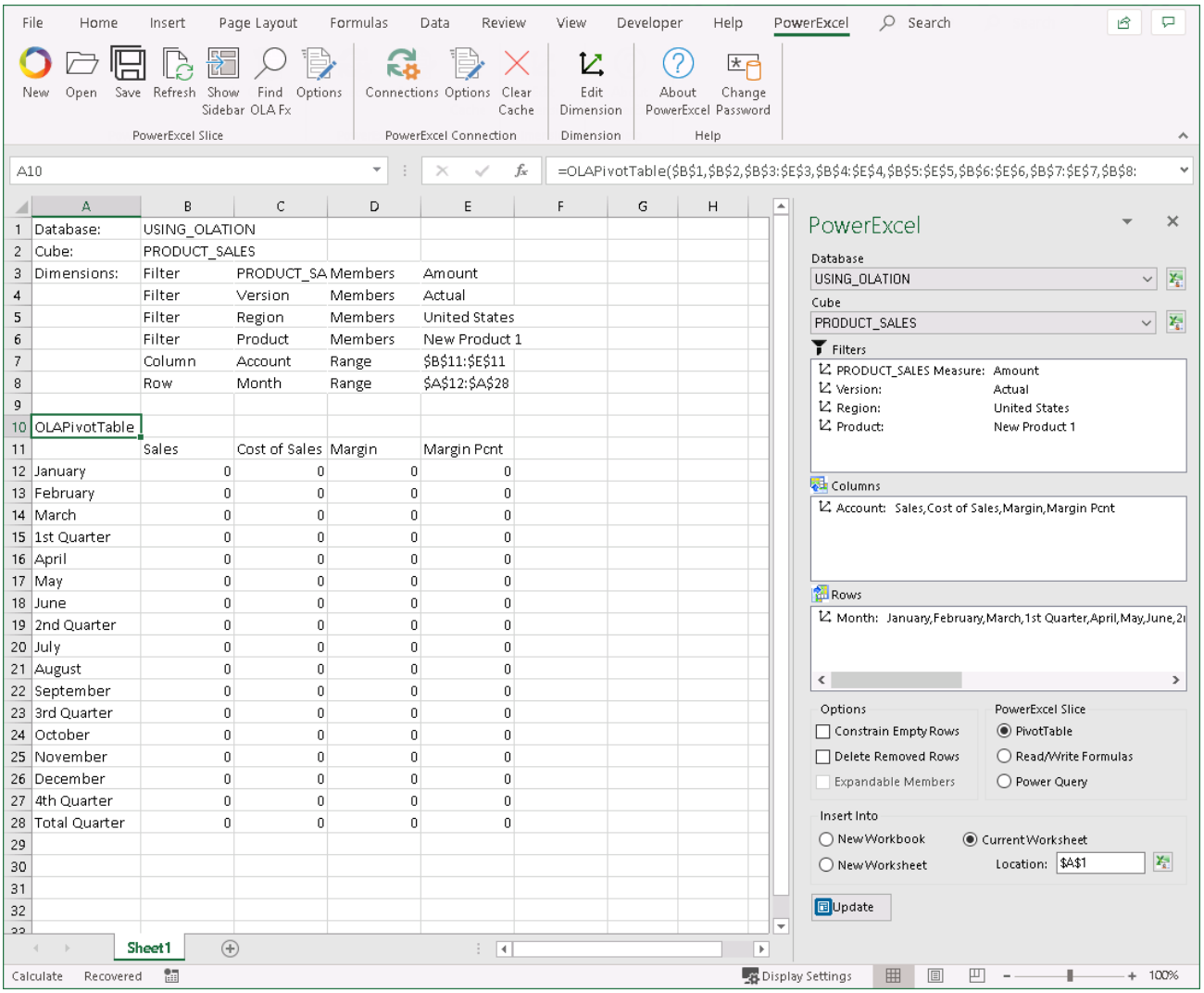


	A	B	C	D	E
1	Database:	USING_OLATION			
2	Cube:	SALES			
3	Dimensions:	Filter	SALES Measure	Members	Amount
4		Filter	Version	Members	Actual
5		Filter	Region	Members	Canada
6		Column	Account	Range	$B$10:$E$10
7		Row	Month	Range	$A$11:$A$27
8					
9	OLAPivotTable				
10		Sales	Cost of Sales	Margin	Margin Pcnt
11	January	500	500	0	0%
12	February	750	500	250	33%
13	March	2793	500	2293	82%
14	1st Quarter	4043	1500	2543	63%
15	April	2952	500	2452	83%
16	May	3121	500	2621	84%
17	June	3298	500	2798	85%
18	2nd Quarter	9371	1500	7871	84%
19	July	3487	500	2987	86%
20	August	3685	500	3185	86%
21	September	3895	500	3395	87%
22	3rd Quarter	11067	1500	9567	86%
23	October	1795	500	1295	72%
24	November	1041	500	541	52%
25	December	2513	500	2013	80%
26	4th Quarter	5349	1500	3849	72%
27	Total Quarter	29830	6000	23830	80%
28					
29					
30					
31					

	A	B	C	D	E	F
1	Database:	USING_OLATION				
2	Cube:	PRODUCT_SALES				
3	Dimensions:	Filter	PRODUCT_SAL	Members	Amount	
4		Filter	Version	Members	Actual	
5		Filter	Region	Members	Canada	
6		Filter	Product	Members	Current Product	
7		Column	Account	Range	$B$11:$E$11	
8		Row	Month	Range	$A$12:$A$28	
9						
10	OLAPivotTable					
11		Sales	Cost of Sales	Margin	Margin Pcnt	
12	January	500	500	0	0.00	
13	February	750	500	250	0.33	
14	March	2793	500	2293	0.82	
15	1st Quarter	4043	1500	2543	0.63	
16	April	2952	500	2452	0.83	
17	May	3121	500	2621	0.84	
18	June	3298	500	2798	0.85	
19	2nd Quarter	9371	1500	7871	0.84	
20	July	3487	500	2987	0.86	
21	August	3685	500	3185	0.86	
22	September	3895	500	3395	0.87	
23	3rd Quarter	11067	1500	9567	0.86	
24	October	1795	500	1295	0.72	
25	November	1041	500	541	0.52	
26	December	2513	500	2013	0.80	
27	4th Quarter	5349	1500	3849	0.72	
28	Total Quarter	29830	6000	23830	0.80	
29						
30						
31						

Slices from the *SALES* cube and the *PRODUCT_SALES* cube showing *Canada*, *Actuals*.

5. If, in the Filters box, we change the *Product* Member to show any other Members (e.g., *New Product 1*, as in the example image below), notice that these intersections show empty values:



The Finance Director can use these “templates”—for different products (the “new” products in the PRODUCT_SALES cube)—providing access to a shared Olation planning model for users across all regions.

In sum, we have demonstrated the creation of different data models/Cubes, all working together dynamically, for the various purposes of planning, analytics, and reporting.