



**Getting Started Guide**  
**Argos and OLAP**

Version 2.5

Last Updated 09/13/2006

## Table of Contents

<b>DOCUMENT OVERVIEW</b> .....	<b>3</b>
INTENDED AUDIENCE .....	3
SAMPLE DATABASE .....	3
<b>OLAP OVERVIEW</b> .....	<b>4</b>
<b>BUILDING FIRST OLAP DATABLOCK</b> .....	<b>6</b>
CREATE DATASETS FOR FACT TABLE AND, OPTIONALLY, DIMENSIONS .....	8
CONFIGURE MEASURES.....	10
CONFIGURE DIMENSIONS .....	13
<b>RUNNING YOUR FIRST OLAP DATABLOCK</b> .....	<b>16</b>
DIMENSION TOOLBAR .....	17
Expand/Collapse Toolbar.....	17
Dimension Items .....	17
Dimension Item Labels .....	17
Activate Dimension Editor .....	17
MEASURE TOOLBAR .....	19
Expand/Collapse Toolbar.....	20
Activate global measure manager .....	20
Calculated Measures Manager.....	21
Measure Items .....	21
Activate individual Measure Manager .....	21
Graphs .....	23
.....	23
<b>ADDING PARAMETERS TO THE OLAP CUBE</b> .....	<b>25</b>
<b>USING PARENT/CHILD RELATIONSHIP</b> .....	<b>27</b>
<b>APPENDIX A – DATABASE</b> .....	<b>29</b>

## Getting Started Guide - OLAP

*We encourage you to inform us of any errors you may find in this guide, and we ask that you please forward suggestions that will make this document more relevant/understandable to those that will follow. We listen to our users to always try to improve our literature and products. Welcome to **Argos!** Enjoy!*

### **Document Overview**

This document is NOT intended to be a complete guide to OLAP (On-Line Analytical Processing). Although some terms will be defined, the topic covered will be how to use Argos to produce OLAP reports.

For the purposes of this guide, it is assumed that the software has been fully installed and configured – i.e. ADO connections and users have been added.

If any problems arise in following the examples in this guide, please contact our support staff for assistance. This can be done by submitting a support request to <http://helpdesk.evisions.com>, or you may choose to first visit our knowledgebase at <http://www.evisions.com/suport> .

In addition to this manual, there are additional manuals and multimedia tutorials for the Administrators and End Users.

### **Intended Audience**

This guide is intended for the **Argos** developer and assumes that the reader is familiar with writing SQL (Structure Query Language) statements. The samples in this document are written using SQL designed for Access. Every attempt is made to keep the queries generic enough so they can be transported across platforms. Although ANSI standards do exist, these standards remain theoretical and differ significantly from implementation to implementation. This is NOT a SQL tutorial.

In addition, the user should be familiar with Argos in so far as they can create a folder and datablock. It is also assumed that the user is familiar with OLAP and has gone through the designer training.

### **Sample Database**

All samples used in this guide (unless otherwise indicated) are based on a Microsoft Access Database that can be downloaded from <http://evisions.com/support/argos>. The database contains information on sales for a small company. Descriptions of the tables and how they are joined can be found in Appendix A.

## OLAP Overview

OLAP stands for 'On-Line Analytical Processing'. A good overall discussion of OLAP can be found at: <http://www.olapreport.com/fasmi.htm>

We will try to summarize some key features here.

OLAP is a specific way to represent statistical data for executives, specialists and analysts. It is designed to aid in decision-making and better information understanding. The main idea is to answer the user's questions, arising at the work time, on-the-fly, quickly. A popular definition is "A million spreadsheets in a box." The key to OLAP is its ability to allow the end user to configure different views of the same data.

An OLAP system allows user to get into details and generalize, filter, sort and regroup data at the time of analysis. Intermediate and final totals are recalculated instantly. The user is presented data in an electronic spreadsheet format. Moving rows and columns or clicking them the user makes the system perform calculations and show data in different aspects.

Thus, the user can produce many reports out of a single dataset on his own, without any interference with IT-specialists.

OLAP breaks data into two groups: facts (numbers, also called measures) and dimensions (descriptions). Facts (**Measures**) are aggregated in a given slice by some algorithm while the user defines grouping and aggregation depth by using **Dimensions**.

Fields by which data records are grouped are called **Dimensions**. Dimensions can contain values of various types: strings, dates, numbers, and so on. A dimension lists members, all of which are perceived by the user to be similar types of data. A dimension is the answer to "How do you want to see your data." For example, a Time dimension might include members for years, quarters, months, and weeks. A Student Attribute dimension might include members for Gender, Ethnicity, Major and Residency. When running the cube, the end user can select to group the data by any or all the defined dimensions.

Dimensions have levels that allow for drilling down. For example, if looking at enrollment figures for a term, you could drill down to get more information as to the make up of the enrollment. This capability allows the end user to help analysis the data and determine why enrollment might be up for some terms and down for others.

**Measures** are the numbers in the OLAP spreadsheet or cube. They can also be referred to as Facts. The Measures are displayed in the cells of the cube. Measures are stored in what are called Fact Tables. Whenever creating a new cube, the first question the designer must answer is what fact table will be used?

Fact tables typically contain the following types of fields:

- Key fields to join the Fact table to Dimension tables.
- Measure fields containing numeric values.

Dimensions can be pulled from the Fact Table but typically are stored in Dimension Tables. Dimension tables have the following types of fields:


- Key fields, used to join the dimension tables to the fact table (star schema).
- Level name fields, used to store the member names for the levels. For example, the Time dimension table could have a field called Month, which would have values such as January, February, March, etc.
- Level Order Key fields, used to store integer values used to order the members of the levels (if necessary). For example, the Time dimension table could have a field called Month Order Key, which could have a value of 1 for January, 2 for February, 3 for March, etc.
- Member Property fields, used to store the member property information. A Time dimension could have a field called Day Count, which would store the number of days for each month.

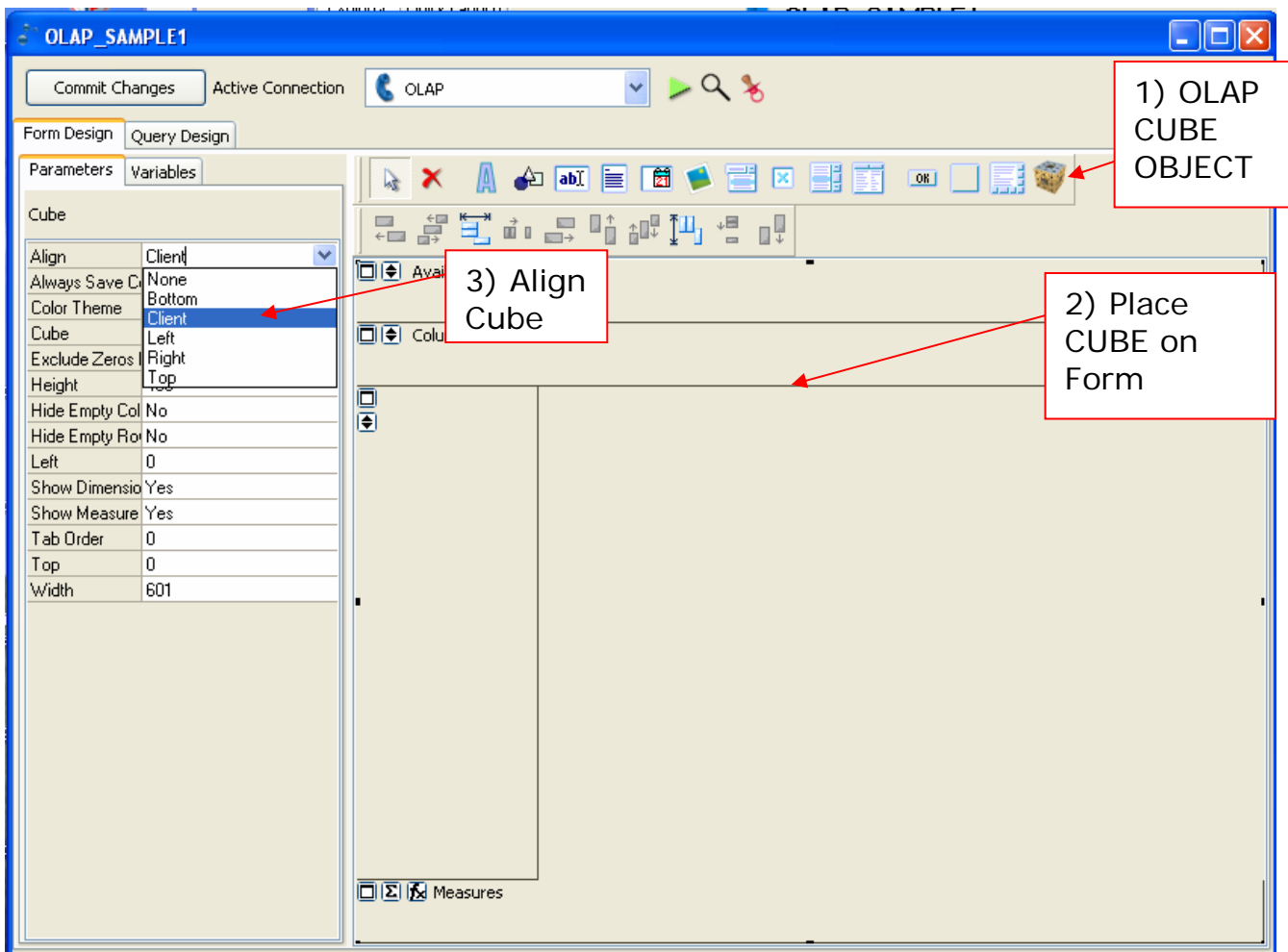
The combination of a fact table with its dimensions is called a **star schema**. The fact table is the center of the star, while each dimension table represents a point of the star.

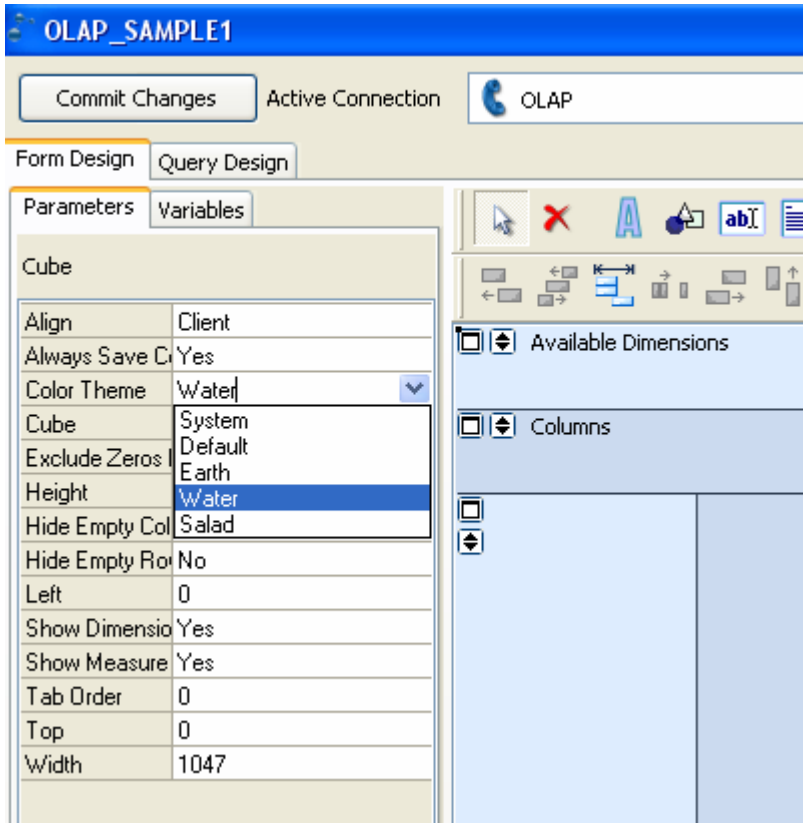
## Building First OLAP Datablock

The first datablock we build will be pretty straight forward, demonstrating how to add a fact table and a dimension. We want to break out sales by Month and Employee to determine if there is any monthly trend and who our top sales person is.

To start we will add a Datablock called OLAP\_SAMPLE1 and point it to the OLAP sample database. We will start on the Form Design Window. a logical structure built like this: in the center of this structure is a *Facts* table. Surrounding this table are the dimension tables which contains the reference data.

To add the OLAP object, we need to click on the OLAP CUBE icon (  ), place it on the Parameter form, and then set the alignment to Client. By setting the alignment to client, the cube will fill the whole page and therefore more data will be visible to the end user when they are running the report associated with this datablock.





Another property we can set at this time is the Color Theme. The default Color Scheme is monnotone gray. To make it easier for the end user, we can set the color theme to different values.

Here we are using the water scheme.

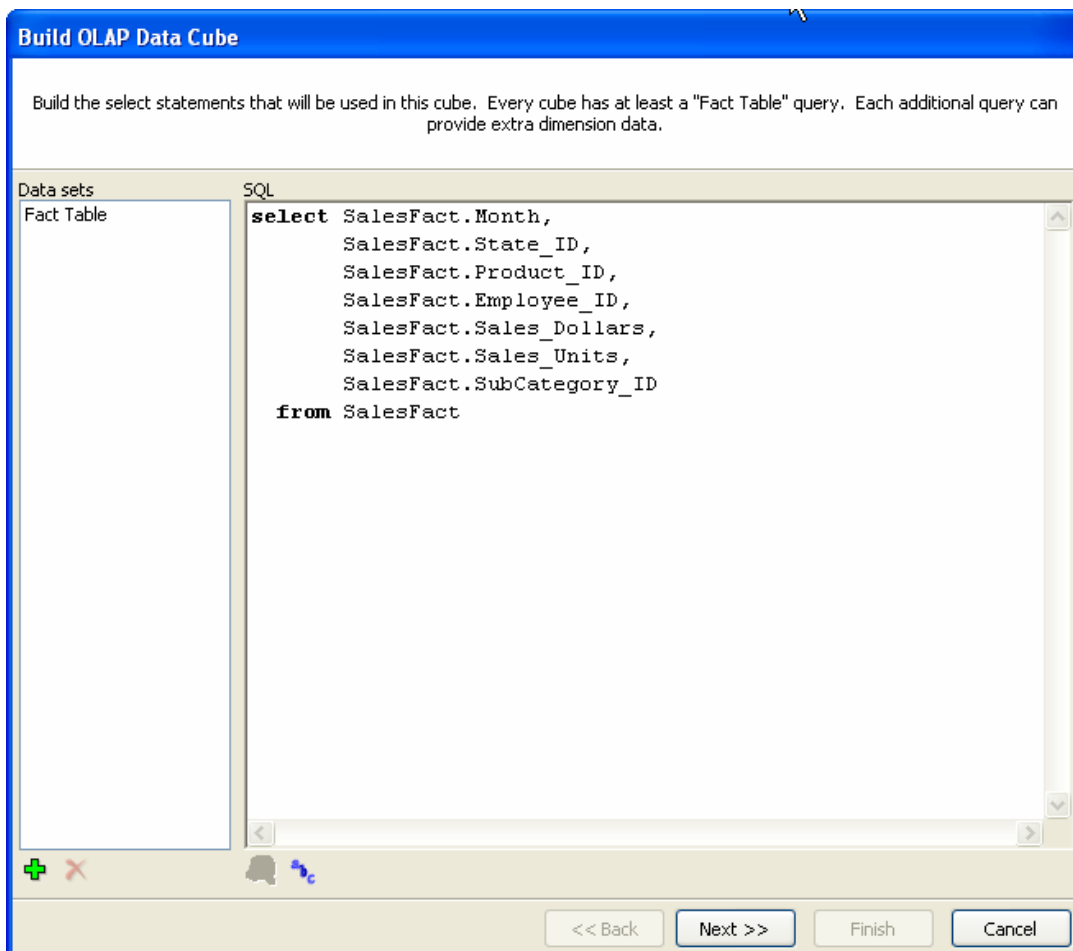
To start editing the properties of the object, double click on the object on the form itself, or click on the ellipses next to the attribute labeled Cube.

## Create Datasets for Fact Table and, optionally, Dimensions

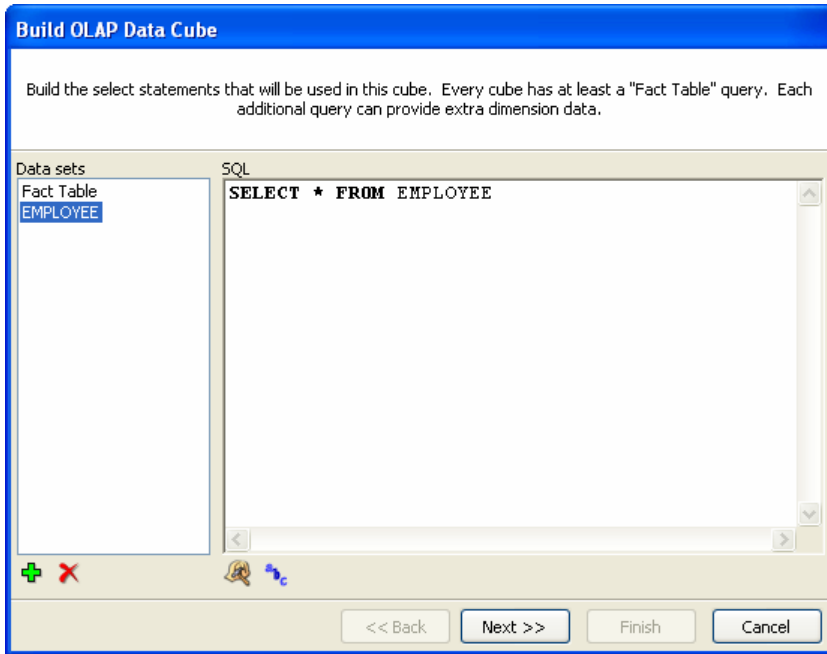
The first form that opens allows us to define the fact and dimension tables. Remember that the Fact table contains the data which we want to add or count or do some calculation with. In addition, it contains links to other tables called dimensions. The dimension tables contain the data by which we will group or organize the fact.

It is possible to create a cube with just a Fact Table. For example, if using Oracle, you might have created a view that contains all the required data.

On the form, note that you can create a SQL statement from scratch, or we can press the hard hat icon (👓) to build the statement using the qui interface. In this example, we used the query builder to pull back all the fields from the SalesFact table in our sample database. Note that one of the fields in the SalesFact table is Employee\_ID which links back to our Employee table.







Next we need to add a dimension dataset. Since we want to group the data by employee, we will be adding the employee table.

To add a new dimension, click on the green plus sign on the bottom of the form. We can quickly add all fields by specifying `SELECT * FROM EMPLOYEE`.

We also want to rename this dataset to `EMPLOYEE` to reflect what data is being returned. To do this, right click on the label `Dataset1` and choose `Rename`. Assign a new name to this object.

At this point, we have set up all the required datasets, establishing where the data is being retrieved. We now need to set up the properties for the measures and dimensions. Press the `Next` button to continue.

The next form allows us to select what fields from the `FactTable` will be used for our Measures.

## Configure Measures

To select a field to be used as a measure, select from the list of Available Fields listed in top pane. Note that only fields from the FactTable query are listed.

When select, the Measure will have be checked Active by default. For the Cube to function, the designer must have at least one active measure. In addition, other default values are set:

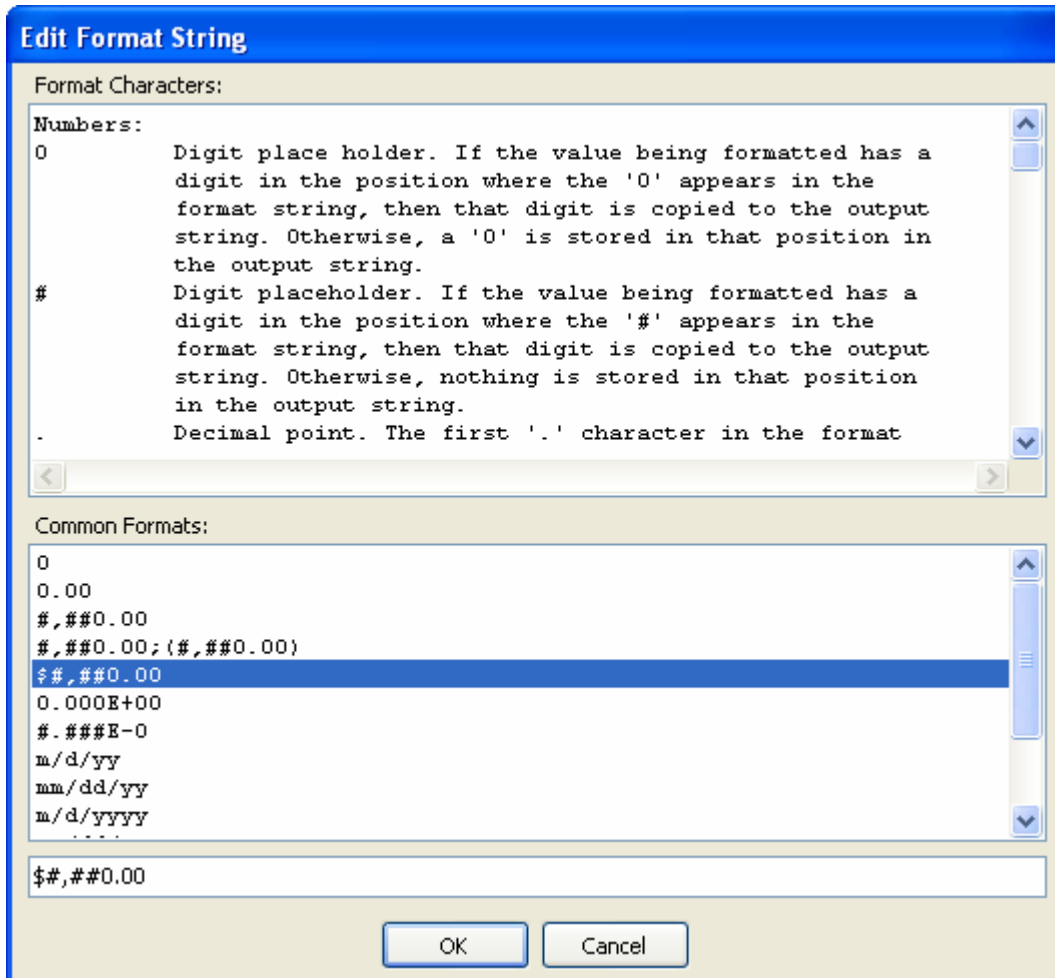
- Display Name will be set to the field name;
- Method of calculation will be Summa (all values will be added together, based on the dimension used for grouping)
- Display format will be set to 0.00.

All of the above values can be changed. To change the Display Name (i.e. what title will be used when this measure is displayed on the report), edit the value in the box.

To change the Method of Calculation, use the drop down arrow and select the method from those listed.

The functions you can choose from are extensive. The most commonly used are listed in the beginning.

Similarly, to change the Display format (how the data will be displayed when user executes the report), use the drop down arrow to select from the list of available formats. Here we are going to change the format to display the dollar amount.



Double click on selected value and click OK to continue.

In addition to adding the Sales\_Dollars, The end user also might want to display the Sales\_Units, i.e. how many units were sold. At this time we will not make the measure active, but we will include in case the user running the report wants to display this information.

**Build OLAP Data Cube**

Configure the measures. Measures represent the numerical data such as price and quantity.

Available Fields

Month	Sales_Dollars
State_ID	Sales_Units
Product_ID	SubCategory_ID
Employee_ID	

Measures (check to make active)

<input checked="" type="checkbox"/> Sales_Dollars
<input type="checkbox"/> Sales_Units

Display Name: Units

Method of Calculation: Summa

Display Format: 0

<< Back   Next >>   Finish   Cancel

Click Next to continue.

## Configure Dimensions

Build OLAP Data Cube

Configure the dimensions. Dimensions modify the drill-down of the data by allowing the user to group data precisely as needed.

Fields available to be used as dimensions

Month	Employee_ID	SubCategory_ID
State_ID	Sales_Dollars	
Product_ID	Sales_Units	

Initial location of dimension (drag-n-drop to other tabs if needed)

Available    Column    Row

Month

↑  
↓  
✖

Display Name

Data for this dimension comes ...

directly from the Fact table

from another data set

Default Sort Method

Forecasting

On this form the designer will select what fields should be used for Grouping. Fields are selected from those listed in the top pane and include all selected fields from the FactTable.

In the above example, we have already selected the column Month to be an available dimension. Values that can be edited are:

- Display Name – by default the name of the column will be used as the Display Name. To change, edit the value in the box.
- Data for this dimension comes...- Allows the designer to select which dataset contains the data that will displayed. It can either come from the FactTable, such as the Month dimension, or from another dataset defined on the first form.
- Default Sort Method – Allows the designer to determine how the data will be sorted when displayed.
- Forecasting – if the Dimension contains more then three entries, you can have the cube predict or forecast preceding and ending values for the measures selected.

In adding a Dimension from another defined dataset, there are other options that need to be provided:

- Date Set – When building the data cube, the first thing the designer did was to define the fact table as well as any additional datasets that will be used for pulling back dimension information. Specify the dataset in this field.
- Key Field – The Value entered here specifies what field in selected data set links the fact table to the selected dimension table.
- Data Field – The value entered here specifies what field in selected data set will be displayed when the cube is run.
- Parent Field – The value entered here specifies what field in selected data set is considered the 'parent' of the Data Field. This value can be left blank if there is no hierarchal value set.

In addition to the Month dimension, we want to be able to report sale figures for the employees. To do this we select the `employee_id` from the list of fields listed in the top pane. Because this field links back to the employee table we can use this field to pull back information about the employee from the employee table.

We could just display the `employee_id`, but from the end users point of view this might not be useful. We really want to display the employee name. To do that, we are going to use the second dataset (Employee) that we added to the definition of this cube. When using a dataset other than the FactTable, we need to specify what field is being used to link the FactTable to the Dimension dataset. This is done by specifying the Key Field.

We also want to sort by the employee name. Finally, we want the display name to be Employee and not `Employee_ID`. When done, the completed form will look like the following:

**Build OLAP Data Cube**

Configure the dimensions. Dimensions modify the drill-down of the data by allowing the user to group data precisely as needed.

Fields available to be used as dimensions

Month	Employee_ID	SubCategory_ID
State_ID	Sales_Dollars	
Product_ID	Sales_Units	

Initial location of dimension (drag-n-drop to other tabs if needed)

Available    Column    Row

Month  
Employee\_ID

Display Name  
Employee\_ID

Data for this dimension comes ...  
 directly from the Fact table  
 from another data set

Data Set  
Employee

Key Field  
Employee\_ID

Data Field  
Employee\_Name

Parent Field

Default Sort Method  
Sort by name

Forecasting  
No forecasting

<< Back    Next >>    Finish    Cancel

The last thing we can do is to assign the dimensions to be either a column or a row when the cube is actually displayed. For the time being we will leave the dimensions as 'Available' and let the end user determine how to display the data.

Click Finish to save your settings and on the parameter form press the Commit Changes button to save the datablock.

## Running your First OLAP datablock

A part of designing any datablock, you should review and make sure it works correctly. We can use the Review button (✓) or run the report as a normal end user. Both display the following form:

The screenshot shows the 'Test Run' window with the following components and callouts:

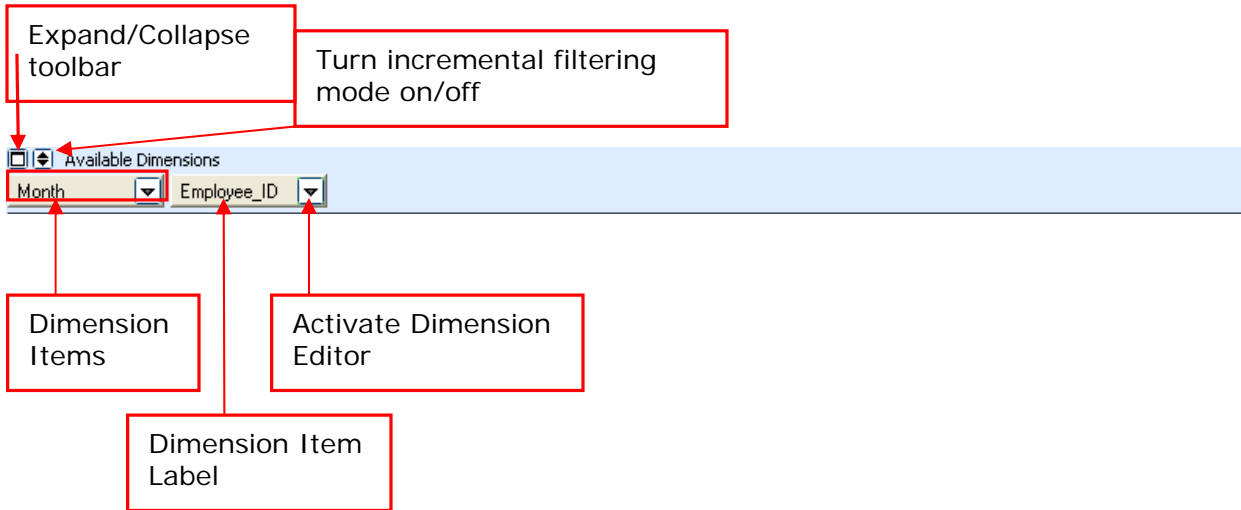
- Dimension Toolbar:** Located at the top right, containing 'Available Dimensions' and dropdowns for 'Month' and 'Employee\_ID'.
- Dimension Column Toolbar:** Located below the Dimension Toolbar, containing a 'Columns' section.
- OLAP Cube Grid:** The central area displaying a table with the following data:

Total by ROWS
Amount Sold
Value
\$76,741.28
- Dimension Rows Toolbar:** Located on the left side of the OLAP Cube Grid.
- Measure Toolbar:** Located at the bottom left, containing a 'Measures' section with a dropdown for 'Amount Sold'.
- Footer:** Contains the 'evisions' logo, a text field 'Retrieve a maximum 100000 record(s)', and 'Get' and 'Close' buttons.




## **Dimension Toolbar**

The dimension toolbar is used to manipulate the dimensions on the cube.



### **Expand/Collapse Toolbar**

You can collapse the dimension toolbar in order to have a bigger display area for the Cube Grid. The button () acts as a toggle.

### **Dimension Items**

The dimensions that are displayed are obtained from the list of available dimensions the designer added to the cube. All dimensions are displayed, except for those already added to the dimension rows or column toolbars. If any of the dimensions were added by the designer to be columns or rows, they will not be listed here.

The order of the items on the dimension toolbar has no significance. The order is determined by the order they were added by the designer. The end user can also rearrange by dragging them to the dimension rows or dimension columns.

### **Dimension Item Labels**

The dimension labels are obtained from the Display Name property of the defined dimension.

### **Activate Dimension Editor**


Click on this button to activate the dimension editor.

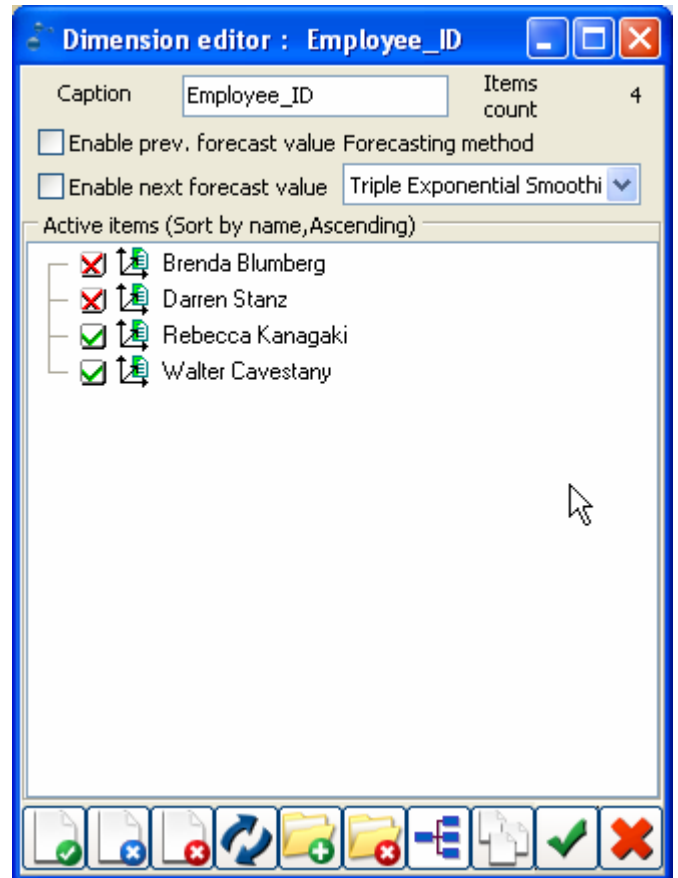
The dimension editor allows the end user running the report to 'override' or change some of the default settings defined by the created of the datablock.

The Dimension Editor allows the user to 'filter' the data or resort the data.

To filter the data to display only the items selected, change the green check mark to a blue or red check mark by clicking on the current value. The example to right will only display the information for Rebecca Kanagaki and Walter Caverstany.

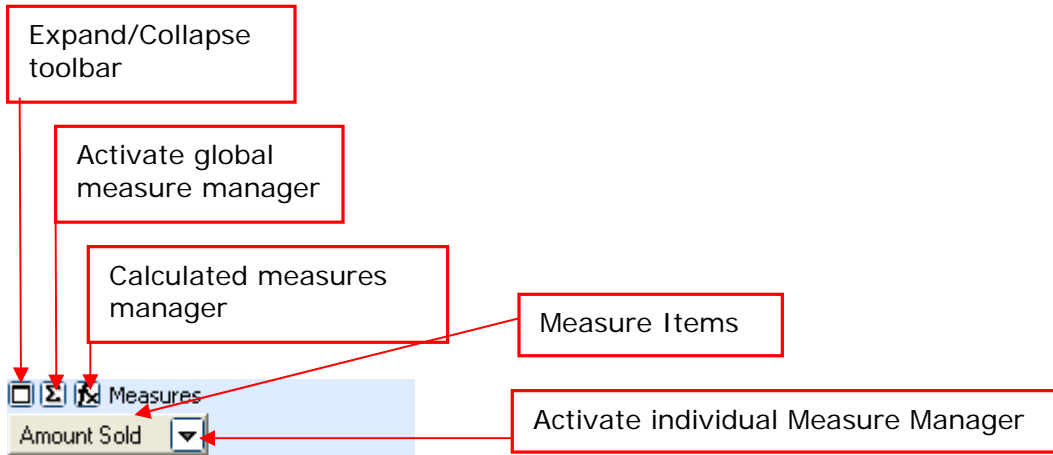
In addition we can change the sort order by

clicking on the 'Sort Dimension' icon () on the bottom tool bar. You can select to sort by Name, Key value, or not sort at all.



## Measure Toolbar

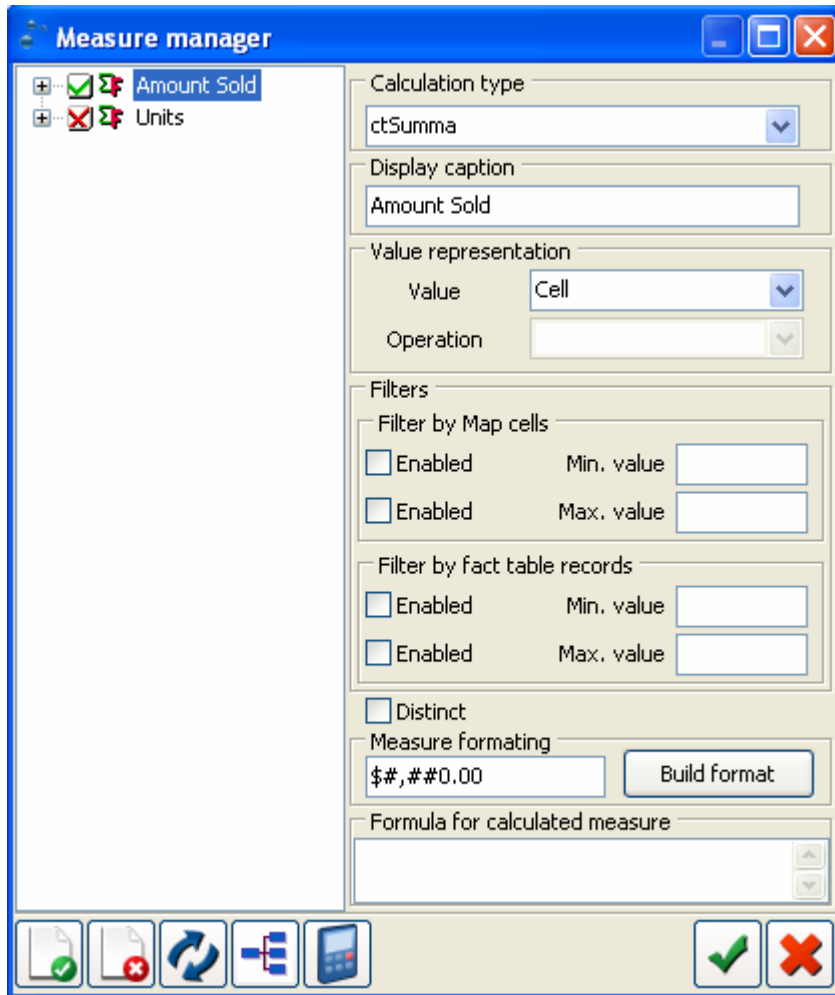
The measure toolbar allows the end user to modify the default settings for the added measures.



### Expand/Collapse Toolbar

You can collapse the measure toolbar in order to have a bigger display area. The button (☐) acts as a toggle.

### Activate global measure manager



The measure manager is used to easily set the attributes of each measure item. Among the attributes you can set are the visibility of each measure, the displayed measure value and the filtered valued. On the example to the left, to display the Units measure, click on the Red x to change to a Green check mark.

When a measure item is not displayed on the measures toolbar, use the global measure manager to set its state to visible again.

You can also add a new calculated measure by using the calculated

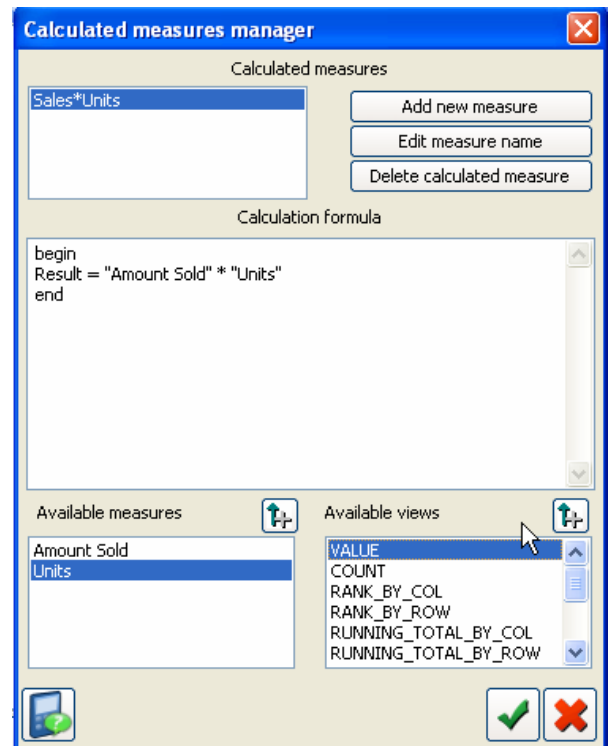


icon (☒) on bottom toolbar. This will bring up the same form as the 'Activate Calculated Measures Manager' button.

## Calculated Measures Manager

The calculated measures manager dialog is used to add additional measures. The value of these calculated measures may use the values of the defined measures, much like how calculated fields are used in a database table.

In this example we have added a new measure that multiplies the Amount sold by the number of units sold.

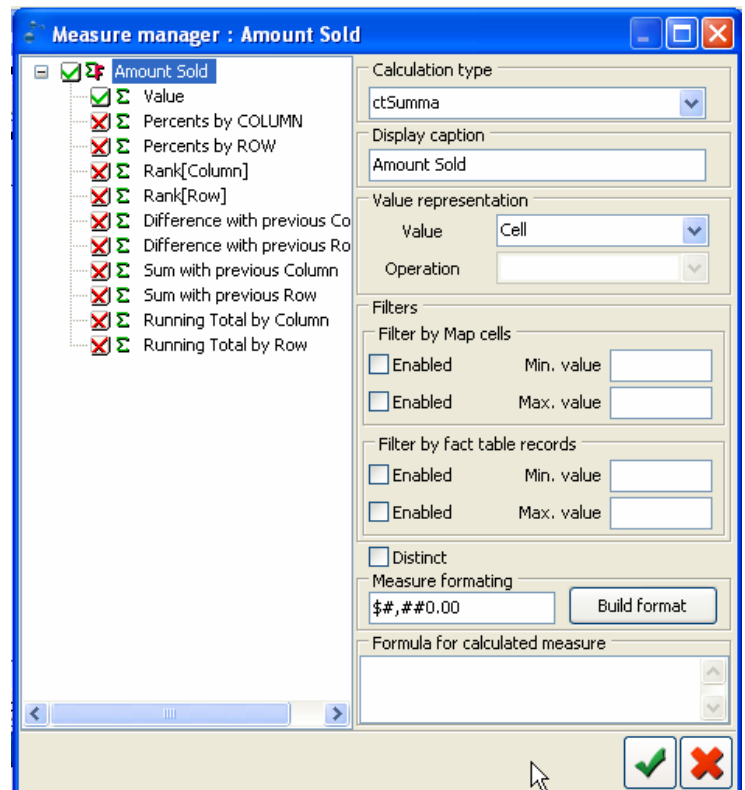


## Measure Items

Display the 'visible' or active measures.

## Activate individual Measure Manager

The individual measure manager is used to set the attributes of the selected measure. It is identical to the global measure manager, except that you can only set the attributes of the selected measure.



In the following sample, we have dragged the Month dimension into the Dimension Row Toolbar, the Employee\_ID dimension into the Dimension Column Toolbar and checked the Units measure to be visible.

The screenshot shows the 'Test Run' window with the following configuration:


- Available Dimensions:** Empty
- Columns:** Employee\_ID
- Measures:** Amount Sold, Units

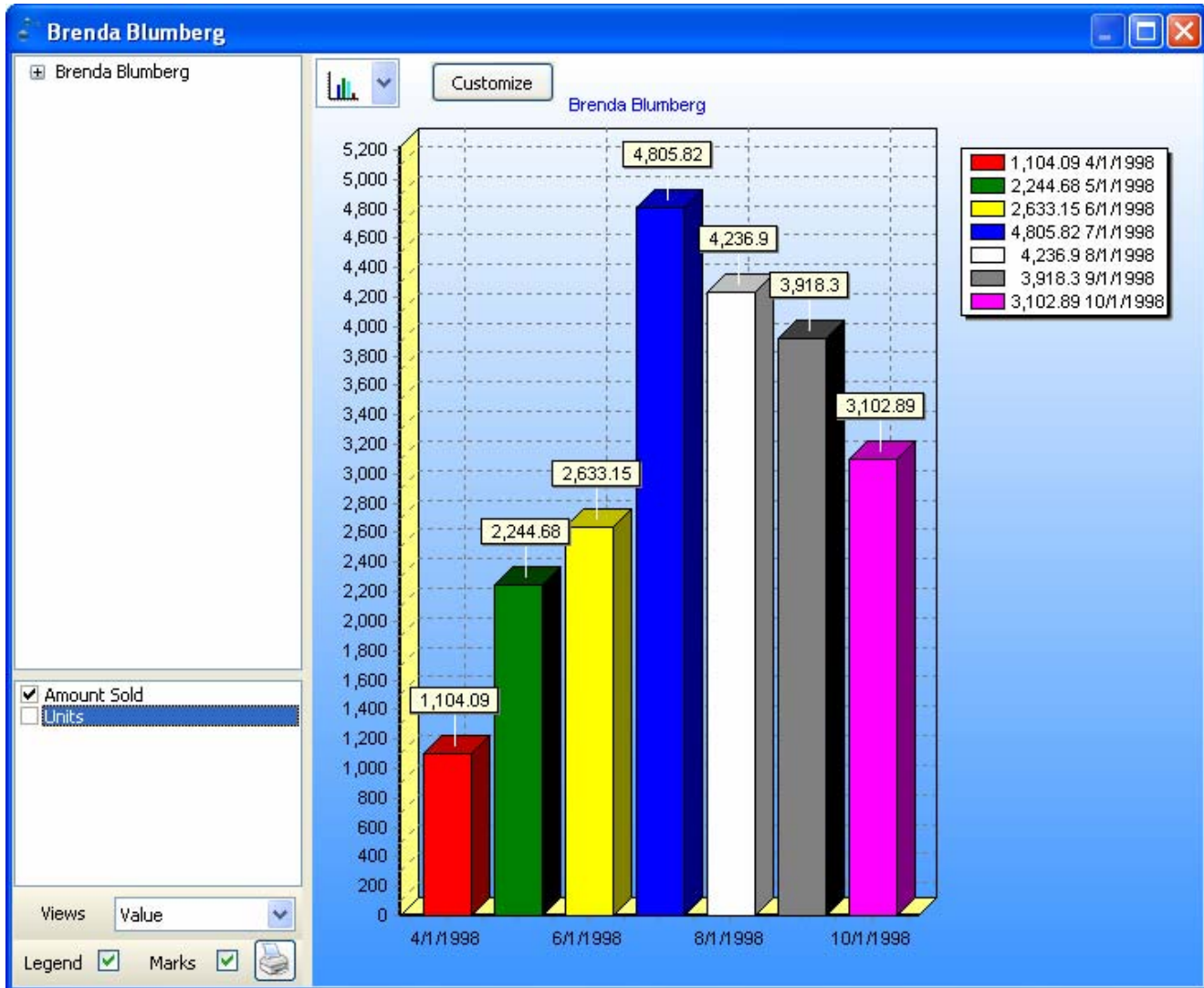
Month	Employee_ID	Brenda Blumberg		Darren Stanz		Rebec
		Amount Sold	Units	Amount Sold	Units	Amou
	Value	Value	Value	Value		
1/1/1997		\$0.00	0	\$486.58	244	
2/1/1997		\$0.00	0	\$675.53	320	
3/1/1997		\$0.00	0	\$540.97	275	
4/1/1997		\$0.00	0	\$587.31	281	
5/1/1997		\$0.00	0	\$964.60	479	
6/1/1997		\$0.00	0	\$878.50	449	
7/1/1997		\$0.00	0	\$1,228.89	564	
8/1/1997		\$0.00	0	\$905.87	446	
9/1/1997		\$0.00	0	\$917.86	453	
10/1/1997		\$0.00	0	\$872.01	425	
11/1/1997		\$0.00	0	\$1,660.16	786	
Total by COLUMNS		\$22,045.83	10171	\$46,012.45	21450	

At the bottom, the 'evisions' logo is visible, along with a field 'Retrieve a maximum 100000 record(s)' and 'Get' and 'Close' buttons.

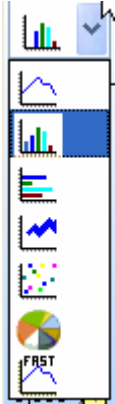
The user has complete control over what is displayed and how displayed.

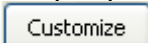
## Graphs

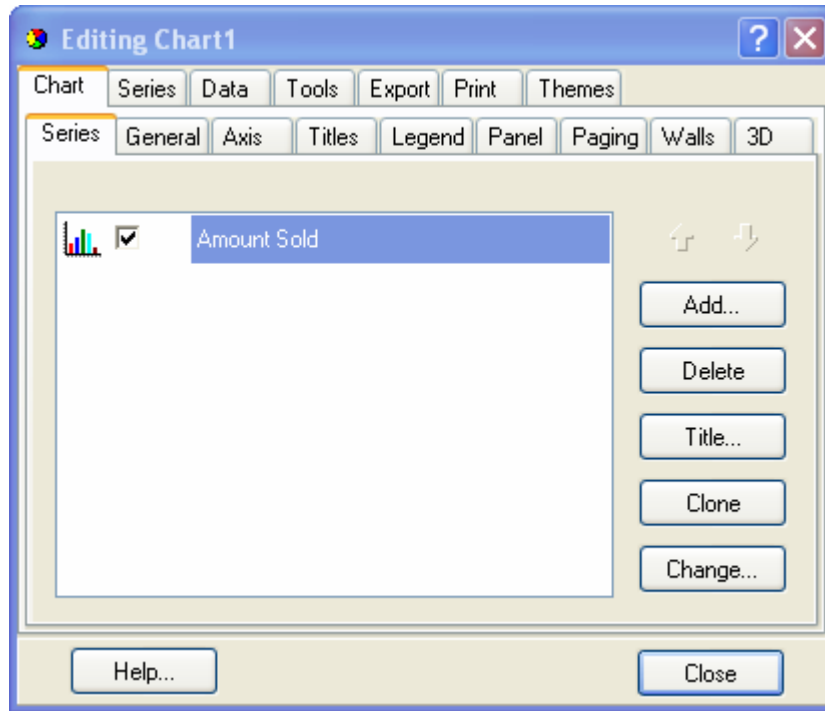
In the above cube, note the  icon on the Column Dimensions. If click, the user will have access to different charts. In this example we clicked on the chart icon next to Brandy Blumberg.



The end user can change the style of the chart by select from the drop down list displayed above the chart.



The user has eight predefined styles to select. In addition, other properties of the chart can be modified by pushing the 'customize' button (  ). This brings up the same interface as found in the Band Editor:





## Adding Parameters to the OLAP cube

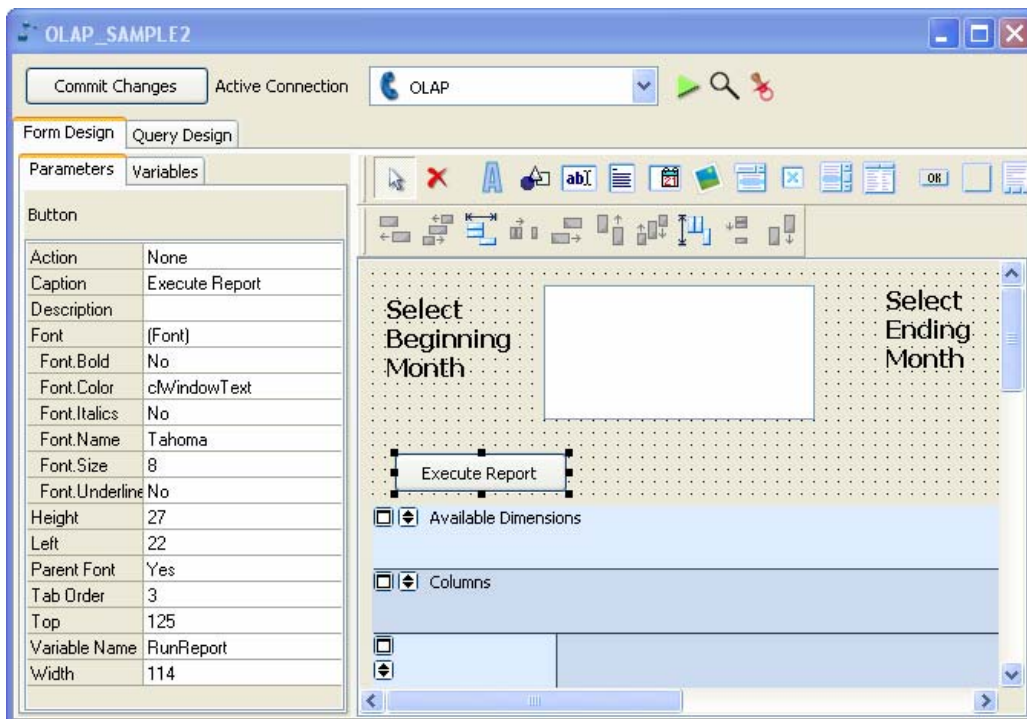
When running an OLAP cube, it is possible to have millions of rows returned. The end user might know ahead of time how they want the data filter and would like the capability of filtering the data before actually running the cube. In this case, we can use Argos to add parameters to the form, and then change the values that are returned by changing the query pulling back the data from the fact table. In addition, because an OLAP cube might return millions of records, we might want to delay the execution of the report based on the end user pressing a 'button' on the parameter form.

In the above example, we might want to limit the output by limiting beginning and ending month. We will add two parameters to do this. First we must change the Alignment of the Cube top allow the addition of the parameters. We will change alignment attribute to None and move the Cube down to provide room for two parameters. We want to add a two list boxes (BeginMonth, EndMonth) to display the Month data so the end user will be able to select the beginning and ending month range to report on. The SQL statement for the BeginMonth is:

```
select DISTINCT SalesFact.Month
from SalesFact
order by SalesFact.Month
```

The EndMonth is similar but the order by will be in Descending order.

Finally we will add a button to the form that when pressed will execute the cube. The button by default has a value of NULL. The attributes for the button we are going to add are shown below:



Now double click on the Cube object and change the select statement for the fact table to include these new variables:

```

select SalesFact.Month,
       SalesFact.State_ID,
       SalesFact.Product_ID,
       SalesFact.Employee_ID,
       SalesFact.Sales_Dollars,
       SalesFact.Sales_Units,
       SalesFact.SubCategory_ID
from SalesFact
where SalesFact.Month between :BeginMonth.Month and :EndMonth.Month
and :RunReport is not NULL
    
```

Now when we run the report, the end user will be forced to select a beginning and ending month value and the data returned will be filtered to only show data for only the months between the selected values. In addition, nothing will be displayed until the Execute Report button is pressed:

The screenshot shows a 'Test Run' window with two date selection dropdowns. The 'Select Beginning Month' dropdown is set to 1/1/1997, and the 'Select Ending Month' dropdown is set to 6/1/1998. Below these is an 'Execute Report' button. Underneath, there are sections for 'Available Dimensions' (Employee\_ID) and 'Columns' (Month). The main area displays a table with the following data:

Month	Value
1/1/1997	\$486.58
2/1/1997	\$675.53
3/1/1997	\$540.97
4/1/1997	\$587.31
5/1/1997	\$964.60
6/1/1997	\$878.50
7/1/1997	\$1,228.89
8/1/1997	\$905.87
9/1/1997	\$917.86
10/1/1997	\$872.01
11/1/1997	\$1,660.16
12/1/1997	\$1,805.03
1/1/1998	\$1,463.49
2/1/1998	\$1,416.26
<b>Total by COLUMNS</b>	<b>\$14,403.06</b>

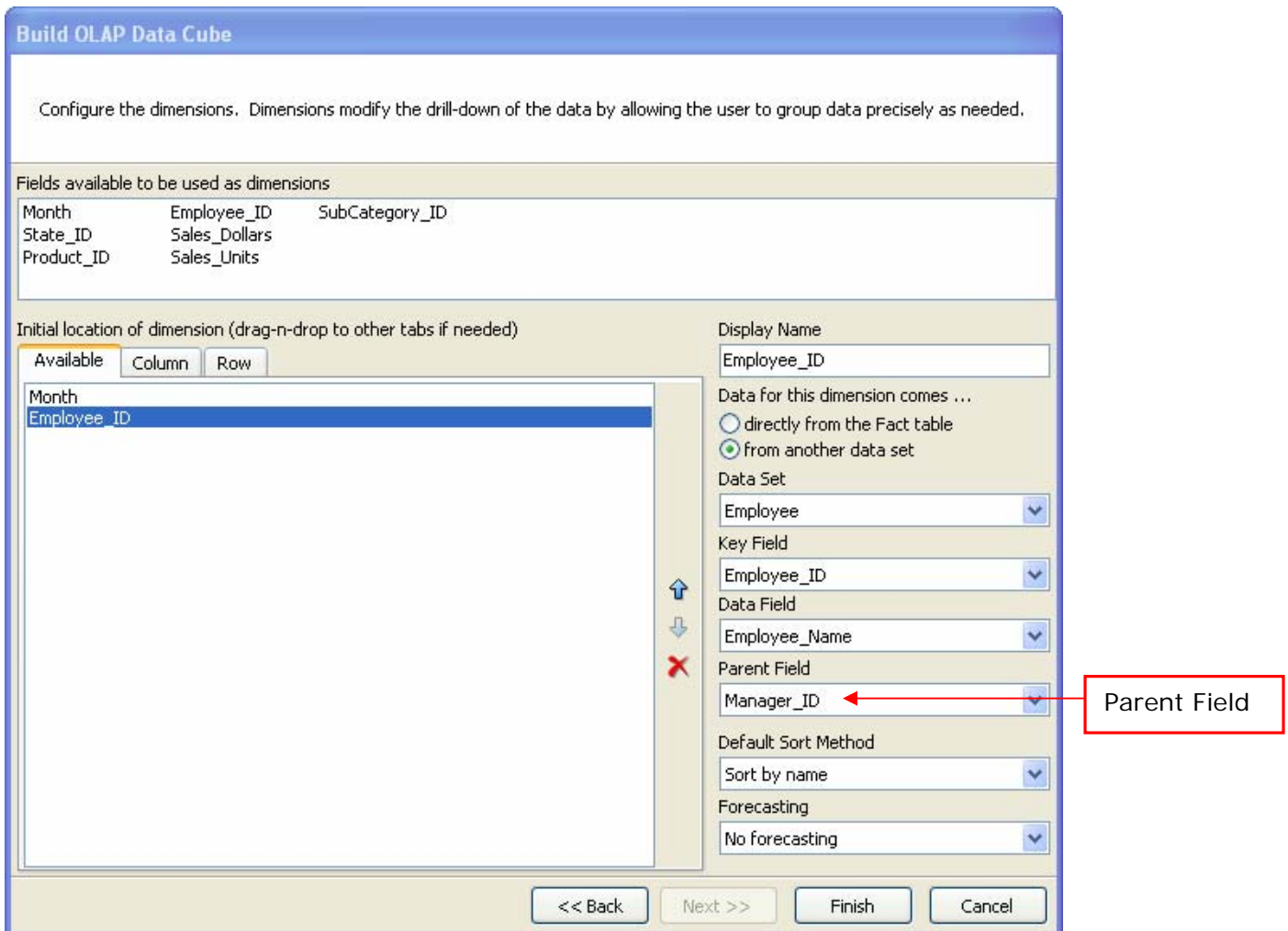
At the bottom of the window, there is a status bar with 'evisions' logo, a text field 'Retrieve a maximum 100000 record(s)', and 'Get' and 'Close' buttons.

## Using Parent/Child relationship

Another capability of OLAP reporting is to build a child – parent relationships between different elements. For example, it would be nice to get a report with a break down of managers and who reports to them.

If we look at the employee table, note that each employee has an associated Manager ID which points back to the employee ID. The hierarchy is built into the table and we can take advantage of that. Note that this only works with this type of built in hierarch.

Using the Manager\_ID field as the parent, we can set up the 'parent/child' relationship so that we will get a break down of Managers with a sub-listing of employees who report to them. We will need to go back into the definition of the cube and change the employee dimension to look like the following:



Note that we have added only the parent field.

When we run the report, notice that all the employees are rolled up to the Manager level.

The screenshot shows the 'Test Run' window with the following configuration:

- Available Dimensions: Month
- Columns: Employee\_ID
- Measures: Amount Sold

Total by ROWS	
Amount Sold	Value
Sheri Nowmer	\$76,741.28
Total by COLUMNS	\$76,741.28

At the bottom, the 'evisions' logo is visible, along with a field for 'Retrieve a maximum 100000 record(s)' and 'Get' and 'Close' buttons.

When we expand, we can not get a break down of employee, manager.

The screenshot shows the 'Test Run' window with the following configuration:

- Available Dimensions: Month
- Columns: Employee\_ID
- Measures: Amount Sold

Total by ROWS	
Employee_ID	Value
Sheri Nowmer	\$76,741.28
Darren Stanz	\$46,012.45
Maya Gutierrez	\$30,728.83
Brenda Blumberg	\$22,045.83
Jonathan Murrain	\$8,683.00
Rebecca Kanagaki	\$8,020.38
Walter Cavestany	\$662.62
Total by COLUMNS	\$76,741.28

At the bottom, the 'evisions' logo is visible, along with a field for 'Retrieve a maximum 100000 record(s)' and 'Get' and 'Close' buttons.

## Appendix A – Database

Following is a description of the Microsoft Access Database used in the above examples:

### Table: SalesFact

#### Columns:

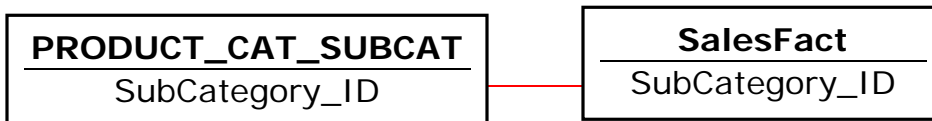
Name	Type
Month	Date/Time
State_ID	Long Integer
Product_ID	Long Integer
Employee_ID	Long Integer
Sales_Dollars	Currency
Sales_Units	Double
SubCategory_ID	Long Integer

#### Relationships:

##### EmployeeSalesFact



##### Product\_CAT\_SUBCATSalesFact



##### ProductSalesFact



##### StateSalesFact



**Table: Employee**

**Columns:**

<b>Name</b>	<b>Type</b>
Employee_ID	Long Integer
Employee_Name	Text
Manager_ID	Long Integer

**Table: Product**

**Columns:**

<b>Name</b>	<b>Type</b>
Product_ID	Long Integer
Brand_Name	Text
Product_NAame	Text
Price	Currency
SubCategory_ID	Long Integer

**Table: Product\_Cat\_Subcat**

**Columns:**

<b>Name</b>	<b>Type</b>
Category_ID	Long Integer
Category	Text
SubCategory_ID	Long Integer
SubCategory	Text

**Table: State**

**Columns:**

<b>Name</b>	<b>Type</b>
State_ID	Long Integer
State_Name	Text
Region	Text
Country	Text

